ibdreg

(version 0.3.6)

Regression methods to test for linkage with covariates using IBD data from relative pairs

Jason P. Sinnwell and Daniel J. Schaid
Mayo Clinic, Rochester MN USA

September 3, 2021

Contents

1 Description 3

2 Operating System and Installation 3

3 Installing and Loading the Packae 3

4 Data Setup 4

4.1 Covariate Data Frame (*data*) 5

4.2 Creating IBD Files with Merlin and PERL Scripts 5

4.3 Identical-By-Descent (IBD) Data (*ibd.dat*) 6

4.4 Covariance Matrices for IBD Statistics (*ibd.var*) 7

4.4.1 Simulation-Based *ibd.var* 7

4.4.2 Exact *ibd.var* (via Merlin) 8

5 Creating Pair-Specific Covariates and Scaling for Relationships 8

5.1 Incorporating Covariates 8

5.2 Scaling Factors 9

6 Tests for Linkage 9

6.1 Linkage without Covariates 10

6.1.1 Linkage on AA Pairs 10

6.1.2 Linkage on ALL pairs 11

6.2 Linkage with Covariates 14

6.2.1 Linkage with One Covariate 14

6.2.2 Linkage with Multiple Covariates 15

7 Troubleshooting 16

7.1 Unexpected IBD Sharing 16

7.2 Unexpected IBD Sharing within Pedigrees 16

7.3 Correlation Between IBD Statistics 19

7.4 What Happened to My Data? 19

8 License and Warranty 21

9 Acknowledgements 21

1 Description

The ibdreg package is a suite of S-PLUS/R routines for the analysis of genetic linkage with covariates by regression methods that use identity by descent (IBD) sharing probabilities for relative pairs. The methods account for correlations of IBD statistics for relative pairs within the same pedigree.

The tests for genetic linkage use quasi-likelihood score statistics, formulated in terms of weighted least squares regression. The covariates in the regression framework are scaled according to the degree of relationship between relatives in a pair, so relatives pairs have an appropriate weight in the regression. The method yields the following tests:

- *Linkage*
- *Linkage with covariate effects*
- *Effect of covariate on IBD sharing*

These tests are applied to relative pairs that are of specific affection status: Affected-Affected (AA), Unaffected-Unaffected (UU), and Affected-Unaffected (AU). In the same framework, it is possible to test linkage using all relative pairs while imposing constraints on excess allele sharing in the direction expected under linkage.

2 Operating System and Installation

The ibdreg package version 0.3.6 is written for R R (version 4.0.3) and has been placed on the Comprehensive R Archive Network (CRAN), where it is made available on other systems. It is assumed that users have a general working knowledge of the R language.

3 Installing and Loading the Packae

To install ibdreg refer to instructions given in *README.ibdreg*. The package may be installed for global or local use on your system. After installing the ibdreg library, the routines are available by starting a session and attaching the appropriate directory using *library()* as illustrated below.

```
## if local library, use:
         library(ibdreg, lib.loc="/your/local/path/")
```

```
R> # load the library, load and preview at demo dataset
R> library(ibdreg, lib="~/Rlib403")
R> sessionInfo()
```

```
R version 4.0.3 (2020-10-10)
Platform: x86_64-pc-linux-gnu (64-bit)
```

```
Running under: CentOS Linux 7 (Core)

Matrix products: default
BLAS:   /usr/lib64/libblas.so.3.4.2
LAPACK: /usr/lib64/liblapack.so.3.4.2

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] tools     stats     graphics  grDevices utils
[6] datasets  methods   base

other attached packages:
[1] ibdreg_0.3.6

loaded via a namespace (and not attached):
[1] compiler_4.0.3
```

4 Data Setup

The methods within ibdreg require initial setup of three data objects:

- *data* A data frame that includes a row for each person, with pedigree ID, person ID, affection status, and other covariates.
- *ibd.dat* An object that includes a row for each relative pair with the IBD sharing probabilities. This object includes pedigree and person identifiers for the relative pairs, the null prior probabilities of sharing 0, 1, or 2 alleles IBD for the relative pairs, and the estimated posterior probabilities of sharing 0, 1, or 2 alleles IBD at each position on a chromosome.
- *ibd.var* A list containing one entry for each pedigree. The entry comprises pedigree and person identifiers, and the variance-covariance matrix of IBD statistics for pairs of relative pairs.

The examples in this manual use data from a linkage study for prostate cancer. The data show linkage on chromosome 20, and use the covariates diagnosis age (*dxage*) and tumor grade (*grade*).

The ibdreg package is not provided with example data because the files are too large.

4.1 Covariate Data Frame (*data*)

First, because ibdreg uses data for each person to construct pair-specific covariates, a data frame containing the covariates for each person is needed. This data frame must have columns that represent pedigree id, person id, and affection status. The two identifiers (pedigree id and person id) must uniquely identify each person in the data frame. The column for affection status must be coded as 2 for affected, 1 for unaffected, and *NA* for unknown. The *NA* is a special code for missing values in S-PLUS and R; missing values in the covariate columns must also be coded to *NA*.

In the example below, *read.table* is used to read in fields to *cov.data* from a comma-delimited file, *"cov.data.csv"*, where each row represents one subject, and the first row contains the column names. The *names* function is used on *cov.data* to check that the column names are as expected. The required columns mentioned above are named *ped.id* (pedigree id), *person.id* (personal id), and *pcstat* (affection status).

```
R> cov.data <- read.table("cov.data.csv", sep=",", header=TRUE)
R> names(cov.data)

 [1] "ped.id"     "person.id" "sex"        "pcstat"
 [5] "liab"       "grade"     "dxage"      "stage"
 [9] "nodes"      "dzgroup"   "naff"

R>
```

4.2 Creating IBD Files with Merlin and PERL Scripts

To facilitate use of this S-PLUS/R library, a number of PERL scripts are provided within the *ibdreg/perl* subdirectory. These scripts must be run outside of S-PLUS/R. Two file types are required for ibdreg: **(1)** posterior IBD probability file for relative pairs (a file for each specified chromosome), **(2)** prior IBD probability file for relative pairs (one for autosomes, and one for the X chromosome).

The two file types can be generated by either Genehunter[6], Allegro[4], or Merlin[1]. All provided PERL scripts assume use of Merlin, but they can be altered for use of other software.

The input files to the PERL scripts are the LINKAGE formatted files. We refer to these files as the "par" file (for loci parameter descriptions; called the DATAFILE in LINKAGE documentation) and the "pre" file (the pedigree information along with genotypes; called the PEDFILE in LINKAGE documentation). We call the second input file "pre" because the file format is the pre-MAKEPED format. These pre and par file formats are the standard formatted files for input to GENEHUNTER.

Two PERL scripts perform separate tasks of creating the two file types listed above, *merlin.post.ibd.pl* and *merlin.prior.ibd.pl*. An additional script, *gws.merlin.ibd.pl*, is a ”master” script that controls the other two scripts to make files for the posterior and prior IBD files needed for a genome-wide linkage scan using ibdreg.

The master script uses specific directory names and file name signatures which may need to be changed by the user. The script expects pre and par files to be in a subdirectory named *INPUTDIR*, and the posterior and prior IBD output files are placed in directories *PostIBD* and *PriorIBD*, respectiviely. The "signature" naming patterns expected for the pre and par files for chromosome k are "ch.[k].pre" and "ch.[k].par", respectively. The signature naming pattern for output files are "chr[k].post.ibd" and "chr[k].prior.ibd". Note that prior probabilities are only generated for chromosomes 1 and 23 (X) because the prior probabilities are the same for all autosomes (chromosomes 1 - 22).

The other two scripts should not require editing, unless one wishes to change the optional parameters that control Merlin, such as the spacing of calculations along the chromosomes, or how probable genotype errors are handled. Currently, calculations are for 1 cM intervals along each chromosome, and probable genotype errors are "wiped out" using Merlin's pedwipe step. These scripts are briefly described below.

The *merlin.post.ibd.pl* script sets up the files needed for Merlin to compute the posterior IBD probabilities. Note that this script runs Merlin and Pedwipe to use only "cleaned" genotypes: first Merlin is run to check for likely errors, then Pedwipe is run to remove them, then Merlin is run a second time, without the likely erroneous genotypes, to dump the IBD probabilties to a file. These steps and the Merlin options can easily be modifed as desired.

The *merlin.prior.ibd.pl* script sets up the files needed for Merlin to compute null prior IBD probabilities. This is achieved by creating a dummy homozygous (non-informative) marker for all subjects. The script then runs Merlin to dump the IBD information to a file. The required input is the name of a pre file and the chromosome number (1-23). The Merlin IBD file is then output.

4.3 Identical-By-Descent (IBD) Data (*ibd.dat*)

The IBD data created by *gws.merlin.ibd.pl* or *merlin.post.ibd.pl* in section 4.2 must be read into S-PLUS/R by the function *create.ibd.dat*. This function makes an object of class *ibd.dat* containing IBD sharing probabilities for relative pairs. The relative pairs within each pedigree are distinguished by a pedigree identifier (*ped.id*) and identifiers for the two relatives (*person1.id, person2.id*). To make an *ibd.dat* object, specify the two file names with probabilities for posterior IBD (*postfile*) and prior IBD (*priorfile*) created in section 4.2. Two files used here are named "chr20.post.ibd" and "chr1.prior.ibd", as placed in the *PostIBD* and *PriorIBD* directories, respectively, by *gws.merlin.ibd.pl*.

```
R> ibd.dat.obj <- create.ibd.dat(postfile="./PostIBD/chr20.post.ibd",
+                                priorfile="./PriorIBD/chr1.prior.ibd",
+                                x.linked=FALSE,
+                                cov.data=NULL,
+                                rm.noninform=TRUE)
R> names(ibd.dat.obj)
```

```
[1] "ped.id"      "person1.id" "person2.id" "post0"
[5] "post1"       "post2"      "prior0"     "prior1"
[9] "prior2"

R>
```

The named elements *post0*, *post1*, and *post2*, are N x L data frames containing posterior probabilities of sharing 0, 1, or 2 alleles IBD for each of the N relative pairs at L positions. The corresponding prior IBD sharing probabilities are vectors *prior0*, *prior1*, and *prior2*.

Some relative pairs are not informative for linkage. By setting the parameter *rm.noninform=TRUE*, *create.ibd.dat* removes uninformative relative pairs. Uninformative pairs are detected if their posterior IBD sharing probabilities, at all positions, are equivalent to their prior IBD sharing probabilities. The *ibdreg()* function also removes uninformative relative pairs if it detects them.

4.4 Covariance Matrices for IBD Statistics (*ibd.var*)

The analyses in ibdreg require an object containing the covariance matrices for IBD sharing statistics within families under the null hypothesis of no linkage. This will be referred to as an *ibd.var* class object. This class of object can be estimated using simulations or calculated exactly via Merlin. The simulation routine is recommended for large pedigrees, because Merlin cannot handle them. In addition, the extra steps between Merlin and S-PLUS/R leave more room for pitfalls for the user.

4.4.1 Simulation-Based *ibd.var*

Within an R or S-PLUS session, the *sim.ibd.var* function makes an *ibd.var* object using the information from a pre file (see section 4.2). The function approximates the joint distribution of IBD sharing statistics by simulating gene-dropping for each pedigree (see Schaid et al. [9]). From many simulated IBD sharing values a sample variance-covariance matrix can be estimated under the assumption of no linkage. The *sim.ibd.var* function needs the pre file name (*prefile*) and allows the user to choose the number of simulations *n.sim*. Sufficient accuracy is usually reached after 10,000 simulations for most pedigrees.

```
R> ibd.var.obj <- sim.ibd.var("./INPUTDIR/ch.20.pre", n.sim=10000,
+                                  x.linked=FALSE)
R> names(ibd.var.obj[[1]])

[1] "ped.id"      "person1.id" "person2.id" "sm"
[5] "sv"
```

The *ibd.var* object is a list with an element for each pedigree. For each pedigree, in addition to the covariance matrix of the IBD sharing statistics (*sv*), the *ibd.var* object contains the pedigree id (*ped.id*), personal identifiers for persons within each relative pair (*person1.id, person2.id*), and the mean IBD sharing (assuming no linkage) for each relative pair (*sm*).

4.4.2 Exact *ibd.var* (via Merlin)

Exact values for the variance-covariance matrices of ibd sharing statistics within families are made possible through Merlin. The steps are managed by a PERL script named *exact.ibd.var.pl*. The script first adds to the pre file a dummy homozygous marker (as done in *merlin.prior.ibd.pl* in section 4.2). It then runs Merlin with the options *“–ibd –matrices”* to make null prior probabilities for all possible IBD sharing configurations of relatives within each pedigree. These probabilities are used to create the mean vector and covariance matrix for IBD sharing statistics, assuming no linkage, for all relative pairs within a pedigree. This script uses as input a pre file, the chromosome number (1 or 23) and the name of the output file. The *gws.merlin.ibd.pl* script has an option to run *exact.ibd.var.pl* for chromosomes 1 and 23 if *doExactVar* is set to 1.

To create the *ibd.var* object, assume you have a file named *“ibd.var.out”* made from *exact.ibd.var.pl* then use *exact.ibd.var()* as done below.

```
R> ibd.var.ex <- exact.ibd.var("./PriorIBD/ibd.var.out")
R>
```

5 Creating Pair-Specific Covariates and Scaling for Relationships

When using the *ibdreg()* routine, one must define pair-specific covariates for the regression framework, as well as choose a scaling factor, both of which are described below.

5.1 Incorporating Covariates

The *ibdreg()* function is built to work with the modeling capabilities of the S language. This typically means that the function expects the first parameter to be a formula where the dependent variable is on the left side, followed by a tilde (˜), followed by the independent variables on the right side. The *formula* parameter is accompanied by the *data* parameter, which contains all the variables specified in *formula*.

The *formula* parameter in *ibdreg()* differs from the above standard because *data* contains covariate data for individuals, while the dependent variable is more complex than that typically used in formulas (i.e., a linear combination of two matrices with IBD posteriors). The *formula* parameter in *ibdreg()* is used to create covariates for relative pairs. This provides enormous flexibility, allowing users to create their own functions for making pair-specific covariates. Note that the dependent variable, estimated IBD sharing, is calculated from posterior probabilities in *ibd.dat*. Thus, the *formula* parameter contains only the right side of the regression formula, which can either be *˜1* (for intercept-only, used for linkage tests without covariates), or user-defined functions that make pair-specific covariates from person-specific covariates in *data*. Below are examples of how to make user-defined functions to use in *formula*, and a simple illustration of how they work.

```
R> pairSum <- function(cov1,cov2) { cov1 + cov2 }
R> pairDiff <- function(cov1,cov2) { abs(cov1 - cov2) }
```

```
R> pairSum(20,30)

[1] 50

R> pairDiff(20,30)

[1] 10

R>
```

These two functions are included in ibdreg, but *formula* can accept any user-defined function that meets the following two restrictions. (1) Input two vectors, which are the same covariate measured on the two persons in a relative pair. (2) Output a single vector which does not depend upon order of the input vectors. Furthermore, the syntax in the call of *ibdreg()* allows the user to specify the covariate name once or twice, but only the first one is used. Two different covariate names are not allowed. Here are examples of formulas that *ibdreg()* accepts:

- *~1*
- *~pairDiff(dxage)*
- *~pairDiff(dxage,dxage) + pairSum(grade, grade)*

5.2 Scaling Factors

The scaling factor described in Schaid et al, 2006[9] is used as a weight for all covariates, so that all relative pairs can be used in the same regression model. The basic idea is that relative pairs who are closely related should have a greater weight in a regression than more distant relative pairs. The parameter to specify scaling factor, *c.scale*, can be either of two options:

- *"minimax"* constraint from Goddard and Olson [2]
- *"nodom"* no dominance variance from Greenwood and Bull [3]

For all examples in this manual, the no dominance variance option is used.

6 Tests for Linkage

One can test for linkage on either of the AA, UU, or AU relative pair subsets. When linkage exists, pairs within each group are expected to have homogeneous linkage signals, where both AA and UU pairs are expected to share an excess of alleles IBD in the presence of linkage, and AU pairs are expected to share a deficit of alleles IBD.

The null hypothesis is no linkage and no covariate effect, and the alternative hypothesis is linkage. In the presence of linkage, IBD sharing is expected to deviate from the null hypothesis in only one direction, leading to one-sided, or constrained, tests. For all of the tests, *ibdreg()* performs both the unconstrained and constrained tests.

The object returned from *ibdreg()* is given the class *ibdreg*. The tests contained in an *ibdreg* object depend on what the user specifies. The selection of tests is controlled by two parameters, *formula* and *status.method*. *status.method* is the parameter that specifies which affection status group (AA, UU, AU, ALL) to test. For *status.method* of AA, UU, and AU, the possible tests are linkage without covariates, linkage with covariates, and covariate effects on IBD sharing. If *formula=˜1*, linkage without covariates is performed. If a pair-specific covariate is given, then all three tests are performed. If *status.method=ALL*, these same principles apply to each of the status groups where the tests are performed on each group. Additionally for ALL, regardless of the formula, a constrained test for linkage on ALL pairs is performed.

Table 1 below shows which tests are performed for each combination of these parameters. The rows are the tests, the columns are the choice of *status.method*, and in each box is a code indicating a specified formula. In the table, *1* represents *formula=˜1* and *X* represents a formula that includes pair-specific covariates, as defined in section 5.1.

Table 1. Linkage tests performed according to *status.method*

	AA	UU	AU	ALL
Linkage without Covariate	1, X	1, X	1, X	1, X
Linkage with Covariate(s)	X	X	X	X
Covariate Effect on IBD	X	X	X	X
All Pairs Linkage				1, X

6.1 Linkage without Covariates

To illustrate testing for linkage without covariates, the basic tests are performed first on only AA pairs, and then ALL pairs.

6.1.1 Linkage on AA Pairs

The first example is the basic test for linkage without incorporating covariates on just the AA relative pairs. The two parameters used for this analysis are *status.method=”AA”* and *formula = ˜1*. For scaling factor of no dominance variance, set *c.scale=”nodom”*. To specify the data objects for this example, use *data=cov.data*, *ibd.dat=ibd.dat.obj*, and *ibd.var=ibd.var.obj*. To specify the required column names of *cov.data*, use *ped.id=ped.id*, *person.id=person.id*, and *status=pcstat*.

The example below saves the test results of *ibdreg()* in *nocov.AA*, and prints the results using the print method (*digits=3* means print 3 significant digits). An explanation of the results follows.

```
R> #  AA pairs, no covariates (linkage only)
R>
R> nocov.AA <- ibdreg(formula = ~1, status.method="AA", c.scale="nodom",
+                            data=cov.data, status=pcstat, ped.id=ped.id,
+                       person.id=person.id,
+                       ibd.dat=ibd.dat.obj, ibd.var=ibd.var.obj, epsilon=1e-5)
R> print(nocov.AA, digits=3)

ibdreg(formula = ~1, status.method = "AA", c.scale = "nodom",
    data = cov.data, status = pcstat, ped.id = ped.id, person.id = person.id,
    ibd.dat = ibd.dat.obj, ibd.var = ibd.var.obj, epsilon = 1e-05)


============================================================
                          AA PAIRS
============================================================
Number of pedigrees used:             159
Number of relative pairs:             429

============================================================
                  Score Tests for Linkage
============================================================
                             pos(cM) Score d.f.   pvalue
            Linkage w/o Cov       76  11.4    1 0.000724
constrained Linkage w/o Cov       76  11.4  0:1 0.000362

R>
```

Explanation of Results

First, the print method displays the call to *ibdreg()* made by the user. Then it recognizes which tests were performed and were saved in the object. Since the results in *nocov.AA* were the linkage tests on AA pairs, only details for those tests are given. The first section shows the number of pedigrees and relative pairs used in the tests. The "Score Tests for Linkage" section displays test results for both unconstrained and constrained linkage tests. The above table shows results for only the maximum score statistic over all chromosome positions (although results for all positions are stored in the returned object, *nocov.AA*). Each line in the above table shows the name of the test, the chromosome position (in centiMorgans) where the maximum score test occured, the score statistic, degrees of freedom, and P-value. Note that the constrained test for linkage is a mixture of chi-square distributions with 0 and 1 degrees of freedom, whereas the unconstrained statistic has a chi-square distribution with 1 degree of freedom.

6.1.2 Linkage on ALL pairs

To test linkage without covariates for all relative pairs, use the same parameters as above, except for *status.method="ALL"*. The tests for linkage without covariates will be performed on AA, UU, and

AU subsets, then another constrained test for linkage on ALL pairs. This last test constrains the affection-status groups to IBD allele sharing under the assumption of linkage. Therefore, the test constrains IBD sharing for AU pairs to share less than the null sharing, UU pairs are constrained to share more than the null, and AA pairs are constrained to share more than UU pairs. For more details on this test see Schaid et al, 2006 [9].

```
R> # status method = 'ALL' for all pairs
R> # linkage-only tests are done on AA, AU, UU,
R> # then another constrained linkage test on all pairs at once
R>
R> nocov.ALL <- ibdreg(formula=~1, status.method="ALL", c.scale="nodom",
+                        status=pcstat, ped.id=ped.id, person.id=person.id,
+                        data=cov.data, ibd.dat=ibd.dat.obj,
+                        ibd.var=ibd.var.obj, epsilon=1e-5)
R> print(nocov.ALL, digits=3)
```

```
ibdreg(formula = ~1, status.method = "ALL", c.scale = "nodom",
    data = cov.data, status = pcstat, ped.id = ped.id, person.id = person.id,
    ibd.dat = ibd.dat.obj, ibd.var = ibd.var.obj, epsilon = 1e-05)


============================================================
                          AA PAIRS
============================================================
Number of pedigrees used:             159
Number of relative pairs:             429

============================================================
                  Score Tests for Linkage
============================================================
                             pos(cM) Score d.f.   pvalue
            Linkage w/o Cov       76  11.4    1 0.000724
constrained Linkage w/o Cov       76  11.4  0:1 0.000362


============================================================
                          UU PAIRS
============================================================
Number of pedigrees used:               7
Number of relative pairs:               7
```

```
============================================================
                  Score Tests for Linkage
============================================================
                             pos(cM) Score d.f. pvalue
            Linkage w/o Cov       56 1.303    1  0.254
constrained Linkage w/o Cov      102 0.119  0:1  0.365


============================================================
                          AU PAIRS
============================================================
Number of pedigrees used:              27
Number of relative pairs:              59

============================================================
                  Score Tests for Linkage
============================================================
                             pos(cM) Score d.f. pvalue
            Linkage w/o Cov       96  1.07    1  0.301
constrained Linkage w/o Cov       69  0.34  0:1  0.280

============================================================
                         ALL PAIRS
============================================================
Number of pedigrees used:             159
Number of relative pairs:             495
                AA pairs:             429
                UU pairs:               7
                AU pairs:              59

============================================================
                   Score Test for Linkage
============================================================
                     pos(cM) Score Test    d.f.   pvalue
constrained Linkage       76       12.4 0:1:2:3 0.000732

R>
```

Explanation of Results

The print results follow the same format as in section 6.1.1 above, except that now there are three sections that show the linkage without covariate results for AA, UU, and AU relative pairs. The last section, “ALL PAIRS”, contains output of a similar format, showing first the counts of pedigrees and relative pairs for all three groups of relative pairs, then the results for the constrained test that uses all relative pairs. Again, the test shown is the maximum score statistic over all chromosome positions. Note that this test statistic is a mixture of chi-square distributions with 0, 1, 2, and 3 degrees of freedom.

Instead of printing all test results at all chromosome positions, one can view test results at all positions in a plot, with the x-axis the chromosome position, and the y-axis the -log10(pvalue) of the test statistic. An example is given in Figure 1 at the end of this document. For *nocov.ALL*, the plot method plots the constrained tests for AA, UU, and AU pairs, as well as the constrained test that uses all pairs. The color codes are given in the legend in the upper left corner of Figure 1.

6.2 Linkage with Covariates

6.2.1 Linkage with One Covariate

To demonstrate linkage tests that include a covariate for relative pairs, the sum of the variable in *data* called *dxage*, “age of diagnosis” is used. Since this covariate could not be measured for unaffected relatives, this example only applies to AA relative pairs. The call to *ibdreg* is similar to the two previous calls, but with *formula= pairSum(dxage)* to specify the covariate.

```
R> sum.dxage.AA <- ibdreg(formula=~pairSum(dxage), status.method="AA", c.scale="no
+                  status=pcstat, ped.id=ped.id, person.id=person.id, data=cov.dat
+                  ibd.dat=ibd.dat.obj, ibd.var=ibd.var.obj)
R> print(sum.dxage.AA, digits=2)
```

```
ibdreg(formula = ~pairSum(dxage), status.method = "AA", c.scale = "nodom",
    data = cov.data, status = pcstat, ped.id = ped.id, person.id = person.id,
    ibd.dat = ibd.dat.obj, ibd.var = ibd.var.obj)


==========================================================
                         AA PAIRS
==========================================================
Number of pedigrees used:            159
Number of relative pairs:            429

==========================================================
         Score Tests for Linkage with Covariate(s)
==========================================================
                                pos(cM) Score d.f.  pvalue
                Linkage w/o Cov      76  11.4    1 0.00072
    constrained Linkage w/o Cov       76  11.4  0:1 0.00036
                 Linkage w/ Cov       82  16.3    2 0.00029
     constrained Linkage w/ Cov       82  16.3  1:2 0.00017
            Cov Effect (robust)       82   8.8    1 0.00302
constrained Cov Effect (robust)       82   8.8    1 0.00302

R>
```

Explanation of Results

The print results again follow the format of results illustrated in section 6.1.1, except that additional tests were performed. In the table of maximum score statistics for each test, notice first that the tests for linkage without covariates (labeled *w/o Cov*) are the same as in *nocov.AA* results. The next two lines (labeled *w/ Cov*) show the unconstrained and constrained tests for linkage with covariate effect, against the null of no linkage. The distribution of the constrained test is a mixture of chi-square distributions with 0 and 1 degrees of freedom. The final two tests are the unconstrained and constrained tests for covariate effect on IBD sharing (using a robust variance in the score test). Both of these test statistics are distributed as chi-square with 1 degree of freedom. See Schaid et al, 2006 [9] for more details on the tests performed.

A plot of *sum.dxage.AA* results is illustrated in Figure 2 at the end of this document. The three lines illustrate the constrained tests for linkage without the covariate, linkage with covariate *pairSum(dxage)*, and effect of *pairSum(dxage)* on IBD sharing.

6.2.2 Linkage with Multiple Covariates

To demonstrate how multiple covariates can be specified in *ibdreg()* using the *formula*, *pairSum* is used for *dxage* and *pairSqDiff* (created below) is applied to *grade* (grade of cancer tissue). The printed results will be similar in format to the results in section 6.2.

```
R> pairSqDiff <- function(cov1, cov2) { (cov1-cov2)^2 }
R> dxage.grade.AA <- ibdreg(formula=~pairSum(dxage) + pairSqDiff(grade),
+                          status.method="AA", c.scale="nodom",
+                          status=pcstat, ped.id=ped.id, person.id=person.id,
+                          data=cov.data, ibd.dat=ibd.dat.obj, ibd.var=ibd.var.obj)
R> print(dxage.grade.AA, digits=3)
```

```
ibdreg(formula = ~pairSum(dxage) + pairSqDiff(grade), status.method = "AA",
    c.scale = "nodom", data = cov.data, status = pcstat, ped.id = ped.id,
    person.id = person.id, ibd.dat = ibd.dat.obj, ibd.var = ibd.var.obj)


============================================================
                          AA PAIRS
============================================================
Number of pedigrees used:             135
Number of relative pairs:             308

============================================================
         Score Tests for Linkage with Covariate(s)
============================================================
                                 pos(cM) Score d.f.  pvalue
                Linkage w/o Cov       69  9.41    1 0.00216
    constrained Linkage w/o Cov       69  9.41  0:1 0.00108
                  Linkage w/ Cov       82 11.13    3 0.01102
     constrained Linkage w/ Cov       82 11.13  2:3 0.00742
            Cov Effect (robust)       82  5.59    2 0.06098
constrained Cov Effect (robust)       82  5.59    2 0.06098

R>
```

7 Troubleshooting

7.1 Unexpected IBD Sharing

Many calculations in the linkage tests are useful for checking model assumptions. For example, it is assumed that the markers segregate in Mendelian fashion. If this is violated, linkage results could be biased. One way for markers to deviate from Mendelian expectation is when the parental genotypes are missing and allele frequencies are misspecified. This can cause unexpected IBD sharing for all types of relative pairs.

One way to find evidence for unexpected allele sharing is to compare the constrained and unconstrained linkage tests. If all the allele sharing deviates in the direction assumed under linkage, the test statistics will be the same, yet the p-values for the constrained test will be smaller because the constrained test is distributed as a mixture chi-square. If there is unexpected allele sharing, the unconstrained test for linkage can have a smaller p-value than the constrained test. The two tests are illustrated in Figure 3, at the end of the document. The figure was created using the *plot.ibdreg.unexpect* function for *status.method="AA"* relative pairs. The plot shows there is very little evidence of unexpected allele sharing over the whole chromosome for AA pairs.

7.2 Unexpected IBD Sharing within Pedigrees

Further details of unexpected allele sharing are available. The ibdreg object contains a matrix with z-scores for allele sharing for each pedigree (columns) at each chromosome position (rows).

One way to use this matrix is to find pedigrees that may have patterns of unexpected allele sharing. For the AA pairs within each family, the IBD sharing is unexpected if the z-score is less than zero. To find a pedigree that has unexpected sharing within its AA pairs, one can compute the mean z-scores of AA pairs within pedigrees over all positions, illustrated in *ped.mean* below. These can then be sorted, and the pedigree identifier can be examined to look at information on the pedigrees whose mean z-score was low for AA pairs.

```
R> ped.mean <- apply(nocov.ALL$AA.linkage$ped.zscore,2,mean)
R> sort(ped.mean)[1:10]

        39        110        100        176        141
-1.2459832 -0.9493389 -0.8066873 -0.7916735 -0.7868101
         6        170         55         49        117
-0.7467697 -0.7429451 -0.7281821 -0.6984625 -0.6944832
```

Information on the pedigree with the most unexpected IBD allele sharing for AA pairs (ped.id = 39) can be found in both *ibd.var.obj* and *cov.data* as done below. The *print.ibd.var* function prints the *ibd.var* information for specified pedigrees in *ped.id*. As shown below, output from *print.ibd.var* is long; the *sinkfile* parameter can be used to name a file to sink more pedigrees output.

```
R> cov.data[cov.data$ped.id==39,]

    ped.id person.id sex pcstat liab grade dxage stage
277     39         1   1      2    4    NA    58     3
278     39       102   2      1    1    NA    NA    NA
279     39       105   1      1    1    NA    NA    NA
280     39       106   1      2    2    NA    56     1
281     39      1000   1      1    1    NA    NA    NA
282     39      1001   2      1    1    NA    NA    NA
    nodes dzgroup naff
277     1       2    3
278    NA      NA    3
279    NA      NA    3
280     0       0    3
281    NA      NA    3
282    NA      NA    3
```

```
R> print.ibd.var(ibd.var.obj, ped.id=39, sinkfile=NULL, digits=3)

$ped.id
[1] 39

$person1.id
 [1]    1    1    1    1    1  102  102  102  102  105  105
```

```
[12]  105  106  106 1000

$person2.id
 [1]  102  105  106 1000 1001  105  106 1000 1001  106 1000
[12] 1001 1000 1001 1001

$sm
 [1] 1.004 1.011 0.999 1.000 1.000 0.995 1.001 1.000 1.000
[10] 0.999 1.000 1.000 1.000 1.000 0.000

$sv
           [,1]     [,2]     [,3] [,4] [,5]     [,6]
 [1,]  0.498632 -0.00745  0.00350    0    0  0.00162
 [2,] -0.007446  0.50433 -0.00159    0    0  0.00205
 [3,]  0.003503 -0.00159  0.48705    0    0  0.00170
 [4,]  0.000000  0.00000  0.00000    0    0  0.00000
 [5,]  0.000000  0.00000  0.00000    0    0  0.00000
 [6,]  0.001619  0.00205  0.00170    0    0  0.49983
 [7,]  0.006296 -0.00281  0.00110    0    0 -0.00109
 [8,]  0.000000  0.00000  0.00000    0    0  0.00000
 [9,]  0.000000  0.00000  0.00000    0    0  0.00000
[10,] -0.000696  0.00461  0.00780    0    0  0.00350
[11,]  0.000000  0.00000  0.00000    0    0  0.00000
[12,]  0.000000  0.00000  0.00000    0    0  0.00000
[13,]  0.000000  0.00000  0.00000    0    0  0.00000
[14,]  0.000000  0.00000  0.00000    0    0  0.00000
[15,]  0.000000  0.00000  0.00000    0    0  0.00000
          [,7] [,8] [,9]     [,10] [,11] [,12] [,13] [,14]
 [1,]  0.00630    0    0 -0.000696     0     0     0     0
 [2,] -0.00281    0    0  0.004611     0     0     0     0
 [3,]  0.00110    0    0  0.007800     0     0     0     0
 [4,]  0.00000    0    0  0.000000     0     0     0     0
 [5,]  0.00000    0    0  0.000000     0     0     0     0
 [6,] -0.00109    0    0  0.003496     0     0     0     0
 [7,]  0.49665    0    0 -0.002499     0     0     0     0
 [8,]  0.00000    0    0  0.000000     0     0     0     0
 [9,]  0.00000    0    0  0.000000     0     0     0     0
[10,] -0.00250    0    0  0.501449     0     0     0     0
[11,]  0.00000    0    0  0.000000     0     0     0     0
[12,]  0.00000    0    0  0.000000     0     0     0     0
[13,]  0.00000    0    0  0.000000     0     0     0     0
```

```
[14,]  0.00000    0    0  0.000000     0     0     0     0
[15,]  0.00000    0    0  0.000000     0     0     0     0
      [,15]
 [1,]     0
 [2,]     0
 [3,]     0
 [4,]     0
 [5,]     0
 [6,]     0
 [7,]     0
 [8,]     0
 [9,]     0
[10,]     0
[11,]     0
[12,]     0
[13,]     0
[14,]     0
[15,]     0

R>
```

7.3 Correlation Between IBD Statistics

There is known to be dependence on the IBD statistics between sets of relative pairs, which is why *ibdreg()* uses the covariance matrix for these statistics. One specific set of relative pairs is known to have complete negative correlation: a grandchild-grandmother and the same grandchild with its grandfather. In this case, if the grandchild shares an allele with the grandmother, the grandchild cannot share an allele with the grandfather.

This scenario can cause the variance-covariance matrix to be less than full rank for some pedigrees. The genaralized inverse of these matrices is handled by *Ginv()*, which uses singular value decomposition. In some instances, warnings may occur that indicate a negative variance estimate. The variance estimate could be negative because of a very small singular value that should be considered zero. In this case, the user can specify the cut-off value for singular values; this can be managed in *ibdreg()* with the *epsilon* parameter. It is used in section 6.1.2 when making *nocov.ALL*.

7.4 What Happened to My Data?

Several elements of the *ibdreg* object were included to keep track of relative pairs in the analysis. It is possible to see how many pairs are within each pedigree in the *relpair.tbl*. See below the number of relative pairs in the first 10 pedigrees, stored in *relpair.tbl*. Also, the counts of relative pair groups by status of AA, UU, AU are in *status.tbl*.

```
R> names(nocov.ALL)

[1] "Call"        "relpair.tbl" "status.tbl"  "data.rm.df"
[5] "pairs.rm.df" "AA.linkage"  "UU.linkage"  "AU.linkage"
[9] "ALL.linkage"

R> nocov.ALL$relpair.tbl[1:10]

 1  2  3  4  5  6  7  8  9 10
 6  1  6 15  3  6  3  3  6  6

R> nocov.ALL$status.tbl

  AA.count UU.count AU.count
1      429        7       59

R>
```

Relative pairs are not used in the analysis when they are either not informative for linkage, missing affection status, or missing a covariate. Within the *ibdreg* result, *pairs.rm.df* keeps the identifiers for the relative pairs that are removed because they were uninformative or one of the persons had a missing value in *data*. Another data frame with removed data is *data.rm.df*, which has the identifiers for persons who appear in *data*, but not in IBD calculations.

Since the number of removed pairs can be many, the output below presents a print of the number of rows in *pairs.rm.df*, just to show the number of relative pairs removed. Likewise, the number of persons in *data.rm.df* is given.

```
R> nrow(nocov.ALL$pairs.rm.df)

[1] 216

R> nrow(nocov.ALL$data.rm.df)

[1] 449

R>
```

8 License and Warranty

License:

For other licensing arrangements, please contact Daniel J. Schaid.
Daniel J. Schaid, Ph.D.
Division of Biostatistics
Harwick Building - Room 775
Mayo Clinic
200 First St., SW
Rochester, MN 55905
phone: 507-284-0639
fax: 507-284-9542
email: schaid@mayo.edu

9 Acknowledgements

This research was supported by United States Public Health Services, National Institutes of Health, contract grant numbers GM67768.

References

[1] Abecasis GR, Cherny SS, Cookson WO, Cardon LR. (2002). Merlin - Rapid Analysis of Dense Genetic Maps Using Sparse Gene Flow Trees. *Nat Gen* 30:97-101.

[2] Goddard KA, Witte JS, Suarez BK, Catalona WJ, Olson JM (2001) Model-Free Linkage Analysis with Covariates Confirms Linkage of Prostate Cancer to Chomosomes 1 and 4. *Am J Hum Genet* 68:1197-1206.

[3] Greenwood CMT, Bull SB (1999) Analysis of Affected Sib Pairs, With Covariates - With and Without Constraints. *Am J Hum Genet* 64:871-885.

[4] Gudbjartsson DF, Thorvaldsson T, Kong A, Gunnarsson G, Ingolfsdottir A (2005). Allegro Version 2. *Nat Gen.* 37: 1015-1016.

[5] Insightful Corporation. (2006, August 16). *Insightful Corporation: Statistical Analysis Software and Data Mining Software.* Retrieved August 16, 2006 from Insightful Corporation S-PLUS Version 6 Documentation Pages on the World Wide Web: http://www.insightful.com/support/splus60unix/unixug.pdf

[6] Kruglyak L, Daly MJ, Reeve-Daly MP, Lander ES. (1996). Parametric and Nonparametric Linkage Analysis: A Unified Multipoint Approach. *Am J Hum Gen* 58:1347-1363.

[7] Kruglyak L, Lander ES. (1998) Faster Multipoint Linkage Analysis Using Fourier Transforms. *Journal of Computational Biology* 5:1-7.

[8] The R Foundation for Statistical Computing. (2003, March 6). *The R Foundation for Statistical Computing.* Software home page on World Wide Web: <*http://www.r-project.org*>.

[9] Schaid DJ, Sinnwell JP, Thibodeau SN (2006) Testing Genetic Linkage with Relative Pairs and Covariates by Quasi-Likelihood Score Statistics. *Submitted.*

[10] Silvapulle M, Silvapulle P. (1995) A Score Test Against One-Sided Alternatives. *J Am Statist Assoc* 90:342-349.

[11] Venebles WN, Ripley BD (1994) Modern Applied Statistics with S-Plus. *Springer-Verlag* New York, NY.

```
R> plot(nocov.ALL)
R>
```

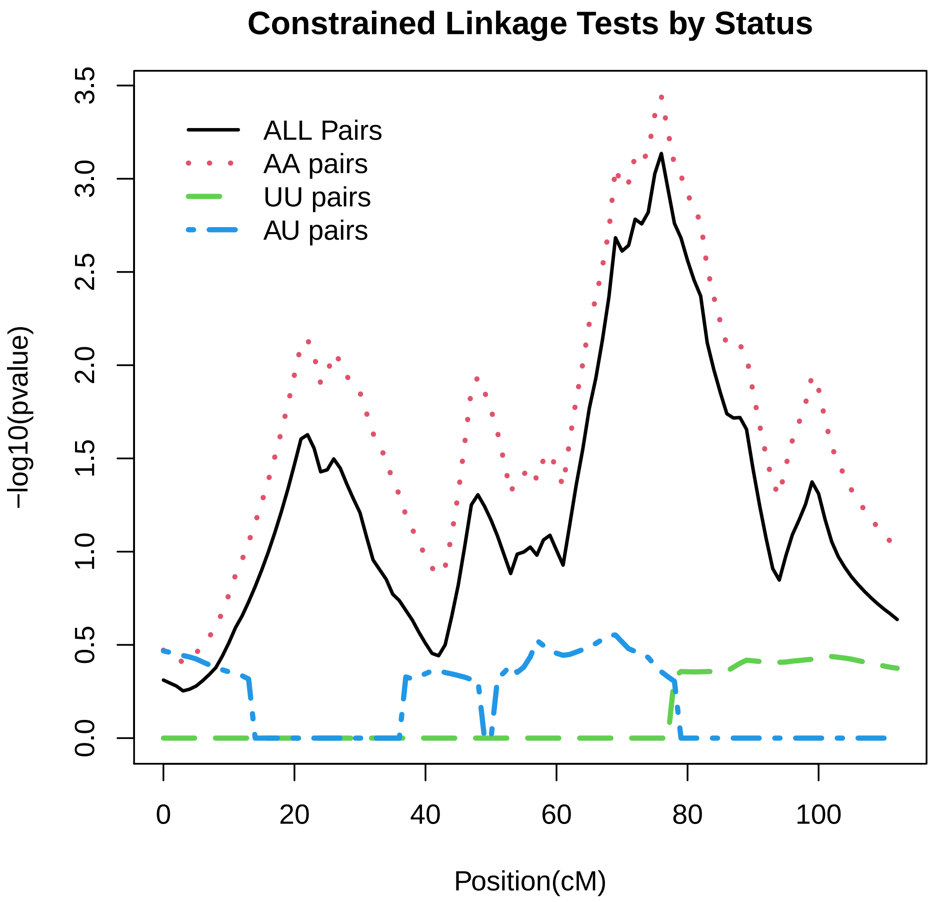

Figure 1: Linkage without covariates for all ALL, AA, UU, AU pairs

```
R> plot(sum.dxage.AA)
R>
```

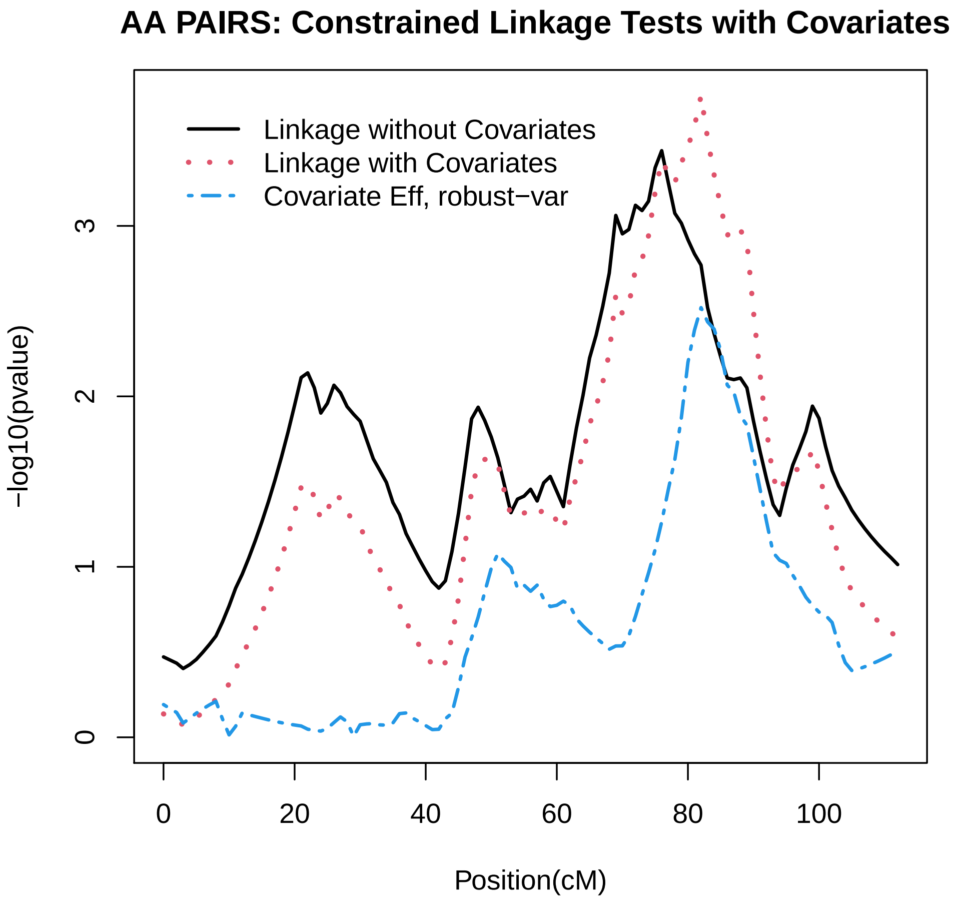

Figure 2: Plot of tests from pairSum(dxage) on AA pairs

```
R> ## to plot difference in constrained and unconsrained
R> plot.ibdreg.unexpect(nocov.ALL, status.method="AA")
R>
```

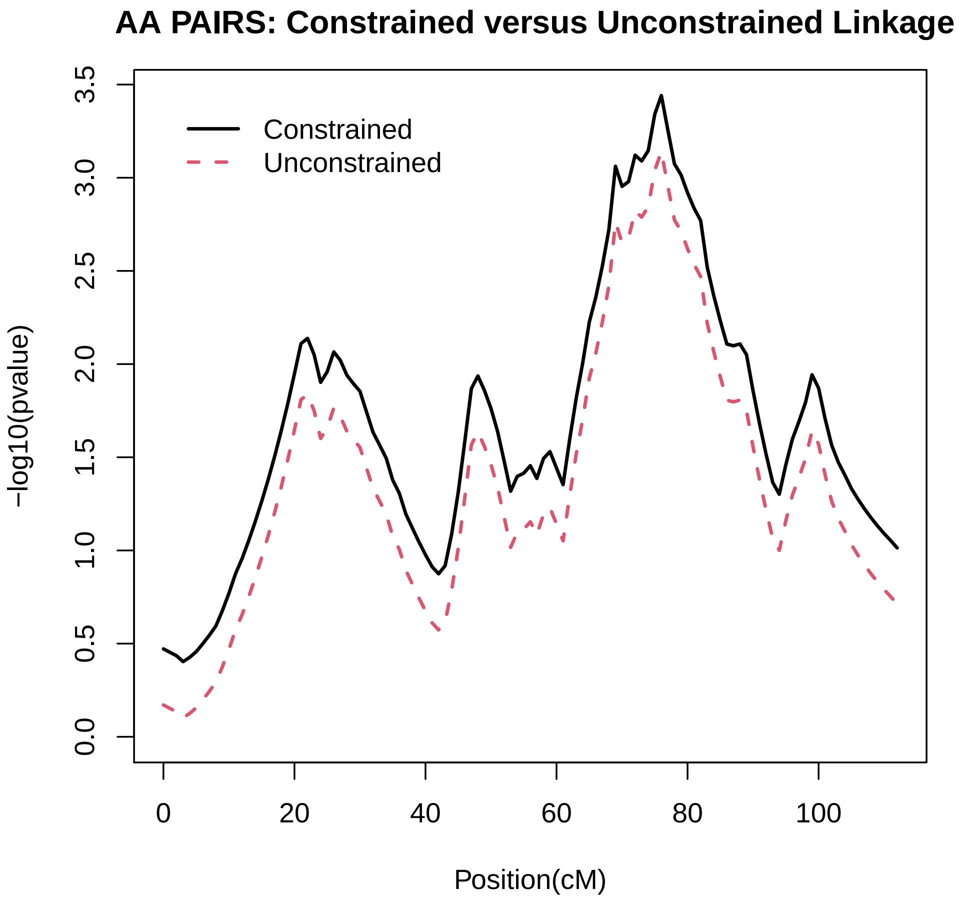

Figure 3: Plot of Unconstrained versus Constrained Linkage