IBD_Haplo R Tools

Marshall Brown, updated by E. A. Thompson

November 7, 2011

1 Introduction

`IBDhaploRtools` consists of several functions to store, analyze, and plot the output of the `IBD_Haplo` software package. More information regarding `IBD_Haplo` can be found at www.stat.washington.edu/thompson/Genepi/pangaea.shtml.

2 Tutorial

This tutorial provides a description of the main functions in this package, and demonstrates how to implement these functions on the `IBD_Haplo` gold standards. The gold standards used by this tutorial were created by running `IBD_Haplo` on four sets of four simulated haplotypes/two individuals. Specifically, the `IBD_Haplo` output files were created by running the MORGAN program `ibd_haplo`, as described in the file `README_ibdhap`, which can be found in the "Gold" subdirectory of the `IBD\_Haplo` directory in MORGAN. If the executable is linked to the current directory, the linux command

```
./ibd_haplo ibd_haplo.par > ibd_haplo.out
```

creates the file "qibd_h_gold.out." This file, along with "compu_4hap.dat" and "trueibd_h_Gold.txt" are the only files needed to follow along in this tutorial. These three files, together with the gzipped tar file of the R package should be placed in the directory where you will open your R session.

To begin, we can install the package into `R` by downloading "`IBDhaploRtools\_1.2.tar.gz`," opening an `R` session, and running:

```
> install.packages("IBDhaploRtools_1.2.tar.gz", repos = NULL)
> library("IBDhaploRtools")
```

Now that we have installed the package, we can use the help pages by typing `?[function name]` (for example `?ibdhap.make.states`.) Please see these for more information on function arguments. It is important to note that all the functions listed below besides `ibdhap.makes.states` take the output of `ibdhap.make.states` as input. Thus, it is imperative to run this `ibdhap.make.states` first.

ibdhap.make.states: stores and simplifies the main output files (called "qibd_h.out" in the Gold examples) created by IBD Haplo. This is acheived by "calling" a marker to be in an ibd state if the probability of the state for the marker (conditional on the data and the model) meets the "cutoff" value. If the cuttoff is met, the value of the marker is set to the ibd state (integer value from 1–15 or 1–9), otherwise value of the marker is set to 0 (which means this marker is a "no call".) `ibdhap.make.states` outputs a `R` `data.frame` where each row is a marker, and each column is a set of four haplotypes. The value at a marker for a set of haplotypes is as described above. The `R` `data.frame` that this function creates is expected by other functions in this package.

To run this function on the gold standards, run:

```
> qibd.gold<- ibdhap.make.states( qibd.filename = "qibd_h_gold.out",
+                       dat.filename="compu_4hap.dat" , cutoff = 0.9)

[1] "Data read in successfully"
[1] "pair 1 of 4 complete..."
[1] "pair 2 of 4 complete..."
[1] "pair 3 of 4 complete..."
[1] "pair 4 of 4 complete..."
```

This data frame has four columns and 2,000 rows (one for each marker). The first 20 markers for each haplotype are:

```
> qibd.gold[1:20,]

   V1 V2 V3 V4
1   0  0 15  0
2   0  0 15  0
3   0  0 15  0
```

```
4    0  0 15  0
5    0  0 15  0
6    0  0 15  0
7    0  0 15  0
8    5 15 15 15
9    5 15 15 15
10   5 15 15 15
11   5 15 15 15
12   5 15 15 15
13   0 15 15 15
14  15 15 15 15
15  15 15 15 15
16  15 15 15 15
17  15 15 15 15
18  15 15 15 15
19  15 15 15 15
20  15 15 15 15
```

For marker eight, we see that `IBD_Haplo` inferred with greater than 0.9 probability that the first set of haplotypes were in IBD state 5 while the other three sets of haplotypes were in IBD state 15 (no IBD shared).

ibdhap.summary : summarizes the data created by `ibdhap.make.states` by calculating mean lengths of ibd segments, mean proportions of ibd shared, and counts on ibd segments. Averages are taken over all markers and all sets of haplotypes.

```
> summary.gold <- ibdhap.summary( qibd.gold, data.type="h")
> summary.gold$mean.prop

 any.ibd  not.ibd  no.call
0.193750 0.670875 0.135375

> summary.gold$mean.length

    len.ibd len.not.ibd  len.nocall
   51.66667   103.15385    13.19512

> summary.gold$seg.counts
```

```
    ibd  no.ibd no.call
     30      52      82
```

This shows, for example, that on average 67.0875 % of the chromosome is in IBD state 15 (no ibd shared among the four haplotypes), and that the mean length (number of markers) of a continuous segment of the chromosome with no ibd shared is 103.15 markers.

ibdhap.seg.lengths : given the ibd states from a set of haplotypes/pair of genotypes (taken from a column of the output of ibdhap.make.states), this function creates a `data.frame` consisting of all segments of differing ibd state, paired with their respective length. This function is called by many of the other functions, and provides equivalent information held in a column of "qibd.gold" but just in a different form.

```
> seg.lengths.gold<-ibdhap.seg.lengths(qibd.gold[,4])
> seg.lengths.gold
```

```
   ibd.state seg.lengths
1          0           7
2         15          27
3          0          27
4         15          10
5          0          15
6         15         488
7          0          14
8         15          69
9          0          23
10        15         110
11         0           3
12        12         466
13         0          14
14        12          35
15         0          16
16        12           1
17         0           6
18        12         174
19         0          18
```

```
20         12           6
21          0          21
22         15         115
23          0           7
24         13           5
25          0          14
26         15          14
27          0          19
28         12         187
29          0           2
30         15          86
```

For the fourth set of haplotypes, this shows that there are 30 segments of differing ibd state as inferred by `IBD_Haplo`. The sixth segment is in ibd state 15 and consists of 488 markers, while the eighteenth segment is in ibd state 12 and consists of 174 markers. If a vector of positions is given, segment lengths will be given in its respective units. Notice how this portrays the same information as `qibd[,4]`, but in a more compact form.

ibdhap.transitions: creates a matrix of transition counts from when ibd state switches along the chromosome.

```
> options(width=100) #set the display so the matrix displays nicely
> transitions.gold<-ibdhap.transitions(qibd.gold, data.type="h")
> #split the transitions matrix into two parts so it fits on the
> #document
> transitions.gold[,1:8]
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    0    0    0    0    0    0    0    0
[2,]    0    0    0    0    0    0    0    0
[3,]    0    0    0    0    0    0    0    0
[4,]    0    0    0    0    0    0    0    0
[5,]    0    0    0    0    0    0    0    0
[6,]    0    0    0    0    0    0    0    0
[7,]    0    0    0    0    0    0    0    0
[8,]    0    0    0    0    0    0    0    0
```

```
 [9,]    0    0    0    0    0    0    0    0
[10,]    0    0    0    0    0    0    0    0
[11,]    0    0    0    0    0    0    0    0
[12,]    0    0    0    0    0    0    0    0
[13,]    0    0    0    0    0    0    0    0
[14,]    0    0    0    0    0    0    0    0
[15,]    0    0    2    0    3    0    0    3
```

```
> transitions.gold[,9:15]
```

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

Hence, across all sets of haplotypes, there where four instances when ibd state 12 transitioned to ibd state 15.

ibdhap.barplot : Graphically displays regions of any ibd sharing, no ibd sharing and no calls along a chromosome for a set of haplotypes / pair of genotypes. The default colors are red, white, and grey for any state with ibd sharing, no ibd sharing and no calls, respectively.

```
> par(mfrow=c(4,1))
> ibdhap.barplot(qibd.gold[,1], data.type="h", xlab="", ylab="")
> ibdhap.barplot(qibd.gold[,2], data.type="h", xlab="", ylab="")
```

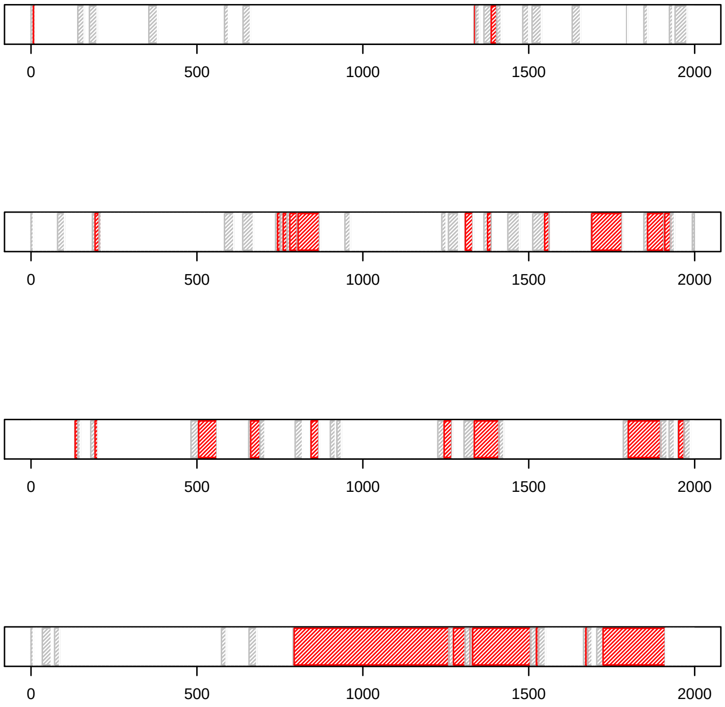

Figure 1: Barplots for four sets of four haplotypes using IBD_Haplo output. Red means some ibd shared, white is no ibd shared and grey means “no call.”

```
> ibdhap.barplot(qibd.gold[,3], data.type="h", xlab="", ylab="")
> ibdhap.barplot(qibd.gold[,4], data.type="h", xlab="", ylab="")
```

We can see that the first set of haplotypes does not have much (inferred) ibd shared among the four haplotypes, while the fourth set has a large portion of the chromosome in an ibd state where ibd is shared among the four haplotypes. Since we simulated these haplotypes, and therefore know the true ibd states for these four sets of haplotypes, it is interesting to compare the previous barplots to similar barplots created from using the known ibd states. The true ibd patterns shared among the four sets of haplotypes are contained in the file “trueibd_h_Gold.txt”.

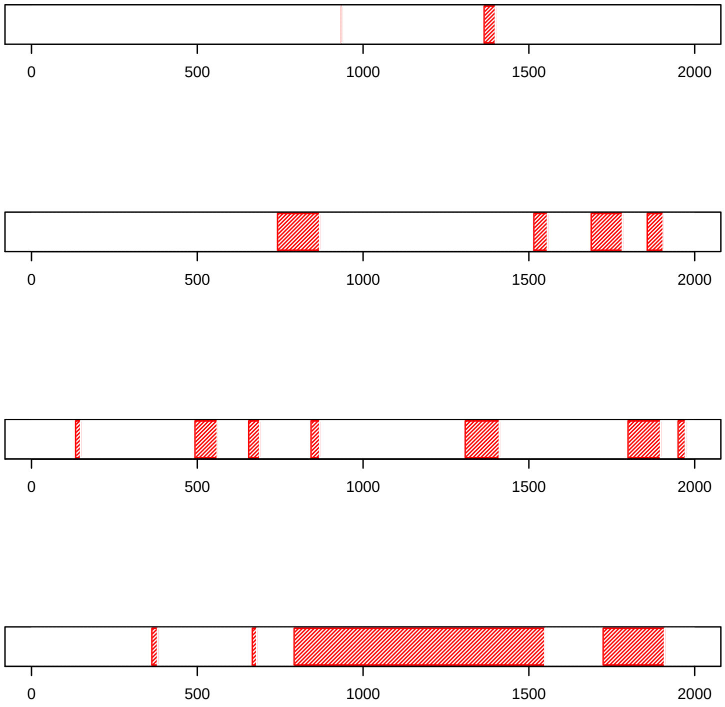

Figure 2: Barplots for four sets of four haplotypes using simulated data. Red means some ibd shared, and white is no ibd shared

```
> trueibd.gold<-read.table("trueibd_h_Gold.txt")
> trueibd.gold<-t(trueibd.gold) #transpose the data
> #get rid of extra "names" rows
> trueibd.gold<-trueibd.gold[-(1:2),]

> par(mfrow=c(4,1))
> ibdhap.barplot(trueibd.gold[,1], data.type="h", xlab="", ylab="")
> ibdhap.barplot(trueibd.gold[,2], data.type="h", xlab="", ylab="")
> ibdhap.barplot(trueibd.gold[,3], data.type="h", xlab="", ylab="")
> ibdhap.barplot(trueibd.gold[,4], data.type="h", xlab="", ylab="")
```

Upon inspection, we see that `IBD_Haplo` does well in inferring ibd in this example.

ibdhap.correct.bylen : if simulated data (true ibd state) is available, this function creates a dataframe summarizing how well `IBD_haplo` infers segments based on segment length. Each row in the dataframe represents a segment of true (simulated) ibd (shared or not shared). If `ibd.only` is true, then only segments where ibd is shared are returned. The first column of resulting dataframe gives the length of the segment (as specified by the position vector), the second column gives the proportion of markers inferred correctly by ibdhaplo, and the third column gives the ibd state of the segment.

```
>  gold.corr.bylen<-ibdhap.correct.bylen( qibd.gold,
+           trueibd.gold, data.type="h",
+           ibd.only=FALSE, position=NA)
```

We can use the output of `ibdhap.correct.bylen` to create figure 3. The x axis is the length of the segment, and the y axis is proportion snps called correctly within the true segment.

```
>  plot( gold.corr.bylen[,1], gold.corr.bylen[,2],
+        xlab="segment length", ylab="proportion correct")
> lines(lowess(gold.corr.bylen[,1],
+       gold.corr.bylen[,2]), col="blue")
```

We can see by figure 3 that `IBD\_Haplo` does very well with ibd segments that are above 25 to 50 snps in length. In these data, this corresponds with ibd segments that are roughly one centimorgan in length.

This completes our quick run through of the main functions of `IBDHaploRtools`. Note that all functions that we ran above for haplotypic data can also be run on genotypic data by setting `data.type=``g''`. Please see the man pages for more information regarding function arguments, descriptions, and examples.

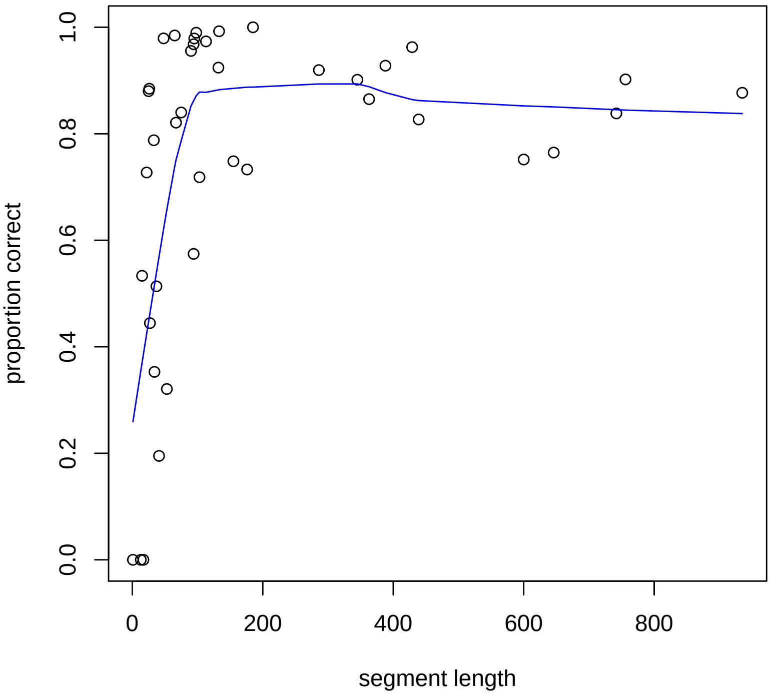

Figure 3: Each point in the plot is a segment of true (simulated) ibd (shared or not shared).The x axis is the length of the segment, and the y axis is proportion snps called correctly within the true segment