

ASSET(Association analysis for SubSETs) Package

May 3, 2016

Introduction

ASSET is a suite of statistical tools specifically designed to be powerful for pooling association signals across multiple studies when true effects may exist only in a subset of the studies and could be in opposite directions across studies. The method explores all possible subsets (or a restricted set if user specifies so) of studies and evaluates fixed-effect meta-analysis-type test-statistics for each subset. The final test-statistic is obtained by maximizing the subset-specific test-statistics over all possible subsets and then evaluating its significance after efficient adjustment for multiple-testing, taking into account the correlation between test-statistics across different subsets due to overlapping subjects. The method not only returns a p-value for significance for the overall evidence of association of a SNP across studies, but also outputs the "best subset" containing the studies that contributed to the overall association signal. For detection of association signals with effects in opposite directions, ASSET allows subset search separately for positively- and negatively- associated studies and then combines association signals from two directions using a chi-square test-statistic. The method can take into account correlation due to overlapping subjects across studies (e.g. shared controls). Although the method is originally developed for conducting genetic association scans, it can also be applied for analysis of non-genetic risk factors as well.

The ASSET package consists of two main functions: (1) `h.traits` and (2) `h.types`. The function `h.traits` is suitable for conducting meta-analysis of possibly different traits when summary level data are available from individual studies. The function allows for correlation among different studies/traits, which, for example, may arise due to shared subjects across studies. This function can also be used to conduct "meta-analysis" across multiple correlated traits on the same individuals by appropriately specifying the correlation matrix for the multivariate trait. Input arguments to this function are vectors/matrices of the estimated log-odds ratios, standard errors and number of cases and controls for each SNP and study. The function `h.types` is suitable for analysis of case-control studies when cases consist of distinct disease subtypes. This function assumes individual level data are available. The main input argument for `h.types` is a data frame containing the SNP variables, response variable and covariates for all subjects.

```
> library(ASSET)
```

Examples of h.traits

Get the path to the data.

```
> datafile <- system.file("sampleData", "vdata.rda", package="ASSET")
```

Load the data frames. There are 4 data frames, data1 - data4 for the 4 independent studies. Each study has the SNPs SNP1-SNP3 genotyped, and information on each subject's age and case-control status. Each SNP is coded as the number of copies of the minor allele or NA for missing genotypes.

```
> load(datafile)
> data1[1:5, ]
```

	CC	AGE	TYPE	SNP1	SNP2	SNP3
456	1	70	TYPE_3	1	0	2
457	1	55	TYPE_3	1	0	1
458	1	48	TYPE_3	1	0	0
459	1	72	TYPE_1	0	0	0
460	1	74	TYPE_1	1	0	1

```
> SNPs <- paste("SNP", 1:3, sep="")
> nSNP <- length(SNPs)
> studies <- paste("STUDY", 1:4, sep="")
> nStudy <- length(studies)
```

Let us determine the number of non-missing cases and controls for each SNP and study.

```
> case <- matrix(data=NA, nrow=nSNP, ncol=nStudy)
> control <- matrix(data=NA, nrow=nSNP, ncol=nStudy)
> for (i in 1:nStudy) {
+   data <- eval(parse(text=paste("data", i, sep="")))
+   caseVec <- data[, "CC"] == 1
+   controlVec <- !caseVec
+   for (j in 1:nSNP) {
+     temp <- !is.na(data[, SNPs[j]])
+     case[j, i] <- sum(caseVec & temp, na.rm=TRUE)
+     control[j, i] <- sum(controlVec & temp, na.rm=TRUE)
+   }
+ }
> case
```

	[,1]	[,2]	[,3]	[,4]
[1,]	1905	1359	1719	690
[2,]	1909	1371	1716	683
[3,]	1871	1379	1421	428

```
> control
```

	[,1]	[,2]	[,3]	[,4]
[1,]	1953	1793	1260	667
[2,]	1960	1807	1256	659
[3,]	1934	1805	1023	400

Run a logistic regression for each SNP and study

```
> beta <- matrix(data=NA, nrow=nSNP, ncol=nStudy)
> sigma <- matrix(data=NA, nrow=nSNP, ncol=nStudy)
> for (i in 1:nStudy) {
+   data <- eval(parse(text=paste("data", i, sep="")))
+   for (j in 1:nSNP) {
+     data[, "SNP"] <- data[, SNPs[j]]
+     fit <- glm(CC ~ AGE + SNP, data=data, family=binomial())
+     coef <- summary(fit)$coefficients
+     beta[j, i] <- coef["SNP", 1]
+     sigma[j, i] <- coef["SNP", 2]
+   }
+ }
> beta
```

	[,1]	[,2]	[,3]	[,4]
[1,]	0.23723176	-0.0578681	-0.019393947	0.1608221
[2,]	-0.18872480	0.2359282	0.081331889	0.2804106
[3,]	-0.01422937	0.1285913	0.005024673	-0.1900723

```
> sigma
```

	[,1]	[,2]	[,3]	[,4]
[1,]	0.04614171	0.05538078	0.05768809	0.07800331
[2,]	0.09483914	0.08662322	0.09076995	0.11773901
[3,]	0.04902533	0.05698004	0.06469438	0.10539042

```
>
```

Call the `h.traits` function. Since the studies are independent, we do not need to specify the `cor` option.

```
> res <- h.traits(SNPs, studies, beta, sigma, ncase=case, ncntl=control, meta=TRUE)
```

Compute a summary table. Notice that in the `Subset.2sided` results, the first 2 SNPs have missing values for `OR.2`, `CI.low.2`, and `CI.high.2` since the estimated betas were all positive for these SNPs.

```
> h.summary(res)
```

\$Meta

	SNP	Pvalue	OR	CI.low	CI.high
1	SNP1	0.001436399	1.094	1.092	1.096
2	SNP2	0.052613004	1.097	1.092	1.102
3	SNP3	0.583876904	1.017	1.015	1.019

\$Subset.1sided

	SNP	Pvalue	OR	CI.low	CI.high	Pheno
1	SNP1	4.589243e-07	1.243	1.142	1.352	STUDY1,STUDY4
2	SNP2	3.231510e-03	1.286	1.088	1.520	STUDY2,STUDY4
3	SNP3	1.568320e-01	1.137	0.952	1.359	STUDY2

```

$Subset.2sided
  SNP      Pvalue      Pvalue.1      Pvalue.2      OR.1      CI.low.1      CI.high.1      OR.2
1 SNP1 1.575549e-06 1.658121e-07 0.55224459 1.243      1.143      1.351 0.944
2 SNP2 1.337943e-03 2.897340e-03 0.04659685 1.286      1.076      1.538 0.828
3 SNP3 6.374845e-02 6.825304e-02 0.17143707 1.137      0.960      1.347 0.827
  CI.low.2      CI.high.2      Pheno.1      Pheno.2
1      0.398      2.238      STUDY1,STUDY4      STUDY2
2      0.664      1.032      STUDY2,STUDY4      STUDY1
3      0.558      1.225      STUDY2      STUDY4

```

Intead of searching over all possible subsets, let us define our own subset function to determine which nsubsets to search over. We will only consider subsets where the first m traits are in the subset (m = 1, 2, ...). The DLM p-value will also be computed using only these subsets.

```

> sub.def <- function(logicalVec, args) {
+   sum <- sum(logicalVec)
+   ret <- all(logicalVec[1:sum])
+   ret
+ }

```

Call the h.traits function with the zmax.args pval.args options defined

```

> res <- h.traits(SNPs, studies, beta, sigma, ncase=case, ncntl=control, meta=TRUE,
+               zmax.args=list(sub.def=sub.def), pval.args=list(sub.def=sub.def))
> h.summary(res)

```

```

$Meta
  SNP      Pvalue      OR      CI.low      CI.high
1 SNP1 0.001436399 1.094      1.092      1.096
2 SNP2 0.052613004 1.097      1.092      1.102
3 SNP3 0.583876904 1.017      1.015      1.019

```

```

$Subset.1sided
  SNP      Pvalue      OR      CI.low      CI.high      Pheno
1 SNP1 7.363041e-07 1.268      1.154      1.393      STUDY1
2 SNP2 1.192172e-01 0.828      0.653      1.050      STUDY1
3 SNP3 4.378651e-01 1.048      0.931      1.178      STUDY1,STUDY2

```

```

$Subset.2sided
  SNP      Pvalue      Pvalue.1      Pvalue.2      OR.1      CI.low.1      CI.high.1      OR.2
1 SNP1 1.142435e-06 1.218183e-07 0.53445363 1.243      1.144      1.350 0.944
2 SNP2 1.923189e-03 4.341935e-03 0.04659685 1.207      1.049      1.390 0.828
3 SNP3 1.235183e-01 5.731355e-02 0.46625237 1.137      0.969      1.334 0.956
  CI.low.2      CI.high.2      Pheno.1      Pheno.2
1      0.254      3.504      STUDY1,STUDY4      STUDY2
2      0.664      1.032      STUDY2,STUDY3,STUDY4      STUDY1
3      0.333      2.739      STUDY2      STUDY1,STUDY4

```

Examples of h.types

The disease subtype variable in each study data frame is called TYPE, which has values "TYPE_1", "TYPE_2", "TYPE_3" and "CONTROL". First, we will combine the individual study data into a single data frame

```
> data <- NULL
> for (i in 1:nStudy) {
+   temp <- eval(parse(text=paste("data", i, sep="")))
+   temp[, "STUDY"] <- i
+   data <- rbind(data, temp)
+ }
```

In addition to age, we will also adjust for study. Create the study indicator variables:

```
> for (i in 1:nStudy) {
+   dvar <- paste("STUDY_", i, sep="")
+   data[, dvar] <- as.numeric(data[, "STUDY"] %in% i)
+ }
```

Define the SNP variables, adjustment variables and disease subtype labels.

```
> snp.vars <- paste("SNP", 1:3, sep="")
> adj.vars <- c("AGE", "STUDY_1", "STUDY_2", "STUDY_3")
> types.lab <- paste("TYPE_", 1:3, sep="")
```

Call the h.types function with option logit=TRUE to also run the overall logistic regression for each SNP.

```
> ret <- h.types(dat=data, response.var="TYPE", snp.vars=snp.vars, adj.vars=adj.vars, type
+               logit=TRUE)
```

Summarize the results

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

\$Overall.Logistic

	SNP	Pvalue	OR	CI.low	CI.high
1	SNP1	0.03149882	1.060	1.005	1.118
2	SNP2	0.01244612	1.121	1.025	1.225
3	SNP3	0.33157785	1.029	0.971	1.091

\$Subset.Case.Control

	SNP	Pvalue	OR	CI.low	CI.high	Pheno
1	SNP1	0.02142154	1.060	1.009	1.114	TYPE_1,TYPE_2,TYPE_3
2	SNP2	0.11204760	1.153	0.967	1.374	TYPE_1
3	SNP3	0.69068137	1.048	0.833	1.318	TYPE_1

\$Subset.Case.Complement

	SNP	Pvalue	OR	CI.low	CI.high	Pheno
1	SNP1	0.02677336	1.060	1.007	1.116	TYPE_1,TYPE_2,TYPE_3
2	SNP2	0.17692269	1.126	0.948	1.337	TYPE_1
3	SNP3	0.67116586	1.046	0.850	1.288	TYPE_1

Session Information

```
> sessionInfo()
```

```
R version 3.3.0 RC (2016-04-26 r70550)  
Platform: x86_64-apple-darwin13.4.0 (64-bit)  
Running under: OS X 10.9.5 (Mavericks)
```

```
locale:
```

```
[1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

```
attached base packages:
```

```
[1] grid      stats      graphics  grDevices  utils      datasets  methods  
[8] base
```

```
other attached packages:
```

```
[1] ASSET_1.10.0  tmvnsim_1.0-1 mvtnorm_1.0-5 rmeta_2.16    msm_1.6.1  
[6] MASS_7.3-45
```

```
loaded via a namespace (and not attached):
```

```
[1] Matrix_1.2-6  tools_3.3.0    expm_0.999-0   survival_2.39-2  
[5] splines_3.3.0  lattice_0.20-33
```