# mlPhaser

Dave T. Gerrard
david.gerrard@manchester.ac.uk

September 3, 2012

**Abstract**

Select combinations of known haplotypes to fully explain multi-locus genotypes. Suited to highly diverse but well characterised systems such as HLA. Ambiguous cases can be ranked using known haplotype frequencies.

## 1 Introduction

The mlPhaser package attempts to find groups of haplotypes that are consistent with and fully explain a genotype. This task can be difficult when there are very many possible haplotypes. The package does not examine all possible haplotypes (see other packages for that, e.g. haplo.stats) but instead works from a list of 'known' haplotypes. Optionally, independently determined haplotype frequencies are used to select from ambiguous cases.

The package was developed for multi-locus HLA data for which there may be hundreds of candidate alleles at each locus and thousands of known haplotypes to choose from. The functions take in genotypes and haplotypes in a simple format and should be amenable to most multi-locus datasets. The algorithms are designed to work with an arbitrary ploidy level so should be useable with triploid, tetraploid and systems of greater ploidy.

## 2 Installation

Install the package from your R session. Use the packages menu or the following command.

```
> install.packages("mlPhaser_0.01.zip", repos=NULL)
```

## 3 Using mlPhaser

Start by loading the library.

```
> library(mlPhaser)
```

## 3.1 Load haplotype and genotype data

Typical usage will require a table of genotypes and a table of known haplotypes.

The haplotypes are easy to create

```
> haplotypes <- data.frame(        A= c("a","b","c","a","b","c","b"),
+                                  B= c("a","b","c","b","c","a","a"),
+                                  C= c("a","b","c","b","c","a","a") )
> rownames(haplotypes) <- apply(haplotypes, 1, paste,sep="" , collapse="")
> haplotypes
```

```
    A B C
aaa a a a
bbb b b b
ccc c c c
abb a b b
bcc b c c
caa c a a
baa b a a
```

In many cases, these known haplotypes will come with frequency data, this can be used later to sort candidate haplotype groups.

```
> haploFreqs <- c(0.4, 0.3, 0.15, 0.07,0.05, 0.02, 0.01)
> names(haploFreqs) <- rownames(haplotypes)
> haploFreqs
```

```
 aaa  bbb  ccc  abb  bcc  caa  baa
0.40 0.30 0.15 0.07 0.05 0.02 0.01
```

Genotypes are encoded similarly but with multiple columns per locus. Here is a simple genotype heterozygous at each of three loci.

```
> thisGenotype <- data.frame(A.1="a", A.2="b", B.1="a", B.2="b",C.1="a", C.2="b")
> thisGenotype
```

```
  A.1 A.2 B.1 B.2 C.1 C.2
1   a   b   a   b   a   b
```

The function simGenoFromHaplo can be used to simulate a set of genotypes.

```
> my.genotypes <- simGenoFromHaplo(haploTable=haplotypes, haploFreqs=haploFreqs ,
+                 n=20, ploidy=2)
> head(my.genotypes)
```

```
  A.1 A.2 B.1 B.2 C.1 C.2
1   a   c   a   c   a   c
2   b   a   b   a   b   a
3   a   c   a   c   a   c
4   b   c   b   a   b   a
5   c   a   c   a   c   a
6   c   a   c   a   c   a
```

## 3.2 Valid haplotype groups.

The funciton getValidHaploGroups returns groups of haplotypes that fully explain the genotype. There may be none, one or more groups per genotype.

```
> my.valid.groups <- getValidHaploGroups(thisGenotype,haplotypes)
> my.valid.groups

[[1]]
[[1]]$aaa
[[1]]$aaa$A
[1] "a"

[[1]]$aaa$B
[1] "a"

[[1]]$aaa$C
[1] "a"


[[1]]$bbb
[[1]]$bbb$A
[1] "b"

[[1]]$bbb$B
[1] "b"

[[1]]$bbb$C
[1] "b"



[[2]]
[[2]]$abb
[[2]]$abb$A
[1] "a"

[[2]]$abb$B
[1] "b"

[[2]]$abb$C
[1] "b"


[[2]]$baa
[[2]]$baa$A
[1] "b"

[[2]]$baa$B
[1] "a"
```

```
[[2]]$baa$C
[1] "a"
```

The object returned by getValidHaploGroups is a list of lists. Another function, phaseReport, is available to get valid groups for a set of genotypes. phaseReport also formats the results into a table.

```
> phaseReport(thisGenotype,haplotypes)
```

```
  id n.validGroups haploGroup likelihood
1  1             2    aaa/bbb         NA
2  1             2    abb/baa         NA
```

```
> phaseReport(my.genotypes,haplotypes)         # a full table of genotypes
```

```
   id n.validGroups haploGroup likelihood
1   1             1    aaa/ccc         NA
2   2             2    aaa/bbb         NA
3   2             2    abb/baa         NA
4   3             1    aaa/ccc         NA
5   4             1    bbb/caa         NA
6   5             1    aaa/ccc         NA
7   6             1    aaa/ccc         NA
8   7             1    aaa/aaa         NA
9   8             1    aaa/abb         NA
10  9             1    aaa/aaa         NA
11 10             2    aaa/bbb         NA
12 10             2    abb/baa         NA
13 11             2    aaa/bbb         NA
14 11             2    abb/baa         NA
15 12             1    aaa/aaa         NA
16 13             1    bbb/bcc         NA
17 14             1    abb/ccc         NA
18 15             1    abb/bbb         NA
19 16             2    aaa/bbb         NA
20 16             2    abb/baa         NA
21 17             1    abb/ccc         NA
22 18             2    aaa/bbb         NA
23 18             2    abb/baa         NA
24 19             2    aaa/bbb         NA
25 19             2    abb/baa         NA
26 20             1    bbb/bbb         NA
```

A set of haplotype frequencies can be used to order groups of haplotypes. Currently this is done by simply multiplying the probabilities of each haplotype within the group.

```
> # use haplotype frequencies to rank candidate haplotype groups.
> phaseReport(thisGenotype,haplotypes, haploFreqs)
```

```
  id n.validGroups haploGroup likelihood
1  1             2    aaa/bbb     0.1200
2  1             2    abb/baa     0.0007
```

```
> # return only the best haplotype group for each genotype.
> # outFormat="all" is the default
> phaseReport(thisGenotype,haplotypes, haploFreqs, outFormat="top")

  id n.validGroups haploGroup likelihood
1  1             2    aaa/bbb       0.12

> # get phase report on all genotypes
> phaseReport(my.genotypes,haplotypes, haploFreqs, outFormat="all")

   id n.validGroups haploGroup likelihood
1   1             1    aaa/ccc     0.0600
2   2             2    aaa/bbb     0.1200
3   2             2    abb/baa     0.0007
4   3             1    aaa/ccc     0.0600
5   4             1    bbb/caa     0.0060
6   5             1    aaa/ccc     0.0600
7   6             1    aaa/ccc     0.0600
8   7             1    aaa/aaa     0.1600
9   8             1    aaa/abb     0.0280
10  9             1    aaa/aaa     0.1600
11 10             2    aaa/bbb     0.1200
12 10             2    abb/baa     0.0007
13 11             2    aaa/bbb     0.1200
14 11             2    abb/baa     0.0007
15 12             1    aaa/aaa     0.1600
16 13             1    bbb/bcc     0.0150
17 14             1    abb/ccc     0.0105
18 15             1    abb/bbb     0.0210
19 16             2    aaa/bbb     0.1200
20 16             2    abb/baa     0.0007
21 17             1    abb/ccc     0.0105
22 18             2    aaa/bbb     0.1200
23 18             2    abb/baa     0.0007
24 19             2    aaa/bbb     0.1200
25 19             2    abb/baa     0.0007
26 20             1    bbb/bbb     0.0900

> # Same but only return the top result for each genotype
> phaseReport(my.genotypes,haplotypes, haploFreqs, outFormat="top")

   id n.validGroups haploGroup likelihood
1   1             1    aaa/ccc     0.0600
2   2             2    aaa/bbb     0.1200
4   3             1    aaa/ccc     0.0600
5   4             1    bbb/caa     0.0060
6   5             1    aaa/ccc     0.0600
7   6             1    aaa/ccc     0.0600
8   7             1    aaa/aaa     0.1600
9   8             1    aaa/abb     0.0280
10  9             1    aaa/aaa     0.1600
```

```
11 10             2    aaa/bbb     0.1200
13 11             2    aaa/bbb     0.1200
15 12             1    aaa/aaa     0.1600
16 13             1    bbb/bcc     0.0150
17 14             1    abb/ccc     0.0105
18 15             1    abb/bbb     0.0210
19 16             2    aaa/bbb     0.1200
21 17             1    abb/ccc     0.0105
22 18             2    aaa/bbb     0.1200
24 19             2    aaa/bbb     0.1200
26 20             1    bbb/bbb     0.0900
```

# 4 TO-DO

More sophisticated frequency/probability integration. Export results.

# 5 R Session

```
> sessionInfo()

R version 2.14.1 (2011-12-22)
Platform: i386-pc-mingw32/i386 (32-bit)

locale:
[1] LC_COLLATE=English_United Kingdom.1252  LC_CTYPE=English_United Kingdom.1252    LC_MONETARY=English_United Kingdom.1252
[4] LC_NUMERIC=C                            LC_TIME=English_United Kingdom.1252

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

other attached packages:
[1] mlPhaser_0.01

loaded via a namespace (and not attached):
[1] tools_2.14.1
```