CrypticIBDcheck : An R package for checking cryptic relatedness in nominally unrelated individuals

Annick Nembot-Simo
Simon Fraser University

Jinko Graham
Simon Fraser University

Brad McNeney
Simon Fraser University

Abstract

In population association studies, standard methods of statistical inference assume that study subjects are independent samples. In genetic association studies, it is therefore of interest to diagnose undocumented close relationships in nominally unrelated study samples. We describe the R package **CrypticIBDcheck** to identify pairs of closely-related subjects based on genetic marker data from single-nucleotide polymorphisms (SNPs). The package is able to accommodate SNPs in linkage disequibrium (LD), without the need to thin the markers so that they are approximately independent in the population. Sample pairs are identified by superposing their estimated identity-by-descent (IBD) coefficients on plots of IBD coefficients for pairs of simulated subjects from one of several common close relationships. The methods are particularly relevant to candidate-gene association studies, in which dependent SNPs cluster in a relatively small number of genes spread throughout the genome. The accommodation of LD allows the use of all available genetic data, a desirable property when working with a modest number of dependent SNPs within candidate genes. **CrypticIBDcheck** is available from the Comprehensive R Archive Network (CRAN).

Keywords: cryptic relatedness, IBD estimation, linkage disequilibrium, gene drop simulation.

1. Introduction

It is well known that the results of genetic association studies may be confounded by the presence of undocumented relationships – a phenomenon referred to as cryptic relatedness (e.g., Devlin and Roeder 1999; Voight and Pritchard 2005). Before making any inference with the data, it is therefore important to understand cryptic relatedness in the study sample. To facilitate this understanding, we introduce **CrypticIBDcheck**, an R package for exploring the presence of close relationships in a homogeneous sample of nominally unrelated individuals. Although several methods for exploring cryptic relatedness have been implemented (reviewed below), none are geared for data from candidate-gene association studies. **CrypticIBDcheck** fills this need. For ease of interpretation, the package implements exploratory displays based on popular measures of gene-identity by descent. However, a unique feature of these displays is that they accommodate population linkage disequilibrium (LD) amongst genetic markers. The accommodation of LD allows the use of data on all available markers rather than on a subset whose alleles are approximately independent in the population. This feature is attractive in candidate-gene association studies, where markers within genes are in LD but the number of genes is too small to select an independent subset of markers that is informative

for relationship.

The relatedness between two individuals may be defined in terms of the proportion of loci at which they share zero, one or two alleles that are identical-by-descent (IBD). We refer to these proportions as the actual IBD-sharing coefficients, or IBD coefficients. The alleles from two individuals are IBD if they are descended from a common ancestor in a given reference population (e.g., Weir, Anderson, and Hepler 2006). Though alleles from each of two individuals may match or be identical-by-state (IBS), they are not necessarily IBD.

CrypticIBDcheck uses estimated IBD coefficients to summarize possible relationships among pairs of study subjects. The approach is exploratory and graphically-based, similar to the GRR approach of Abecasis, Cherny, Cookson, and Cardon (2001) and the approach of Gogarten, Laurie, Bhangale, Conomos, Laurie, McHugh, Painter, Zheng, Shen, and Swarnkar (2012) implemented in the `ibdPlot()` function of the **GWASTools** Bioconductor package.

GRR calculates and displays the mean and variance of IBS allele sharing over polymorphic loci for each pair of individuals. Pairs of known relationships form reference clusters on the plot, allowing the user to identify errors in reported relationships. In association studies of nominally unrelated individuals, however, there are no reference clusters available. In principle, reference clusters could be obtained theoretically from the joint distribution of the IBS mean and variance estimators, but it is unclear how to derive this distribution in the presence of LD.

The `ibdPlot()` function in **GWASTools** may be applied to view estimated IBD coefficients along with reference clusters for the unobserved, true IBD coefficients based on theoretical moments of their distribution (Hill and Weir 2011). However, in candidate-gene studies with a modest number of SNPs, errors introduced by estimation of IBD coefficients cannot be ignored. Hence, reference distributions for the true IBD coefficients do not adequately represent those for estimated IBD coefficents.

The idea behind **CrypticIBDcheck** is to identify closely-related study pairs by displaying their estimated IBD coefficients together with those from *simulated pairs* of known relationships. The simulated reference pairs provide an empirical joint distribution of the IBD estimators under selected relationships which, in turn, suggest possible relationships amongst study pairs. Working with simulated pairs from known relationships avoids having to derive the joint distribution of the IBD estimators when the genetic markers are in LD. Simulated pairs are obtained by gene drop on a relationship-specific pedigree, with pedigree-founder haplotypes drawn from a fitted haplotype model that accounts for LD (Thomas 2010). We have implemented simulation of the following common relationships: monozygotic twins/sample duplicates, parent-offspring, full siblings, second degree (i.e., half siblings, avuncular or grandparent-grandchild) and first cousins. However, users may also specify their own custom relationships (see Section 4.2).

The paper is structured as follows. In Section 2 we describe the IBD estimators, and methods for gene drop simulation in the presence of LD to obtain reference clusters. Section 3 discusses implementation details. In Section 4, two examples showing how to use the package are provided. Finally, Section 5 includes a discussion and ideas for future work.

2. Methods

2.1. IBD estimation

There are two common approaches to estimating IBD coefficients: maximum likelihood (Thompson 1975; Milligan 2003; Choi, Wijsman, and Weir 2009) and the method of moments (Ritland 1996; Lynch and Ritland 1999; Purcell, Neale, Todd-Brown, Thomas, Ferreira, Bender, Maller, Sklar, de Bakker, Daly, and Sham 2007). Typically, maximum-likelihood estimators (MLEs) are more biased than method-of-moments estimators (MMEs), especially when the number of loci is small; they are also more computationally expensive (Lynch and Ritland 1999). However, MMEs are less precise than MLEs and can fall outside the biologically meaningful parameter space (Milligan 2003).

In this section, we review a popular method of moments approach to estimating IBD coefficients introduced by Purcell *et al.* (2007) and implemented in PLINK. This approach assumes that the individuals are from the same homogeneous, random-mating and non-inbred population. Alleles from two individuals are considered to be IBD if they are descended from a common ancestor in some base population that we take to be relatively recent. All alleles in this base population are defined to be non-IBD. Given a SNP with alleles A and a, a pair of individuals that are, say, AA and aa, respectively, will be denoted (AA, aa).

Identity-by-state (IBS) for a pair of subjects is denoted by the random variable I and identity-by-descent by the random variable Z, with possible states being 0, 1, and 2 for both random variables. The IBD coefficients to be estimated are the proportions of genome shared IBD, denoted by $P(Z = 0)$, $P(Z = 1)$, and $P(Z = 2)$. For a given SNP m, the procedure begins by expressing the prior probability of IBS sharing as

$$P_m(I = i) = \sum_{z=0}^{i} P_m(I = i|Z = z)P(Z = z). \tag{1}$$

$P(Z = z)$ and $P_m(I = i)$ are specific to the pair of subjects being considered, while the conditional SNP-specific IBS probabilities $P_m(I = i|Z = z)$ apply to all pairs. For a given pair of individuals at a given SNP, the above equation specifies three identities for the IBS states 0, 1, and 2. These three identities are summed over SNPs and then rearranged to express $P(Z = 0)$, $P(Z = 1)$, and $P(Z = 2)$ for the pair in terms of marginal and conditional IBS probabilities. For example, in the case of $i = z = 0$, we obtain

$$P(Z = 0) = \sum_m P_m(I = 0) / \sum_m P_m(I = 0|Z = 0).$$

The method-of-moments estimators of IBD coefficients for a given pair are obtained by substituting estimators of the conditional SNP-specific IBS probabilities, $P_m(I = i|Z = z)$, pertaining to any pair and the pair's marginal SNP-specific IBS probabilities, $P_m(I = i)$, into the identities and then solving for $P(Z = i)$.

The marginal SNP-specific IBS probabilities, $P_m(I = i)$ for a pair of subjects may be estimated by the indicator function for whether the pair has $I = i$ at the SNP. An unbiased estimator

of $\sum_m \hat{P}_m(I = i)$ is therefore the count of SNPs at which the pair shares i alleles IBS. Estimates of the SNP-specific conditional IBS probabilities, $P_m(I = i|Z = z)$, are based on data from all subjects in the sample. Derivation of unbiased estimators of $P_m(I = i|Z = z)$ is more involved. To simplify notation, we temporarily drop the SNP subscript m. If p and $q = 1 - p$ denote the frequencies of A and a in the base population, then $P(I = i|Z = z)$ is a function of p and q. For example, two individuals share 0 alleles IBS if they are either (AA, aa) or (aa, AA). Given that $Z = 0$, the probabilities of these genotypes are p^2q^2 and q^2p^2, respectively, leading to $P(I = 0|Z = 0) = 2p^2q^2$. The plug-in estimators of conditional IBS probabilities, such as $P(I = 0|Z = 0)$, obtained by inserting estimators $\hat{p}$ and $\hat{q}$ are biased (Appendix A). Unbiased estimators, expressed as the plug-in estimator multiplied by a correction factor, may be derived as described next.

Let X and Y be the counts of the alleles A and a, respectively, so that the allele frequency estimators are $\hat{p} = X/T$ and $\hat{q} = Y/T$, where T is twice the number of observed genotypes in the population random sample. The estimators of the conditional IBS probabilities $P(I = i|Z = z)$ may be motivated by the following model. The genotype of each individual in the present population is obtained from two independent draws from an infinite base population of alleles. Consequently, the T alleles of a population random sample of study subjects can be viewed as a random sample from the base population. Moreover, conditional on IBD status, any pair of individuals in the present population can be viewed as independent allelic draws from the base population, with the number of draws determined by their IBD status.

For example, in the case of $Z = 0$, a random pair of individuals results from randomly drawing two pairs of alleles from the base population. An indicator variable of whether this sampling process results in $I = 0$ is an unbiased estimator of $P(I = 0|Z = 0)$. An unbiased estimator is therefore the average of these indicator variables over all possible draws from the T alleles on which we have data; i.e., the proportion of pairs of allelic pairs with $I = 0$. The proportion can be computed as follows. The number of ways of selecting four distinct alleles from a total of T is $T(T-1)(T-2)(T-3)$. Without loss of generality, suppose the first two alleles are assigned to the first individual in a pair and the last two alleles to the second individual. Then the number of pairs that are (AA, aa) and (aa, AA) are $X(X-1)Y(Y-1)$ and $Y(Y-1)X(X-1)$, respectively. Hence,

$$\hat{P}(I = 0|Z = 0) = \frac{2X(X-1)Y(Y-1)}{T(T-1)(T-2)(T-3)}, \tag{2}$$

is an unbiased estimator of $P(I = 0|Z = 0)$ (see Appendix A for verification by direct computation). After algebra, the unbiased estimator may be expressed in terms of the allele frequency estimators and a correction factor as:

$$\hat{P}(I = 0|Z = 0) = 2\hat{p}^2\hat{q}^2 \left(\frac{X-1}{X} \times \frac{Y-1}{Y} \times \frac{T}{T-1} \times \frac{T}{T-2} \times \frac{T}{T-3} \right).$$

For $Z = 1$, we consider a pair of individuals to be the result of drawing three alleles from the base population, one of which is shared by the pair of individuals. The proportion of such pairs of individuals with IBS state $I = 1$ in our data is an unbiased estimator of $P(I = 1|Z = 1)$. The number of ways to select three distinct alleles from a total of T is $T(T-1)(T-2)$. Among

these, the genotype pairs that are $I = 1$ are the $X(X-1)Y$, $YX(X-1)$, $Y(Y-1)X$, and $XY(Y-1)$ that are (AA, Aa), (Aa, AA), (aa, Aa), and (Aa, aa), respectively. Thus,

$$\begin{aligned}
\hat{P}(I=1|Z=1) &= \frac{2X(X-1)Y + 2XY(Y-1)}{T(T-1)(T-2)} \\
&= \frac{2XYX}{TT^2}\left(\frac{X-1}{X}\times\frac{T}{T-1}\times\frac{T}{T-2}\right) + \frac{2XY^2}{TT^2}\left(\frac{X-1}{X}\times\frac{T}{T-1}\times\frac{T}{T-2}\right) \\
&= 2\hat{p}^2\hat{q}\left(\frac{X-1}{X}\times\frac{T}{T-1}\times\frac{T}{T-2}\right) + 2\hat{p}\hat{q}^2\left(\frac{X-1}{X}\times\frac{T}{T-1}\times\frac{T}{T-2}\right).
\end{aligned}$$

The other conditional IBS probabilities are estimated in an analogous manner and their expressions are provided in Table 1 of Purcell *et al.* (2007).

With estimates $\hat{P}_m(I=i)$ and $\hat{P}_m(I=i|Z=z)$ for each SNP, we sum over SNPs to obtain estimates of the IBD coefficients for a given pair in the sample. Let

$$\begin{aligned}
\hat{N}(I=i|Z=z) &= \sum_{m=1}^{L}\hat{P}_m(I=i|Z=z) \text{ and} \\
\hat{N}(I=i) &= \sum_{m=1}^{L}\hat{P}_m(I=i), \qquad (3)
\end{aligned}$$

where L is the total number of SNPs with genotype data on both individuals. For any pair of subjects, summing equation (1) over all the SNPs and using equation (3) gives the following method-of-moment estimators of the IBD coefficients:

$$\begin{aligned}
\hat{P}(Z=0) &= \frac{\hat{N}(I=0)}{\hat{N}(I=0|Z=0)} \\
\hat{P}(Z=1) &= \frac{\hat{N}(I=1) - \hat{P}(Z=0)\times\hat{N}(I=1|Z=0)}{\hat{N}(I=1|Z=1)} \\
\hat{P}(Z=2) &= \frac{\hat{N}(I=2) - \hat{P}(Z=0)\times\hat{N}(I=2|Z=0) - \hat{P}(Z=1)\times\hat{N}(I=2|Z=1)}{\hat{N}(I=2|Z=2)}.
\end{aligned}$$

Purcell *et al.* (2007) proposed adjustments to bound these estimators to values consistent with their interpretation as IBD proportions. We have not made these adjustments in our graphical displays.

2.2. Gene drop simulation with LD

The package provides a graphical display that can be used to identify related sample pairs by plotting the estimated IBD coefficients $\hat{P}(Z=1)$ versus $\hat{P}(Z=0)$. To assess the variability of these estimators the points of the IBD plot are superposed on reference clusters obtained from one of the following relationships: unrelated, monozygotic twins/duplicates, parent-offspring, full siblings, half siblings and first cousins. These reference clusters are obtained by gene drop simulation that accounts for LD (Thomas 2009b). A strength of this approach is that we do not need to assume independence of marker loci. In candidate-gene association studies, this feature is important because of the dependence among a relatively small number of SNPs.

Ignoring the dependence among SNPs within genes produces reference clusters that are too tight relative to the true variability, and can lead to false-positive results. We return to this point in the examples.

A graphical model is an approach to modeling the joint distribution of a set of dependent random variables when many independences or conditional independences exist between subsets of the variables. In the case of LD, it is expected that the joint distribution of alleles allong haplotypes shows such a structure. Thomas (2009a) describes a flexible graphical model of haplotype frequencies that captures LD between loci. The model is fit to data from subjects that can be regarded as a population random sample; e.g., the controls in a case-control study of a rare disease. Model parameters are estimated by use of a stochastic optimization algorithm (Thomas 2009b).

Once the LD model is fit, it is used to sample haplotypes for the founders of a pedigree. Data on the remaining members of the pedigree are simulated by gene drop. Gene drop is a method for randomly generating the genotypes of related individuals in a pedigree. Alleles are "dropped" from the founders through the pedigree according to Mendel's laws. Multi-locus gene drop incorporates the process of recombination. To illustrate the simulation procedure, consider a parent-offspring relationship. A pedigree that encompasses this relationship is one comprised of two parents and the offspring. The founders are the parents. Parental haplotypes are simulated from the fitted LD model and are then dropped to the offspring. To mimic real data with missing genotypes, selected genotypes for a simulated individual are set to missing according to the missing genotype pattern of a randomly-sampled study subject.

Programs for fitting LD models and performing gene drop simulations are available in the Java Programs for Statistical Genetics and Computational Statistics (JPSGSC) library developed by Alun Thomas (`http://balance.med.utah.edu/wiki/index.php/JPSGCS`). We use the R package **rJPSGCS** (Blay, Graham, McNeney, and Nembot-Simo 2011) to access these programs from R.

3. Implementation

The main function in **CrypticIBDcheck** is `IBDcheck()`, which estimates IBD coefficients for pairs of study subjects and optionally for simulated pairs of subjects and returns an object of class `IBD`. The plot method for the `IBD` class displays the IBD coefficients for pairs of study subjects, along with prediction ellipses for known relationship pairs.

The arguments of `IBDcheck()` are constructed by the functions `new.IBD()`, `filter.control()` and `sim.control()`. The function `new.IBD` produces an object of class `IBD` suitable for input to `IBDcheck()`. At a minimum, such an object includes the genetic data as a `snp.matrix` object from the **chopsticks** package (Leung 2011), a data frame of SNP information that includes chromosome and physical map positions of each SNP, and a data frame of subject information that includes a logical vector indicating whether (`TRUE`) or not (`FALSE`) each subject is to be used to estimate the conditional IBS probabilities and fit the LD model. The documentation for `new.IBD()` and the examples below provide further details. The function `filter.control()` sets options for quality control filtering of data by SNPs and by subjects,

while the function `sim.control()` sets options that control simulation of subjects by gene drop. The respective help files and the examples below provide further details. As the fitting of LD models in `IBDcheck()` can be computationally demanding, users have the option of splitting computations across a **snow** cluster (Tierney, Rossini, Li, and Sevcikova 2011), as described in Appendix B. The output of `IBDcheck()` is an object of class `IBD`, which includes the estimated IBD coefficients for pairs of study subjects and for simulated pairs with known relationship.

`IBD` objects are graphically displayed by the plot method of the class; the documentation for this method is available through `help("plot.IBD")`. Plots are of $\hat{P}(Z = 1)$ versus $\hat{P}(Z = 0)$ for pairs of study subjects, with prediction ellipses for known relationships superposed, if requested by the user. The prediction ellipses are produced from estimated IBD coefficients for a user-specified number (default 200) of simulated pairs of known relationships, assuming the distribution of estimated IBD coefficients is approximately bivariate Normal. The default setting for `IBDcheck()` is to omit simulated pairs from the object. When simulated pairs are omitted, plotting produces a single interactive display of estimated IBD coefficients for pairs of study subjects, on which points may be identified by clicking with the mouse. On the other hand, when the `IBD` object includes simulated pairs, the function returns a series of plots, which the user is prompted to view and interact with successively. The first plot to appear is non-clickable and shows the estimated IBD coefficients for all pairs of study subjects, along with the prediction ellipse for unrelated, simulated pairs. Subsequent plots are clickable and correspond to each relationship requested in the call to `IBDcheck()`. These relationship-specific plots are for identifying pairs of study subjects which could have the relationship. The plotting regions are restricted to the neighborhood of the prediction ellipse for the simulated pairs of that relationship, which is also drawn. If, however, the plotting region overlaps with the prediction ellipse for simulated unrelated pairs, the ellipse for simulated unrelated pairs is drawn as well. Points falling within the prediction ellipse for the relationship and outside the prediction ellipse for unrelated pairs are automatically flagged. In addition, users may click on points of study pairs that appear to be related but are not automatically flagged. The plot method produces a data frame of information on pairs that have been flagged on the different plots, either automatically or interactively by the user through clicking the mouse.

4. Examples

In this section we illustrate the features of the `CrypticIBDcheck` package using the genetic data `Nhlsim` that comes with the package. These data were simulated to mimic the characteristics of SNP genotypes in subjects of European ancestry from a candidate-gene, case-control study of non-Hodgkin Lymphoma (Schuetz, Daley, Graham, Berry, Gallagher, Connors, Gascoyne, Spinelli, and Brooks-Wilson 2012). The data set is a list comprised of (i) a `snp.matrix` object called `snp.data` with genotypes for 108 controls and 100 cases; (ii) a vector `chromosome` of chromosome numbers for each SNP; (iii) a vector `physmap` of physical map positions of each SNP, from build 36 of the human genome; and (iv) a binary vector `csct` with value one for cases and zero for population controls. The binary vector `csct` is used to select controls for fitting LD models and estimating conditional IBS probabilities. All of the information in `Nhlsim` is required to run `IBDcheck()`.

We present two examples. In the first (Section 4.1), we illustrate basic use of `IBDcheck()`

to fit LD models and do gene drop simulations. Once the user requests simulations, there are a number of parameters, such as the types of relationships to simulate, that control the simulations. Each simulation parameter has a default value, as described in the help file for `sim.control()`. In the analysis presented in Section 4.1 we use these default settings. In the second example (Section 4.2), we illustrate re-use of fitted LD models to perform additional gene drop simulations, this time for a user-specified relationship. For examples of how to use `IBDcheck()` to explore genome-wide data, we refer readers to the package vignette `IBDcheck-hapmap` that illustrates an analysis of genome-wide data from HapMap, using thinning of markers to reduce the computational burden.

4.1. Default analysis with LD model fitting and gene drops

We first load the package and the `Nhlsim` data set.

```
R> library(CrypticIBDcheck)

Loading required package: rJPSGCS
Loading required package: rJava
Loading required package: chopsticks
Loading required package: survival
Loading required package: splines
Loading required package: car
Loading required package: MASS
Loading required package: nnet
Loading required package: ellipse

Attaching package: `ellipse'

The following object(s) are masked from `package:car':

    ellipse

R> data(Nhlsim)
```

Next we create an object of class `IBD` that can be used as input to the `IBDcheck()` function. The `Nhlsim` data does not include genetic map positions for the SNPs, so these will be inferred from the physical positions, assuming the physical positions are from build 36 of the human genome. We use subjects with case-control status 0 (controls) for estimating conditional IBS probabilities and fitting LD models.

```
R> popsam<-Nhlsim$csct==0
R> dat<-new.IBD(Nhlsim$snp.data,Nhlsim$chromosome,Nhlsim$physmap,popsam)

Note: Input does not include genetic map locations (Gen_loc).
Inferring genetic map from physical position (Position),
assuming build 36 of the human genome.
```

```
Note: Using population sample subjects (popsam==TRUE) to fill in pvalues from tests of HWE

Note: Input does not include subject ids (subids). Using rownames of snp.data.
```

In this illustration, we leave all QC filtering options (set by `filter.control()`) at their default values. We use `sim.control()` to modify the default value of `simulate=FALSE` to `simulate=TRUE`, so that reference clusters are simulated.

```
R> ss<-sim.control(simulate=TRUE)
R> cibd<-IBDcheck(dat,simparams=ss)
```

This call to `IBDcheck()` will generate 22 plain-text files in the user's working directory that contain details of the fitted LD models for each of the 22 autosomal chromosomes. The names of these files are stored in the output `IBD` object:

```
R> cibd$simparams$LDfiles
```

```
 [1] "GD1.ld.par"  "GD2.ld.par"  "GD3.ld.par"  "GD4.ld.par"  "GD5.ld.par"
 [6] "GD6.ld.par"  "GD7.ld.par"  "GD8.ld.par"  "GD9.ld.par"  "GD10.ld.par"
[11] "GD11.ld.par" "GD12.ld.par" "GD13.ld.par" "GD14.ld.par" "GD15.ld.par"
[16] "GD16.ld.par" "GD17.ld.par" "GD18.ld.par" "GD19.ld.par" "GD20.ld.par"
[21] "GD21.ld.par" "GD22.ld.par"
```

Section 4.2 gives an example of how to re-use these fitted LD models for performing additional gene drops. The output includes estimated IBD coefficients for pairs of subjects in the input data and for simulated pairs of subjects from the following relationships: unrelated, duplicates/MZ twins, parent-offspring, full siblings and half-siblings. Simulation of first-cousin or user-specified relationship pairs is also possible, but is not done by default. First cousins are typically not distinguishable from unrelated pairs with data from a candidate-gene association study. The estimated IBD coefficients can be plotted with the plot method for the `IBD` class.

```
R> ibdpairs=plot(cibd)
R> ibdpairs
```

```
  member1 member2       pz0       pz1     relationship
1    sub1  sub201 0.0000000 1.0170209 parent-offspring
2    sub2  sub202 0.0000000 1.0049532 parent-offspring
3  sub203  sub206 0.1941393 0.5161277        full sibs
4  sub204  sub207 0.2294063 0.5217202        full sibs
5  sub205  sub208 0.2057583 0.4434503        full sibs
6   sub31   sub44 0.5264858 0.4505440        half sibs
```

In this example, the plotting function produces five plots, shown in Figures 1–3, and an output data frame `ibdpairs` that contains information on study pairs flagged with the last four plots in Figures 2 and 3.

Figure 1 shows the non-clickable plot of the estimated IBD coefficients, $P(Z = 1)$ versus $P(Z = 0)$, for all pairs of study subjects, with the prediction ellipse for unrelated pairs superposed. The level of the prediction ellipse is left at its default value of `ellipse.coverage=0.95`

and, for unrelated pairs, is adjusted to account for the majority of study pairs being unrelated. Specifically, a Bonferroni-type adjustment, $1-(1-$`ellipse.coverage`$)/n_p$, is applied, where n_p is the number of pairs of study subjects. One purpose of the prediction ellipse is to avoid confusing the display by adding points for simulated pairs. Another purpose is to avoid having to manually click points for study pairs that appear within a cloud of points from simulated pairs. We adopted a bivariate normal approximation to the prediction ellipse because it correctly identified the majority of points in experiments with simulated data (e.g. Figure 1). However, in Figure 1, several unrelated pairs appear outside the prediction ellipse, indicating that the distribution of estimated IBD coefficients is slightly heavier-tailed than the bivariate normal approximation.

For the four other plots, shown in Figures 2 and 3, points that lie within a 95% prediction ellipse (the default level for `ellipse.coverage`) for the given relationship *and* outside the prediction ellipse for unrelated pairs are automatically flagged. In addition, these plots are clickable, and points flagged manually are added to the output dataframe. For example, on the plot for half-siblings, the point corresponding to the pair `sub35` and `sub95` has been manually flagged (Figure 3, right panel); this pair appears in both the prediction ellipse for unrelated pairs and the upper portion of the prediction ellipse for half siblings. Manually clicking on the point for this pair adds the following row to the output dataframe `ibdpairs`:

```
7   sub35   sub95 0.4568146 0.6351182        half sibs
```

In this data set, there are no duplicate/MZ twins pairs and no pairs flagged as such (Figure 2, left panel). The two parent-offspring pairs in the `Nhlsim` data fall in the prediction ellipse for parent-offspring pairs (Figure 2, right panel). Similarly, all three full-sibling pairs in the `Nhlsim` data fall in the prediction ellipse for full siblings (Figure 3, left panel). The substantial overlap of the prediction ellipses for half siblings and unrelated pairs (Figure 3, right panel) indicates insufficient data to distinguish these two relationships. Though there are no half-sibling pairs in `Nhlsim`, one pair of unrelated subjects, `sub31` and `sub44`, has atypical estimated IBD coefficients that fall within the prediction ellipse for half siblings but outside the prediction ellipse for unrelated pairs.

The unrelated pair flagged as a potential half-sibling pair is a false-positive result. We observed (results not shown) that the number of false-positive related pairs is greatly increased if we fail to take the LD between SNPs into account. Specifically, if we repeat the simulation of unrelated and half-sibling pairs of subjects assuming independent SNPs (`fitLD=FALSE`), we obtain 16 false-positive half sibling pairs. These observations highlight that naïvely ignoring the dependence among SNPs produces reference clusters that are too tight relative to the true variability.

4.2. Additional gene drops using previously-fit LD models

By far the most computationally-demanding step of `IBDcheck()` is the fitting of LD models. The fitted LD models are stored in plain-text files in the working directory and can be re-used for future gene drops using the argument `LDfiles` of `sim.control()`, as we now illustrate. We also demonstrate how users can create their own relationships to use as reference clusters on the IBD plot.

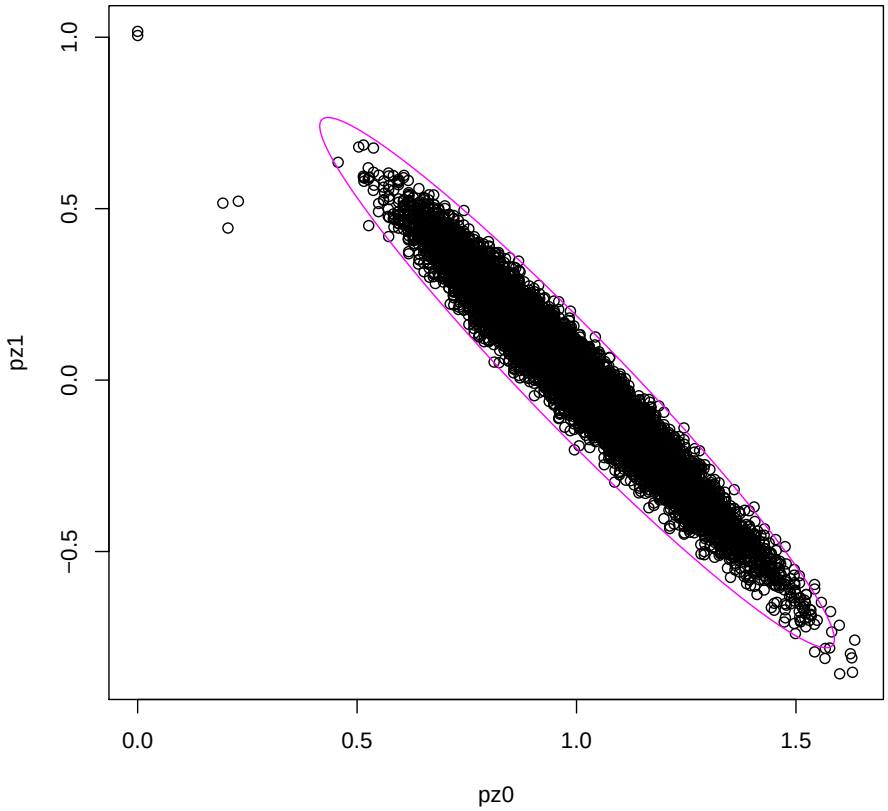

Figure 1: Estimated IBD coefficients for all pairs of study subjects, with the prediction ellipse for unrelated pairs superposed.

Setting of simulation parameters, such as the names of fitted LD model files and specification of the relationships to simulate, is done with the `sim.control()` function. Recall that the names of the LD files are stored in the `IBD` object created by a call to `IBDcheck()`; for example:

```
R> cibd$simparams$LDfiles
```

```
 [1] "GD1.ld.par"  "GD2.ld.par"  "GD3.ld.par"  "GD4.ld.par"  "GD5.ld.par"
 [6] "GD6.ld.par"  "GD7.ld.par"  "GD8.ld.par"  "GD9.ld.par"  "GD10.ld.par"
[11] "GD11.ld.par" "GD12.ld.par" "GD13.ld.par" "GD14.ld.par" "GD15.ld.par"
[16] "GD16.ld.par" "GD17.ld.par" "GD18.ld.par" "GD19.ld.par" "GD20.ld.par"
[21] "GD21.ld.par" "GD22.ld.par"
```

These fitted models are re-used by specifying their names as the argument `LDfiles` to `sim.contol`:

```
R> ss<-sim.control(simulate=TRUE, LDfiles=cibd$simparams$LDfiles)
```

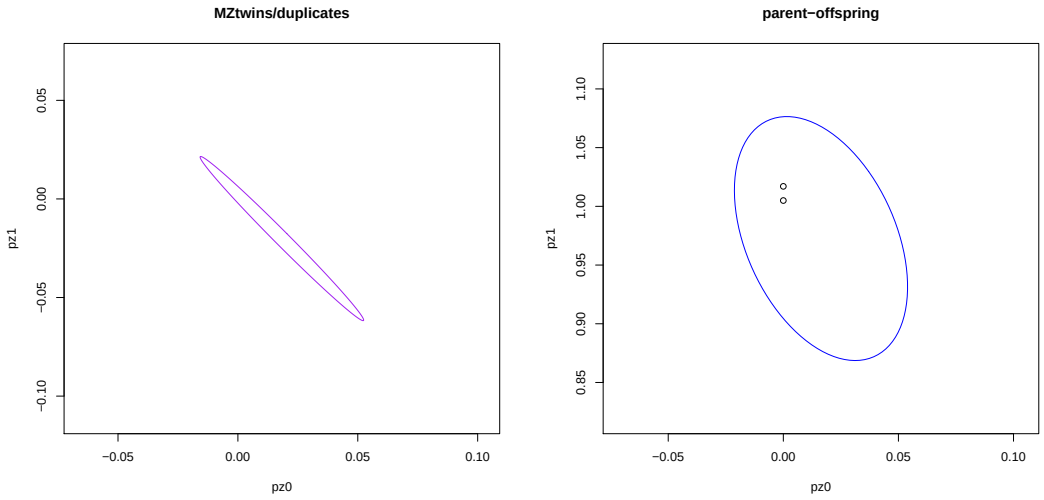

Figure 2: Observed pairs with prediction ellipses for MZ twins/duplicates (left panel) and parent-offspring pairs (right panel) superposed.

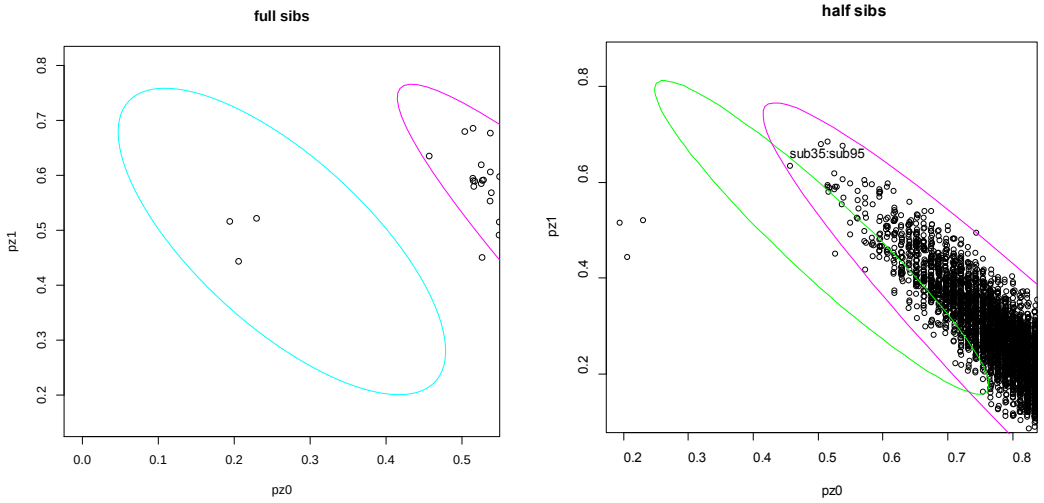

Figure 3: Observed pairs with prediction ellipses for full-siblings pairs (left panel) and half-sibling pairs (right panel) superposed. The magenta ellipse for unrelated subjects appears on each panel.

The `sim.control()` function can also be used to specify the relationships to simulate; e.g., one can obtain simulated cousin pairs with

```
R> ss<-sim.control(simulate=TRUE, rship="cousin",
                   LDfiles=cibd$simparams$LDfiles)
```

It is also possible to obtain pairs simulated according to a user-specified relationship. In the following, the relationship of interest is parent-offspring with first cousins parents. The relationship is depicted in Figure 4, which was drawn using Pedfiddler (Loredo-Osti and

Morgan 2010). To simulate according to this relationship, it is necessary to specify a minimal

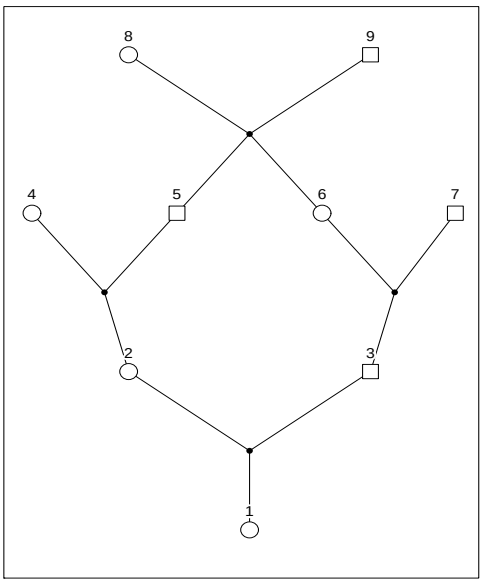

Figure 4: Pedigree for an offspring of a first-cousin marriage. Circles represent females, squares represent males. Lines of descent are indicated by connections between nodes. The mother and daughter of interest are labelled as 2 and 1, respectively.

pedigree that captures the relationship between the mother and daughter *and* to have the mother and daughter be the first two members of the pedigree. The pedigree drawn in Figure 4 has parents (nodes 2 and 3) that are first cousins. Pedigree information is specified in a data frame whose rows describe subjects. The columns of the data frame are member IDs, the IDs of each member's father and mother, and gender, coded as 1 for male and 2 for female. For pedigree founders, the father and mother IDs are set to zero. Specification of the pedigree in Figure 4 is as follows:

```
userdat<-data.frame(ids=1:9,
                    dadids=c(3,5,7,0,9,9,0,0,0),
                    momids=c(2,4,6,0,8,8,0,0,0),
                    gender=c(2,2,1,2,1,2,1,2,1))
```

The call to `IBDcheck()` would then be:

```
ss<-sim.control(simulate=TRUE,
                rships="user", userdat=userdat,
                LDfiles=cibd$simparams$LDfiles)
ff<-filter.control(filter=FALSE) # no need to re-filter data
cibd.user<-IBDcheck(cibd,simparams=ss,filterparams=ff)
```

On the plot of `cibd.user` (not shown) the prediction ellipse for simulated mother-daughter pairs where the daughter is inbred is very similar to that from simulated pairs where the daughter is not inbred (Figure 2, right panel). However, relative to the non-inbred case, the prediction ellipse in the inbred case is shifted slightly downward on the plot, reflecting the

fact that the probability of 2 genes IBD is now non-zero and the probability of 1 gene IBD is therefore smaller.

5. Summary

CrypticIBDcheck is an R package for exploring cryptic relatedness in a homogeneous sample of nominally unrelated individuals. The main function of the package, `IBDcheck()`, computes estimates of IBD coefficients for pairs of study subjects and, optionally, for pairs of subjects simulated to have one of several known relationships. Simulated data for a given relationship are obtained by gene drop simulation on a pedigree that captures the relationship, with founder haplotypes simulated according to an LD model fit to the data. Objects of class `IBD` returned by `IBDcheck()` are displayed by the plot method of the class. Pairs of study subjects whose estimated IBD coefficients are consistent with one of the relationships requested in the call to `IBDcheck()` are flagged, either automatically or interactively by user mouse-clicks, and returned in a data frame.

The methods implemented in **CrypticIBDcheck** are geared specifically towards exploring cryptic relatedness with data from candidate-gene association studies. These studies involve a relatively modest number of SNPs which are correlated because they are clustered within candidate genes. With a modest number of SNPs, the variability in the estimator of IBD coefficients cannot be ignored. Hence, reference distributions for true IBD coefficients do not adequately represent those for estimated IBD coefficients. In addition, thinning to an approximately independent and yet informative set of SNPs is not an option. Nor is ignoring LD and assuming SNPs are approximately independent. As illustrated in the Examples, ignoring LD leads to reference clusters that are too tight.

Fitting LD models can be computationally-demanding for data sets with more than a few thousand SNPs. To minimize the time spent fitting LD models, `IBDcheck()` can distribute fitting of LD models across a **snow** cluster, and LD files from one call to `IBDcheck()` can be re-used in future calls to add simulated pairs from a known relationship to an `IBD` object. We offer the user complete flexibility with respect to the type of relationships and number of pairs of each relationship to be simulated. Users can choose from a number of close relationships built-in to `IBDcheck()`, or specify their own relationships, as illustrated in Section 4.2.

Acknowledgements

We thank John Spinelli for access to the non-Hodgkin lymphoma study data. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and by the Mathematics of Information Technology and Complex Systems (Mitacs), Canadian Networks of Centres of Excellence.

References

Abecasis G, Cherny S, Cookson W, Cardon L (2001). "GRR: graphical representation of relationship errors." *Bioinformatics*, **17**(8), 742–743.

Blay S, Graham J, McNeney B, Nembot-Simo A (2011). *rJPSGCS: R-interface to gene drop Java Programs for Statistical Genetics and Computational Statistics (JPSGCS)*. R package version 0.2-5.

Choi Y, Wijsman E, Weir B (2009). "Case-control association testing in the presence of unknown relationships." *Genetic Epidemiology*, **33**, 668–678.

Devlin B, Roeder K (1999). "Genomic control for association studies." *Biometrics*, **55**, 997–1004.

Gogarten SM, Laurie C, Bhangale T, Conomos MP, Laurie C, McHugh C, Painter I, Zheng X, Shen J, Swarnkar R (2012). *GWASTools: Tools for Genome Wide Association Studies*. R package version 1.2.0.

Hill WG, Weir BS (2011). "Variation in actual relationship as a consequence of Mendelian sampling and linkage." *Genet Res (Camb)*, **93**(1), 47–64.

Leung HT (2011). *chopsticks: The snp.matrix and X.snp.matrix classes*. R package version 1.18.3, URL http://outmodedbonsai.sourceforge.net/.

Loredo-Osti J, Morgan K (2010). *Pedfiddler: A set of programs to manipulate pedigree graphs*. Version 0.5, URL http://www.stat.washington.edu/thompson/Genepi/Pedfiddler.shtml.

Lynch M, Ritland K (1999). "Estimation of pairwise relatedness with molecular markers." *Genetics*, **152**, 1753–1766.

Milligan B (2003). "Maximum-likelihood estimation of relatedness." *Genetics*, **163**, 1153–1167.

Purcell S, Neale B, Todd-Brown K, Thomas L, Ferreira M, Bender D, Maller J, Sklar P, de Bakker P, Daly M, Sham P (2007). "PLINK: a tool set for whole-genome association and population-based linkage analyses." *The American Journal of Human Genetics*, **81**, 559–575.

Ritland K (1996). "Estimators for pairwise relatedness and individual inbreeding coefficients." *Genetical Research*, **67**, 175–185.

Schuetz JM, Daley D, Graham J, Berry BR, Gallagher RP, Connors JM, Gascoyne RD, Spinelli JJ, Brooks-Wilson AR (2012). "Genetic Variation in Cell Death Genes and Risk of Non-Hodgkin Lymphoma." *PLoS ONE*, **7**(2), e31560. doi:10.1371/journal.pone.0031560. URL http://dx.doi.org/10.1371%2Fjournal.pone.0031560.

Thomas A (2009a). "Estimation of graphical models whose conditional independence graphs are interval graphs and its application to modeling linkage disequilibrium." *Computational Statistics and Data Analysis*, **53**, 1818–1828.

Thomas A (2009b). "A method and program for estimating graphical models for linkage disequilibrium that scale linearly with the number of loci, and their application to gene drop simulation." *Bioinformatics*, **25**, 1287–1292.

Thomas A (2010). "Assessment of SNP streak statistics using gene drop simulation with linkage disequilibrium." *Genetic Epidemiology*, **34**, 119–124.

Thompson E (1975). "The estimation of pairwise relationships." *Annals of Human Genetics*, **39**, 173–188.

Tierney L, Rossini AJ, Li N, Sevcikova H (2011). *snow: Simple Network of Workstations*. R package version 0.3-8, URL http://CRAN.R-project.org/package=snow.

Voight B, Pritchard J (2005). "Confounding from cryptic relatedness in case-control association studies." *Plos Genetics*, **1**, e32.

Weir B, Anderson A, Hepler A (2006). "Genetic relatedness analysis: modern data and new challenges." *Nature Reviews Genetics*, **7**, 771–780.

Appendix A: Bias of conditional IBS estimators

We calculate the bias of the plug-in and unbiased estimators of $P(I = 0|Z = 0)$. Bias calculations for estimators of other conditional IBS probabilities are similar. Throughout, consider SNPs with alleles A and a, and let p be the population allele frequency of A. Let T be twice the number of observed genotypes for the SNP in the population random sample. Let X be the number of A alleles among the T sampled and $\hat{p} = X/T$.

Bias of the plug-in estimator

We first prove the bias of the estimator $2\hat{p}^2(1-\hat{p})^2$ of $P(I = 0|Z = 0)$. Since $X \sim$ Binomial (T, p).

$$\begin{aligned} E[\hat{P}(I=0|Z=0)] &= 2E[\hat{p}^2(1-\hat{p})^2] \\ &= \frac{2}{T^4}E(X^2(T-X)^2) \\ &= \frac{2}{T^4}[T^2E(X^2) - 2TE(X^3) + E(X^4)]. \end{aligned}$$

This calculation requires the second, third and fourth moments of the binomial distribution which can be obtained from the moment generating function:

$$\begin{aligned} E(X^2) &= Tp + T(T-1)p^2 \\ E(X^3) &= Tp + 3T(T-1)p^2 + T(T-1)(T-2)p^3 \\ E(X^4) &= Tp + 7T(T-1)p^2 + 6T(T-1)(T-2)p^3 + T(T-1)(T-2)(T-3)p^4. \end{aligned}$$

Thus,

$$\begin{aligned} E(2\hat{p}^2\hat{q}^2) = \frac{2}{T^4}[&Tp(T^2-2T+1) + p^2T(T-1)(T^2-6T+7) + \\ &p^3T(T-1)(T-2)(-2T+6) + T(T-1)(T-2)(T-3)p^4], \end{aligned}$$

which is not equal to $2p^2q^2$.

Unbiased estimator

We now show that the estimator of $P(I = 0|Z = 0)$. in equation (2) is unbiased. Let $W = X(X-1)Y(Y-1)$ so that $E[\hat{P}(I = 0|Z = 0)] = 2E(W)/[T(T-1)(T-2)(T-3)]$. Then

$$\begin{aligned}
E(W) &= E[(X^2 - X)(Y^2 - Y)] \\
&= E[(X^2 - X)((T - X)^2 - T + X)] \\
&= E[X^4 - 2TX^3 + X^2(T^2 + T - 1) - X(T^2 - T)] \\
&= E(X^4) - 2TE(X^3) + (T^2 + T - 1)E(X^2) - T(T-1)E(X) \\
&= p(T - T^3 + T^2 - 2T^2 + T^3 + T^2 - T) + p^2[7T(T-1) \\
&- 6T^2(T-1) + T(T-1)(T^2 + T - 1)] \\
&+ p^3[6T(T-1)(T-2) - 2T^2(T-1)(T-2)] + T(T-1)(T-2)(T-3)p^4 \\
&= p^2[T(T-1)(T^2 - 5T + 6) + pT(T-1)(T-2)(6 - 2T) + p^2T(T-1)(T-2)(T-3)].
\end{aligned}$$

Therefore,

$$\begin{aligned}
E[\hat{P}(I = 0|Z = 0)] &= \frac{2E(W)}{T(T-1)(T-2)(T-3)} \\
&= 2p^2\left[\frac{T^2 - 5T + 6}{(T-2)(T-3)} + 2p\frac{3-T}{T-3} + p^2\right] \\
&= 2p^2(1 - 2p + p^2) \\
&= 2p^2(1-p)^2 \\
&= 2p^2q^2.
\end{aligned}$$

Appendix B: Splitting computations over a snow cluster

When the simulation parameters `simulate=TRUE` and `fitLD=TRUE`, the function `IBDcheck()` can be computationally demanding for data sets with more than about 1000 SNPs. For example, analysis of the `Nhlsim` data presented in section 4.1 took about 40 minutes on an X86-based PC with a 2.4GHz, dual-core, Intel processor. Our package offers the option to use a **snow** cluster in order to split the fitting of LD models and gene drop simulations across different processors. In this appendix we distinguish between a **snow** cluster running in an interactive R session on a single computer, and a **snow** cluster running in batch mode on a compute cluster.

As an example of interactive use on a computer with, say, eight processors, one might create an eight-node socket cluster and use this cluster in a call to `IBDcheck()` as follows:

```
R> library(snow)
R> cl<-makeCluster(8,type="SOCK")
R> clusterEvalQ(cl,library("CrypticIBDcheck"))
R> ss<-sim.control(simulate=TRUE,cl=cl)
R> cibd.cl<-IBDcheck(dat,simparams=ss)
R> stopCluster(cl)
```

To produce the set of plots displayed in Figures 1–3:

```
R> plot(cibd.cl)
```

The steps required to run `IBDcheck()` in batch mode on a compute cluster will depend on the setup of the compute cluster. We describe the necessary steps for a compute cluster at our institution as an example. Our local cluster uses the Torque Portable Batch System (PBS; see the documentation at http://www.clusterresources.com/torquedocs21/usersmanual.shtml) for running and submitting jobs and the Maui scheduler for scheduling the submitted jobs. Job submission is through a PBS file, submitted to the head node of the cluster, which in turn invokes an R script to carry out the computations. The following PBS file was used to fit the LD models and do gene drops with the `Nhlsim` data.

```
-------------------------Nhlsim.pbs-----------------------
## submit this job to the cluster head node with
##    qsub -V Nhlsim.pbs
#!/bin/sh
#PBS -S /bin/bash
## Specify resources: 22 nodes, one processor per node, total of 16GB
## memory, estimate total time to complete of 50 minutes
#PBS -l nodes=22:ppn=1,mem=16gb,walltime=00:50:00
## Capture messages to stdout and stderr in files.
#PBS -o Nhlsimrun.out
#PBS -e Nhlsimrun.err
## Execute any shell commands needed to give the job access to R.
## On our cluster:
##     source /hpc/software/etc/colony-login
##     module load LANG/R/2.14.0
## Change to the directory from which the job was submitted
cd $PBS_O_WORKDIR
## Run the R script
R --vanilla -f Nhlsim.R
------------------------------------------------------------
```

The PBS script calls the following R script to carry out the analysis. The script assumes the R packages **snow** and **Rmpi** have been installed on the cluster.

```
-------------------------Nhlsim.R-------------------------
library(snow)
cl<-makeCluster(22,type="MPI")
clusterEvalQ(cl,library("CrypticIBDcheck"))
ss<-sim.control(simulate=TRUE,cl=cl)
cibd.cl<-IBDcheck(dat,simparams=ss)
save(cibd.cl,file="cibd.cl.RData")
stopCluster(cl)
------------------------------------------------------------
```

After the `Nhlsim.pbs` job has finished, one can start an interactive R session in the directory that contains the file `cibd.cl.RData` and plot the results as follows:

```
R> library(CrypticIBDcheck)
R> load("cibd.cl.RData")
R> plot(cibd.cl)
```

These commands produce the set of plots displayed in Figures 1–3.

Affiliation:

Brad McNeney
Department of Statistics and Actuarial Science
Simon Fraser University
Burnaby, BC, Canada
E-mail: mcneney@sfu.ca
URL: http://stat.stat.sfu.ca