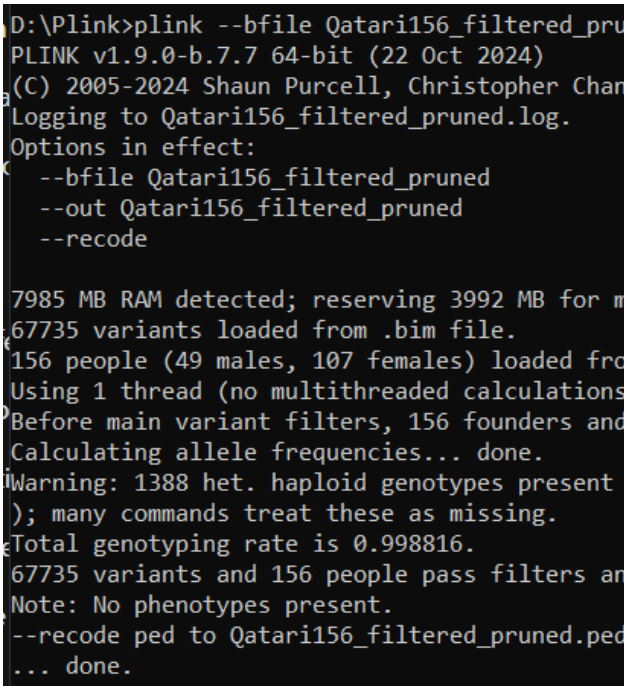
Bioinformatics Project

Ibrahim Mohamed Elsayed Moustafa Omar | 21010023 Mohamed Mohamed Mohamed Abdelmonaem | 2

```
# Run before commiting
save.image(file = "my_workspace.RData")
```



#Reading the data ##Converting .bed .bim .fam files to .map .ped files

##Converting .map .ped files to .gds files

```
# install.packages("BiocManager")
# BiocManager::install(c("SNPRelate", "GENESIS", "GWASTools"), force = TRUE)
# install.packages("coxme")
# install.packages("GeneNet")
# install.packages("openxlsx")
# install.packages("qqman")
# BiocManager::install("SummarizedExperiment")
# install.packages('tinytex')
# tinytex::install_tinytex()
```

```
library(SNPRelate)
```

```
## Loading required package: gdsfmt
```

```
## SNPRelate -- supported by Streaming SIMD Extensions 2 (SSE2)
```

```r
# Define the file names
ped.fn <- "Qatari156_filtered_pruned.ped"
map.fn <- "Qatari156_filtered_pruned.map"
gds.fn <- "Qatari156_filtered_pruned.gds"

# Convert to GDS format
snpgdsPED2GDS(ped.fn, map.fn, out.gdsfn = gds.fn)
```

```
## PLINK PED/MAP to GDS Format:
## Import 67735 variants from 'Qatari156_filtered_pruned.map'
## Chromosome:
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 5224 5144 4369 4124 4254 3976 3517 3346 3043 3553 3131 3412 2577 2303 2204 2316
##   17   18   19   20   21   22   23
## 1990 2214 1283 1895 1074 1129 1657
## Reading 'Qatari156_filtered_pruned.ped'
## Output: 'Qatari156_filtered_pruned.gds'
## Import 156 samples
## Transpose the genotypic matrix ...
## Done.
## Optimize the access efficiency ...
## Clean up the fragments of GDS file:
##     open the file 'Qatari156_filtered_pruned.gds' (5.7M)
##     # of fragments: 52
##     save to 'Qatari156_filtered_pruned.gds.tmp'
##     rename 'Qatari156_filtered_pruned.gds.tmp' (3.1M, reduced: 2.5M)
##     # of fragments: 26
```

```r
# Open the GDS file
genofile <- snpgdsOpen(gds.fn)

# Check the GDS file
snpgdsSummary(genofile)
```

```
## The file name: C:\Users\KimoStore\Desktop\Bioinformatics-Project\Qatari156_filtered_pruned.gds
## The total number of samples: 156
## The total number of SNPs: 67735
## SNP genotypes are stored in SNP-major mode (Sample X SNP).
```

```r
# Closing the GDS file
snpgdsClose(genofile)
```

##Reading the metabolites data from .csv file

```r
metabolites <- read.csv("qatari_metabolites_2025.csv", header = TRUE)

# Check the first few rows of the data
head(metabolites)
```

```
##     Sample Metabolite1 Metabolite2 Metabolite3 Metabolite4 Metabolite5
## 1  QBC-092    47.42229    77.15461    83.85559    18.78791    45.22587
```

```
## 2  QBC-256    77.68558    62.73445    81.58909    58.52761    38.10181
## 3  QBC-107    93.58374    37.05494    68.86348    90.03878    45.68474
## 4  QBC-171    66.40978    80.26947    65.04509    40.99331    57.83323
## 5 QPRC-110    82.63487    30.33938    75.84115    74.08630    57.54897
## 6  QBC-240    61.60166    49.75713    73.35939    79.39279    72.88979
##   Metabolite6 Metabolite7 Metabolite8 Metabolite9 Metabolite10 Metabolite11
## 1    81.55610    76.75280    83.30655    58.71107     68.57458     70.81031
## 2    64.82085    67.58087    53.68596    75.61975     67.63228     79.60736
## 3    70.80035    69.19010    65.55272    60.80358     64.57745     72.86564
## 4    61.23251    66.31334    60.35562    51.15013     75.49514     60.13897
## 5    36.50677    66.09879    45.80220    78.71158     61.96916     71.64201
## 6    55.52628    79.88726    78.44529    57.12925     61.93932     89.71886
##   Metabolite12 Metabolite13 Metabolite14 Metabolite15 Metabolite16 Metabolite17
## 1     64.54457     66.16694     52.89463     63.32401     89.99696     41.59273
## 2     45.33667     67.75164     75.44470     46.46178     60.52969     71.81184
## 3     71.56960     69.16945     18.09773     78.81906     63.58670     73.21806
## 4     50.10261     85.18790     70.69852     70.93467     46.05606     58.61268
## 5     78.11993     50.75500     80.29826     91.72456     59.73534     60.61903
## 6     90.80418     75.04140     78.16688     86.63198     70.44270     81.00818
##   Metabolite18 Metabolite19 Metabolite20
## 1     62.88971     77.03925     54.96671
## 2     46.94935     65.27621     64.24428
## 3     91.85770     68.09990     80.66064
## 4     42.94329     77.29895     68.50738
## 5     70.19732     66.39820     41.59022
## 6     57.84487     62.88352     49.58120
```

#Task 1: Compute Kinship using SNPRelate and GENESIS ###Compute IBD Coefficients (Kinship Estimation)

```
# Complete Kinship Analysis Workflow (Verified 2024)
library(SNPRelate)
library(GENESIS)
library(GWASTools)
```

```
## Loading required package: Biobase

## Loading required package: BiocGenerics

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, saveRDS, setdiff,
##     table, tapply, union, unique, unsplit, which.max, which.min
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
library(BiocParallel)

# Compute KING kinship with SNPRelate
gds <- snpgdsOpen(gds.fn)
king_kinship <- snpgdsIBDKING(gds, num.thread = 2)
```

```
## IBD analysis (KING method of moment) on genotypes:
## Excluding 1,657 SNPs on non-autosomes
## Excluding 0 SNP (monomorphic: TRUE, MAF: NaN, missing rate: NaN)
##     # of samples: 156
##     # of SNPs: 66,078
##     using 2 threads
## No family is specified, and all individuals are treated as singletons.
## Relationship inference in the presence of population stratification.
## KING IBD:    the sum of all selected genotypes (0,1,2) = 16145834
## CPU capabilities: Double-Precision SSE2
## Tue May 20 02:33:00 2025    (internal increment: 65536)
## [..................................................]  0%, ETC: ---        [==========================
## Tue May 20 02:33:01 2025    Done.
```

```
closefn.gds(gds)

# Prepare GENESIS objects
geno <- GdsGenotypeReader(gds.fn)
genoData <- GenotypeData(geno)

# Add sample IDs to kinship matrix
sample_ids <- getScanID(genoData)
kinship_matrix <- king_kinship$kinship
dimnames(kinship_matrix) <- list(sample_ids, sample_ids)

# Run PC-AiR
pca <- pcair(genoData,
        kinobj = kinship_matrix,
        divobj = kinship_matrix,
        kin.thresh = 0.0442,
        div.thresh = -0.0442
)
```

```
## Using kinobj and divobj to partition samples into unrelated and related sets

## Working with 156 samples

## Identifying relatives for each sample using kinship threshold 0.0442

## Identifying pairs of divergent samples using divergence threshold -0.0442
```

```
## Partitioning samples into unrelated and related sets...

## Unrelated Set: 153 Samples
## Related Set: 3 Samples

## Performing PCA on the Unrelated Set...

## Principal Component Analysis (PCA) on genotypes:
## Excluding 1,657 SNPs on non-autosomes
## Excluding 0 SNP (monomorphic: TRUE, MAF: NaN, missing rate: NaN)
##     # of samples: 153
##     # of SNPs: 66,078
##     using 1 thread
##     # of principal components: 32
## PCA:    the sum of all selected genotypes (0,1,2) = 15841547
## CPU capabilities: Double-Precision SSE2
## Tue May 20 02:33:01 2025    (internal increment: 6612)
## [..................................................]  0%, ETC: ---        [============================
## Tue May 20 02:33:01 2025    Begin (eigenvalues and eigenvectors)
## Tue May 20 02:33:01 2025    Done.

## Predicting PC Values for the Related Set...

## SNP Loading:
##     # of samples: 153
##     # of SNPs: 66,078
##     using 1 thread
##     using the top 32 eigenvectors
## SNP Loading:    the sum of all selected genotypes (0,1,2) = 15841547
## Tue May 20 02:33:01 2025    (internal increment: 52896)
## [..................................................]  0%, ETC: ---        [============================
## Tue May 20 02:33:02 2025    Done.
## Sample Loading:
##     # of samples: 3
##     # of SNPs: 66,078
##     using 1 thread
##     using the top 32 eigenvectors
## Sample Loading:    the sum of all selected genotypes (0,1,2) = 304287
## Tue May 20 02:33:02 2025    (internal increment: 65536)
## [..................................................]  0%, ETC: ---        [============================
## Tue May 20 02:33:02 2025    Done.
```

```r
# Prepare for PC-Relate with current syntax
iterator <- GenotypeBlockIterator(genoData)

# Current working version of pcrelate (as of GENESIS 2.22.0)
pcrelate_results <- pcrelate(iterator,
        pcs = pca$vectors[, 1:2], # Note 'pcs' parameter
        training.set = pca$unrels,
        verbose = TRUE
)
```

```
## Using 6 CPU cores
```

```
## 156 samples to be included in the analysis...

## Betas for 2 PC(s) will be calculated using 153 samples in training.set...

## Running PC-Relate analysis for 156 samples using 67735 SNPs in 7 blocks...

## Performing Small Sample Correction...
```

```
# Get kinship matrix
kinship_matrix <- pcrelateToMatrix(pcrelate_results)
```

```
## Using 156 samples provided

## Identifying clusters of relatives...

##     156 relatives in 1 clusters; largest cluster = 156

## Creating block matrices for clusters...

## 0 samples with no relatives included
```

```
# Clean up
close(genoData)
```

##Report the number of individuals who have a kinship > 0.1

```
# Convert kinship matrix to a regular matrix if it's a Matrix object
if (inherits(kinship_matrix, "Matrix")) {
        kinship_matrix <- as.matrix(kinship_matrix)
}

# Get all pairwise combinations
sample_ids <- rownames(kinship_matrix)
kinship_pairs <- data.frame(
        ID1 = rep(sample_ids, each = length(sample_ids)),
        ID2 = rep(sample_ids, times = length(sample_ids)),
        kinship = as.vector(kinship_matrix),
        stringsAsFactors = FALSE
)

# Remove self-comparisons and keep only unique pairs (upper triangle)
kinship_pairs <- kinship_pairs[kinship_pairs$ID1 != kinship_pairs$ID2, ]
kinship_pairs <- kinship_pairs[!duplicated(t(apply(kinship_pairs[, 1:2], 1, sort))), ]

# Filter for kinship > 0.1
high_kinship <- kinship_pairs[kinship_pairs$kinship > 0.1, ]

# Count results
num_pairs <- nrow(high_kinship)
num_individuals <- length(unique(c(high_kinship$ID1, high_kinship$ID2)))

# Report results
cat("Number of individual pairs with kinship > 0.1:", num_pairs, "\n")
```

```
## Number of individual pairs with kinship > 0.1: 9
```

```
cat("Number of unique individuals involved:", num_individuals, "\n")
```

```
## Number of unique individuals involved: 11
```

#Task 2: Compute mQTLs with Mixed Models

load needed packages

```
library(SNPRelate)
library(SeqVarTools)
```

```
## Loading required package: SeqArray
```

```
##
## Attaching package: 'SeqVarTools'
```

```
## The following objects are masked from 'package:GWASTools':
##
##     alleleFrequency, duplicateDiscordance, getGenotype, iterateFilter,
##     lastFilter, lastFilter<-, mendelErr, resetIterator
```

```
library(GENESIS)
library(SummarizedExperiment)
```

```
## Loading required package: MatrixGenerics
```

```
## Loading required package: matrixStats
```

```
## Warning: package 'matrixStats' was built under R version 4.4.3
```

```
##
## Attaching package: 'matrixStats'
```

```
## The following object is masked from 'package:SeqArray':
##
##     rowRanges
```

```
## The following objects are masked from 'package:Biobase':
##
##     anyMissing, rowMedians
```

```
##
## Attaching package: 'MatrixGenerics'
```

```
## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## The following object is masked from 'package:SeqArray':
##
##     rowRanges

## The following object is masked from 'package:Biobase':
##
##     rowMedians

## Loading required package: GenomicRanges

## Loading required package: stats4

## Loading required package: S4Vectors

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:utils':
##
##     findMatches

## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:grDevices':
##
##     windows
```

```
## Loading required package: GenomeInfoDb
```

```
##
## Attaching package: 'SummarizedExperiment'
```

```
## The following object is masked from 'package:SeqArray':
##
##     colData
```

```
library(writexl)
```

```
## Warning: package 'writexl' was built under R version 4.4.3
```

```
metab_t <- metabolites
rownames(metab_t) <- metabolites$Sample
setdiff(covar$sample.id, rownames(metab_t))
```

```
## character(0)
```

```
# metab_t <- as.data.frame(t(metab_t[ , -1]))

metab_t <- (metab_t[, -1])
```

```
# Make sample-level table
sampleTable <- merge(covar, data.frame(sample.id = rownames(metab_t)), by = "sample.id")

# Check overlap
common_ids <- intersect(sampleTable$sample.id, rownames(metab_t))
length(common_ids) # Should match nrow(sampleTable)
```

```
## [1] 156
```

```
# Subset both to common IDs
sampleTable_sub <- sampleTable[sampleTable$sample.id %in% common_ids, ]
metab_sub <- metab_t[common_ids, ]

# Reorder both to the same sample ID order
sampleTable_sub <- sampleTable_sub[order(sampleTable_sub$sample.id), ]
metab_sub <- metab_sub[order(rownames(metab_sub)), ]

metab_fixed <- t(metab_sub) # Now 20 rows × 156 columns

# Check alignment again
stopifnot(identical(colnames(metab_fixed), sampleTable_sub$sample.id))

# Build the SE object
se <- SummarizedExperiment(
        assays = SimpleList(metab = metab_fixed),
        colData = sampleTable_sub
)
```

```r
all(rownames(kinMat) %in% pd$sample.id) # Should be TRUE
```

```
## [1] TRUE
```

```r
all(pd$sample.id %in% rownames(kinMat)) # Should be TRUE
```

```
## [1] TRUE
```

```r
all(rownames(kinMat) == colnames(kinMat)) # Should be TRUE (kinship matrix must be symmetric)
```

```
## [1] TRUE
```

```r
library(Matrix)
```

```
##
## Attaching package: 'Matrix'
```

```
## The following object is masked from 'package:S4Vectors':
##
##     expand
```

```r
kinMat_PD <- as.matrix(nearPD(kinMat)$mat)
nullMods <- list()

for (m in rownames(assay(se, "metab"))) {
        pheno <- data.frame(
                sample.id = colnames(assay(se, "metab")),
                y = assay(se, "metab")[m, ]
        )

        pd <- merge(sampleTable, pheno, by = "sample.id")
        rownames(pd) <- pd$sample.id
        pd <- pd[rownames(kinMat_PD), ]

        nullMods[[m]] <- GENESIS::fitNullModel(
                x = pd,
                outcome = "y",
                covars = c("PC1", "PC2", "PC3"),
                family = "gaussian",
                cov.mat = kinMat_PD
        )
}
```

```
## Computing Variance Component Estimates...
```

```
## Sigma^2_A     log-lik     RSS
```

```
## [1]  140.150908  140.150908 -646.632734    1.984113
## [1]  101.772002  210.540835 -646.614895    1.326416
## [1]   47.926627  237.012041 -646.608954    1.181129
## [1]    3.537727  255.606143 -646.606627    1.097031
## [1]    1.745208  256.328736 -646.606567    1.094007
## [1]    0.8725459  256.6807124 -646.6065391    1.0925406
## [1]    0.000000  257.028128 -646.606511    1.091097
## [1]    0.00000  278.48771 -646.60651    1.00702
## [1]    0.000000  280.428985 -646.606511    1.000049

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  142.314187  142.314187 -648.857455    2.011996
## [1]  756.335958  207.463159 -648.350150    1.341832
## [1] 2125.563758  248.453931 -647.817040    1.074666
## [1] 3707.812576  254.242246 -647.550730    1.009152
## [1] 4479.936797  251.803625 -647.511476    1.000907
## [1] 4655.31317  251.03527 -647.50974    1.00004
## [1] 4682.941497  250.889157 -647.509698    1.000001
## [1] 4686.9790  250.8665 -647.5097    1.0000
## [1] 4687.5642  250.8632 -647.5097    1.0000
## [1] 4687.6489  250.8627 -647.5097    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  138.269542  138.269542 -646.178645    1.999129
## [1]  301.248760  206.510795 -646.164217    1.333046
## [1]  525.452023  256.784381 -646.154176    1.066693
## [1]  659.510674  271.980798 -646.152142    1.003967
## [1]  676.402863  272.937402 -646.152111    1.000016
## [1]  676.3396  272.9425 -646.1521    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  160.421757  160.421757 -655.314173    1.943159
## [1]  779.060471  232.072719 -654.921638    1.310801
## [1] 1999.533440  276.946347 -654.581235    1.063263
## [1] 3136.560974  285.339581 -654.463695    1.005672
## [1] 3504.567545  284.593007 -654.455251    1.000228
## [1] 3546.712592  284.393729 -654.455147    1.000003
## [1] 3549.9444  284.3740 -654.4551    1.0000
## [1] 3550.1837  284.3725 -654.4551    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  172.144976  172.144976 -659.594902    1.915751
## [1]   48.05478  214.53536 -659.53522    1.54650
## [1]    7.638025  224.354940 -659.522802    1.481097
```

```
## [1]    2.547094  225.535724 -659.521380    1.473618
## [1]    0.000000  226.123004 -659.520679    1.469928
## [1]    0.000000  298.413276 -659.520679    1.113839
## [1]    0.000000  328.912470 -659.520679    1.010556
## [1]    0.000000  332.348214 -659.520679    1.000109
## [1]    0.0000  332.3845 -659.5207    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  121.245311  121.245311 -634.870112    1.964631
## [1]   53.007890  151.780425 -634.832491    1.577564
## [1]    0.7816268  166.1412632 -634.8151855    1.4453321
## [1]    0.3417381  166.2449546 -634.8150604    1.4444628
## [1]    0.1217424  166.2967636 -634.8149979    1.4440289
## [1]    0.000000  166.322659 -634.814963    1.443813
## [1]    0.000000  217.448503 -634.814963    1.104348
## [1]    0.000000  237.994848 -634.814963    1.009008
## [1]    0.00000  240.11968 -634.81496    1.00008

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  156.525958  156.525958 -655.487361    1.996066
## [1]   54.77254  167.03301 -655.38561    1.87902
## [1]   26.621003  169.675039 -655.359627    1.851957
## [1]   12.238729  170.996388 -655.346647    1.838743
## [1]    4.973625  171.657027 -655.340163    1.832214
## [1]    1.322925  171.987323 -655.336922    1.828969
## [1]    0.4080029  172.0698929 -655.3361122    1.8281602
## [1]    0.1791326  172.0905352 -655.3359096    1.8279580
## [1]    0.06468002  172.10085628 -655.33580832    1.82785693
## [1]    0.000000  172.106017 -655.335751    1.827807
## [1]    0.000000  250.052190 -655.335751    1.258044
## [1]    0.00000  301.34165 -655.33575    1.04392
## [1]    0.000000  314.019753 -655.335751    1.001773
## [1]    0.000000  314.575590 -655.335751    1.000003

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  123.9966  123.9966 -638.9131    2.0260
## [1]   39.350761  132.416988 -638.781846    1.905585
## [1]   17.153904  134.531174 -638.750452    1.877712
## [1]    5.962451  135.588021 -638.735044    1.864101
## [1]    0.3460602  136.1162795 -638.7274148    1.8573751
## [1]    0.1702453  136.1327842 -638.7271770    1.8571658
## [1]    0.08233317  136.14103649 -638.72705818    1.85706121
## [1]    0.03837597  136.14516262 -638.72699876    1.85700890

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS
```

```
## [1]  161.385244  161.385244 -655.745165    1.942543
## [1]   59.779712  171.837511 -655.614407    1.831665
## [1]    8.472631  177.048463 -655.555954    1.781168
## [1]    2.062261  177.696878 -655.548976    1.775086
## [1]    0.4601473  177.8588671 -655.5472423    1.7735737
## [1]    0.05965213  177.89935686 -655.54680971    1.77319606
## [1]    0.000000  177.904418 -655.546745    1.773149
## [1]    0.000000  255.476394 -655.546745    1.234756
## [1]    0.000000  304.048511 -655.546745    1.037503
## [1]    0.000000  315.038958 -655.546745    1.001308
## [1]    0.000000  315.450590 -655.546745    1.000002

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  119.22200  119.22200 -634.46345    1.98731
## [1]   92.949641  179.151192 -634.448380    1.327811
## [1]    5.774130  224.319701 -634.440474    1.065011
## [1]    2.795792  225.202327 -634.440378    1.060963
## [1]    1.390946  225.619323 -634.440334    1.059061
## [1]    0.0262863  226.0247601 -634.4402940    1.0572188

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  126.182791  126.182791 -639.615095    2.009373
## [1]   53.629979  142.655042 -639.543349    1.785387
## [1]   11.22789  150.83727 -639.51119    1.69238
## [1]    0.1336108  152.8516998 -639.5036156    1.6710194
## [1]    0.04622723  152.86735819 -639.50355717    1.67085563
## [1]    0.000000  152.875187 -639.503526    1.670774
## [1]    0.000000  214.250751 -639.503526    1.192154
## [1]    0.000000  248.784151 -639.503526    1.026673
## [1]    0.000000  255.247487 -639.503526    1.000675
## [1]    0.0000  255.4198 -639.5035    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  150.583257  150.583257 -652.972044    2.007294
## [1]   42.808919  170.345757 -652.874674    1.783818
## [1]   12.247512  175.255799 -652.852367    1.736156
## [1]    4.472788  176.475202 -652.846978    1.724731
## [1]    0.5715212  177.0837062 -652.8443144    1.7190877
## [1]    0.08304424  177.15969095 -652.84398281    1.71838574
## [1]    0.0000  177.1692 -652.8439    1.7183
## [1]    0.000000  251.231104 -652.843926    1.211752
## [1]    0.000000  295.133368 -652.843926    1.031499
## [1]    0.000000  304.145869 -652.843926    1.000933
## [1]    0.000000  304.429488 -652.843926    1.000001

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS
```

```
## [1]  151.852679  151.852679 -651.795001    1.959923
## [1]   28.700242  162.018844 -651.662815    1.847053
## [1]   11.799984  163.296663 -651.645721    1.833916
## [1]    3.266184  163.935669 -651.637169    1.827425
## [1]    1.122499  164.095426 -651.635028    1.825810
## [1]    0.04939006  164.17530407 -651.63395832    1.82500408
## [1]    0.00000  164.17780 -651.63391    1.82498
## [1]    0.000000  238.394182 -651.633909    1.256831
## [1]    0.000000  287.109590 -651.633909    1.043578
## [1]    0.000000  299.098759 -651.633909    1.001747
## [1]    0.000000  299.620310 -651.633909    1.000003

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  126.628339  126.628339 -638.191343    1.965142
## [1]   48.951130  130.991756 -638.056499    1.906723
## [1]    9.035393  133.183195 -637.989508    1.878830
## [1]    3.992535  133.457499 -637.981147    1.875404
## [1]    1.468123  133.594668 -637.976970    1.873696
## [1]    0.2051854  133.6632569 -637.9748820    1.8728429
## [1]    0.04727291  133.67183077 -637.97462108    1.87273633
## [1]    0.00000  133.67397 -637.97454    1.87271
## [1]    0.000000  195.967997 -637.974543    1.277416
## [1]    0.000000  238.526290 -637.974543    1.049497
## [1]    0.000000  249.775818 -637.974543    1.002229
## [1]    0.000000  250.331398 -637.974543    1.000005

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  122.896207  122.896207 -636.853237    1.989481
## [1]  196.080581  183.914174 -636.853042    1.328664
## [1]  267.944026  229.204752 -636.852799    1.065179
## [1]  311.711906  242.993150 -636.852613    1.003761
## [1]  332.420094  243.735468 -636.852532    1.000015
## [1]  343.9466  243.6404 -636.8525    1.0000
## [1]  350.6229  243.5834 -636.8525    1.0000
## [1]  354.4809  243.5505 -636.8525    1.0000
## [1]  356.7068  243.5315 -636.8525    1.0000
## [1]  357.9898  243.5206 -636.8525    1.0000
## [1]  358.7290  243.5143 -636.8525    1.0000
## [1]  359.1548  243.5107 -636.8525    1.0000
## [1]  359.4000  243.5086 -636.8525    1.0000
## [1]  359.5412  243.5074 -636.8525    1.0000
## [1]  359.6224  243.5067 -636.8525    1.0000
## [1]  359.6692  243.5063 -636.8525    1.0000
## [1]  359.6962  243.5060 -636.8525    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  190.736956  190.736956 -670.090364    1.985061
```

```
## [1]  249.569216  285.695232 -670.089548    1.326707
## [1]  250.938449  356.568817 -670.088803    1.064564
## [1]  214.307334  378.647480 -670.088502    1.003697
## [1]  194.197157  380.226499 -670.088460    1.000014
## [1]  188.1523  380.2855 -670.0885    1.0000
## [1]  186.4321  380.3008 -670.0885    1.0000
## [1]  185.9417  380.3051 -670.0885    1.0000
## [1]  185.8017  380.3064 -670.0885    1.0000
## [1]  185.7618  380.3067 -670.0885    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  155.733467  155.733467 -654.837211    1.989134
## [1]   29.833800  195.730240 -654.780659    1.593309
## [1]    5.016507  200.493484 -654.772637    1.557077
## [1]    1.819076  201.080262 -654.771641    1.552737
## [1]    0.2149232  201.3730828 -654.7711446    1.5505807
## [1]    0.000000  201.409649 -654.771078    1.550313
## [1]    0.000000  272.903809 -654.771078    1.144169
## [1]    0.000000  307.290484 -654.771078    1.016133
## [1]    0.000000  312.169254 -654.771078    1.000252
## [1]    0.0000  312.2479 -654.7711    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  157.322856  157.322856 -653.592445    1.937051
## [1]  748.403078  226.606885 -653.002933    1.309506
## [1] 1898.423596  269.829102 -652.568399    1.064122
## [1] 2862.62549  279.22067 -652.46908    1.00567
## [1] 3003.647463  279.953744 -652.467916    1.000065
## [1] 2991.2775  280.0547 -652.4679    1.0000
## [1] 2992.8578  280.0443 -652.4679    1.0000
## [1] 2992.6612  280.0456 -652.4679    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  136.049954  136.049954 -644.600796    1.989998
## [1]   51.238871  145.143376 -644.508595    1.873631
## [1]    3.835604  149.716403 -644.461639    1.820612
## [1]    0.7281062  150.0022550 -644.4586534    1.8174121
## [1]    0.3385804  150.0379846 -644.4582799    1.8170131
## [1]    0.1437496  150.0558492 -644.4580932    1.8168137
## [1]    0.04631729  150.06478153 -644.45799979    1.81671396

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  142.960225  142.960225 -648.746662    1.999986
## [1]   14.159152  152.648184 -648.529362    1.882681
```

```
## [1]    6.367699  153.253439 -648.517208    1.875812
## [1]    2.481184  153.555974 -648.511188    1.872397
## [1]    0.5402581  153.7072163 -648.5081912    1.8706946
## [1]    0.05531955  153.74502360 -648.50744357    1.87026959
## [1]    0.000000  153.747387 -648.507358    1.870245
## [1]    0.000000  225.287682 -648.507358    1.276347
## [1]    0.000000  274.065619 -648.507358    1.049184
## [1]    0.000000  286.913352 -648.507358    1.002202
## [1]    0.000000  287.543866 -648.507358    1.000005
```

```r
kinMat_PD <- as.matrix(nearPD(kinMat)$mat)
nullMods <- list()
pheno_data_list <- list() # Store phenotype data frames for creating SeqVarData

for (m in rownames(assay(se, "metab"))) {
        # Create phenotype data frame for the current metabolite
        pheno <- data.frame(
                sample.id = colnames(assay(se, "metab")),
                y = assay(se, "metab")[m, ]
        )

        # Merge with sample metadata and ensure sample order matches kinship matrix
        pd <- merge(sampleTable, pheno, by = "sample.id")
        rownames(pd) <- pd$sample.id
        pd <- pd[rownames(kinMat_PD), ] # Ensure order matches kinMat_PD

        # Store the prepared phenotype data frame for later use with SeqVarData
        pheno_data_list[[m]] <- pd

        # Fit the null model for the current metabolite
        nullMods[[m]] <- GENESIS::fitNullModel(
                x = pd, # Use the prepared data frame
                outcome = "y",
                covars = c("PC1", "PC2", "PC3"),
                family = "gaussian",
                cov.mat = kinMat_PD
        )
}
```

```
## Computing Variance Component Estimates...

## Sigma^2_A     log-lik     RSS

## [1]  140.150908  140.150908 -646.632734    1.984113
## [1]  101.772002  210.540835 -646.614895    1.326416
## [1]   47.926627  237.012041 -646.608954    1.181129
## [1]    3.537727  255.606143 -646.606627    1.097031
## [1]    1.745208  256.328736 -646.606567    1.094007
## [1]    0.8725459  256.6807124 -646.6065391    1.0925406
## [1]    0.000000  257.028128 -646.606511    1.091097
## [1]    0.00000  278.48771 -646.60651    1.00702
## [1]    0.000000  280.428985 -646.606511    1.000049
```

```
## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  142.314187  142.314187 -648.857455    2.011996
## [1]  756.335958  207.463159 -648.350150    1.341832
## [1] 2125.563758  248.453931 -647.817040    1.074666
## [1] 3707.812576  254.242246 -647.550730    1.009152
## [1] 4479.936797  251.803625 -647.511476    1.000907
## [1] 4655.31317  251.03527 -647.50974    1.00004
## [1] 4682.941497  250.889157 -647.509698    1.000001
## [1] 4686.9790  250.8665 -647.5097    1.0000
## [1] 4687.5642  250.8632 -647.5097    1.0000
## [1] 4687.6489  250.8627 -647.5097    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  138.269542  138.269542 -646.178645    1.999129
## [1]  301.248760  206.510795 -646.164217    1.333046
## [1]  525.452023  256.784381 -646.154176    1.066693
## [1]  659.510674  271.980798 -646.152142    1.003967
## [1]  676.402863  272.937402 -646.152111    1.000016
## [1]  676.3396  272.9425 -646.1521    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  160.421757  160.421757 -655.314173    1.943159
## [1]  779.060471  232.072719 -654.921638    1.310801
## [1] 1999.533440  276.946347 -654.581235    1.063263
## [1] 3136.560974  285.339581 -654.463695    1.005672
## [1] 3504.567545  284.593007 -654.455251    1.000228
## [1] 3546.712592  284.393729 -654.455147    1.000003
## [1] 3549.9444  284.3740 -654.4551    1.0000
## [1] 3550.1837  284.3725 -654.4551    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  172.144976  172.144976 -659.594902    1.915751
## [1]   48.05478  214.53536 -659.53522    1.54650
## [1]    7.638025  224.354940 -659.522802    1.481097
## [1]    2.547094  225.535724 -659.521380    1.473618
## [1]    0.000000  226.123004 -659.520679    1.469928
## [1]    0.000000  298.413276 -659.520679    1.113839
## [1]    0.000000  328.912470 -659.520679    1.010556
## [1]    0.000000  332.348214 -659.520679    1.000109
## [1]    0.0000  332.3845 -659.5207    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS
```

```
## [1]  121.245311  121.245311 -634.870112    1.964631
## [1]   53.007890  151.780425 -634.832491    1.577564
## [1]    0.7816268  166.1412632 -634.8151855    1.4453321
## [1]    0.3417381  166.2449546 -634.8150604    1.4444628
## [1]    0.1217424  166.2967636 -634.8149979    1.4440289
## [1]    0.000000  166.322659 -634.814963    1.443813
## [1]    0.000000  217.448503 -634.814963    1.104348
## [1]    0.000000  237.994848 -634.814963    1.009008
## [1]    0.00000  240.11968 -634.81496    1.00008

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  156.525958  156.525958 -655.487361    1.996066
## [1]   54.77254  167.03301 -655.38561    1.87902
## [1]   26.621003  169.675039 -655.359627    1.851957
## [1]   12.238729  170.996388 -655.346647    1.838743
## [1]    4.973625  171.657027 -655.340163    1.832214
## [1]    1.322925  171.987323 -655.336922    1.828969
## [1]    0.4080029  172.0698929 -655.3361122    1.8281602
## [1]    0.1791326  172.0905352 -655.3359096    1.8279580
## [1]    0.06468002  172.10085628 -655.33580832    1.82785693
## [1]    0.000000  172.106017 -655.335751    1.827807
## [1]    0.000000  250.052190 -655.335751    1.258044
## [1]    0.00000  301.34165 -655.33575    1.04392
## [1]    0.000000  314.019753 -655.335751    1.001773
## [1]    0.000000  314.575590 -655.335751    1.000003

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  123.9966  123.9966 -638.9131    2.0260
## [1]   39.350761  132.416988 -638.781846    1.905585
## [1]   17.153904  134.531174 -638.750452    1.877712
## [1]    5.962451  135.588021 -638.735044    1.864101
## [1]    0.3460602  136.1162795 -638.7274148    1.8573751
## [1]    0.1702453  136.1327842 -638.7271770    1.8571658
## [1]    0.08233317  136.14103649 -638.72705818    1.85706121
## [1]    0.03837597  136.14516262 -638.72699876    1.85700890

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  161.385244  161.385244 -655.745165    1.942543
## [1]   59.779712  171.837511 -655.614407    1.831665
## [1]    8.472631  177.048463 -655.555954    1.781168
## [1]    2.062261  177.696878 -655.548976    1.775086
## [1]    0.4601473  177.8588671 -655.5472423    1.7735737
## [1]    0.05965213  177.89935686 -655.54680971    1.77319606
## [1]    0.000000  177.904418 -655.546745    1.773149
## [1]    0.000000  255.476394 -655.546745    1.234756
## [1]    0.000000  304.048511 -655.546745    1.037503
## [1]    0.000000  315.038958 -655.546745    1.001308
## [1]    0.000000  315.450590 -655.546745    1.000002
```

```
## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  119.22200  119.22200 -634.46345    1.98731
## [1]   92.949641  179.151192 -634.448380    1.327811
## [1]    5.774130  224.319701 -634.440474    1.065011
## [1]    2.795792  225.202327 -634.440378    1.060963
## [1]    1.390946  225.619323 -634.440334    1.059061
## [1]    0.0262863  226.0247601 -634.4402940    1.0572188

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  126.182791  126.182791 -639.615095    2.009373
## [1]   53.629979  142.655042 -639.543349    1.785387
## [1]   11.22789  150.83727 -639.51119    1.69238
## [1]    0.1336108  152.8516998 -639.5036156    1.6710194
## [1]    0.04622723  152.86735819 -639.50355717    1.67085563
## [1]    0.000000  152.875187 -639.503526    1.670774
## [1]    0.000000  214.250751 -639.503526    1.192154
## [1]    0.000000  248.784151 -639.503526    1.026673
## [1]    0.000000  255.247487 -639.503526    1.000675
## [1]    0.0000  255.4198 -639.5035    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  150.583257  150.583257 -652.972044    2.007294
## [1]   42.808919  170.345757 -652.874674    1.783818
## [1]   12.247512  175.255799 -652.852367    1.736156
## [1]    4.472788  176.475202 -652.846978    1.724731
## [1]    0.5715212  177.0837062 -652.8443144    1.7190877
## [1]    0.08304424  177.15969095 -652.84398281    1.71838574
## [1]    0.0000  177.1692 -652.8439    1.7183
## [1]    0.000000  251.231104 -652.843926    1.211752
## [1]    0.000000  295.133368 -652.843926    1.031499
## [1]    0.000000  304.145869 -652.843926    1.000933
## [1]    0.000000  304.429488 -652.843926    1.000001

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  151.852679  151.852679 -651.795001    1.959923
## [1]   28.700242  162.018844 -651.662815    1.847053
## [1]   11.799984  163.296663 -651.645721    1.833916
## [1]    3.266184  163.935669 -651.637169    1.827425
## [1]    1.122499  164.095426 -651.635028    1.825810
## [1]    0.04939006  164.17530407 -651.63395832    1.82500408
## [1]    0.00000  164.17780 -651.63391    1.82498
## [1]    0.000000  238.394182 -651.633909    1.256831
## [1]    0.000000  287.109590 -651.633909    1.043578
## [1]    0.000000  299.098759 -651.633909    1.001747
## [1]    0.000000  299.620310 -651.633909    1.000003
```

```
## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  126.628339  126.628339 -638.191343    1.965142
## [1]   48.951130  130.991756 -638.056499    1.906723
## [1]    9.035393  133.183195 -637.989508    1.878830
## [1]    3.992535  133.457499 -637.981147    1.875404
## [1]    1.468123  133.594668 -637.976970    1.873696
## [1]    0.2051854  133.6632569 -637.9748820    1.8728429
## [1]    0.04727291  133.67183077 -637.97462108    1.87273633
## [1]    0.00000  133.67397 -637.97454    1.87271
## [1]    0.000000  195.967997 -637.974543    1.277416
## [1]    0.000000  238.526290 -637.974543    1.049497
## [1]    0.000000  249.775818 -637.974543    1.002229
## [1]    0.000000  250.331398 -637.974543    1.000005

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  122.896207  122.896207 -636.853237    1.989481
## [1]  196.080581  183.914174 -636.853042    1.328664
## [1]  267.944026  229.204752 -636.852799    1.065179
## [1]  311.711906  242.993150 -636.852613    1.003761
## [1]  332.420094  243.735468 -636.852532    1.000015
## [1]  343.9466  243.6404 -636.8525    1.0000
## [1]  350.6229  243.5834 -636.8525    1.0000
## [1]  354.4809  243.5505 -636.8525    1.0000
## [1]  356.7068  243.5315 -636.8525    1.0000
## [1]  357.9898  243.5206 -636.8525    1.0000
## [1]  358.7290  243.5143 -636.8525    1.0000
## [1]  359.1548  243.5107 -636.8525    1.0000
## [1]  359.4000  243.5086 -636.8525    1.0000
## [1]  359.5412  243.5074 -636.8525    1.0000
## [1]  359.6224  243.5067 -636.8525    1.0000
## [1]  359.6692  243.5063 -636.8525    1.0000
## [1]  359.6962  243.5060 -636.8525    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  190.736956  190.736956 -670.090364    1.985061
## [1]  249.569216  285.695232 -670.089548    1.326707
## [1]  250.938449  356.568817 -670.088803    1.064564
## [1]  214.307334  378.647480 -670.088502    1.003697
## [1]  194.197157  380.226499 -670.088460    1.000014
## [1]  188.1523  380.2855 -670.0885    1.0000
## [1]  186.4321  380.3008 -670.0885    1.0000
## [1]  185.9417  380.3051 -670.0885    1.0000
## [1]  185.8017  380.3064 -670.0885    1.0000
## [1]  185.7618  380.3067 -670.0885    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS
```

```
## [1]  155.733467  155.733467 -654.837211    1.989134
## [1]   29.833800  195.730240 -654.780659    1.593309
## [1]    5.016507  200.493484 -654.772637    1.557077
## [1]    1.819076  201.080262 -654.771641    1.552737
## [1]    0.2149232  201.3730828 -654.7711446    1.5505807
## [1]    0.000000  201.409649 -654.771078    1.550313
## [1]    0.000000  272.903809 -654.771078    1.144169
## [1]    0.000000  307.290484 -654.771078    1.016133
## [1]    0.000000  312.169254 -654.771078    1.000252
## [1]    0.0000  312.2479 -654.7711    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  157.322856  157.322856 -653.592445    1.937051
## [1]  748.403078  226.606885 -653.002933    1.309506
## [1] 1898.423596  269.829102 -652.568399    1.064122
## [1] 2862.62549  279.22067 -652.46908    1.00567
## [1] 3003.647463  279.953744 -652.467916    1.000065
## [1] 2991.2775  280.0547 -652.4679    1.0000
## [1] 2992.8578  280.0443 -652.4679    1.0000
## [1] 2992.6612  280.0456 -652.4679    1.0000

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  136.049954  136.049954 -644.600796    1.989998
## [1]   51.238871  145.143376 -644.508595    1.873631
## [1]    3.835604  149.716403 -644.461639    1.820612
## [1]    0.7281062  150.0022550 -644.4586534    1.8174121
## [1]    0.3385804  150.0379846 -644.4582799    1.8170131
## [1]    0.1437496  150.0558492 -644.4580932    1.8168137
## [1]    0.04631729  150.06478153 -644.45799979    1.81671396

## Computing Variance Component Estimates...
## Sigma^2_A     log-lik     RSS

## [1]  142.960225  142.960225 -648.746662    1.999986
## [1]   14.159152  152.648184 -648.529362    1.882681
## [1]    6.367699  153.253439 -648.517208    1.875812
## [1]    2.481184  153.555974 -648.511188    1.872397
## [1]    0.5402581  153.7072163 -648.5081912    1.8706946
## [1]    0.05531955  153.74502360 -648.50744357    1.87026959
## [1]    0.000000  153.747387 -648.507358    1.870245
## [1]    0.000000  225.287682 -648.507358    1.276347
## [1]    0.000000  274.065619 -648.507358    1.049184
## [1]    0.000000  286.913352 -648.507358    1.002202
## [1]    0.000000  287.543866 -648.507358    1.000005
```

```
library(SeqVarTools)
library(SeqArray)
library(SNPRelate)
```

```r
library(Biobase)


# Open the SeqArray GDS file
seq_file <- "Qatari156_filtered_pruned_SeqArray.gds"
geno_file <- seqOpen(seq_file)


seqData <- SeqVarData(geno_file)
iterator <- SeqVarBlockIterator(seqData)
```

```
## # of selected variants: 10,000
```

```r
allResults <- do.call(rbind, lapply(names(nullMods), function(m) {
        at <- assocTestSingle(
                null.model = nullMods[[m]],
                gdsobj = iterator,
                test = "Score"
        )
        if (nrow(at) > 0) {
                cbind(metabolite = m, at)
        } else {
                NULL # skip if empty
        }
}))
```

```
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
## Using 6 CPU cores
```

```r
seqClose(iterator)
```

```r
colnames(allResults)
```

```
##  [1] "metabolite"   "variant.id"   "chr"          "pos"          "allele.index"
```

```
##  [6] "n.obs"        "freq"         "MAC"          "Score"        "Score.SE"
## [11] "Score.Stat"   "Score.pval"   "Est"          "Est.SE"       "PVE"
```

```r
# 1. Filter significant results
significantResults <- allResults[allResults$Score.pval < 0.0001, ]

# 2. Create a clean data frame with only relevant columns
output <- data.frame(
        metabolite     = significantResults$metabolite,
        beta           = significantResults$Est,
        SE             = significantResults$Est.SE,
        effect.allele  = significantResults$allele.index,
        n              = significantResults$n.obs,
        pval           = significantResults$Score.pval
)

# 3. Write to Excel
write_xlsx(output, path = "mQTL_significant_p1e4.xlsx")
```

#Task 3: Inflation factor calculation

```r
library(SummarizedExperiment)

# 1. Get unique metabolites from results
unique_metabolites <- unique(output$metabolite)
n_metab <- length(unique_metabolites)

# 2. Initialize results data frame
lambda_results <- data.frame(
        metabolite = unique_metabolites,
        lambda = numeric(n_metab),
        stringsAsFactors = FALSE
)

# 3. Calculate lambda for each metabolite
for (i in 1:n_metab) {
        current_metab <- unique_metabolites[i]

        # Get p-values for current metabolite
        pvals <- output$pval[output$metabolite == current_metab]

        # Remove NA p-values
        pvals <- pvals[!is.na(pvals)]

        # Handle extreme p-values
        pvals[pvals < 1e-20] <- 1e-20
        pvals[pvals > (1 - 1e-10)] <- 1 - 1e-10

        # Calculate chi-square statistics
        chisq <- qchisq(1 - pvals, df = 1)

        # Calculate lambda
        lambda_results$lambda[i] <- median(chisq) / qchisq(0.5, df = 1)
```

```
}

# 4. Calculate counts per metabolite for weighting
counts <- sapply(unique_metabolites, function(m) sum(output$metabolite == m))

# 5. Calculate weighted average lambda
average_lambda <- sum(lambda_results$lambda * counts) / sum(counts)

# 6. Report results
cat("=== mQTL Inflation Factors ===\n")
```

```
## === mQTL Inflation Factors ===
```

```
print(lambda_results[order(lambda_results$lambda, decreasing = TRUE), ])
```

```
##    metabolite   lambda
## 1 Metabolite1 37.09991
```

```
cat("\nAverage inflation factor (weighted):", round(average_lambda, 4), "\n")
```

```
##
## Average inflation factor (weighted): 37.0999
```

#Task 4: Manhattan Plot

```
library(qqman)
```

```
## Warning: package 'qqman' was built under R version 4.4.3
```

```
##
```

```
## For example usage please run: vignette('qqman')
```

```
##
```

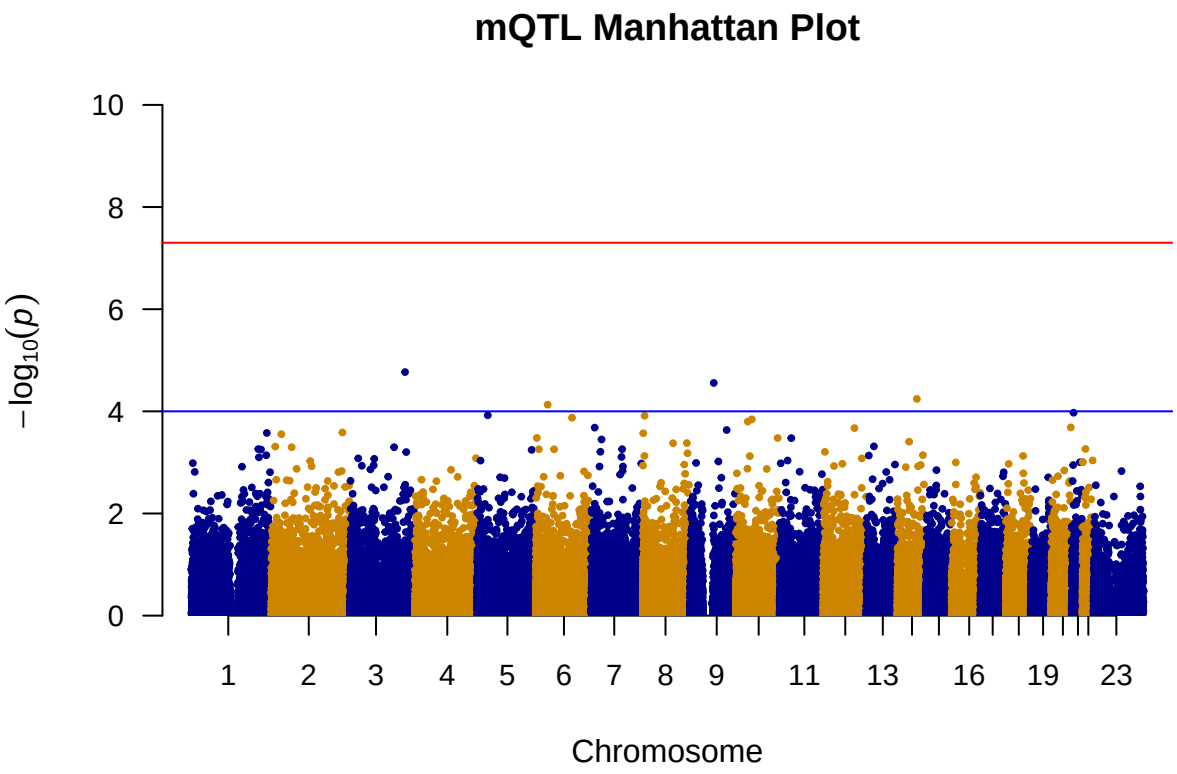
```
## Citation appreciated but not required:
```

```
## Turner, (2018). qqman: an R package for visualizing GWAS results using Q-Q and manhattan plots. Jour
```

```
##
```

```
# 1. Prepare your data
manhattan_data <- data.frame(
        SNP = allResults$variant.id, # SNP IDs
        CHR = as.numeric(allResults$chr), # Chromosome as numeric
        BP = allResults$pos, # Base pair position
        P = allResults$Score.pval # P-values
)
```

```r
# 2. Create Manhattan plot
manhattan(manhattan_data,
        main = "mQTL Manhattan Plot",
        ylim = c(0, 10), # Adjust based on your p-values
        suggestiveline = -log10(1e-4), # Suggestive threshold
        genomewideline = -log10(5e-8), # Genome-wide significant
        col = c("blue4", "orange3"), # Alternating colors
        cex = 0.6, # Point size
        cex.axis = 0.9
) # Axis text size
```

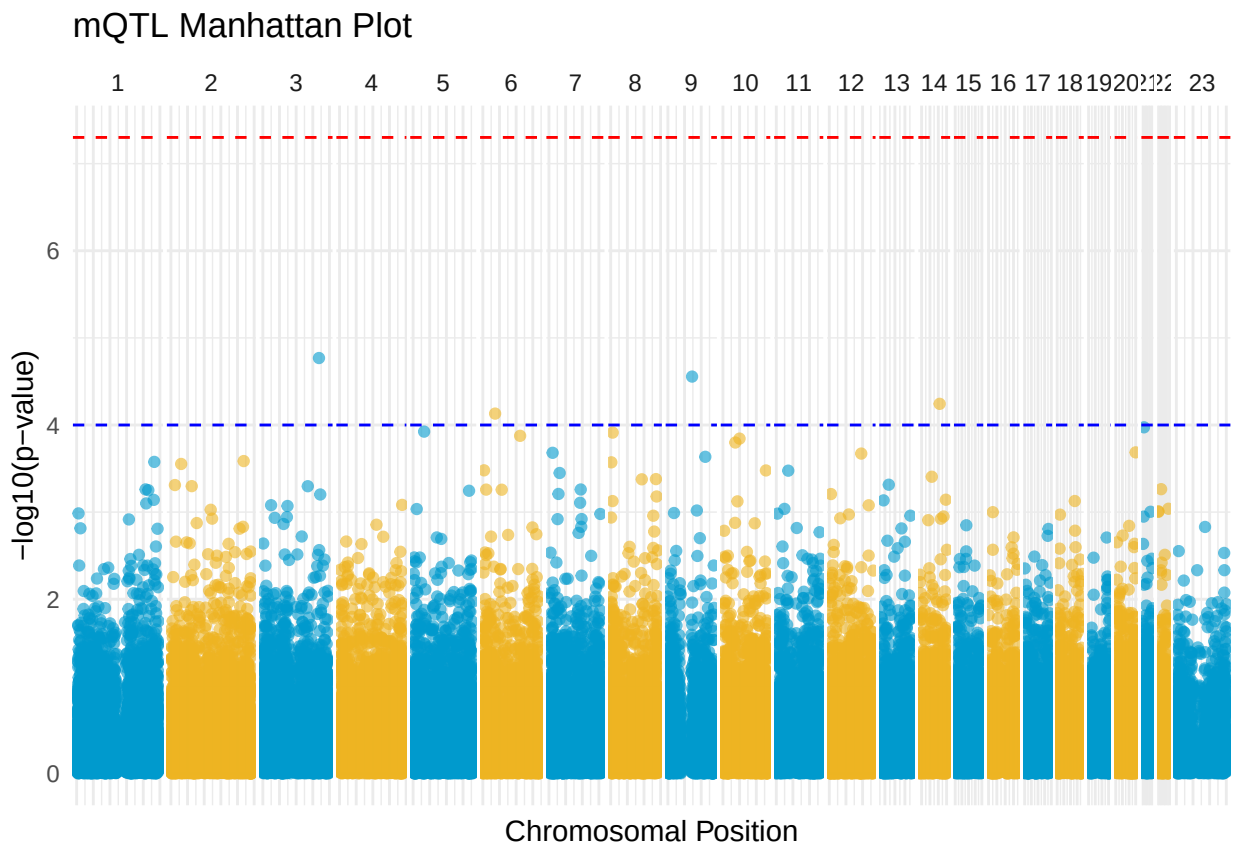



```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:GenomicRanges':
##
##     intersect, setdiff, union

## The following object is masked from 'package:GenomeInfoDb':
##
##     intersect
```

```
## The following objects are masked from 'package:IRanges':
##
##     collapse, desc, intersect, setdiff, slice, union

## The following objects are masked from 'package:S4Vectors':
##
##     first, intersect, rename, setdiff, setequal, union

## The following object is masked from 'package:matrixStats':
##
##     count

## The following object is masked from 'package:Biobase':
##
##     combine

## The following objects are masked from 'package:BiocGenerics':
##
##     combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
# 1. Prepare data with chromosome colors
plot_data <- manhattan_data %>%
        group_by(CHR) %>%
        mutate(color = if_else(CHR %% 2 == 1, "odd", "even"))

# 2. Create plot
ggplot(plot_data, aes(x = BP, y = -log10(P), color = color)) +
        geom_point(alpha = 0.6, size = 1.5) +
        facet_grid(. ~ CHR, scales = "free_x", space = "free_x") +
        geom_hline(yintercept = -log10(5e-8), color = "red", linetype = "dashed") +
        geom_hline(yintercept = -log10(1e-4), color = "blue", linetype = "dashed") +
        scale_color_manual(values = c("odd" = "deepskyblue3", "even" = "goldenrod2")) +
        labs(
                title = "mQTL Manhattan Plot",
                x = "Chromosomal Position",
                y = "-log10(p-value)"
        ) +
        theme_minimal() +
        theme(
                legend.position = "none",
                panel.spacing.x = unit(0.1, "lines"),
                axis.text.x = element_blank()
        )
```



#Task 5: Metabolic Networks ##Correct metabolites for covariates and kinship using polygenic() residuals

```
library(Matrix)
library(coxme) # Load coxme for lmekin
```

```
## Warning: package 'coxme' was built under R version 4.4.3
```

```
## Loading required package: survival
```

```
## Loading required package: bdsmatrix
```

```
##
## Attaching package: 'bdsmatrix'
```

```
## The following object is masked from 'package:base':
##
##     backsolve
```

```
# Create a kinship matrix suitable for lmekin
kin_mat <- as.matrix(kinship_matrix)
kin_mat <- kin_mat[pca$sample.id, pca$sample.id] # Align rows/cols
# Add a small value to the diagonal to ensure positive definiteness
kin_mat <- kin_mat + diag(1e-6, nrow(kin_mat))
# Force kin_mat to be positive semi-definite using nearPD
```

```r
kin_mat <- as.matrix(Matrix::nearPD(kin_mat)$mat)

corrected_metabs <- data.frame(sample.id = pca$sample.id)

for (metab in colnames(metabolites)) {
        df <- data.frame(
                value = as.numeric(metabolites[[metab]]),
                PC1 = as.numeric(pca$vectors[, 1]),
                PC2 = as.numeric(pca$vectors[, 2]),
                PC3 = as.numeric(pca$vectors[, 3]),
                id = pca$sample.id
        )
        # Remove rows with NA values
        df <- na.omit(df)
        if (nrow(df) == 0) {
                warning(paste("No valid data for metabolite:", metab))
                next
        }
        # Mixed model using kinship as random effect
        model <- lmekin(value ~ PC1 + PC2 + PC3 + (1 | id),
                data = df,
                varlist = kin_mat
        )
        # Residuals = corrected metabolite
        corrected_metabs[[metab]] <- resid(model)
}
```

```
## Warning in data.frame(value = as.numeric(metabolites[[metab]]), PC1 =
## as.numeric(pca$vectors[, : NAs introduced by coercion
```

```
## Warning: No valid data for metabolite: Sample
```

```r
head(corrected_metabs)
```

```
##   sample.id Metabolite1 Metabolite2 Metabolite3 Metabolite4 Metabolite5
## 1   QBC-092  -18.362924  13.5458621   21.077946 -0.52926744  -17.145662
## 2   QBC-256   12.430672  -0.1173253   17.701093 -0.04148716  -21.173199
## 3   QBC-107   28.357161 -25.7134955    4.786697  0.36023119  -13.641657
## 4   QBC-171    3.878960  15.5913465    2.661467 -0.35676532   -9.306989
## 5  QPRC-110   16.375083 -33.7206928   14.868404  0.11449925   -2.862163
## 6   QBC-240   -4.128167 -14.4969184   11.494094  0.12194974    8.076153
##   Metabolite6 Metabolite7 Metabolite8 Metabolite9 Metabolite10 Metabolite11
## 1   17.868663   2.0139745  0.23297477   -5.368613    0.4426044     8.213959
## 2    3.099056   1.1224822 -0.09037276   12.475230    0.3031681    16.587895
## 3    8.967723   1.7937117  0.02247785   -2.059328   -0.1432662     9.886799
## 4   -2.615191  -0.1243338 -0.04195218  -10.621306    1.6950372    -4.632207
## 5  -25.290992   0.1259195 -0.23214386   11.656198   -1.0483343     8.655712
## 6   -9.245038   3.1345686  0.19358085   -7.482304   -0.9239052    27.049914
##   Metabolite12 Metabolite13 Metabolite14 Metabolite15 Metabolite16 Metabolite17
## 1     2.347975   0.00618478   -11.596077 -0.007590152    3.3730807   -21.313770
## 2   -15.828226   0.03479607    13.787603 -0.111856055   -0.3235906    10.782228
## 3    10.507655   0.05217191   -43.147655  0.094690665    0.1621564    12.208349
```

```
## 4   -15.547544   0.12844436     2.049231  0.040543310   -3.2323507    -7.379139
## 5    15.260132  -0.09442212    12.727181  0.150471706   -0.8521100    -1.633287
## 6    27.425768   0.05178014    11.049034  0.113628990    0.5384435    16.531011
##   Metabolite18 Metabolite19 Metabolite20
## 1   -0.6272656   13.8645290    -6.400697
## 2  -17.0880334    1.3938062     4.923532
## 3   26.9764323    4.3039889    21.340225
## 4  -17.4040786   16.3136009     7.325493
## 5   10.7663644    1.8257239   -19.196941
## 6   -4.4329743    0.3428462   -12.998350
```

##Use GeneNet to Compute Partial Correlation Network

```
# Load required package
library(GeneNet)
```

```
## Warning: package 'GeneNet' was built under R version 4.4.3
```

```
## Loading required package: corpcor
```

```
## Loading required package: longitudinal
```

```
## Loading required package: fdrtool
```

```
# 1. Prepare metabolite data (samples=rows, metabolites=columns)
metab_data <- corrected_metabs[, -1] # Exclude sample.id column
rownames(metab_data) <- corrected_metabs$sample.id

# 2. Clean data - remove invariant metabolites and samples with NAs
metab_clean <- metab_data[, apply(metab_data, 2, var, na.rm = TRUE) > 0]
metab_clean <- na.omit(metab_clean)
metabolite_names <- colnames(metab_clean)

# 3. Compute partial correlations between METABOLITES (columns)
# Note: We DON'T transpose here - metabolites should be columns
pcor_matrix <- ggm.estimate.pcor(as.matrix(metab_clean), method = "dynamic")
```

```
## Estimating optimal shrinkage intensity lambda (correlation matrix): 0.9187
```

```
rownames(pcor_matrix) <- colnames(pcor_matrix) <- metabolite_names

# 4. Test for significant metabolite-metabolite edges
network_test <- network.test.edges(pcor_matrix, fdr = TRUE, direct = TRUE)
```

```
## Estimate (local) false discovery rates (partial correlations):
```

```
## Warning in fdrtool(w[, 1], statistic = "correlation", plot = plot, ...): There
## may be too few input test statistics for reliable FDR calculations!
```

```
## Step 1... determine cutoff point
## Step 2... estimate parameters of null distribution and eta0
## Step 3... compute p-values and estimate empirical PDF/CDF
## Step 4... compute q-values and local fdr
## Step 5... prepare for plotting

##
## Estimate (local) false discovery rates (log ratio of spvars):

## Warning in fdrtool(log.spvar, statistic = "normal", plot = plot, ...): There
## may be too few input test statistics for reliable FDR calculations!

## Step 1... determine cutoff point
## Step 2... estimate parameters of null distribution and eta0
## Step 3... compute p-values and estimate empirical PDF/CDF
## Step 4... compute q-values and local fdr
## Step 5... prepare for plotting
```

```
# 5. Create edges data frame with metabolite names
edges <- data.frame(
        source = metabolite_names[network_test$node1],
        target = metabolite_names[network_test$node2],
        pcor = network_test$pcor,
        pval = network_test$pval,
        stringsAsFactors = FALSE
)

# 6. Remove any NA values and filter significant edges
edges <- edges[complete.cases(edges), ]
sig_edges <- edges[edges$pval < 0.05, ]

# 7. Create nodes data frame
nodes <- data.frame(
        id = metabolite_names,
        nodeType = "metabolite",
        stringsAsFactors = FALSE
)

# 8. Save results
write.csv(sig_edges, "metabolite_network_edges.csv", row.names = FALSE)
write.csv(nodes, "metabolite_network_nodes.csv", row.names = FALSE)

# 9. Verification output
cat("=== Network Analysis Results ===\n")
```

```
## === Network Analysis Results ===
```

```
cat("Samples analyzed:", nrow(metab_clean), "\n")
```

```
## Samples analyzed: 156
```

```r
cat("Metabolites analyzed:", length(metabolite_names), "\n")
```

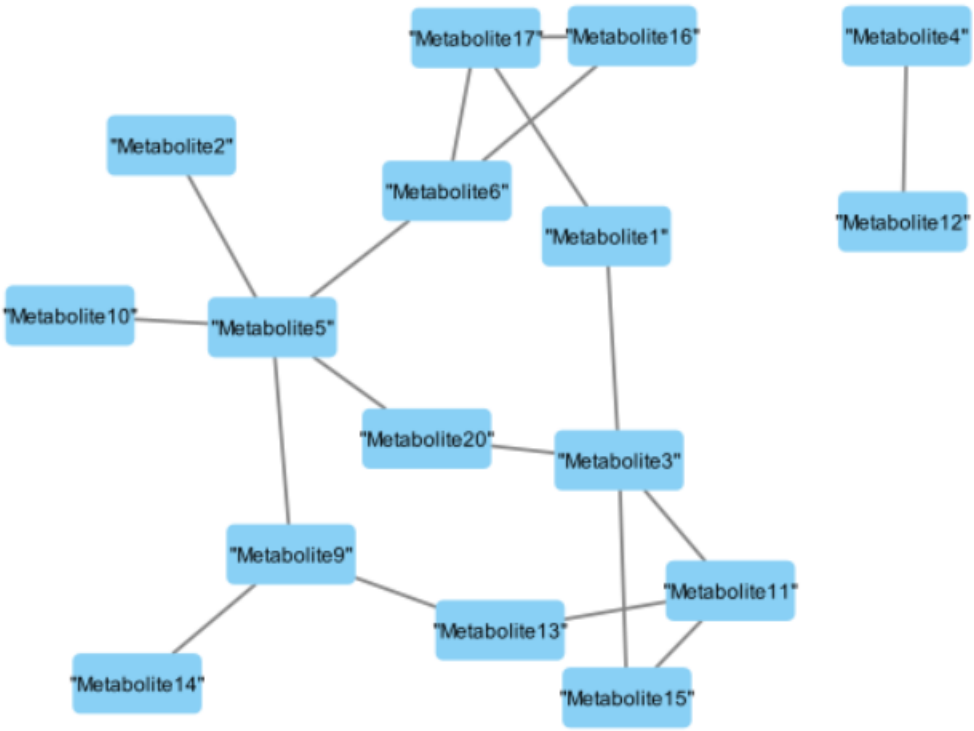
```
## Metabolites analyzed: 20
```

```r
cat("Significant edges (q < 0.05):", nrow(sig_edges), "\n")
```

```
## Significant edges (q < 0.05): 18
```

```r
if (nrow(sig_edges) > 0) {
        cat("\nSample of significant edges:\n")
        print(head(sig_edges, 5))
} else {
        cat("\nNo significant edges found at q < 0.05\n")
}
```

```
##
## Sample of significant edges:
##         source       target        pcor        pval
## 1  Metabolite6 Metabolite16 -0.01850526 0.001605452
## 2  Metabolite3 Metabolite20 -0.01728112 0.003217245
## 3  Metabolite3 Metabolite15 -0.01725230 0.003268731
## 4 Metabolite11 Metabolite13  0.01611235 0.006016226
## 5  Metabolite4 Metabolite12 -0.01540358 0.008638265
```



##Visualize in Cytoscape
##Analyze Network in Cytoscape

```r
cytoscapeNetworkAnalysis <- read.csv("cytoscapeNetworkAnalysis.csv", header = TRUE)
print(cytoscapeNetworkAnalysis)
```

```
##              name selected    shared.name
## 1   "Metabolite6"    false  "Metabolite6"
## 2  "Metabolite16"    false "Metabolite16"
## 3   "Metabolite3"    false  "Metabolite3"
## 4  "Metabolite20"    false "Metabolite20"
## 5  "Metabolite15"    false "Metabolite15"
## 6  "Metabolite11"    false "Metabolite11"
## 7  "Metabolite13"    false "Metabolite13"
## 8   "Metabolite4"    false  "Metabolite4"
## 9  "Metabolite12"    false "Metabolite12"
## 10  "Metabolite9"    false  "Metabolite9"
## 11 "Metabolite17"    false "Metabolite17"
## 12  "Metabolite1"    false  "Metabolite1"
## 13  "Metabolite2"    false  "Metabolite2"
## 14  "Metabolite5"    false  "Metabolite5"
## 15 "Metabolite10"    false "Metabolite10"
## 16 "Metabolite14"    false "Metabolite14"
```

#Task 6: Annotate Significant SNPs

```r
library(dplyr)

top20_snps <- allResults %>%
        arrange(Score.pval) %>%
        distinct(variant.id, .keep_all = TRUE) %>%
        slice(1:20)

annovar_input <- data.frame(
        chr = top20_snps$chr,
        start = top20_snps$pos,
        end = top20_snps$pos,
        ref = substr(top20_snps$allele.index, 1, 1), # First allele as ref
        alt = substr(top20_snps$allele.index, 3, 3), # Second allele as alt
        snp_id = top20_snps$variant.id,
        stringsAsFactors = FALSE
)

# 3. Write ANNOVAR input (tab-delimited, no header)
write.table(annovar_input, "top_snps_annovar_input.txt",
        sep = "\t", quote = FALSE, row.names = FALSE, col.names = FALSE
)
```

#Task 7: Regional plots using SNIPA

```r
top5_snps <- allResults %>%
        arrange(Score.pval) %>%
        distinct(variant.id, .keep_all = TRUE) %>%
        slice(1:5)
```