Case analysis using the **DNAmixtures** package

Therese Graversen
University of Oxford
graversen@stats.ox.ac.uk

April 6, 2014

This document is a companion to Cowell et al. (2014), which presents a statistical model for forensic DNA mixtures. We give the details of their analysis relating to mixtures MC15 and MC18. All analyses in the paper were performed using R-package **DNAmixtures** (Graversen, 2014), which may be found at the package web-page

http://dnamixtures.r-forge.r-project.org

along with a guide to installation. The analysis in this document were performed using version 0.1.3 of **DNAmixtures**; the package version can be checked by

```
> packageVersion("DNAmixtures")

[1] '0.1.3'
```

Details on the computational approach may be found in Graversen and Lauritzen (2014) as well as in the package help pages.

Contents

1 Introduction

The package and relevant datasets are loaded as

```
> library(DNAmixtures)
> data(MC15, MC18, USCaucasian)
```

The peak height information is for each of the two DNA mixtures given as a `data.frame` containing the `marker`, the `allele`, and the `height` of the detected peaks.

```
> MC15[MC15$marker == "TH01",]

   marker allele height K1 K2 K3
35   TH01    7.0    727  1  0  0
36   TH01    8.0    625  1  0  0
37   TH01    9.0      0  0  2  0
38   TH01    9.3    165  0  0  2
```

Three individuals `K1`, `K2`, and `K3` are associated to the case, and for these we have their full DNA profile. The genotype of a contributor i is represented by a vector of allele counts $(n_{i1}, \ldots, n_{iA})$, denoting by n_{ia} the number of alleles a that the contributor possesses.

A *hypothesis* specifies a set of contributors to the mixture. We distinguish between *known* and *unknown* contributors, depending on whether their DNA profiles are known to us or not.

We shall focus on hypotheses that include as known contributors the individuals `K1` and `K2`. Individual `K3` is the defendant, and the *prosecution hypotheses* thus include `K1`, `K2`, and `K3` as well as a number of unknown contributors. We shall use $H_p(k)$ to denote a prosecution hypothesis involving a total of k contributors. Similarly, we use $H_d(k)$ for the various *defence hypotheses* involving k contributors of which contributors `K1` and `K2` are known.

The allele counts of an unknown contributor follow a multinomial distribution with some specified allele frequencies; we use

```
> data(USCaucasian)
> db <- USCaucasian
> db[db$marker == "TH01",]

   marker allele   frequency
22   TH01    5.0 0.001659967
23   TH01    6.0 0.231785364
24   TH01    7.0 0.190396192
25   TH01    8.0 0.084438311
26   TH01    9.0 0.114237715
27   TH01    9.3 0.367542649
28   TH01   10.0 0.008279834
29   TH01   11.0 0.001659967
```

Unknown contributors are assumed mutually unrelated and unrelated to the known contributors. The genotypes are assumed independent across markers and the two alleles to be inherited independently.

When $R \geq 1$ mixtures are modelled jointly, we include in the model the joint set of contributors, assuming that a person i has contributed with some fraction ϕ_{ri} to mixture r, allowing $\phi_{ri} = 0$.

In a hypothesis involving p unknown contributors these are named `U1`, ..., `Up` and they are ordered in terms of non-increasing contributions to the first mixture, i.e. so that

$$\phi_{1,\texttt{U1}} \geq \cdots \geq \phi_{1,\texttt{Up}}$$

Peak height distribution Consider a model of $R \geq 1$ mixtures and a set of k contributors. Given the DNA profiles of the k contributors, the peak heights are assumed mutually independent and for mixture r, allele a, the peak height H_{ra} is gamma distributed as

$$H_{ra} \sim \Gamma\left(\rho_r \sum_{i=1}^{k} \left\{(1-\xi_r)n_{ia} + \xi_r n_{i,a+1}\right\} \phi_{ri}, \eta_r\right)$$

Applying a detection threshold $C_r \geq 0$ for each mixture r we observe

$$Z_{ra} = \begin{cases} H_{ra}, & H_{ra} \geq C_r \\ 0, & H_{ra} < C_r \end{cases}$$

There is one set of model parameters for each of the R mixtures, and so the total set of parameters are

$$\begin{array}{c} \\ 1 \\ \vdots \\ R \end{array} \begin{array}{c} \begin{array}{cccc} \rho & \eta & \xi & \phi \end{array} \\ \begin{pmatrix} \rho_1 & \eta_1 & \xi_1 & \phi_{1,1}, \ldots, \phi_{1,k} \\ \vdots & \vdots & \vdots & \vdots \\ \rho_R & \eta_R & \xi_R & \phi_{R,1}, \ldots, \phi_{R,k} \end{pmatrix} \end{array}.$$

2 Analysis of MC15

2.1 Four contributors

Firstly, consider the prosecution hypothesis $H_p(4)$: `K1` & `K2` & `K3` & `U1`.

```
> mix15P.4 <- DNAmixture(
      list(MC15),             ## Peak heights and known profiles
      C = list(50),           ## Detection threshold
      k = 4,                  ## Number of contributors
      K = c("K1", "K2", "K3"),## Names of known contributors
      database = db           ## Allele frequencies
      )
> mix15P.4

A DNA mixture model with 4 contributors.

Known: K1 K2 K3
Unknown: U1
```

```
Mixtures included: list(MC15)
Detection threshold(s): 50
```

A parameter for this model is specified as

```
> p <- mixpar(rho = list(25), eta = list(20), xi = list(0.07),
              phi = list(c(K1 = 0.25, K2 = 0.25, K3 = 0.25, U1 = 0.25)))
```

Starting the maximisation from the fairly arbitrary point `p` we get

```
> ml15P.4 <- mixML(mix15P.4, p)
> ml15P.4$mle

      rho     eta       xi     phi.U1   phi.K1    phi.K2   phi.K3
1   34.24   26.67   0.0737   0.008434   0.8205   0.04735   0.1237

> ml15P.4$lik          ## logL

[1] -271.8025

> ml15P.4$lik/log(10)  ## log10(L)

[1] -118.0423
```

As the competing explanation, we consider the defence hypothesis $H_d(4)$: K1 & K2 & U1 & U2.

```
> mix15D.4 <- DNAmixture(list(MC15),       ## Peak heights and known profiles
                         C = list(50),     ## Detection threshold
                         k = 4,            ## Number of contributors
                         K = c("K1", "K2"),## Names of known contributors
                         database = db)    ## Allele frequencies
> p <- mixpar(rho = list(30), eta = list(30), xi = list(0.07),
              phi = list(c(K1 = 0.8, K2 = 0.05, U2 = 0.1, U1 = 0.05)))
> ml15D.4 <- mixML(mix15D.4, p)
> ml15D.4$mle

      rho     eta        xi    phi.U1    phi.U2   phi.K1    phi.K2
1   25.54   35.81   0.07186   0.08114   0.08113   0.7983   0.03941

> ml15D.4$lik          ## logL

[1] -297.8047

> ml15D.4$lik/log(10)  ## log10(L)

[1] -129.3349
```

The weight of evidence against K3 may now be found as

```
> (ml15P.4$lik - ml15D.4$lik)/log(10)

[1] 11.2926
```

2.1.1 Variance estimates for the MLE

The function `varEst` estimates the variance matrix for the MLE based on the Hessian, as computed by `numDeriv` in the maximum point. Through the argument `npars` we specify whether each of the four parameters `rho, eta, xi`, and `phi` are fixed (0), equal for all traces (1), or different for all traces (as many parameters as there are traces). In our case, we analyse a single mixture with free parameters, so all values are simply 1.

```
> var15P.4 <- varEst(mix15P.4, ml15P.4$mle,
                     npars = list(rho = 1, eta = 1, xi = 1, phi = 1))
> summary(var15P.4)

            Estimate    StdErr
rho.1      34.237349   7.13107
eta.1      26.668601   5.61851
xi.1        0.073704   0.01441
phi.U1.1    0.008434   0.01852
phi.K1.1    0.820514   0.02014
phi.K2.1    0.047349   0.01361
phi.K3.1    0.123703   0.01532
```

We shall consider the parametrisation using $\mu = \rho\eta$ and $\sigma = 1/\sqrt{\rho}$ rather than (ρ, η). The corresponding MLE and their estimated standard errors are

```
> summary(var15P.4, transform = TRUE)

             Estimate     StdErr
mu.1       913.062223   35.04813
sigma.1      0.170903    0.01780
xi.1         0.073704    0.01441
phi.U1.1     0.008434    0.01852
phi.K1.1     0.820514    0.02014
phi.K2.1     0.047349    0.01361
phi.K3.1     0.123703    0.01532
```

The variance estimate for the defence is more complicated, since the maximum is on the boundary $\phi_{\texttt{U1}} = \phi_{\texttt{U2}}$. We condition on this event as follows. Firstly, we compute the unconstrained variance matrix

```
> var15D.4 <- varEst(mix15D.4, ml15D.4$mle,
                     npars = list(rho = 1, eta = 1, xi = 1, phi = 1))
```

We then transform to obtain the variance matrix for the MLE in the parametrisation using $(\phi_{\texttt{U1}}, \phi_{\texttt{U2}} - \phi_{\texttt{U1}})$ rather than $(\phi_{\texttt{U1}}, \phi_{\texttt{U2}})$. We denote by `dif` the new parameter `phi.U2.1-phi.U1.1` indicating the difference in contributions.

```
> ## dif = phi.U2.1 - phi.U1.1, all other parameters unchanged
> A <- diag(nrow(var15D.4$cov.trans))
> dimnames(A) <- dimnames(var15D.4$cov.trans)
> rownames(A)[rownames(A) == "phi.U2.1"] <- "dif"
> A["dif", "phi.U1.1"] <- -1
> newV <- A %*% var15D.4$cov.trans %*% t(A)
```

The variance matrix `newV` is singular due to the restriction that mixture proportions sum to 1. We therefore remove one parameter, $\phi_{\texttt{K2}}$, by removing the corresponding row and column

phi.K2.1 in the variance matrix. Inverting this, we get the concentration matrix.

```
> v <- newV[rownames(newV) != "phi.K2.1", colnames(newV) != "phi.K2.1"]
> conc <- solve(v)
```

Now, the concentration matrix conditionally on $\phi_{\texttt{U2}} - \phi_{\texttt{U1}} = 0$ is obtained simply by removing the corresponding row and column dif. Inverting the concentration matrix, we obtain the conditional variance.

```
> condV <- solve(conc[rownames(conc) != "dif", colnames(conc) != "dif"])
```

Finally, we transform to include the parameter $\phi_{\texttt{K2}} = \phi_{\texttt{U1}} + \phi_{\texttt{U2}} + \phi_{\texttt{K1}}$.

```
> B <- cbind(diag(5), rep(0:1, times = c(3,2)))
> dimnames(B) <- list(dimnames(condV)[[1]],
                      c(dimnames(condV)[[2]], "phi.K2.1"))
> condV <- t(B) %*% condV %*% B
```

The MLE and their estimated standard errors are

```
> var15D.4$mle.trans

      mu    sigma       xi    phi.U1    phi.U2   phi.K1    phi.K2
1  914.4   0.1979  0.07186   0.08114   0.08113   0.7983  0.03941

> sqrt(diag(condV))

      mu.1     sigma.1        xi.1    phi.U1.1    phi.K1.1    phi.K2.1
40.63032385  0.02299631  0.01897350  0.01319188  0.02766587  0.01994643
```

2.2 Three contributors

Consider now $H_p(3)$: K1 & K2 & K3. Although the hypothesis does not involve any unknown contributors, we still need to specify a database of allele frequencies – this is because the database also defines the range of possible alleles for each marker.

```
> mix15P.3 <- DNAmixture(list(MC15), C = list(50), k = 3,
                         K = c("K1", "K2", "K3"), database = db)
> p <- mixpar(rho = list(30), eta = list(30), xi = list(0.07),
              phi = list(c(K1 = 0.82, K2 = 0.05, K3 = 0.13)))
> ml15P.3 <- mixML(mix15P.3, p)
> ml15P.3$mle

     rho     eta       xi   phi.K1    phi.K2   phi.K3
1  33.86   26.94  0.07583   0.8248   0.04932   0.1259
```

For the defence hypothesis, consider $H_d(3)$: K1 & K2 & U1.

```
> mix15D.3 <- DNAmixture(list(MC15), C = list(50), k = 3,
                         K = c("K1", "K2"), database= db)
> p <- mixpar(rho = list(30), eta = list(30), xi = list(0.07),
              phi = list(c(K1 = 0.82, K2 = 0.05, U1 = 0.13)))
> ml15D.3 <- mixML(mix15D.3, p)
> ml15D.3$mle
```

```
      rho     eta         xi   phi.U1   phi.K1    phi.K2
1   26.95   33.86   0.08616   0.1222   0.8232   0.05462
```

The WoE against K3 in the case of 3 contributors is

```
> (ml15P.3$lik - ml15D.3$lik)/log(10)

[1] 12.11822
```

2.3 Identification of U1 under $H_d(3)$: K1 & K2 & U1

The DNAmixture object contains a full representation of the statistical model in terms of one Bayesian network per marker. If a marker has A alleles, then allele counts $(n_{11}, \ldots, n_{1A})$ for contributor U1 are represented by network variables n_1_1, ..., n_1_A.

Firstly, we condition on the observed peak heights, specifying also ml15D.3$mle as the parameter for the peak height model.

```
> setPeakInfo(mix15D.3, ml15D.3$mle)
```

Now, for the prediction of genotypes for U1, we compute for each marker the list of configurations of genotypes with probability above pmin = 0.001.

```
> mp15D.3 <- map.genotypes(mix15D.3, type = "all", pmin = 0.001)
> mp15D.3$D2S1338

   n_1_1 n_1_2 n_1_3 n_1_4 n_1_5 n_1_6 n_1_7 n_1_8 n_1_9 n_1_10 n_1_11 n_1_12
1      0     1     1     0     0     0     0     0     0      0      0      0
2      0     0     2     0     0     0     0     0     0      0      0      0
3      0     0     1     0     0     0     0     0     0      1      0      0
4      0     0     1     0     0     1     0     0     0      0      0      0
5      0     0     1     0     0     0     0     0     1      0      0      0
6      0     0     1     0     1     0     0     0     0      0      0      0
7      0     0     1     0     0     0     0     0     0      0      1      0
8      0     0     1     1     0     0     0     0     0      0      0      0
9      0     0     1     0     0     0     1     0     0      0      0      0
10     0     0     1     0     0     0     0     0     0      0      0      1
11     0     0     1     0     0     0     0     1     0      0      0      0
12     0     1     0     1     0     0     0     0     0      0      0      0
   n_1_13        Prob
1       0 0.527587167
2       0 0.169683453
3       0 0.064032228
4       0 0.052704759
5       0 0.050908355
6       0 0.041324583
7       0 0.031558539
8       0 0.030488501
9       0 0.014972203
10      0 0.010779697
11      0 0.003159506
12      0 0.001730493
```

We can summarise the output from map.genotypes to get the genotypes rather than allele counts

```
> s <- summary(mp15D.3)
> print(s, markers = "D2S1338")

D2S1338:
     U1.1   U1.2   Prob
1    16     17     0.52759
2    17     17     0.16968
3    17     24     0.06403
4    17     20     0.05270
5    17     23     0.05091
6    17     19     0.04132
7    17     25     0.03156
8    17     18     0.03049
9    17     21     0.01497
10   17     26     0.01078
11   17     22     0.00316
12   16     18     0.00173

Total probability: 0.9989
```

Due to independence between markers, the posterior probability of the most likely DNA profile is the product of probabilities for the marginally most likely genotypes. The most likely DNA profile and its posterior probability is

```
> sapply(s, function(x)x[1,])

     D16S539   D18S51    D19S433   D21S11    D2S1338   D3S1358   D8S1179
U1.1 12        12        14        28        16        15        10
U1.2 13        16        15        30        17        19        11
Prob 0.4937995 0.4607899 0.4453883 0.6420297 0.5275872 0.4423433 0.8986166
     FGA       TH01      VWA
U1.1 20        9.3       15
U1.2 23        9.3       19
Prob 0.4356195 0.5855462 0.6701804

> prod(sapply(mp15D.3, function(x)x$Prob[1]))

[1] 0.002332576
```

Similarly, we can find the probabilities of the five most likely DNA profiles.

3 Analysis of MC18

The analysis of `MC18` is completely analogous to that of `MC15`.

3.1 Four contributors

```
> mix18P.4 <- DNAmixture(list(MC18), C = list(50), k = 4,
                         K = c("K1", "K2", "K3"), database = db)
> p <- mixpar(rho = list(25), eta = list(20), xi = list(0.07),
              phi = list(c(K1 = 0.25, K2 = 0.25, K3 = 0.25, U1 = 0.25)))
```

```
> ml18P.4 <- mixML(mix18P.4, p)
> ml18P.4$mle

      rho     eta        xi     phi.U1   phi.K1    phi.K2   phi.K3
1   36.25   29.13   0.08536   0.009443   0.7057   0.09055   0.1943

> ml18P.4$lik/log(10) ## log10(L)

[1] -130.0918

> mix18D.4 <- DNAmixture(list(MC18), C = list(50), k = 4,
                         K = c("K1", "K2"), database = db)
> p[,"phi"] <- list(c(K1 = 0.25, K2 = 0.25, U1 = 0.25, U2 = 0.25))
> ml18D.4 <- mixML(mix18D.4, p)
> ml18D.4$mle

      rho     eta        xi   phi.U1    phi.U2   phi.K1    phi.K2
1   33.84   31.21   0.08469   0.1926   0.01343   0.6978   0.09617

> ml18D.4$lik/log(10) ## log10(L)

[1] -143.3619

> ## WoE
> (ml18P.4$lik - ml18D.4$lik)/log(10)

[1] 13.27014

> var18P.4 <- varEst(mix18P.4, ml18P.4$mle,
                     npars = list(rho = 1, eta = 1, xi = 1, phi = 1))
> summary(var18P.4, transform = TRUE)

              Estimate     StdErr
mu.1       1055.921129   39.33692
sigma.1       0.166102    0.01659
xi.1          0.085360    0.01562
phi.U1.1      0.009443    0.01847
phi.K1.1      0.705743    0.02205
phi.K2.1      0.090547    0.01602
phi.K3.1      0.194268    0.01820

> var18D.4 <- varEst(mix18D.4, ml18D.4$mle,
                     npars = list(rho = 1, eta = 1, xi = 1, phi = 1))
> summary(var18D.4, transform = TRUE)

             Estimate     StdErr
mu.1       1056.02050   40.71227
sigma.1       0.17192    0.01948
xi.1          0.08469    0.01702
phi.U1.1      0.19257    0.02046
phi.U2.1      0.01343    0.02119
phi.K1.1      0.69782    0.02554
phi.K2.1      0.09617    0.01830
```

3.2 Three contributors

```
> mix18P.3 <- DNAmixture(list(MC18), C = list(50), k = 3,
                         K = c("K1", "K2", "K3"), database = db)
> p[,"phi"] <- list(c(K1 = 0.82, K2 = 0.05, K3 = 0.13))
> ml18P.3 <- mixML(mix18P.3, p)
> ml18P.3$mle

      rho     eta        xi   phi.K1    phi.K2   phi.K3
1   35.77   29.49   0.08838   0.7101   0.09283   0.1971

> mix18D.3 <- DNAmixture(list(MC18), C = list(50), k = 3,
                         K = c("K1", "K2"), database = db)
> p[,"phi"] <- list(c(K1 = 0.82, K2 = 0.05, U1 = 0.13))
> ml18D.3 <- mixML(mix18D.3, p)
> ml18D.3$mle

      rho     eta        xi   phi.U1   phi.K1    phi.K2
1   33.37   31.61   0.08897   0.1963   0.7042   0.09956

> ## Weight of evidence
> (ml18P.3$lik - ml18D.3$lik)/log(10)

[1] 13.30398
```

3.3 Identification of U1 under $H_d(3)$: K1 & K2 & U1

```
> setPeakInfo(mix18D.3, ml18D.3$mle)
> mp18D.3 <- map.genotypes(mix18D.3, type = "all", pmin = 0.001)
> print(summary(mp18D.3), markers = "D2S1338")

D2S1338:
    U1.1   U1.2   Prob
1   16     17     0.988461
2   17     17     0.005299
3   17     23     0.003452
4   17     24     0.002306

Total probability: 0.9995

> ## The most probable DNA profile and its probability
> sapply(s, function(x)x[1,])

     D16S539   D18S51    D19S433   D21S11    D2S1338   D3S1358   D8S1179
U1.1 12        12        14        28        16        15        10
U1.2 13        16        15        30        17        19        11
Prob 0.4937995 0.4607899 0.4453883 0.6420297 0.5275872 0.4423433 0.8986166
     FGA       TH01      VWA
U1.1 20        9.3       15
U1.2 23        9.3       19
Prob 0.4356195 0.5855462 0.6701804
```

```
> prod(sapply(mp18D.3, function(x)x$Prob[1]))

[1] 0.1081666
```

4 Joint analysis of MC15 and MC18

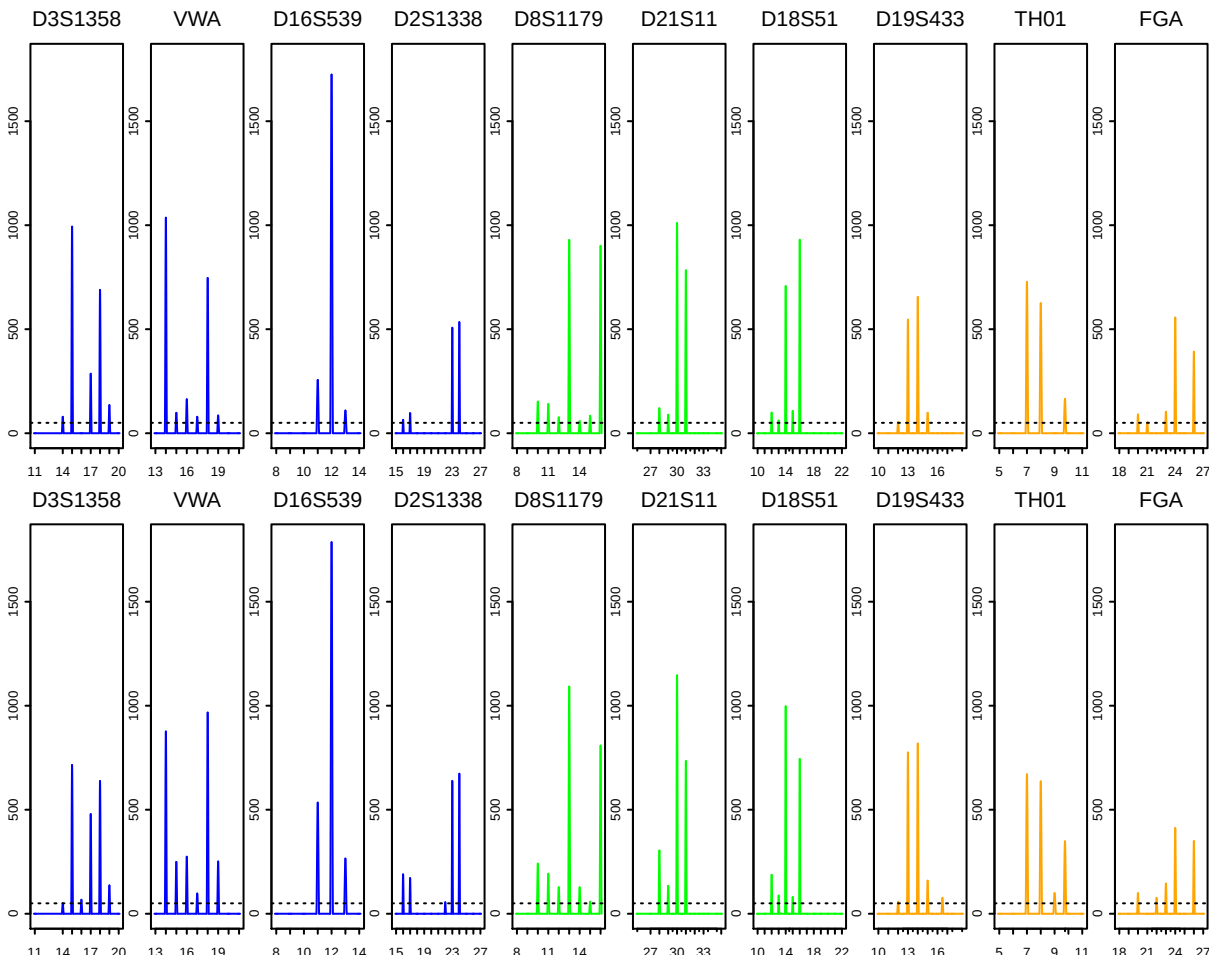
We now consider joint models for mixtures `MC15` and `MC18`. Firstly, let us see what the EPGs for the two mixtures look like.

```
> data(SGMplusDyes) ## dyes for each marker using SGMplus
> dyes <- SGMplusDyes
> dyes$green <- dyes$green[-1] ## Remove Amelogenin
> dyes

$blue
[1] "D3S1358" "VWA"     "D16S539" "D2S1338"

$green
[1] "D8S1179" "D21S11"  "D18S51"

$yellow
[1] "D19S433" "TH01"    "FGA"

> cols <- c("blue", "green", "orange") ## Define plot colors
```

We plot one row for each mixture, choosing the same order of the markers for easy marker-wise comparison.

```
> par(mfcol = c(2, 10), mar = c(1.1,1,2,0.1), mgp = c(1.3,0.2,0),
      font.main = 1, cex.main = 1, cex.axis=0.6, tcl = -0.2)
> for (d in 1:length(dyes)){
      for (m in dyes[[d]]){
          plot(mix15P.4, markers = m, col = cols[d], ylim = c(0,1800))
          plot(mix18D.4, markers = m, col = cols[d], ylim = c(0,1800))
      }
  }
```

We see that the two mixtures mostly share the observed alleles, and also that the heights of the peaks are very similar for the two EPGs.



If we consider a model in which the unknowns are different and unrelated for the two mixtures, we simply multiply the likelihoods for the models fitted separately to the two mixtures, i.e. add the log-likelihoods. The WoE is then

```
> (log10L.Hp <- (ml15P.4$lik + ml18P.4$lik)/log(10))

[1] -248.1341

> (log10L.Hd <- (ml15D.4$lik + ml18D.4$lik)/log(10))

[1] -272.6969

> log10L.Hp - log10L.Hd

[1] 24.56274
```

In the following we use common scale (`eta`) and stutter (`xi`) parameters for the two mixtures. Equality constraints are included in `mixML` by specifying the constraint in terms of a vector-valued function of a `mixpar` and a vector of values for it to take:

```
> eq.eta.xi <- function(q){
    c(q[[1,"xi"]]-q[[2,"xi"]], q[[1,"eta"]]-q[[2,"eta"]])
  }
```

Our constraint can now be phrased as `eq.eta.xi(q) == c(0,0)`.

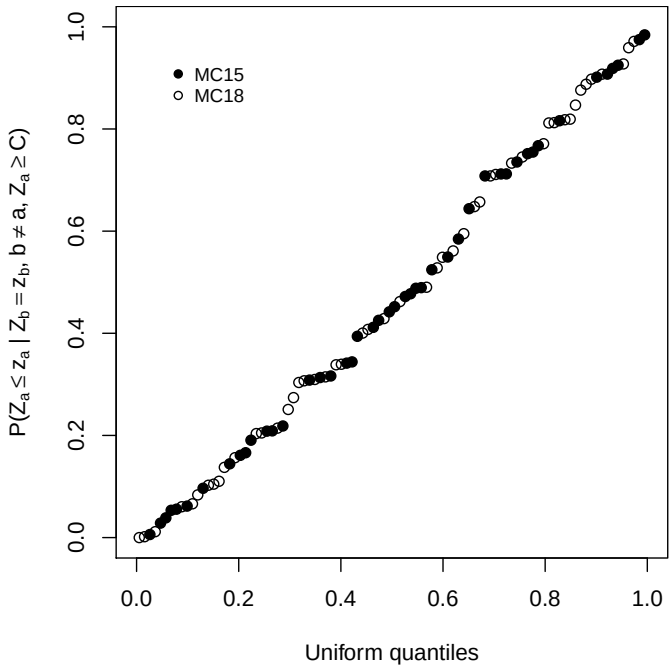
4.1 Four contributors

The prosecution hypothesis is $H_p(4)$: K1 & K2 & K3 & U1.

```
> mix1518P.4 <- DNAmixture(list(MC15, MC18), C = list(50, 50),
                           k = 4,  K = c("K1", "K2", "K3"),
                           database = db)
> p <- mixpar(rho = list(32, 37),
              eta = list(27, 27),
              xi = list(0.08, 0.08),
              phi = list(c(K1 = 0.8, K2 = 0.05, K3 = 0.1, U1 = 0.05),
                         c(K1 = 0.7, K2 = 0.09, K3 = 0.9, U1 = 0.01)))
> ml1518P.4 <- mixML(mix1518P.4, p, constr = eq.eta.xi, val = c(0,0))
> ml1518P.4$mle

      rho     eta        xi     phi.U1   phi.K1    phi.K2   phi.K3
1   32.66   27.99   0.07935   0.006043   0.8218   0.04764   0.1245
2   37.68   27.99   0.07935   0.012268   0.7045   0.08989   0.1933
```

To assess whether $H_p(4)$ explains the peak height distribution well, we can make a quantile-quantile plot.

```
> par(pty = "s", mar = c(4.5, 4.5, 0, 0))
> qq <- qqpeak(mix1518P.4, pars = ml1518P.4$mle, dist = "conditional",
               plot = FALSE)
> plot(ppoints(qq$q), qq$q, pch = ifelse(qq$trace==1, 19, 21),
       xlim = c(0,1), ylim = c(0,1), xlab = "Uniform quantiles",
       ylab = expression(paste(
           "P(",Z[a]<=z[a]," | ",Z[b]==z[b],", ",b!=a,", ",Z[a]>=C,")"
           )))
> legend(0.05, 0.95, c("MC15", "MC18"), pch = c(19, 21), bty = "n", cex = 0.8)
```



As the defence hypothesis, we consider $H_d(4)$: K1 & K2 & U1 & U2.

```
> mix1518D.4 <- DNAmixture(list(MC15, MC18), C = list(50, 50),
                           k = 4, K = c("K1", "K2"), database = db)
> p <- mixpar(rho = list(25, 35), eta = list(27, 27),
              xi = list(0.08, 0.08),
              phi = list(c(K1 = 0.82, K2 = 0.05, U1 = 0.12, U2 = 0.008),
                         c(K1 = 0.7, K2 = 0.09, U1 = 0.19, U2 = 0.001)))
> ml1518D.4 <- mixML(mix1518D.4, p, constr = eq.eta.xi, val = c(0,0))
> ml1518D.4$mle

      rho    eta        xi   phi.U1     phi.U2   phi.K1    phi.K2
1   31.74   28.8   0.07929   0.1233   0.007729   0.8202   0.04876
2   36.62   28.8   0.07929   0.1929   0.013646   0.7021   0.09135
```

4.1.1 Variance estimates

As we are analysing two mixtures, there are two parameters for each of `rho` and `phi`; this is specified through the list `npars`. Both `eta` and `xi` are assumed common for the two mixtures, so there is only a single parameter of each of these.

```
> var1518P <- varEst(mix1518P.4, ml1518P.4$mle,
                     npars = list(rho = 2, eta = 1, xi = 1, phi = 2))
> summary(var1518P, transform = TRUE)

              Estimate     StdErr
mu.1        914.218403   35.69128
mu.2       1054.689827   38.30652
sigma.1       0.174982    0.01286
sigma.2       0.162913    0.01192
xi.1          0.079346    0.01069
xi.2          0.079346    0.01069
phi.U1.1      0.006043    0.01848
phi.K1.1      0.821775    0.02033
phi.K2.1      0.047640    0.01386
phi.K3.1      0.124541    0.01567
phi.U1.2      0.012268    0.01743
phi.K1.2      0.704538    0.02150
phi.K2.2      0.089893    0.01562
phi.K3.2      0.193300    0.01772

> var1518D <- varEst(mix1518D.4, ml1518D.4$mle,
                     npars = list(rho = 2, eta = 1, xi = 1, phi = 2))
> summary(var1518D, transform = TRUE)

              Estimate     StdErr
mu.1        914.112283   36.20040
mu.2       1054.868402   38.86138
sigma.1       0.177510    0.01346
sigma.2       0.165243    0.01249
xi.1          0.079288    0.01128
xi.2          0.079288    0.01128
phi.U1.1      0.123258    0.01617
phi.U2.1      0.007729    0.01926
```

```
phi.K1.1      0.820249    0.02132
phi.K2.1      0.048764    0.01453
phi.U1.2      0.192886    0.01839
phi.U2.2      0.013646    0.01815
phi.K1.2      0.702122    0.02293
phi.K2.2      0.091346    0.01669
```

4.2 Three contributors

Fitting $H_p(3)$: K1 & K2 & K3.

```
> mix1518P.3 <- DNAmixture(list(MC15, MC18),
                           C = list(50, 50),
                           k = 3,
                           K = c("K1", "K2", "K3"),
                           database = db)
> p <- mixpar(rho = list(32, 37),
              eta = list(27, 27),
              xi = list(0.08, 0.08),
              phi = list(c(K1 = 0.8, K2 = 0.05, K3 = 0.1),
                         c(K1 = 0.7, K2 = 0.09, K3 = 0.9)))
> ml1518P.3 <- mixML(mix1518P.3, p, constr = eq.eta.xi, val = c(0,0))
> ml1518P.3$mle

      rho     eta        xi   phi.K1    phi.K2   phi.K3
1   32.23   28.35   0.08216   0.8249   0.04896   0.1262
2   37.16   28.35   0.08216   0.7103   0.09284   0.1969
```

Fitting $H_d(3)$: K1 & K2 & U1.

```
> mix1518D.3 <- DNAmixture(list(MC15, MC18),
                           C = list(50, 50),
                           k = 3,
                           K = c("K1", "K2"),
                           database = db)
> p <- mixpar(rho = list(31, 36),
              eta = list(29, 29),
              xi = list(0.082, 0.082),
              phi = list(c(K1 = 0.82, K2 = 0.05, U1 = 0.12),
                         c(K1 = 0.71, K2 = 0.09, U1 = 0.20)))
> ml1518D.3 <- mixML(mix1518D.3, p, constr = eq.eta.xi, val = c(0,0))
> ml1518D.3$mle

      rho     eta        xi   phi.U1   phi.K1    phi.K2
1   31.29   29.18   0.08257   0.1252   0.8243   0.05043
2   36.09   29.18   0.08257   0.1969   0.7086   0.09454
```

The weight of evidence against K3 is under the assumption of at most three contributors

```
> (ml1518P.3$lik - ml1518D.3$lik)/log(10)

[1] 14.10439
```

4.2.1 Test for equal mixture proportions

We wish to investigate whether the contributors have contributed the same proportion of DNA to each of the two mixtures. We therefore fit the model $H_p(3)$ under the constraint that $\phi^{\text{MC15}} = \phi^{\text{MC18}}$. To speed up the maximisation, we start the maximisation from the previously found MLE. Setting the mixture proportions to be equal in R mixtures with k contributors reduces the dimension of the parameter space by $(R-1)(k-1)$, which in our case with $R = 2$ mixtures and $k = 3$ contributors is 2.

```
> ml1518P.3.eq <- mixML(mix1518P.3, ml1518P.3$mle,
                        constr = eq.eta.xi, val = c(0,0), phi.eq = TRUE)
> (QP <- 2*(ml1518P.3$lik - ml1518P.3.eq$lik))

[1] 16.55927

> pchisq(QP, df = 2, lower.tail = FALSE)

[1] 0.0002536301
```

```
> ml1518D.3.eq <- mixML(mix1518D.3, ml1518D.3$mle,
                        constr = eq.eta.xi, val = c(0,0), phi.eq = TRUE)
> (QD <- 2*(ml1518D.3$lik - ml1518D.3.eq$lik))

[1] 16.0553

> pchisq(QD, df = 2, lower.tail = FALSE)

[1] 0.0003263135
```

4.3 Identification of U1 under $H_d(3)$: K1 & K2 & U1

```
> setPeakInfo(mix1518D.3, ml1518D.3$mle)
> mp1518D.3 <- map.genotypes(mix1518D.3, type = "all", pmin = 0.001)
> print(summary(mp1518D.3), markers = "D2S1338")

D2S1338:
    U1.1   U1.2   Prob
1   16     17     0.9997

Total probability: 0.9997

> ## Posterior most likely profile
> sapply(s, function(x)x[1,])

     D16S539   D18S51    D19S433   D21S11    D2S1338   D3S1358   D8S1179
U1.1 12        12        14        28        16        15        10
U1.2 13        16        15        30        17        19        11
Prob 0.4937995 0.4607899 0.4453883 0.6420297 0.5275872 0.4423433 0.8986166
     FGA       TH01      VWA
U1.1 20        9.3       15
U1.2 23        9.3       19
Prob 0.4356195 0.5855462 0.6701804
```

```
> prod(sapply(mp1518D.3, function(x)x$Prob[1]))

[1] 0.4358329
```

4.4 Interpretation of artefacts under $H_d(4)$: K1 & K2 & U1 & U2

We are interested in computing the posterior probabilities of an observed peak being due to stutter and of an absent peak being due to dropout given the peak height observations.

Firstly, we define a function `addY`, which can modify the networks in a `DNAmixture` to include binary auxiliary variables `Y_a`; these are `TRUE` if and only if at least one contributor possesses allele a.

```
> addY <- function(mixture){

      ## Function for setting the conditional probability tables of Y_a-s
      set.CPT.Y <- function(domain, n.unknown, n_K, Y){
          present_in_U <- c(0, rep(1, 3^n.unknown-1))
          present_in_K <- rowSums(n_K)
          one.allele <- function(a){
              ## indicator of allele presence
              present <- (present_in_U + present_in_K[a] > 0)
              ## Alternates Y_a = FALSE and Y_a = TRUE, starting with FALSE
              cptfreqs <- as.numeric(rbind(1-present, present))
              set.table(domain, Y[a], cptfreqs, type = "cpt")
          }
          sapply(seq_along(Y), one.allele)
          invisible(NULL)
      }

      ## For each network: Add nodes Y_a
      for (m in mixture$markers){
          ## Save the old elimination order for fast triangulation
          o <- names(.Call("RHugin_domain_get_elimination_order",
                           mixture$domains[[m]]))
          alleles <- seq_along(mixture$data[[m]]$allele)
          Y <- paste("Y", alleles, sep = "_")
          for (a in alleles){
              add.node(mixture$domains[[m]], Y[a], subtype = "boolean")
              ## add edges to parents n_i_a
              for (i in 1:mixture$n.unknown){
                  add.edge(mixture$domains[[m]], Y[a],
                           attr(mixture$domains[[m]], "n")[a,i])
              }
          }
          set.CPT.Y(mixture$domains[[m]], mixture$n.unknown,
                    mixture$data[[m]][,mixture$K], Y)
          ## First eliminate Y_a-s then follow the old order
          triangulate(mixture$domains[[m]], order = c(Y, o))
          compile(mixture$domains[[m]])
      }
  }
```

We add auxiliary variables `Y_a` to the networks in `mix1518D.4` and make sure that these network represent the posterior distribution given peak heights (using the MLE).

```
> addY(mix1518D.4)
> setPeakInfo(mix1518D.4, ml1518D.4$mle)
```

Next, we define a function, which computes for each marker and allele the distribution of `Y_a`.

```
> get.allele.presence <- function(mixture){
    one.marker <- function(m){
      dat <- mixture$data[[m]]
      one.allele <- function(a){
          as.numeric(get.belief(mixture$domains[[m]],
                                paste("Y", a, sep="_")))
      }
      ps <- sapply(1:nrow(dat), one.allele)
      df <- dat[,1:(mixture$ntraces+1)]
      df$Y_eq_TRUE <- ps[1,]
      df$Y_eq_FALSE <- ps[2,]
      df
    }
    out <- lapply(mixture$markers, one.marker)
    names(out) <- mixture$markers
    out
  }
```

`height1` and `height2` denote the observed peak heights for `MC15` and `MC18`. For an allele where `height > 0`, the probability that the peak is due to stutter is the probability of `Y_eq_TRUE`. For an allele where `height == 0` the probability that the allele has dropped out, is the probability of `Y_eq_FALSE`.

```
> ap <- get.allele.presence(mix1518D.4)
> ap[c("D2S1338", "TH01")]

$D2S1338
   allele height1 height2    Y_eq_TRUE  Y_eq_FALSE
1      15       0       0 9.969228e-01 0.003077173
2      16      64     189 1.940907e-04 0.999805909
3      17      96     171 2.072320e-08 0.999999979
4      18       0       0 8.439697e-01 0.156030297
5      19       0       0 7.811174e-01 0.218882564
6      20       0       0 7.257969e-01 0.274203123
7      21       0       0 9.173419e-01 0.082658104
8      22       0      55 9.265260e-01 0.073474049
9      23     507     638 0.000000e+00 1.000000000
10     24     534     673 0.000000e+00 1.000000000
11     25       0       0 8.218994e-01 0.178100595
12     26       0       0 9.400760e-01 0.059923970
13     27       0       0 9.966091e-01 0.003390897

$TH01
  allele height1 height2 Y_eq_TRUE  Y_eq_FALSE
```

```
1    5.0       0       0 0.9962503 0.003749739
2    6.0       0       0 0.6722531 0.327746888
3    7.0     727     670 0.0000000 1.000000000
4    8.0     625     636 0.0000000 1.000000000
5    9.0       0      99 0.0000000 1.000000000
6   10.0       0       0 0.9818283 0.018171699
7   11.0       0       0 0.9962252 0.003774758
8    9.3     165     348 0.0000000 1.000000000
```

5 Comparison to likeLTD

5.1 FST and sampling adjustment

To compare our analysis to that obtained using likeLTD, we change to use the database UKCaucasian as found in likeLTD. Following Balding (2013) we perform some further alterations to accommodate an F_{st}-correction as well as a sampling adjustment.

```
> data(UKCaucasian)
> ## Selecting only the markers used in MC15 and MC18
> db <- UKCaucasian[UKCaucasian$marker %in% MC15$marker,]
> db$marker <- droplevels(db$marker)
> db[db$marker == "TH01",]

    marker allele counts   frequency
121   TH01    5.0      1 0.002617801
122   TH01    6.0     77 0.201570681
123   TH01    7.0     57 0.149214660
124   TH01    8.0     46 0.120418848
125   TH01    9.0     50 0.130890052
126   TH01    9.3    151 0.395287958

> db$oldfreq <- db$frequency ## Save frequencies for comparison
```

Sampling adjustment is done by adding the alleles of K3 to the database.

```
> ## Add the alleles of K3 to the database
> db <- merge(db, subset(MC15, select = c("marker", "allele", "K3")),
              all = TRUE, by = c("marker", "allele"))
> db$K3[is.na(db$K3)] <- 0 ## NA means 0 alleles for K3 of this type
> db$newcounts <- db$counts + db$K3
> ## Normalise with total allele counts for each marker
> total <- tapply(db$newcounts, db$marker, sum)
> db$frequency <- db$newcounts/total[db$marker]
> db[db$marker == "TH01",]

   marker allele counts   frequency     oldfreq K3 newcounts
91   TH01    5.0      1 0.002604167 0.002617801  0         1
92   TH01    6.0     77 0.200520833 0.201570681  0        77
93   TH01    7.0     57 0.148437500 0.149214660  0        57
94   TH01    8.0     46 0.119791667 0.120418848  0        46
95   TH01    9.0     50 0.130208333 0.130890052  0        50
96   TH01    9.3    151 0.398437500 0.395287958  2       153
```

We also do an F_{st} correction using $\theta = 0.02$.

```
> theta <- 0.02
> db$frequency <- (1-theta)/(1+theta)*db$frequency + db$K3*theta/(1+theta)
> db[db$marker == "TH01",]

   marker allele counts   frequency     oldfreq K3 newcounts
91   TH01    5.0      1 0.002502042 0.002617801  0         1
92   TH01    6.0     77 0.192657271 0.201570681  0        77
93   TH01    7.0     57 0.142616422 0.149214660  0        57
94   TH01    8.0     46 0.115093954 0.120418848  0        46
95   TH01    9.0     50 0.125102124 0.130890052  0        50
96   TH01    9.3    151 0.422028186 0.395287958  2       153

> ## Clean up the data.frame
> db <- subset(db, select = c("marker", "allele", "frequency"))
```

Now we can simply change to use this new definition of a database when setting up a `DNAmixture`.

5.2 Three contributors and equal mixture proportions

WoE for $H_p(3)$ vs $H_d(3)$ under the restriction of common ϕ, ξ, and η for the two mixtures.

```
> mixHp <- DNAmixture(list(MC15, MC18), C = list(50,50),
                      k = 3, K = c("K1", "K2", "K3"),
                      database = db)
> mlHp <- mixML(mixHp,
                mixpar(rho = list(20,30), eta = list(30),
                       xi = list(0.07),
                       phi = list(c(K1=0.7, K2=0.1, K3=0.2))),
                constr = eq.eta.xi, val = c(0,0), phi.eq = TRUE)
> mixHd <- DNAmixture(list(MC15, MC18), C = list(50,50),
                      k = 3, K = c("K1", "K2"),
                      database = db)
> mlHd <- mixML(mixHd,
                mixpar(rho = list(20,30), eta = list(30),
                       xi = list(0.07),
                       phi = list(c(K1=0.7, K2=0.1, U1=0.2))),
                constr = eq.eta.xi, val = c(0,0), phi.eq = TRUE
                )
```

Using the peak height information, the WoE is

```
> (mlHp$lik - mlHd$lik)/log(10)

[1] 12.74489
```

In comparison, using `db = USCaucasian` we got a WoE of 14.1.

5.3 Ignoring peak heights

The WoE using estimates from peak heights, but using peak presence only as observations.

```
> (log10L.Hp <- logL(mixHp, presence.only = TRUE)(mlHp$mle)/log(10))

[1] -10.13045

> (log10L.Hd <- logL(mixHd, presence.only = TRUE)(mlHd$mle)/log(10))

[1] -20.09479

> log10L.Hp - log10L.Hd

[1] 9.964338
```

References

Balding, D. (2013). Evaluation of mixed-source, low-template DNA profiles in forensic science. *Proceedings of the National Academy of Sciences of the United States of America*, 110(30):12241–12246.

Cowell, R. G., Graversen, T., Lauritzen, S., and Mortera, J. (2014). Analysis of forensic DNA mixtures with artefacts. To appear in JRSS C.

Graversen, T. (2014). *DNAmixtures: Statistical Inference for Mixed Traces of DNA*. R-package version 0.1-3, dnamixtures.r-forge.r-project.org/.

Graversen, T. and Lauritzen, S. (2014). Computational aspects of DNA mixture analysis. *Statistics and Computing*. doi: 10.1007/s11222-014-9451-7.