Pleiotropy tests for quantitive, binary, and ordinal traits with covariates

Dan Schaid, Jason Sinnwell

23 June, 2017

Example Data

The pleio package contains a demonstration dataset with three traits simulated from a normal, binomial, and ordinal distribution, respectively, stored in a matrix. Additionally, the dataset contains a simulated set of five covariates simulated from a normal distribution, and genotypes simulated based on minor allele frequency of 0.2. We assume that the traits and the covariates are not asssociated with dose of minor allele.

Here, we load the simulated dataset and show matrix y for traits, matrix x for covariates, and geno for dose of minor allele.

```
## load package and dataset
require(pleio)
data(pleio.demo)

## preview simulated data
head(y)
```

```
##         y.gaus y.bin y.ord
## [1,]  1.322771     0     3
## [2,]  1.552195     0     3
## [3,]  1.066899     0     2
## [4,]  1.597667     0     2
## [5,] -0.684680     1     1
## [6,] -1.612002     1     1
```

```
head(x)
```

```
##            var1        var2        var3        var4        var5
## [1,] -1.5717502  1.80268280 -0.04479937 -1.92511172  1.34489856
## [2,] -0.9178921 -0.18137723  0.24515112 -0.84067111  0.33476325
## [3,] -0.1959975 -0.29648506 -1.32939952 -0.08900092 -0.62272925
## [4,]  0.1258191 -0.20398319  1.58518754 -0.26349043 -1.58102977
## [5,]  1.2446132  1.58435872  0.91316989  1.60291419  2.33437675
## [6,] -1.1709365 -0.04318023  1.00328579  0.91223179  0.02197139
```

```
table(geno)
```

```
## geno
##   0   1   2
## 320 154  26
```

Sequential Pleiotropy Tests

The *pleio.glm.sequential* function is a high-level way to perform sequential tests of the number of traits (and which traits) are associated with a genotype. The algorithm starts with testing the usual multivariate null hypthothesis that all betas for the traits are zero. If this test rejects, because the p-value is less than a user-spiecifed threshold, then allow one beta to be non-zero in order to test whether the remaining betas = 0. If the test allowing for one non-zero beta rejects, then allow two non-zero betas (testing all combinations of two non-zero betas). Continue this sequential testing until the p-value for a test is greater than the specified threshold. When the sequential testing stops, one can conclude that the final model contains the non-zero betas, while all other betas are inferred to be zero. For m traits, the sequential testing stops either when the p-value is less than the threshold, or when (m-1) traits are tested. If the p-value remains less than *pval.threshold* when testing (m-1) traits, this implies that all m traits are associated with the genotype.

Below we run two functions, *pleio.glm.fit*, which performs pre-calculations on the models to be tested for the trait families in *glm.family*, and *pleio.glm.sequential*, which performs the sequential pleiotropy tests on the pre-computed object from *pleio.glm.fit*.

The final result lists the indices of the non-zero betas (the indices of the traits associated with a genotype), and the p-value that tests the fit of the final model. A p-value greater than *pval.threshold* is expected for the final model, showing that the final model fits the data well. For the simulated, which was simulated assuming all traits are not associated with the genotype, we apply the sequential testing using a large *pval.threshold* for illustration of the method (using an appropriate small threshold, such as 0.05, would terminate the sequential steps at the initial step). For this example, the sequential approach stopped at 2 traits because the p-value is greater than the *pval.threshold* argument of 0.5.

```
fit <- pleio.glm.fit(y, geno, glm.family=c("gaussian","binomial", "ordinal"))
out <- pleio.glm.sequential(fit, pval.threshold=.5)
out
```

```
## $pval
## [1] 0.531151
##
## $count
## [1] 2
##
## $index.nonzero.coef
## [1] 1 2
```

Equivalent Steps to Sequential Fit

The sequential steps above can be performed with more user control using *pleio.glm.test*, with *count.nonzero.coef* as the number of non-zero betas for the null hypothesis. The result of *pleio.glm.test* contains the global test statistic, degrees of freedom (df), p-value for testing the model, the indices of the non-zero betas in the model (the set of tested indices are given in *bk.set*, with columns for the different models tested and rows the indices of non-zero betas.), and a data.frame called *tests* that contains the tests performed for the null hypothesis models (i.e., the indices of the non-zero betas and the corresponding statistic, t_k, for each model). For m traits, and $k = count.nonzero.coef$, there are m-choose-k models in the set that are considered in the null hypothesis. The minimum test statistic over the set provides the global test statistic (*stat*) reported.

```
test0 <- pleio.glm.test(fit, count.nonzero.coef = 0)
test0
```

```
## $stat
## [1] 4.191602
##
## $pval
## [1] 0.241504
##
## $df
## [1] 3
```

```
##
## $index.nonzero.coef
## [1] 0
##
## $vk.set
## [1] 0
##
## $tk
## [1] 4.191602
```

```
test1 <- pleio.glm.test(fit, count.nonzero.coef = 1)
test1
```

```
## $stat
## [1] 1.495151
##
## $pval
## [1] 0.4735131
##
## $df
## [1] 2
##
## $index.nonzero.coef
## [1] 1
##
## $vk.set
##      [,1] [,2] [,3]
## [1,]    1    2    3
##
## $tk
## [1] 1.495151 1.723254 3.732726
```

```
test2 <- pleio.glm.test(fit, count.nonzero.coef = 2)
test2
```

```
## $stat
## [1] 0.392189
##
## $pval
## [1] 0.531151
##
## $df
## [1] 1
##
## $index.nonzero.coef
## [1] 1 2
##
## $vk.set
##      [,1] [,2] [,3]
## [1,]    1    1    2
## [2,]    2    3    3
##
## $tk
## [1] 0.3921890 1.4793814 0.5150495
```

Sequential Tests With Covariates

Both the sequential steps and the user-controlled steps can be run with covariates specific to each of the traits in the y matrix. All Possible covariates are collected into a single matrix called *x.all*. A separate list, called *x.index.list*, is used to define which covariates are to be used for each trait. By this approach, some covariates can be used for multiple traits, allowing traits to overlap with respect to the covariates, or covariates can be mutually exclusive for other traits, and even some traits can be defined without covariates. Below we show how to set the indices for each of the traits from a common data.frame, x, and *pleio.glm.sequential* considers the covariates specific to the trait when running the sequential steps. The indices need to be a list with a vector of indices for each of the traits y. We set *pval.threshold* again to be abnormally high for the purpose of demonstrating how it iterates to test multiple traits.

Each trait has a covariate

The first example shows how to specify varying number of covariates for each trait.

```r
index.cov <- list()
## cols 1 and 2 are covariates for trait 1, etc.
index.cov[[1]] <- c(1:2)
index.cov[[2]] <- c(2:4)
index.cov[[3]] <- c(4,5)

fit.cov <- pleio.glm.fit(y, geno, glm.family=c("gaussian","binomial","ordinal"), x.all=x,
                         x.index.list=index.cov)
cov.out <- pleio.glm.sequential(fit.cov, pval.threshold=.52)
print(cov.out)
```

```
## $pval
## [1] 0.5272492
##
## $count
## [1] 2
##
## $index.nonzero.coef
## [1] 1 2
```

One trait without a covariate

The second example shows how to specify a trait without any covariates, which is to give an index of 0. For this demonstration, *pval.threshold* is set high at 0.55 to illustrate output when all traits are considered to be associated with *geno*.

```r
index.cov <- list()
## cols 1 and 2 are covariates for trait 1, etc.
index.cov[[1]] <- c(1:2)
index.cov[[2]] <- c(2:4)
index.cov[[3]] <- 0

fit.cov <- pleio.glm.fit(y, geno, glm.family=c("gaussian","binomial","ordinal"), x.all=x,
                         x.index.list=index.cov)
cov.out <- pleio.glm.sequential(fit.cov, pval.threshold=.55)
print(cov.out)
```

```
## $pval
## [1] 1
##
## $count
## [1] 3
##
## $index.nonzero.coef
## [1] 1 2 3
```