REBET (subREgion-based BurdEn Test)

July 9, 2018

Introduction

There is an increasing focus to investigate the association between rare variants and common complex diseases in the hope of explaining the missing heritability that remains unexplained from genome-wide association studies (GWAS) of common variants. Recent studies have reported that rare variants contribute to the genetic susceptibility for a number of complex traits or diseases, including human adult height, lipid levels, autism, ischemic stroke, prostate cancer and breast cancer. Detecting rare variant associations is statistically challenging, stemming from two characteristics of rare variants: low frequency and heterogeneous risk effects. The power of detecting a single rare variant would be very low unless the sample size and/or effect size is extremely large. Thus it has been proposed to aggregating rare variants in a gene or genomic region to boost the power, but this can also negatively affect power due to the problem of heterogeneous effects. To take these heterogeneous effects into account, burden tests have been considered which reweight the effects of rare variants based on their frequencies. While these tests have robust power for detecting a susceptibility region containing clusters of causal variants, they do not readily identify which variants, or class of them, in a gene contribute most to the association.

The subREgion-based BurdEn Test (REBET) simultaneously detects the rare variant association of a gene and identifies the most susceptible sub-regions that drive the gene-level significant association. In order to apply REBET, biologically meaningful sub-regions within a gene need to be specified. The rare variants within each sub-region may share common biologic characteristics, such as functional domain or functional impact. REBET then searches all possible combinations of sub-regions, identifies the one with the strongest association signal through linear burden test, and assesses its statistical significance while adjusting for multiple tests involved in the sub-region search. For detecting overall association for a gene, REBET has robust power when risk effects are relatively homogeneous within sub-regions, but potentially heterogeneous across sub-regions.

```
> library(REBET)
```

Example

Get the paths to the data files.

```
> genofile  <- system.file("sampleData", "geno_impute.txt.gz", package="REBET")
> subfile   <- system.file("sampleData", "subjects.txt.gz", package="REBET")
> phenofile <- system.file("sampleData", "pheno.txt.gz", package="REBET")
```

Read in the phenotype data containing the response, covariates, and subject ids.

```
> data <- read.table(phenofile, header=1, sep="\t")
> data[1:5, ]
```

```
  Subject  Response Age Gender
1       1 11.911188  66 FEMALE
2       2  9.540571  59 FEMALE
3       3  9.346940  50 FEMALE
4       4 12.164063  71 FEMALE
5       5 10.495913  51   MALE
```

Our model will be adjusted for age and gender. Let us add a dummy variable for gender.

```
> data[, "MALE"] <- as.numeric(data[, "Gender"] %in% "MALE")
```

For out analysis, we are only interested in the four sub-regions of chromosome 7 defined below.

```
> subRegions <- rbind(c(87654800, 87661050),
+                     c(87661051, 87668870),
+                     c(87668871, 87671945),
+                     c(87671946, 87673200))
> subRegions
```

```
         [,1]     [,2]
[1,] 87654800 87661050
[2,] 87661051 87668870
[3,] 87668871 87671945
[4,] 87671946 87673200
```

We will use the minimum and maximum positions when the genotype data is read.

```
> min.loc <- min(subRegions)
> max.loc <- max(subRegions)
```

We also need the subject order for the genotype data. These ids are in the file subfile. Let us read in the genotype subject ids.

```
> geno.subs  <- scan(subfile, what="character")
```

The set and order of subjects may not be the same in the phenotype and genotype data. We need the common set of subject and the correct order.

```
> tmp   <- data[, "Subject"] %in% geno.subs
> data  <- data[tmp, ]
> order <- match(data[, "Subject"], geno.subs)
```

The genotype data is in a file created from the IMPUTE2 software. Each row of this file has the form: Snpid RSid Position A1 A2 $P1_1P1_2P1_3P2_1P2_2P2_3$... where A1, A2 are the alleles and $Pj_1 = P(a1/a1), Pj_2 = P(a1/a2), Pj_3 = P(a2/a2)$ for the jth subject. We do not know how many variants are in the file and do not know how many variants are in the sub-regions defined above, but we know it should not be more than 100. So we will read in the file row by row instead of attempting to read in the entire file at once. We will initialize some objects to store the necessary information we need from the genotype file. The matrix G will store the expected dosages for the variants we want. The vectors snps and locs will store the variant names and positions.

```
> upper.n <- 100
> G       <- matrix(data=NA, nrow=nrow(data), ncol=upper.n)
> snps    <- rep("", upper.n)
> locs    <- rep(NA, upper.n)
```

Before the genotype file is read we need some vectors that will pick off the probability of each genotype for each subject.

```
> id1 <- seq(from=1, to=3*length(geno.subs), by=3)
> id2 <- id1 + 1
> id3 <- id1 + 2
```

Now we are ready to open the genotype file and read it row by row. In the code below, we are only going to store the variants that are between the min.loc and max.loc defined above. For such variants, we compute the expected dosage for each subject as $Pj_2 + 2 * Pj_3$, which make allele a2 the effect allele. Note that we must check for missing genotypes - if all three probabilities are 0, then the expected dosage is NA (not 0!).

```
> index <- 0
> fid   <- gzfile(genofile, "r")
> while(1) {
+   vec <- scan(fid, what="character", sep=" ", quiet=TRUE, nlines=1)
+   if (!length(vec)) break
+   snp <- vec[2]
+   loc <- as.numeric(vec[3])
+   if ((loc >= min.loc) & (loc <= max.loc)) {
+     geno.probs  <- as.numeric(vec[-(1:5)])
+     probs1      <- geno.probs[id1]
+     probs2      <- geno.probs[id2]
+     probs3      <- geno.probs[id3]
+     dosage      <- probs2 + 2*probs3
+
+     # Check for missing genotypes
+     tmp <- (probs1 == 0) & (probs2 == 0) & (probs3 == 0)
+     tmp[is.na(tmp)] <- TRUE
+     if (any(tmp)) dosage[tmp] <- NA
+
+     index       <- index + 1
+     G[, index]  <- dosage[order]
+     snps[index] <- snp
```

```
+     locs[index] <- loc
+   }
+ }
> close(fid)
```

Subset the objects G, snps, and locs by the number of variants we stored, which is the number index.

```
> G    <- G[, 1:index, drop=FALSE]
> snps <- snps[1:index]
> locs <- locs[1:index]
> colnames(G) <- snps
```

The rebet function requires a vector of sub-region names for the variants in matrix G. The sub-region names will be SR1-SR4 and will be stored in the vector E.

```
> E <- rep("", index)
> for (i in 1:nrow(subRegions)) {
+   tmp <- (locs >= subRegions[i, 1]) & (locs <= subRegions[i, 2])
+   tmp[is.na(tmp)] <- FALSE
+   if (any(tmp)) E[tmp] <- paste("SR", i, sep="")
+ }
```

Define the response vector, matrix of covariates, and call the rebet function.

```
> Y   <- as.numeric(data[, "Response"])
> X   <- as.matrix(data[, c("Age", "MALE")])
> ret <- rebet(Y, G, E, X=X)
```

The result summary shows that sub-region SR3 is highly significant.

```
> print(h.summary(ret))

$Meta
   SNP       Pvalue     OR CI.low CI.high
1 Gene 0.0001554158 43.915  6.186 311.755

$Subset.1sided
   SNP       Pvalue       OR   CI.low  CI.high      Pheno
1 Gene 5.434698e-08 532447.2 4590.188 61762179 Region_SR3

$Subset.2sided
   SNP       Pvalue     Pvalue.1  Pvalue.2     OR.1 CI.low.1 CI.high.1  OR.2
1 Gene 4.336478e-07 4.224093e-08 0.5527489 532447.2 4771.827  59411208 0.401
  CI.low.2 CI.high.2    Pheno.1    Pheno.2
1     0.02     8.199 Region_SR3 Region_SR2
```