polySegratioMM: An R library for Bayesian mixture models for marker dosage in autopolyploids

Peter Baker

February 25, 2026

It is well known that the dosage level of markers in autopolyploids and allopolyploids can be characterised by their observed segregation ratios. The related package `polySegratio` provides functions to allocate dosage by standard approaches and to simulate marker data sets for differing ploidies and levels of overdispersion. Note that these methods could equally well be applied to allopolyploids with specified expected segregation ratios. For details see `polySegratio`.

A Bayesian approach was proposed by Baker et al. (2010) where marker dosage estimation was obtained by fitting a finite mixture distribution.

This library calls the `JAGS` software for Bayesian calculation. `JAGS 1.0` or higher must be installed following instructions from `http://mcmc-jags.sourceforge.net/`. Note that only the most recent version is used for testing with `R`. The `JAGS` executable must be in your path. Currently, no checking is carried out to ascertain whether or not `JAGS` is set up appropriately.

To use the library, you need to attach it with

```
> library(polySegratioMM)
```

1 Simulated data

Library functions are demonstrated on a simulated data set generated using the `sim.autoMarkers` function from the `polySegratio` package.

The following `R` code can be used to generate 500 markers for 200 autohexaploid individuals exhibiting overdispersion with the parameter `shape1 = 25`. The underlying percentages of single double and triple dose markers are 70%, 20% and 10%, respectively.

```
  hexmarkers <- sim.autoMarkers(6,c(0.7,0.2,0.1),n.markers=500,n.individuals=200)

> ##<<simData, cache=true>>=
> ## simulate small autohexaploid data set of 500 markers for 200 individuals
> ## with %70 Single, %20 Double and %10 Triple Dose markers
> ## created with
> ## hexmarkers <- sim.autoMarkers(6,c(0.7,0.2,0.1),n.markers=500,n.individuals=200)
> ## save(hexmarkers, file="../../data/hexmarkers.RData")
> print(hexmarkers)
```

```
Autopolyploid dominant markers generated at Fri Jul 18 14:51:38 2008
with call:
sim.autoMarkers(ploidy.level = 6, dose.proportion = c(0.7, 0.2,
    0.1), n.markers = 500, n.individuals = 200)

Ploidy level is: 6 ( Hexaploid )
Parents were set as heterogeneous for the markers
Theoretical segregation proportions:
       ratio.SD        ratio.DD        ratio.TD    ploidy.level
          "0.5"           "0.8"          "0.95"             "6"
    ploidy.name    type.parents
    "Hexaploid" "heterogeneous"

Proportions in each dosage class:
 SD  DD  TD
0.7 0.2 0.1
No. of markers generated from multinomial distribution:
   No.markers
SD        346
DD        103
TD         51

Data were generated for 200 individuals with 500 markers
A subset is:
     X.1 X.2 X.3 X.4 X.5 X.6 X.7 X.8 X.9 X.10 r   n   ratio dose
M.1  1   0   0   1   0   1   0   1   1   1    108 200 0.54  SD
M.2  0   1   1   1   0   1   0   1   0   1    102 200 0.51  SD
M.3  1   0   0   1   0   0   0   1   1   1    103 200 0.515 SD
M.4  1   1   1   1   1   0   0   0   1   0    97  200 0.485 SD
M.5  0   0   1   1   0   0   1   1   0   1    99  200 0.495 SD
M.6  1   1   0   0   0   0   0   0   0   0    103 200 0.515 SD
M.7  1   1   0   0   0   1   0   0   1   1    101 200 0.505 SD
M.8  1   1   1   1   1   1   0   0   0   1    102 200 0.51  SD
M.9  0   1   1   1   1   0   0   1   1   0    110 200 0.55  SD
M.10 1   0   1   0   1   1   1   0   1   1    108 200 0.54  SD
```

Note that the segregation ratios for simulated or real data may be extracted by using `segregationRatios` which sets up the appropriate objects for testing marker dosage and plotting or summarising the marker data.

```
> sr <-  segregationRatios(hexmarkers$markers)
```

For instance, as seen in Figure 1, segregation ratios may be plotted with

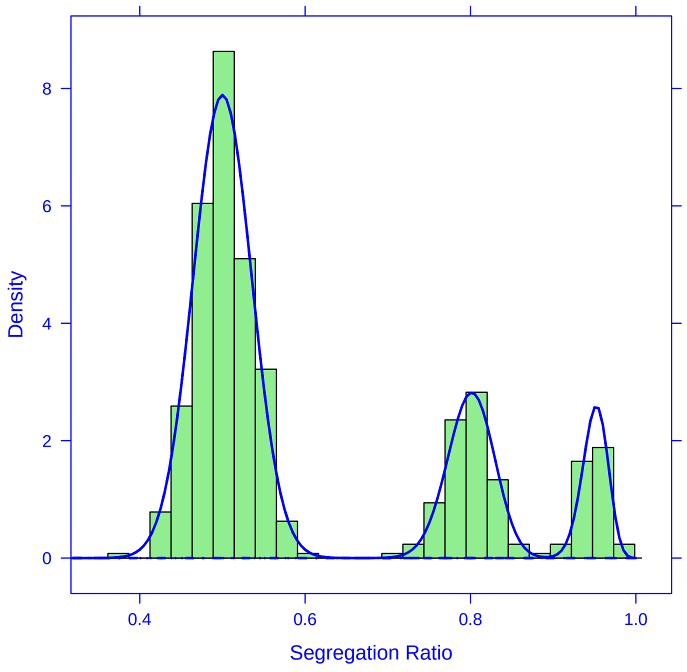

Figure 1: Segregation ratios of 500 simulated markers from 200 autohexaploid individuals. Percentages of single double and triple dose markers are 70%, 20% and 10%, respectively. Data were generated assuming no overdispersion.

```
plotTheoretical(ploidy.level=6, seg.ratios=sr,
  expected.segratio=NULL, proportions=c(0.7,0.2,0.1),
  n.individuals=200)
```

On the other hand, consider a similar data set that exhibits overdispersion. This may be simulated as follows

```
hexmarkers.overdisp <- sim.autoMarkers(6,c(0.7,0.2,0.1),n.markers=500,n.individuals=200,
                                    overdispersion=TRUE, shape1=30)
```

```
> sr.overdisp <-  segregationRatios(hexmarkers.overdisp$markers)
```

The histogram of marker segregation ratios, which is a useful graphical method for identifying overdispersion or outliers, is seen in Figure 2. Note that, due to overdispersion the theoretical distribution is narrower than the observed data.

2 A Bayesian mixture model approach

For the j^{th} marker $j = 1 \ldots n$, we assume the observed number r_j of dominant markers out of N_j lines follows a binomial distribution denoted $Bin(N_j, Pk)$.

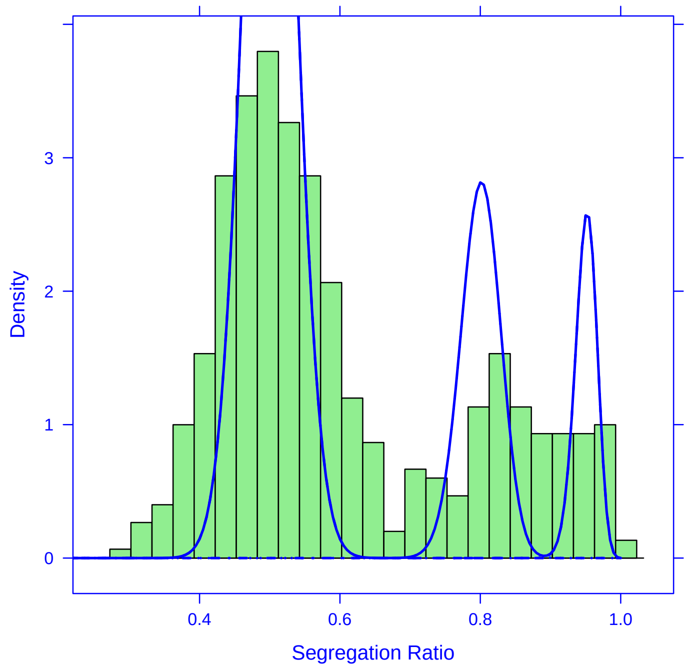

Figure 2: Segregation ratios of 500 simulated markers from 200 autohexaploid individuals. Percentages of single double and triple dose markers are 70%, 20% and 10%, respectively. Data were generated from the Beta–Binomial distribution assuming a shape parameter `shape1` of 30.

If we knew the dosage k then, following Ripol et al. (1999), the expected value of P_k may be written as

$$P_k(k|m,x) = 1 - \frac{\binom{m-k}{mx}}{\binom{m}{mx}}, k = 0 \ldots m/2 \tag{1}$$

where m is the ploidy level or number of homologous chromosomes and the monoploid number x is the number of chromosomes in a basic set. Note that for diploids $m = 2$, tetraploids $m = 4$, octaploids then $m = 8$ and so on and also that if there are no marker data missing then N_j is simply the number of progeny.

Since the dosage of each marker is unknown, we rely on the missing data representation of Dempster et al. (1977) and Tanner and Wong (1987) which is commonly adopted for MCMC computation in finite mixture models. An indicator variable z_j corresponding to unknown marker dosage class k is introduced where $z_j = k$ if the marker has dose k. For the K components with $K \leq m/2$, consider the logit transformation of the true segregation proportions P_k for dose $k, k = 1 \ldots K$. The the logit transformed segregation ratio ω_k

is then

$$\omega_k = \log(\frac{P_k}{1-P_k}). \tag{2}$$

Let $z = (z_1 \dots z_n)^T$ be a vector of unknown dosages (labelled $1, 2 \dots K$ corresponding to simplex, duplex, triplex markers and so on), then r_j is binomially distributed with known size parameter N_j and unknown proportion parameter ω_{Z_j} which is the segregation ratio for marker dosage z_j. Hence, given marker dosage z_j then

$$\begin{aligned} r_j|z_j &\sim Bin\left(N_j, \omega_{Z_j}\right), \\ \text{where} & \\ \text{logit}(\omega_{Z_j}) &= \log(\frac{\omega_{Z_j}}{1-\omega_{Z_j}}) \sim N(\mu_{Z_j}, \tau_{Z_j}^{-1}) \end{aligned} \tag{3}$$

where μ_k and τ_k are the mean and precision ($\tau_k = 1/\sigma_k^2$) of marker dosage class k on the logit scale.

Since the dosage is unknown, for the autohexaploid data generated here then for the logit(ω_{z_k}) can be modelled as a finite mixture of 3 normals

$$\text{logit}(\omega_{Z_j}) \sim \pi_1 N(\mu_1, \tau_1^{-1}) + \pi_2 N(\mu_2, \tau_2^{-1}) + \ldots + \pi_K N(\mu_K, \tau_K^{-1}) \tag{4}$$

where μ_k is the mean and τ_k is the precision of component k on the logit scale, and π_k are the mixing proportions of the three components with $\sum_{k=1}^{K} \pi_k = 1$. The probability density function $f(x)$ of logit(ω_k) is

$$f(x) = \sum_{k=1}^{K} \pi_k \phi(x|\mu_k, \tau_k^{-1}) \tag{5}$$

where ϕ is the normal cumulative distribution function with parameters mean μ_k and variance $\sigma_k^2 = \tau_k^{-1}$.

Simulation studies suggested that incorporating strong prior information, such as the expected distributions of Haldane (1930) provided the best method of allocating dosage. Further details may be found in Baker et al. (2010).

3 Specifying a model

A mixture model may be set up with `setModel`. By default, only two parameters are required, namely the `ploidy.level` or the number of homologous chromosomes set either as a numeric or as a character string and also `n.components` or the number of components for mixture model (less than or equal to maximum number of possible dosages). By default, strong priors are set by using the formulae of Haldane (1930) for the expected numbers and ratios of offspring for various parental configurations of autopolyploids.

For the autohexaploid data generated above, the models are set with

```
> x.mod1 <- setModel(3,6)  # autohexaploid model with 3 components
```

The R object x.mod1 contains components describing aspects of the model such as the number of components, ploidy, expected segregation ratios and so on. Note that the str command is useful for displaying the internal structure of any R object.

4 Fitting a mixture model

While various options are available for fine tuning the MCMC process, the simplest way to fit a mixture model to allocate marker dosages is with the wrapper function runSegratioMM as follows:

```
mcmcHexRun <- runSegratioMM(sr.overdisp, x.mod1)
```

which automatically determines starting values, priors, length of burn in, number of iterations, and other parameters as well as producing summary statistics and diagnostic plots.

To run JAGS without producing plots then set the plots option to FALSE. For the overdispersed data running this command produced the following selected output. While selected output is printed here the simple command print(mcmcHexRun) whould produce the following output and more.

The summary of processing times:

```
> print(mcmcHexRun$run.jags)

CMD File: test.cmd
JAGS started at Fri Jul 18 14:51:41 2008
JAGS run completed successfully at Fri Jul 18 14:55:47 2008
Elapsed times:
   user  system elapsed
  228.0   228.0   246.4
```

And summary statistics for the posterior distributions of selected parameters:

```
> print(mcmcHexRun$summary)

$statistics
         Mean      SD  Naive SE Time-series SE
P[1]  0.72356 0.02064 0.0002919      0.0003616
P[2]  0.19709 0.01906 0.0002695      0.0005650
P[3]  0.07935 0.01336 0.0001890      0.0006636
mu[1] 0.02381 0.01696 0.0002398      0.0006163
mu[2] 1.54959 0.04421 0.0006253      0.0031491
mu[3] 3.16008 0.08879 0.0012557      0.0074792
sigma 0.27012 0.01215 0.0001718      0.0005203

$quantiles
           2.5%     25%     50%     75%   97.5%
P[1]   0.681698 0.70954 0.72439 0.73818 0.76208
```

```
P[2]   0.161445 0.18404 0.19643 0.20946 0.23651
P[3]   0.054776 0.07029 0.07876 0.08805 0.10753
mu[1] -0.008683 0.01177 0.02381 0.03529 0.05805
mu[2]  1.464629 1.51904 1.54875 1.57957 1.63693
mu[3]  2.995540 3.09972 3.15792 3.21625 3.34191
sigma  0.247668 0.26165 0.26998 0.27826 0.29443

$start
[1] 0

$end
[1] 4999

$thin
[1] 1

$nchain
[1] 1

attr(,"class")
[1] "summarySegratioMCMC"
```

Note that MCMC convergence diagnostic output is produced automatically. Assessing convergence is crucial in MCMC and poor convergence may result in mis–allocated marker dosages. The diagnostic statistics indicate that convergence was achieved.

```
> print(mcmcHexRun$diagnostics)
```

```
$raftery
$raftery[[1]]

Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

       Burn-in  Total Lower bound  Dependence
       (M)      (N)   (Nmin)       factor (I)
 P[1]  2        3803  3746         1.020
 P[2]  2        3930  3746         1.050
 P[3]  2        3680  3746         0.982
 mu[1] 4        4713  3746         1.260
 mu[2] 10       11010 3746         2.940
 mu[3] 18       19611 3746         5.240
 sigma 10       10754 3746         2.870


$geweke
```

```
$geweke[[1]]

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

   P[1]    P[2]    P[3]   mu[1]   mu[2]   mu[3]   sigma
 1.3314 -1.9552  1.4159  1.7912 -0.7923 -0.9007 -1.0033




$heidel
$heidel[[1]]

      Stationarity start     p-value
      test         iteration
P[1]  passed       1         0.348
P[2]  passed       1         0.904
P[3]  passed       1         0.547
mu[1] passed       1         0.387
mu[2] passed       1         0.465
mu[3] passed       1         0.374
sigma passed       1         0.913

      Halfwidth Mean   Halfwidth
      test
P[1]  passed    0.7236 0.000709
P[2]  passed    0.1971 0.001107
P[3]  passed    0.0794 0.001301
mu[1] passed    0.0238 0.001208
mu[2] passed    1.5496 0.006172
mu[3] passed    3.1601 0.014659
sigma passed    0.2701 0.001020


$hpd
$hpd[[1]]
         lower   upper
P[1]   0.68391 0.76365
P[2]   0.15950 0.23382
P[3]   0.05438 0.10695
mu[1] -0.01018 0.05593
mu[2]  1.46539 1.63737
mu[3]  2.99198 3.33813
sigma  0.24697 0.29359
attr(,"Probability")
[1] 0.95
```

And finally, summaries of marker dosage allocations are produced:

```
> print(mcmcHexRun$doses)

Dosages for chain: 1
Thresholds set at:
[1] 0.50 0.60 0.70 0.80 0.90 0.95 0.99
A random sample of posterior probabilities and classifications
          SD     DD     TD 0.5 0.6 0.7 0.8 0.9 0.95 0.99 maxPostP
M.13  1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.39  1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.67  1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.123 1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.131 1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.142 1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.208 1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.262 0.9998 0.0002 0.0000   1   1   1   1   1    1    1        1
M.294 1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.307 1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.310 1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.311 1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.333 1.0000 0.0000 0.0000   1   1   1   1   1    1    1        1
M.399 0.0000 1.0000 0.0000   2   2   2   2   2    2    2        2
M.414 0.0008 0.9992 0.0000   2   2   2   2   2    2    2        2
M.427 0.0008 0.9992 0.0000   2   2   2   2   2    2    2        2
M.442 0.0000 0.8470 0.1530   2   2   2   2   .    .    .        2
M.463 0.0000 0.0000 1.0000   3   3   3   3   3    3    3        3
M.480 0.0000 0.0016 0.9984   3   3   3   3   3    3    3        3
M.481 0.0000 0.0002 0.9998   3   3   3   3   3    3    3        3

Maximum posterior probabilities for 500 markers
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.502   1.000   1.000   0.980   1.000   1.000

Proportion of genes classified using maximum posterior probability
   SD    DD    TD
0.724 0.194 0.082
Total proportion of markers classified: 1
Call:
dosagesJagsMix(mcmc.mixture = read.jags, jags.control = jags.control,
    seg.ratio = seg.ratios)
```

Note that simply plotting `mcmcHexRun` will produce a histogram of segregation proportions and the fitted model but that other plots are easily produced.

When the `plots` option of `runSegratioMM` is set to the default value of `TRUE`, numerous plots are produced including trace and density plots from the `CODA` package. These may also be extracted manually but the process is somewhat more complicated. For instance to obtain trace and density plots for the parameters p_1, μ_1, σ_1 and for the 140^{th} marker, as shown in Figure 3, then `CODA` may be used directly by following command.

```
NULL
```

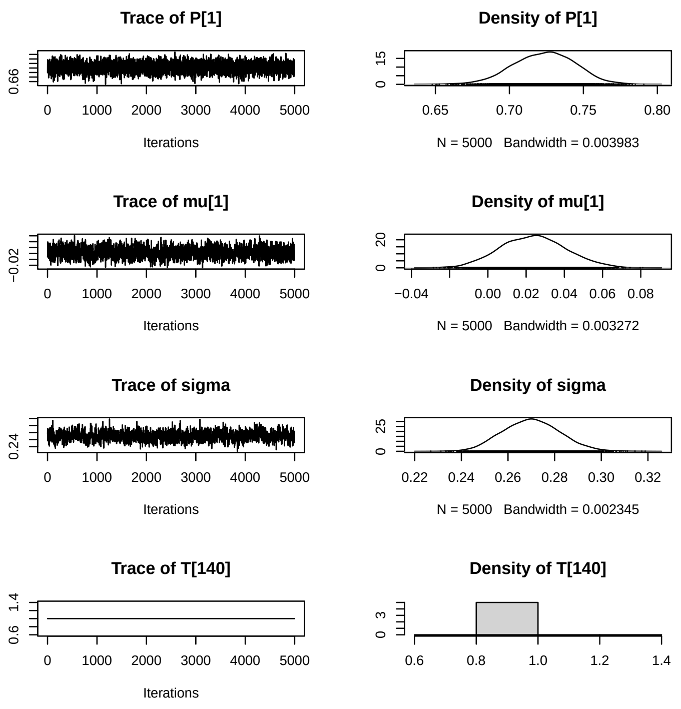

Figure 3: Trace and posterior density plots for the parameters parameters p_1, μ_1, σ_1 and for the 140th marker for the overdispersed data.

```
plot(mcmcHexRun$mcmc.mixture$mcmc.list[[1]][,c("P[1]","mu[1]","sigma","T[140]")])
```

The histogram of segregation proportions with fitted and theoretical values shown in Figure 4 may be obtained by setting the `theoretical` option to `TRUE` as follows.

```
print(plot(mcmcHexRun, theoretical=TRUE))
```

5 Assigning marker dosage

Marker dosages allocations may be obtained directly from the object `mcmcHexRun`. The dosage with maximum posterior probability is simply `mcmcHexRun$doses$max.post.dosage`. A more conservative allocation is obtained by using `mcmcHexRun$doses$dosage[,"0.8"]` whereby the dosage with posterior probability over 0.8 is employed. For instance, to tabulate the number of markers (including those not allocated a dosage which are labelled NA) the `table` command can be employed.

```
> cat("Employing maximum posterior probability\n")

Employing maximum posterior probability
```

```
Warning: component proportions normalised, now:
   P[1]    P[2]    P[3]
0.72356 0.19709 0.07935
PlotTheoretical Warning: Binomial mixture density is only approximate on logit scale
```

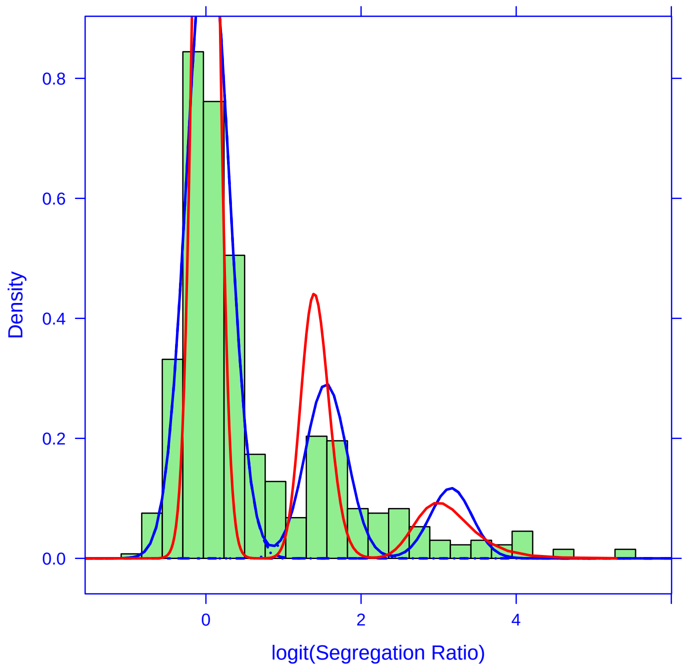

Figure 4: Fitted (blue) and theoretical (red) distributions for simulated segregation ratios with overdispersion for 500 markers from 200 individuals.

```
> table(Dose=mcmcHexRun$doses$max.post.dosage, exclude=NULL)

Dose
  1   2   3
362  97  41

> cat("Employing posterior probability > 0.8\n")

Employing posterior probability > 0.8

> table(Dose=mcmcHexRun$doses$dosage[,"0.8"], exclude=NULL)

Dose
   1    2    3 <NA>
 358   89   34   19
```

And of course since the data were simulated we can compare the estimated and true dosages obtained as `hexmarkers.overdisp$true.doses$dosage` via cross tabulation. Doses can also be obtained for the standard χ^2 test by using the `test.segRatio` command from the `polySegratio` library.

```
> cat("Employing theChi squared test\n")

Employing theChi squared test

> dose.chi <- test.segRatio(sr.overdisp, ploidy.level = 6)
> table(Chi2Dose=dose.chi$dosage, True=hexmarkers.overdisp$true.doses$dosage, exclude=NULL

        True
Chi2Dose   1   2   3
    1    223   2   0
    2      0  54   5
    3      0   3  27
    <NA> 130  39  17

> cat("Employing maximum posterior probability\n")

Employing maximum posterior probability

> table(MixtureDose=mcmcHexRun$doses$max.post.dosage, True=hexmarkers.overdisp$true.doses$
+ exclude=NULL)

           True
MixtureDose   1   2   3
          1 353   9   0
          2   0  86  11
          3   0   3  38

> cat("Employing posterior probability > 0.8\n")

Employing posterior probability > 0.8

> table(MixtureDose=mcmcHexRun$doses$dosage[,"0.8"], True=hexmarkers.overdisp$true.doses$d
+ exclude=NULL)

           True
MixtureDose   1   2   3
       1    352   6   0
       2      0  78  11
       3      0   2  32
       <NA>   1  12   6
```

These tables show that far fewer markers are allocated a dosage using the standard χ^2 test than by the mixture model. Fewer markers were misclassified using a posterior probability threshold of 0.8 rather than the maximum posterior probability as a basis for allocating dosage.

References

Baker, P., Jackson, P., and Aitken, K. (2010). Bayesian estimation of marker dosage in sugarcane and other autopolyploids. *TAG Theoretical and Applied Genetics*, 120(8):1653–1672.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B*, 39:1–38.

Haldane, J. B. S. (1930). Theoretical genetics of autopolyploids. *Journal of Genetics*, 22:359–372.

Ripol, M. I., Churchill, G. A., da Silva, J. A., and Sorrells, M. (1999). Statistical aspects of genetic mapping in autopolyploids. *Gene*, 235(1-2):31–41.

Tanner, M. A. and Wong, W. H. (1987). The calculation of posterior distributions by data augmentation: with discussion. *Journal of the American Statistical Association*, 82:528–550.

5.1 Acknowledgments

Karen Aitken, given her experience in tetraploids and sugarcane marker maps, has provided many valuable insights into marker dosage in autopolyploids. Additionally, Ross Darnell, Andrew George and Kerrie Mengersen provided useful comments and discussions.