Tutorial on haplotype evidence with focus on Y-STR haplotype analysis using the discrete Laplace method

Mikkel Meyer Andersen
mikl@math.aau.dk

May 20, 2014

1 Prerequisites

1. Open R.
2. Next, install the required packages if you have not already done so:

```
install.packages(c("disclapmix", "maps", "mapplots", "RColorBrewer"))
```

3. Load the disclapmix package:

```
library(disclapmix)
```

2 Count methods

```
data(danes)
str(danes)
```

```
## 'data.frame': 136 obs. of  11 variables:
##  $ DYS19   : int  13 13 13 13 13 14 14 14 14 14 ...
##  $ DYS389I : int  13 13 11 14 13 13 13 13 13 13 ...
##  $ DYS389II: int  29 30 27 32 30 30 30 27 29 29 ...
##  $ DYS390  : int  22 25 23 24 24 23 24 23 23 24 ...
##  $ DYS391  : int  10 10 11 10 10 11 11 10 10 10 ...
##  $ DYS392  : int  15 14 14 11 11 13 13 13 13 13 ...
##  $ DYS393  : int  13 13 13 13 13 14 13 13 13 13 ...
##  $ DYS437  : int  14 15 15 14 14 15 15 15 15 15 ...
##  $ DYS438  : int  11 12 12 10 10 12 12 12 12 12 ...
##  $ DYS439  : int  12 12 12 12 12 13 11 12 12 12 ...
##  $ n       : int  1 1 1 1 2 1 1 1 1 4 ...
```

```
N <- sum(danes$n)
N
```

```
## [1] 185
```

If the haplotype of a suspect has not been observed before, we can use the following estimators. First, the traditional count method (very conservative for unobserved haplotypes, but reasonable if a haplotype has been observed relatively often earlier):

```
count_match_prob <- 1/(N + 1)
1/count_match_prob

## [1] 186
```

Then we inspect Brenner's kappa method (remember that the haplotype is not in the database, it is previously unobserved):

```
singletons <- sum(danes$n == 1L)
singletons

## [1] 112

kappa <- (singletons + 1)/(N + 1)
kappa

## [1] 0.6075

brenners_match_prob <- (1 - kappa)/(N + 1)
1/brenners_match_prob

## [1] 473.9
```

We see that Brenner's kappa method inflates the LR by a factor of:

```
(1/brenners_match_prob)/(1/count_match_prob)

## [1] 2.548

count_match_prob/brenners_match_prob

## [1] 2.548

1/(1 - kappa)

## [1] 2.548
```

Hence, the LR using Brenner's kappa is 2.5479 times larger than the count method.

And finally the discrete Laplace method. First the data is prepared:

```
danes_cor <- danes
danes_cor$DYS389II <- with(danes_cor, DYS389II - DYS389I)  # Decouple 389I and 389II
danes_db <- as.matrix(danes_cor[rep(1L:nrow(danes_cor), danes_cor$n),
    1L:10L])
```

Then the model is fitted (should take less than 10 seconds):

```
fits <- lapply(1L:10L, function(clusters) {
    fit <- disclapmix(x = danes_db, clusters = clusters, verbose = 0L,
        iterations = 500L)
    return(fit)
```

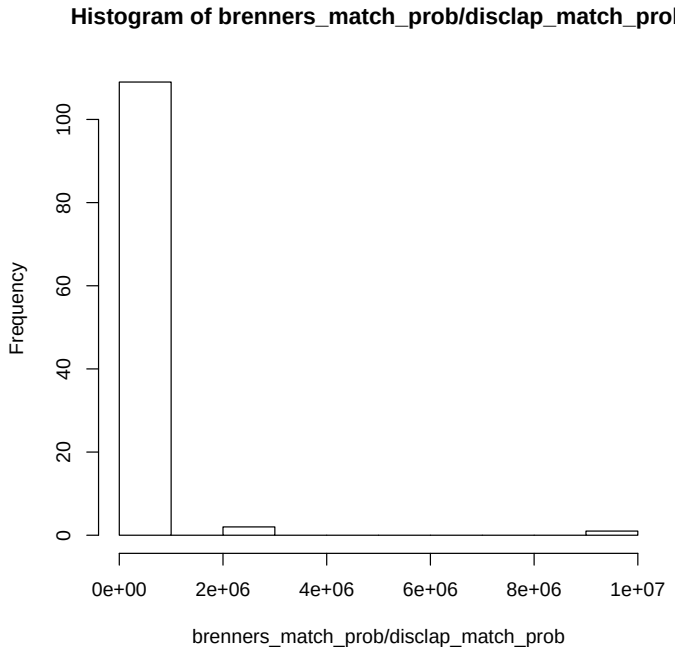
```
})

BICs <- unlist(lapply(fits, function(fit) fit$BIC_marginal))
best_fit_BIC <- fits[[which.min(BICs)]]
best_fit_BIC

## disclapmixfit from 185 observations on 10 loci with 4 clusters.
```

One at a time, we assume that the singletons in the database were that of interest and then added to the database:

```
disclap_match_prob <- predict(best_fit_BIC, newdata = as.matrix(subset(danes_cor,
    n == 1L)[, 1L:10L]))
max(disclap_match_prob)

## [1] 0.007935

1/max(disclap_match_prob)

## [1] 126

min(disclap_match_prob)

## [1] 2.31e-10

1/min(disclap_match_prob)

## [1] 4.33e+09

summary(1/disclap_match_prob)

##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 1.26e+02 1.81e+03 1.27e+04 6.68e+07 9.08e+04 4.33e+09
```

Compare to Brenner's kappa method:

```
hist(brenners_match_prob/disclap_match_prob)
```




```
brenners_match_prob/max(disclap_match_prob)
```

```
## [1] 0.2659
```

```
brenners_match_prob/min(disclap_match_prob)
```

```
## [1] 9136289
```

Let us investigate the two extreme cases. First, we calculate distances to other haplotypes as we are going these in just a moment:

```
dist_danes <- as.matrix(dist(danes_cor, method = "manhattan"))
```

First the one with the maximum match probability:

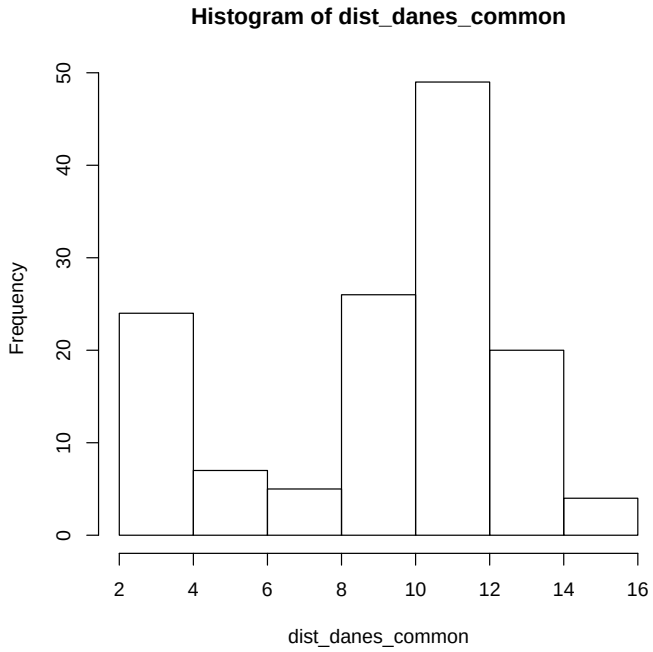
```
h_common <- as.matrix(subset(danes_cor, n == 1L)[which.max(disclap_match_prob),
    1L:10L])
h_common
```

```
##    DYS19 DYS389I DYS389II DYS390 DYS391 DYS392 DYS393 DYS437 DYS438 DYS439
## 58    14      12       16     22     10     11     13     16     10     10
```

```
i_db_common <- which(apply(danes_cor[, 1L:10L], 1, function(h) all(h ==
    h_common)))
dist_danes_common <- dist_danes[i_db_common, -i_db_common]
summary(dist_danes_common)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00    8.00   11.00    9.56   12.00   16.00
```

```
hist(dist_danes_common)
```



And then the one with the minimum match probability:

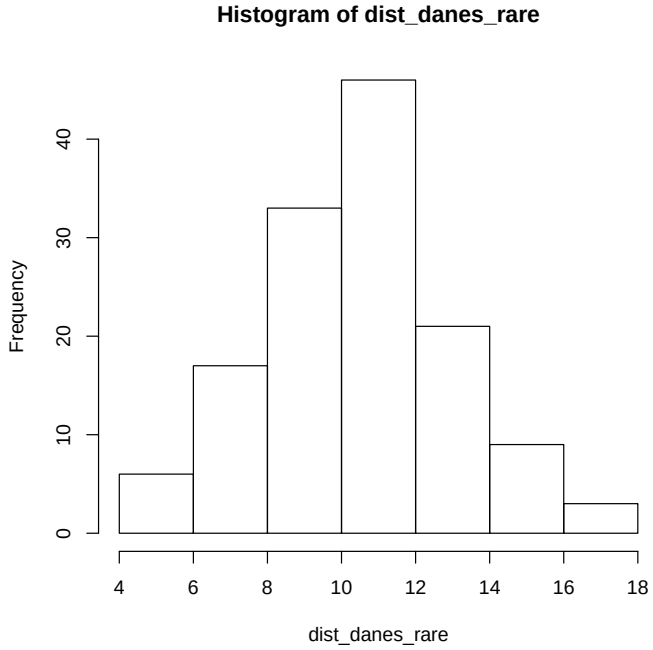
```
h_rare <- as.matrix(subset(danes_cor, n == 1L)[which.min(disclap_match_prob),
    1L:10L])
h_rare
```

```
##   DYS19 DYS389I DYS389II DYS390 DYS391 DYS392 DYS393 DYS437 DYS438 DYS439
## 1    13      13       16     22     10     15     13     14     11     12
```

```
i_db_rare <- which(apply(danes_cor[, 1L:10L], 1, function(h) all(h ==
    h_rare)))
dist_danes_rare <- dist_danes[i_db_rare, -i_db_rare]
summary(dist_danes_rare)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     9.5    11.0    11.0    12.0    18.0
```

```
hist(dist_danes_rare)
```



3 More advanced examples

Please refer to the following files in the material for more detailed examples (also refer to the README.txt file):

1. analyse-hapfreq.R
2. analyse-mixtures.R
3. analyse-cluster.R
4. analyse-phist.R