# MasterBayes: Maximum Likelihood and Markov chain Monte Carlo methods for pedigree reconstruction, analysis and simulation.

**J. D. Hadfield**

September 28, 2006

# Contents

This document provides worked examples for several types of model that can be fitted using MasterBayes. The main emphasis is on the syntax used to specify, estimate and interpret the models, although in some cases I have tried to explain the underlying theory. These explanations are intended for non-statisticians and vary from the patronising to the arcane. I assume a basic familiarity with R.

```
> library("MasterBayes", lib.loc = "~/My_Programs")

Loading required package: coda
Loading required package: genetics
Loading required package: combinat
Loading required package: gdata
Loading required package: gtools
Loading required package: MASS

Attaching package: 'genetics'


        The following object(s) are masked from package:MASS :

         genotype


        The following object(s) are masked from package:base :

         as.factor

Loading required package: gtools
```

Many of the models that can be fitted are computationally intensive, and many routines have been written in compiled C++ code for efficiency. Nethertheless, the amount of computing time required by some models may be a limiting factor, and I incude model run times for many examples. These examples were run on a dual Xeon 2.8GHz Linux machine with 1Gb RAM. In general the length of the Markov chain is much shorter that what would be used in a real analysis. Generally, these examples can be fitted in a few minutes, even seconds, but for real problems I would suggest running multiple chains for as long as possible.

# 1 Markov Chain Monte Carlo

In order to fit and interpret models successfully in MasterBayes it will be necessary to have a baisc understanding of Marvov chain Monte Carlo (MCMC) methods. I will giva a non-technical, heuristic tour of MCMC that should give an operational understanding of MasterBayes, and in particular the function `MCMCped`. I strongly recommend reading one of the many good introductary

texts on the subject; my favourite is Bayesian Data Analysis [**?**].

Let's imagine a model in which there are two parameters of interest, the probability that territorial males gain paternity over non-territorial males, and also the probability that old males gain paternity over young males. We will denote the two parameters as $\beta_1$ and $\beta_2$, and group them in the vector $\boldsymbol{\beta}$. We are interested in the joint posterior distribution of these parameters conditional on the parentage, spatial and age data we have collected ($\boldsymbol{y}$):

$$Pr(\beta_1, \beta_2 | \boldsymbol{y}) \tag{1}$$

The posterior probability distribution is a complete description of our state of knowledge of the true value of $\boldsymbol{\beta}$. The posterior is the product of two types of information, information from the data we have collected (the likelihood) and information that we have gained from prior experience (the prior).

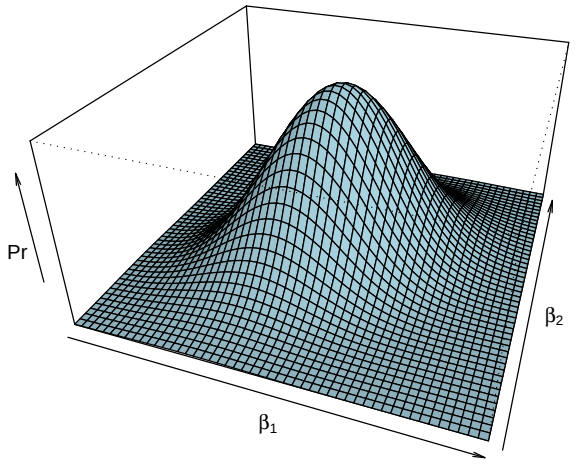


Figure 1: Bivariate posterior distribution of $\boldsymbol{\beta}$

For some very simple models the posterior distribution can be derived analytically. Let's imagine that the equation describing the joint posterior distribution of $\beta_1$ and $\beta_2$ was known and could be plotted (see Figure 1). The area under the

distribution is equal to 1, since we are dealing with a probability, and we can often summarise the posterior distribution using simple statistics. For example we could state the values of $\beta_1$ and $\beta_2$ for the peak of the distribution, which would represent the most likely values of $\boldsymbol{\beta}$. We could state the width of the distribution along the $\beta_1$ axis as a measure of the precision with which we have estimated $\beta_1$, and so on.

For most models, however, we cannot derive the posterior distribution analytically, and we must use MCMC to get an approximation. MCMC relies on the fact that although we cannot derive the complete posterior we can calculate the height of the posterior distribution at a particular set of co-ordinates. In this example there are only two parameters so we may be inclined to systematically go through every likely combination of $\beta_1$ and $\beta_2$ and calculate the height of the distribution at regular distances, and then plot Figure 1. The Markov chain does exactly this, although it does not move systematically through parameter space (the $\beta_1$ and $\beta_2$ co-ordinates in this case), it moves stochastically, hence the name 'Monte Carlo'. There are several ways in which we can get the chain to move in parameter space, and MasterBayes uses a combination of Metropolis-Hastings updates and Gibbs sampling.

## 1.1 Metropolis-Hastings Updates and `tunePed` Objects

First we need to initialise the chain and specify a set of co-ordinates from which the chain can start a journey through parameter space. Ideally we would like to pick a region of high probability, as we do not want to waste time wandering through regions of low probability: we are not so interested in determining the height of the distribution outside of Figure 1 as it is virtually flat and close to zero (or at least we hope so!). MasterBayes uses a mixture of Maximum Likelihood and heuristic techniques to detemine the place from which the chain is launched, the co-ordinates of the starting configuration are denoted $\boldsymbol{\beta}^{t=0}$.

After initialising the chain we need to decide where to go next, and this decision is based on two rules. First we have to generate a candidate destination, and then we need to decide whether to go there or stay where we are. There are many ways in which we could generate the candidate coordinates, and MasterBayes uses a well tested and simple method. A random set of cordinates are picked from a multivariate normal distribution that is centered on the initial co-ordinates $\boldsymbol{\beta}^{t=0}$, and has a user specified variance. We will denote this new set of cordinates as $\boldsymbol{\beta}^{new}$. The question then remains whether to move to this new set of cordinates or remain at our current co-ordinates $\boldsymbol{\beta}^{t=0} = \boldsymbol{\beta}^{old}$. If the height of the distribution at the new set of co-ordinates is greater, then the chain moves from $\boldsymbol{\beta}^{old}$ to $\boldsymbol{\beta}^{new}$. If the new set of coordinates is in a region of low probability then the chain may move there, but not all the time. The probability that the chain moves to low lying areas, is determined by the realtive difference bewteen the heights of the posterior distribution at the two co-ordinates. If the height

of the distribution at $\boldsymbol{\beta}^{new}$ is 5 times less than the height at $\boldsymbol{\beta}^{old}$, then the chain would move to the new set of coordinates 1 in 5 times, and $\boldsymbol{\beta}^{t=1}$ would become $\boldsymbol{\beta}^{new}$. Using these rules we can record where the chain has travelled and generate an approximate posterior distribution. Basically, a histogram of Figure 1.

The speed with which the chain moves through parameter space is critical, and in part depends on the variance of the proposal distribution. Say we initialised the chain at the co-ordinates under the peak of the distribution in Figure **??** and specified the variance of the proposal distribution to be large (much larger than the variance of the posterior distribution itself). Many of the candidate co-ordinates would lie outside of Figure **??** in regions of very low probability, and since the probability of moving there is low the chain would sit at the peak for large amounts of time. We would have to run the chain for many itterations before we could generate a histogram that was an adequate approxiamtion to Figure 1. Alternatively, we could specify the variance to be very small, say a thousand times smaller than the variance of the posterior. In this case the cordintates $\boldsymbol{\beta}^{new}$ and $\boldsymbol{\beta}^{old}$ would be very close, and the chain would move almost every itteration since the difference in heights of the posterior at the two cordintates would be tiny. Although the chain is continuously moving it is travelling such small distances each itteration that the chain would have to pass through many itterations before it adequately covered parameter space.

`tunePed` objects control the variance of the proposal distributions for parameters updated by Metropolis-Hastings upadtes. By default, the standard deviation of the proposal distribution will be the standard error of the parameter estimated using Maximum Likelihood. Scaling constants can then be passed to `tunePed` which are multiplied by the standard error squared to obtain the variance of the proposal distribution. Ideally, the chain should move between 20% and 50% of the time, and this can be assesd by specifying `verbose=TRUE` in `MCMCped`. The Metroplis-Hastings accpetance rates are then printed to the screen during model fitting.

## 1.2 Gibbs Sampling

Gibbs sampling is a special case of Metropolis-Hastings updates, and `MCMCped` uses Gibbs sampling to sample genotypes and parents. In the Metropolis-Hastings example above, the Markov Chain was allowed to move in both directions of parameter space simulateously. An equally valid approach would have been to set up two Metropolis-hastings schemes where the chain was first allowed to move along the $\beta_1$ axis, and then along the $\beta_2$ axis. In Figure 2 we have cut the posterior distribution of Figure 1 in half, and the edge of the surface facing us is the conditional distribution of $\beta_1$ given that $\beta_2 = 0$:

$$Pr(\beta_1|\beta_2 = 0, \boldsymbol{y}). \tag{2}$$

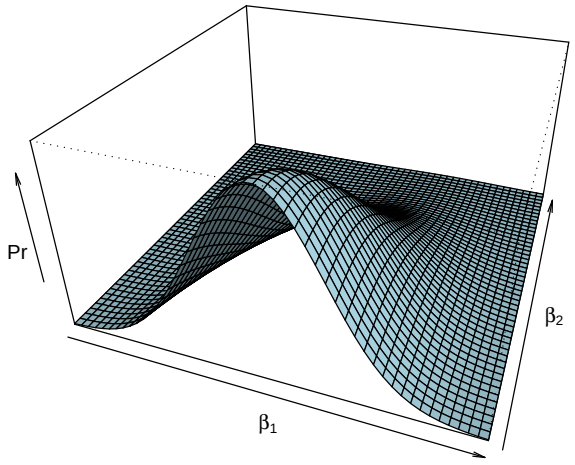


Figure 2: Bivariate posterior distribution of $\boldsymbol{\beta}$, for postive values of $\beta_2$

In some cases, the equation that describes this conditional distribution can be derived despite the equation for the complete joint distribution of Figure **??** remaining unknown. When the conditional distribution of $\beta_1$ is known we can use Gibbs sampling. Lets say the chain at a particular iteration is located at zero for $\beta_2$. If we updated $\beta_1$ using a Metroplis-Hastings algorithm we would generate a candidate value and evaluate its relative probability compared to the old value. This procedure would take place in the slice of posterior that is facing us in Figure 2. However, because we know the actual equation for this slice we can just generate a new value of $\beta_1$ directly. This is Gibbs sampling. If for example, the slice of the posterior that we can see in Figure 2 has a normal distribution with mean of zero and variance of one, then $\beta_1^{new}$ can simply be drawn directly from this distribution. This can be much more efficient than Metropolis-Hastings updates, and avoids the issue of having to specigy the variance of a proposal distribution.

## 1.3 MCMC Diagnostics

When fitting a model using `MCMCped` the parameter values through which the Markov chain has travelled are stored and returned. The length of the chain (the number of iterations) can be specified using the `nitt` argument of `MCMCped`, and should be long enough so that the posterior approximation is valid. If we had known the joint posterior distribution in Figure 1 we could have set up a Markov chain that sampled directly from the posterior. If this had been the case each successive value in the Markov chain would be independent of the previous value after conditioning on the data, $\boldsymbol{y}$, and a thousand iterations of the chain would have produced a histogram that resembled Figure 1 very closely. However, generally we do not know the joint posterior distribution of the parameters, and we use Gibbs sampling and Metropolis-Hastings updates to approximate this distribution. For this reason the parameter values of the Markov chain at sucessive itterations are usually not idependent and care needs to be taken regarding the validity of the approximation. `MCMCped` returns the Markov chain for continous variables as `mcmc` objects, which can be analysed using the `coda` pacakge. The function `autocorr` estimates the level of non-independence between succesive samples in the chain. When autocorrelation is high the chain needs to be run for longer, and this can lead to storage problems for high dimensional problems. The argument `thin` can be passed to `MCMCped` specifying the intervals at which the Markov chain is stored.

The approximation obtained from the Markov chain is conditional on the set of parameter values that was used to initialise the chain. In many cases the first itterations show a strong dependence on the starting parameteristaion, but as the chain progresses this dependence may be lost. As the dependence on the starting parameterisation diminishes the chain is said to converge and the argument `burnin` can be passed to `MCMCped` specifying the number of itterations which must pass before samples are stored. Assessing convergence of the chain is notoriously difficult. The posterior distribution in Figure 1 has a simple form and the convergence of the chain would be easy to acheive. If however, the posterior was multimodal convergence would be harder to acheive and diagnose.

For example, imagine a chain was initailised at $C_1$ in Figure 3. The chain may provide valid approximates for the region of high probablity at samll values of $\beta_1$ and $\beta_2$ but it may fail to pass under the low probability ridge that connects the two high probability regions. Conclusions drawn from a chain initialised a $C_2$ may be very different, and it is good practice to run multiple chains from different starting parametrisations (See Section 3.10 for an interesting example). For `MCMCped` models in which the posterior distribution of genotypes is estimated it is particularly important to assess the sensitivity of the chain to different starting configurations. Genotype configurations with high probability may be separated by configurations of low probability and the chain may mix poorly. See Section 3.1 for an example where high probability genotype configurations are actually separated by configurations of zero probability. In such cases the

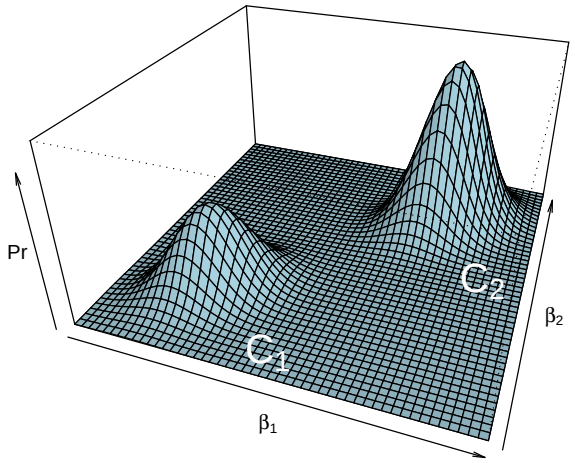


Figure 3: A bimodal bivariate posterior distribution of $\boldsymbol{\beta}$.

chain is said to be reducible and posterior simulation is not possible.

## 1.4 Prior Specifications and `priorPed` Objects

The posterior distribution is the product of the likelihood and the prior. If a prior is not specified using the function `priorPed`, the default is to use an improper uniform prior for all parameters, except allele frequencies. For the size of the unsampled population and the genotyping error rates the prior has zero probability for negative values. When the posterior distribution itself is improper (i.e if the volume under the surfaces plotted in Figures 1 and 3 are not finite) then the posterior can no longer be treated as a probability, and inferences taken from it woud be compromised. For many models the data will contain enough information to make the posterior proper despite an improper prior specification. However, for models in which parameters are confounded a proper prior distribution is required to make the parameters idenitifiable. Imagine that all territorial males were old, and all non-territorial males were young. If we tried to fit the two parameters $\beta_1$ and $\beta_2$ in a single model we would run into difficulties. The parameters are not identifiable because we cannot

distinguish between the two alternatives; whether territorial males gain more paternity beacuse they hold a territory, or because they older. The likelihood surface for $\boldsymbol{\beta}$ would look something like Figure 4, with a ridge of high probability extending to infinity.

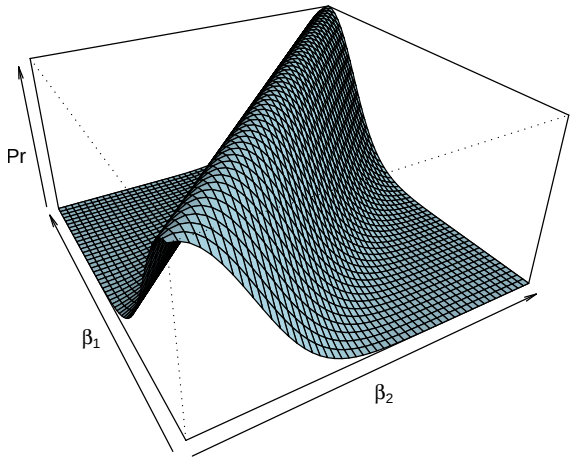


Figure 4: The likelihood of $\boldsymbol{\beta}$, when the two parameters are not identifiable from the data, $\boldsymbol{y}$

In this insatnce the Markov chain would wander aimlessly along the ridge, perhaps even giving the appearance of having converged. MasterBayes will not check whether all parameters are idenitifiable.

## 2 A Worked Example: The Seychelles Warbler

To illustrate the basic methodology, I will use data collected in 1999 from the Cousin Island population of the endangered Seychelles warbler [**?**].

```
> data(WarblerP)
> data(WarblerG)
```

`WarblerP` is a data frame containing phenotypic data for all individuals recorded in that year. This includes a unique identifier for each bird, its sex and the territory it was recorded on. `offspring` is a binary vector with 1 indicdating the bird was born in 1999, and `status` is a factorial vector with 2 levels indidcating whether the bird is a dominant or a subordinate. `lat` and `long` are x and y coordinates for the centre of each bird's territory. WarblerG is a data frame containing the genotypes of each individual at 14 loci.

In this particular year there were 59 offspring, 127 adult females and 121 adult males recorded on the island. We will assume for now that an offspring's mother is always found on the same territory as the offspring, but that an offspring's father could potentially be any one of the 121 sampled males, or even a male that was not sampled. Initially, we will focus on modelling the degree to which offspring and fathers are associated spatially, by estimating the rate at which the probability of paternity drops with distance from the offspring. This rate ($\beta$) is the exponential rate parameter, and we are interested in its probability distribution given the data we have collected:

$$Pr(\beta|\boldsymbol{G}_{obs}, \boldsymbol{l}at, \boldsymbol{l}ong, \boldsymbol{t}err). \tag{3}$$

$\boldsymbol{G}$ represents genotypes, and we use the subscript $_{obs}$ to indicate observed rather than actual genotypes. This distinction is necessary when genotyping error is present. Evaluating the problem in this form is intractable, but the problem can be simplified by augmenting with the pedigree $\boldsymbol{P}$:

$$\int_{\boldsymbol{P}} Pr(\beta, \boldsymbol{P}|\boldsymbol{G}_{obs}, \boldsymbol{l}at, \boldsymbol{l}ong, \boldsymbol{t}err) d\boldsymbol{P}. \tag{4}$$

Equations 3 and 4 are equivalent. In the latter case we estimate the pedigree, but we also integerate out any uncertainity that may remain regarding its structure, leaving us with the marginal distribution of $\beta$. We will start by fitting commonly used approximations to equation 4 and end by a fitting a model that is very close to being exact.

## 2.1 Phenotypic Predictors

To fit the model we need to start by creating a variable that indicates whether particular females and particular offspring occur on the same territory. Unfortunately the nature of these models precludes us from using the R formula mini-language, and so variables are created using `varPed` and then evaluted as part of the model formula in a `PdataPed` object

```
> res1 <- expression(varPed(x = "terr", gender = "Female", relational = "OFFSPRING",
+     restrict = TRUE))
```

`x` is the variable we are interested in, in this case territories. `gender` specifies whether the variable relates to maternity (`gender="Female"`), paternity

(gender="Male") or both (gender=NULL). The argument relational is a little more complex, and sepcifies whether the variable is to be treated as it is (relational=FALSE), or is to be transformed into a distance. relational="OFFSPRING" creates a variable that is the distance between x measured in the offspring and the parent, and relational="MATE" is the distance between x measured in a potential mother and a potential father. When x is numeric the transformed variable is a Euclidean distance. When x is a factor the transformed variable is a logical vector with TRUE indicating that x for the two indivdiuals are the same, and FALSE indicating that they are different. In this example we are dealing with a logical variable that is TRUE when offspring and females are present on the same territory, and FALSE if not. We could estimate a parameter associated with this variable which would be interpreted as the probability of within territory maternity. However, in this case we are assuming that extra territory maternity does not occur and we can use the argument restrict=TRUE to retain mothers that have TRUE (same territory as the offspring) for the transformed variable. Females that are on a different territory to the offspring have FALSE for the transformed variable x, and are excluded as mothers. This is essentially a strong prior on the parameter associated with within territory/extra territory maternity, but is computationaly faster because excluded mothers can be discarded for particular offspring.

We also need to exclude indidvuals that are born in 1999 as potential parents

```
> res2 <- expression(varPed(x = "offspring", gender = NULL, relational = FALSE,
+     restrict = 0))
```

We want to exclude individuals that have 1 in the variable offspring, as they represent chicks born in 1999. We want to exclude offspring irrelevant of their sex so gender=NULL and we want to evaluate x as it is (i.e. as a binary variable). We exclude offspring by specifying restrict=0 which retains individuals that have 0 in the offspring variable.

The goal of the analysis is to estimate $\beta$, the rate at which paternity drops with distance from an offspring

```
> var1 <- expression(varPed(x = c("lat", "long"), gender = "Male",
+     relational = "OFFSPRING"))
```

x in this instance contains two variables that specify the cordinates of each indivdual on the island. Because relational="OFFSPRING" this variable is interpreted as the Euclidean distance between offspring and fathers (gender="Male") in 2 dimensions. restrict is not specified indicdiating that all males are potential fathers.

$$p_{i,j}^{(o)} \propto \exp(\beta\sqrt{(\texttt{lat}_j - \texttt{lat}_o)^2 + (\texttt{long}_j - \texttt{long}_o)^2}) \tag{5}$$

The full notation for a multinomial model is cumbersome so I'll express all models in this form for clarity. A full list of models and the associated varPed

specificiations can be found in Appendix **??**. $p_{i,j}^{(o)}$ is the probability that female $i$ and male $j$ are the parents of offspring $o$.

These variables are to be evaluated inside various functions and they need to be associated with the `data.frame` in which `x` is stored. We also need to provide data on the sex, age, and id of each individual. This is done by creating a `PdataPed` object.

```
> PdP <- PdataPed(formula = list(res1, res2, var1), data = WarblerP,
+     USsire = TRUE)
```

The variables are passed as a list to the argument `formula`, and the relevant data are contained in `WarblerP`. In this case variables named `id`, `sex` and `offspring`, exist in `WarblerP` and they do not need to be specified explicitly. All `PdataPed` objects must contain an `id` and `offspring` variable containing unique identifiers for each individual, and a vetcor indicdating whether records belong to offspring. If `data` does not have a sex column, or `sex=NULL`, the data are assumed to have been collected from a hermaphrodite system. Elements of the sex vector must be either `"Male"`, `"Female"` or `NA`. In this model we also allow for the presence of unsampled males `USsire=TRUE`.

We also need to create a `GdataPed` object for storing the genotype data and some associated information

```
> GdP <- GdataPed(G = WarblerG, categories = NULL)
```

`GdataPed` objects store genotype data (`G`) as a list of `genotype` objects (See the package genetics). A list of `genotype` objects can be directly passed to the argument `G`, or a data frame can be passed that is coerced to a list of genotype objects using the function `genotype.list`. The variable `id` is required, and links genotypes with individuals. Indivdiuals can have multiple genotypes if they have been genotyped more than once. If `G` is a data.frame with a column named `id`, `id` does not have to be explicitly passed to `GdataPed`. If `categories` is not specified then genotyping error rates are assumed not to vary across genotypes, otherwise `categories` must be a vector of factors the same length as `id`.

## 2.2 Approximate Methods and `startPed` Objects

The optional `startPed` object specifies the starting parameterisation for the model, and logical arguments specifying which parts of the model are to be fixed at the starting parameterisation and not to be estimated. Maximum likelihood or heuristic starting parameterisations are used by default, and all estimable parameters are estimated, unless ortherwise specified. These parameters may include the pedigree (`P`), genotypes (`G`), base population allele frequencies (`A`), allelic dropout rates (`E1`), stochastic genotping error rate (`E2`) unsampled male

(USsire) and (female USdam) population sizes, and most importantly the parameters of the multinomial log-linear model (beta). Estimating all unknown parameters can be computationaly intensive, and preliminary analyses can usually be caried out using approximate methods. A commonly used approximation for dealing with genotyping error is to integrate out any uncertainty prior to the parentage anlaysis (See Section 3.1). If we were to implement the exact solution for genotyping error we would be required to estimate the actual genotypes of all individuals, but we can use the approximation by specifying estG=FALSE. When estG=FALSE, error rates and allele frequencies are also unestimable and need to be specified. If the approximation is used, the corrected model of genotyping error used by CERVUS is implemented [**?**], and E2 is the per-allele probability of an error. In CERVUS this is specified as a per-genotype probaility, and the default value of 0.005 is close to the CERVUS default ($0.01 \approx 2(1-0.005)0.005+0.005^2$). Allele frequencies are taken directly from the genotype data using the function extractA, if not specified.

## 2.3 Categorical estimation

The most popular software for parentage anlaysis is CERVUS [**?**]. MasterBayes can fit CERVUS type models as a special case, although two major differences exist between the MasterBayes implementation and the CERVUS implementation. Firstly, mothers and fathers can be estimated simultaneously in MasterBayes, whereas CERVUS was developed for systems where one parent is already known. Secondly, in MasterBayes confidence in parentage assignments is assesed at the level of individual assignments and the measure of confidence uses all the information provided by potential parents. CERVUS on the otherhand asses confidence at the level of the population, and only the information provided by the two most likely parents is used [**?**]. By default, the model defined in Equation 4 is fitted with the minimal amount of approximation. To fit a CERVUS type approximation we need to do a little more work. We need to specify that an approximation is used for genotyping error, and we need to provide point estimates for the allele frequencies and the genotyping error rate (see Sections 2.2 and **??**). We also need to specify that the number of unsampled males is not to be estimated (see Section 2.5), and that a point estimate of 10 is to be used instead.

```
> sP <- startPed(estG = FALSE, E2 = 0.005, A = extractA(WarblerG),
+     estUSsire = FALSE, USsire = 10)
```

CERVUS breaks the problem down into two stages. In the fisrt stage the pedigree is estimated using the genetic data alone, and this pedigree (or part of it) is then passed to the second stage, where the spatial parameter, $\beta$ is estimated. We start by estimating the pedigree using the function MCMCped which returns samples of the posterior distribution of all unkowns.

```
> PdPCervus <- PdataPed(formula = list(res1, res2), data = WarblerP,
+     USsire = TRUE)
```

```
> model1 <- MCMCped(PdP = PdPCervus, GdP = GdP, sP = sP, verbose = FALSE)

[1] "using an approximation for genotyping error"
```

Note that we have created a new `PdataPed` object `PdPCervus` that does not contain the distance variable as we wish to use the genetic data only to infer parentage. Beacuse the only unkown in this model is the pedigree the object `model1` only contains a single element (`P`): the posterior distribution of the pedigree. By default this is the marginal distribution of parental pairs, and we can extract the mode of this distribution

```
> ped1 <- modeP(model1$P, threshold = 0)
> ped1[1:10, ]

      [,1]      [,2]      [,3]
 [1,] "N748835" "K278102" NA
 [2,] "Z993558" "K278102" "K278027"
 [3,] "N748825" "K278115" "N021840"
 [4,] "N748841" "N021991" "K278027"
 [5,] "N748842" "J368410" NA
 [6,] "N748848" "J854303" "K278101"
 [7,] "Z993553" "K278011" "J368403"
 [8,] "N748838" "N021965" "N021760"
 [9,] "N748827" "K278018" "N021839"
[10,] "Z993562" "K278018" NA
```

In this example the most likely pedigree for the first ten offspring is displayed, with the offspring, dam and sire in each column. If the most likely father for one of these offspring is unsampled then the respective element of the sire colimn would be treated as missing (`NA`). For such a simple model, Markov chain Monte Carlo is redundant as the posterior distribution of the pedigree can be calculated analyticly

```
> ped2 <- MLE.ped(getXlist(PdPCervus, GdP), USsire = TRUE, nUSsire = 10)
> ped2[3:12, ]

      [,1]      [,2]      [,3]
 [1,] "N748835" "K278102" NA
 [2,] "Z993558" "K278102" "K278112"
 [3,] "N748825" "K278115" "N021716"
 [4,] "N748841" "N021991" "K278027"
 [5,] "N748842" "J368410" NA
 [6,] "N748848" "J854303" "K278101"
 [7,] "Z993553" "K278011" "J368403"
 [8,] "N748838" "N021965" "N021760"
 [9,] "N748827" "K278018" "N021839"
[10,] "Z993562" "K278018" NA
```

This is the Maximum Likelihood estimate of $\boldsymbol{P}$ and is equivalent, in this model, to the Bayesian estimate when all parental combinations have an equal prior probability of being the parents. We will return to the function `getXlist` later, and proceed to estimate $\beta$ assuming the MLE/posterior mode of the pedigree is the true pedigree. We do this by passing the ML pedigree (`dam=ped2[,2]` and `sire=ped2[,3]`) to a `startPed` object, and by specifying that the pedigree should not be estimated `estP=FALSE`, but should be fixed.

```
> sP <- startPed(estP = FALSE, dam = ped2[, 2], sire = ped2[, 3],
+     estUSsire = FALSE)
> model2 <- MCMCped(PdP = PdP, sP = sP, verbose = FALSE)
> plot(model2$beta)
> summary(model2$beta)

Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
    -0.0707663      0.0094499      0.0002988      0.0002628

2. Quantiles for each variable:

    2.5%      25%      50%      75%    97.5%
-0.09115 -0.07678 -0.07069 -0.06406 -0.05284
```

The genetic data are not required since the pedigree is treated as known, and only the `PdataPed` object needs to be passed to `MCMCped`. Again, with such a simple model, Maximum Likelihood gives valid results, and the large sample approximation for the standard error is close to the Bayesian estimate.

```
> MLElat <- MLE.beta(getXlist(PdP), ped2)
> MLElat$beta

            [,1]
[1,] -0.06987857

> sqrt(MLElat$C)

            [,1]
[1,] 0.009572814
```

Despite the two methods giving identical estimates, they are both severely biased towards zero; the value of $\beta$ that would be observed if fathers were

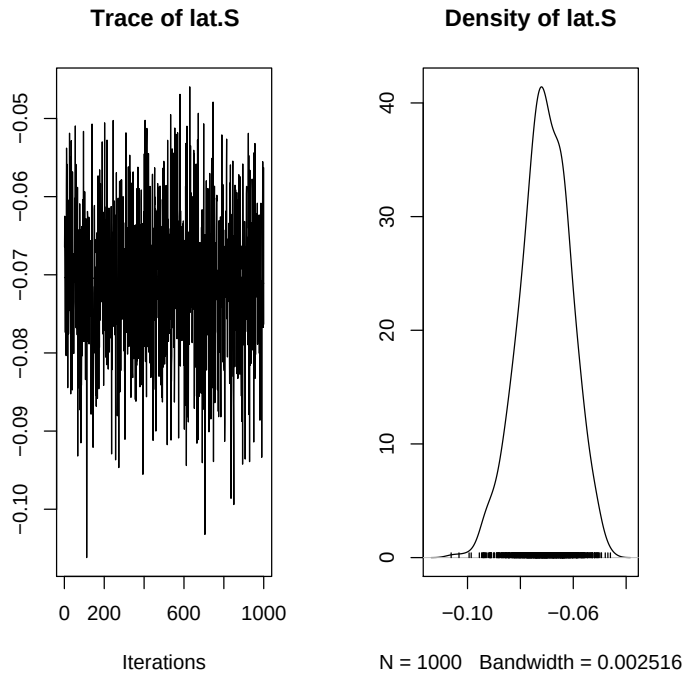


Figure 5: Posterior distribution of $\boldsymbol{\beta}$ from `model2`. The posterior is conditional on the categorical assignment of the pedigree being true. These plots are a way of summarising a Markov Chain using the coda package. The left plot is a trace of the sampled posterior, and can be thought of as a time series. The right plot is a density estimate, and can be thought of a smoothed histogram approximating the posterior. See Section 1 for more details.

distributed randomly across the island with respect to their offspring. We can get a feel for this by estimating $\beta$ for those paternity assignments that had a probability exceeding 0.9.

```
> ped3 <- modeP(model1$P, threshold = 0.9)
> ped3[1:10, ]
```

```
      [,1]       [,2]       [,3]
 [1,] "N748835" "K278102" NA
 [2,] "Z993558" "K278102" NA
 [3,] "N748825" "K278115" NA
 [4,] "N748841" "N021991" NA
 [5,] "N748842" "J368410" NA
 [6,] "N748848" "J854303" "K278101"
```

```
 [7,] "Z993553" "K278011" "J368403"
 [8,] "N748838" "N021965" "N021760"
 [9,] "N748827" "K278018" NA
[10,] "Z993562" "K278018" NA

> MLElat3 <- MLE.beta(getXlist(PdP), ped3)
> MLElat3$beta

           [,1]
[1,] -0.2965067

> sqrt(MLElat3$C)

          [,1]
[1,] 0.0517371
```

The first thing to notice is that by only using males that have a greater than 0.9 probability of being the father, we are excluding 75% of the offspring from the analysis. Because of this the standard error of $\beta$ is more than 5 times larger than that in the previous model. However, the ML estimate of $\beta$ is significantly larger than in the previous model suggesting that our basic model is wrong.

## 2.4 Full Probability Estimation

The reason that it is wrong is beacuse we have failed to use the information contained in the phenotypic data to help us estimate the pedigree. Equation 4 can be rewritten

$$\frac{Pr(\beta, \boldsymbol{P}|\boldsymbol{G}_{obs}, \boldsymbol{l}at, \boldsymbol{l}ong, \boldsymbol{t}err)}{Pr(\boldsymbol{P}|\beta\boldsymbol{G}_{obs}, \boldsymbol{l}at, \boldsymbol{l}ong, \boldsymbol{t}err)}. \tag{6}$$

and can be simplified under the assumption that the genetic data provide no information regarding $\beta$ once the pedigree is known

$$\frac{Pr(\boldsymbol{G}_{obs}|\boldsymbol{P})Pr(\boldsymbol{P}|\beta, \boldsymbol{l}at, \boldsymbol{l}ong, \boldsymbol{t}err)}{Pr(\boldsymbol{P}|\beta, \boldsymbol{G}_{obs}, \boldsymbol{l}at, \boldsymbol{l}ong, \boldsymbol{t}err)} Pr(\boldsymbol{P}, \beta) \tag{7}$$

The fundamental difference between the exact solution implemented in MasterBayes and the approximations used in categorical and fractional type approaches lies in the denominator of equation 7. This equation highlights the importance of using the phenotypic data to aid pedigree reconstruction, and the central role that $\beta$ plays in mediating this information. The CERVUS type approximation makes the mistake of assuming that only the genetic data are required to estimate the pedigree [see the information boxes in **?**].

We can fit a model where the pedigree and $\beta$ are estimated simultanously, although for now we will retain the CERVUS type approxiamtion for genotyping error and the number of unsampled males.

```
> sP <- startPed(estG = FALSE, A = extractA(WarblerG), E2 = 0.005,
+     estUSsire = FALSE, USsire = 10)
> model3 <- MCMCped(PdP = PdP, GdP = GdP, sP = sP, verbose = FALSE)

[1] "using an approximation for genotyping error"

> plot(model3$beta)
> summary(model3$beta)

Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
     -0.286211       0.035635       0.001127       0.002004

2. Quantiles for each variable:

   2.5%     25%     50%     75%   97.5%
-0.3668 -0.3065 -0.2846 -0.2618 -0.2256
```

This estimate is much closer to the ML estimate for the restricted pedigree `ped3`, although the standard error is tighter because the whole pedigree is used. It should also be noted that the ML standard error is artificially low because it does not take into account the uncertainty in the pedigree. This is another draw back of the categorical approach; once the pedigree is estimated you then have to treat it as if it was known with complete certainity, rather than a random variable.

## 2.5 Unsampled parents

Next we will relax the assumption that the number of unsampled males is 10 and actually simultaneously estimate the number from the data

```
> sP <- startPed(estG = FALSE, A = extractA(WarblerG), E2 = 0.005)
> model4 <- MCMCped(PdP = PdP, GdP = GdP, sP = sP, verbose = FALSE)

[1] "using an approximation for genotyping error"

> plot(model4$USsire)
> summary(model4$USsire)
```

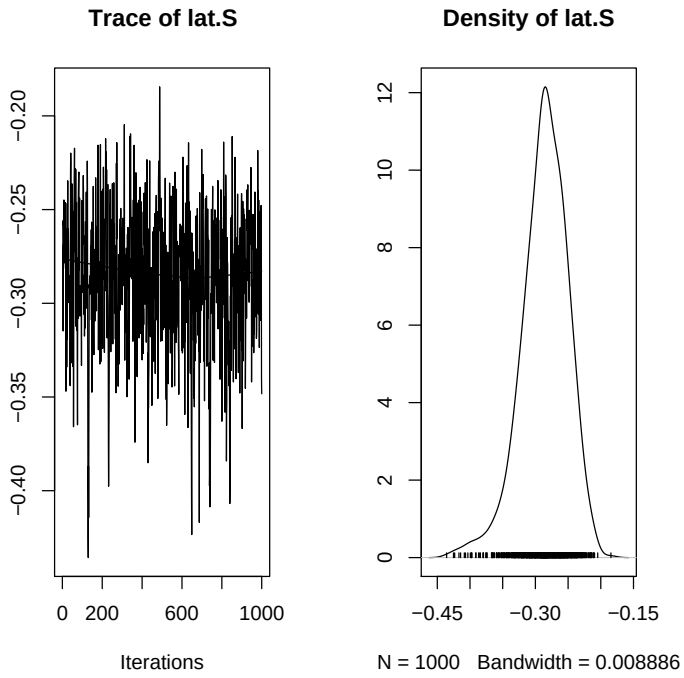


Figure 6: Posterior distribution of $\boldsymbol{\beta}$ from `model3`. The pedigree and $\beta$ have been simultaneously estimated, but an approximation for genotyping error has still been used. It was also assumed that the we size of the unsampled population and the genotyping error rate were known.

```
Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
       51.1102        18.3661         0.5808         0.6213

2. Quantiles for each variable:

 2.5%   25%   50%   75% 97.5%
22.91 38.39 48.54 61.45 92.08
```

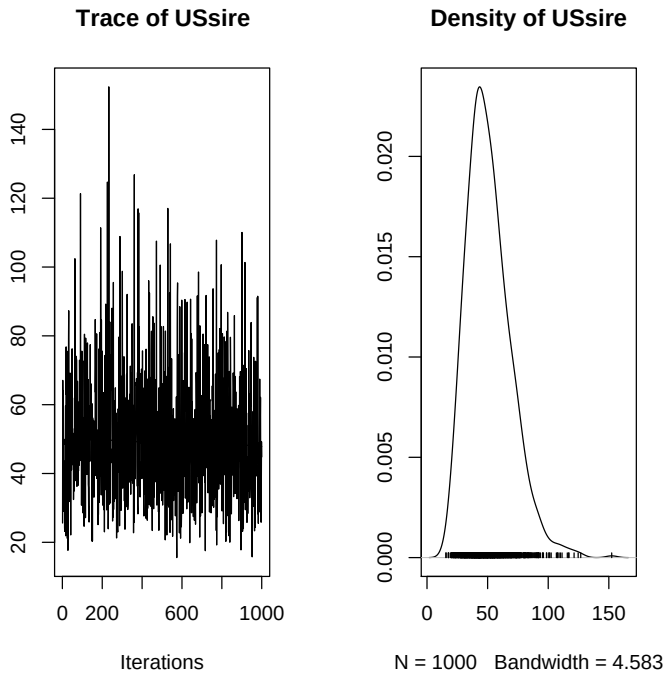


Figure 7: Posterior distribution of the number of unsampled males from `model4`. The number of unsampled males, the pedigree and $\beta$ have all been simultaneously estimated, but we have still used an approximation for genotyping error.

The population of Seychelles Warbler on Cousin Island is small and essentially closed. Given that the island has been extensively sampled we have reasonable prior belief that the number of unsampled males is probably somewhere between 2 and 15, and an estimate of 50 unsampled males seems unreasonably large. We could set up a prior specification that reflected this, however, before doing this it will be usefull to discuss some possible reasons why the model has failed to estimate the number of unsampled males accurately.

The problem of accomodating unsampled parents is a difficult one beacuse the genotypes and phenotypes of unsampled indivduals are by definition unobserved. This does not mean that information does not exist in the observed data regarding the genotypes and phenotypes of unsampled individuals. Imagine, that 3 offspring in the north of the island posses an allele that is not present in the sampled parental population. If genotyping error was low enough we could be reasonbaly confident in stating that an unsampled male was present in the north of the island and that his genotype conatined the rare allele. However,

using this information is difficult and I use two approximations for dealing with unobserved genotypes and phenotypes that are based around the same logic.

**?** gives the likeihood of the total male population size, sampled and unsampled, given the genotype data and known mother-offspring pairs:

$$Pr(\boldsymbol{G}|\boldsymbol{N},\boldsymbol{A}) = \prod_{i}^{n_o}\left[\frac{N-n}{N}Pr(\boldsymbol{O}_i|\boldsymbol{M}_i,\boldsymbol{A}) + \sum_{j=1}^{n}\frac{1}{N}Pr(\boldsymbol{O}_i|\boldsymbol{M}_i,\boldsymbol{F}_j)\right]. \quad (8)$$

$N$ and $n$ are the total and sampled male population sizes, respectively, and $n_o$ is the number of sampled mother-offspring pairs. It is assumed that the genotype data, $\boldsymbol{G}$ are observed without error and that allele frequencies $\boldsymbol{A}$ are known. $\boldsymbol{O}$, $\boldsymbol{M}$ and $\boldsymbol{F}$ are the genotypes of offspring, mothers, and sampled father's respectively. The two genetic likelihoods are the Mendelian transition probability ($Pr(\boldsymbol{O}_i|\boldsymbol{M}_i,\boldsymbol{F}_j)$) and an approximation when the genotype of a parent is unknown ($Pr(\boldsymbol{O}_i|\boldsymbol{M}_i,\boldsymbol{A})$). This approximation makes several assumptions: a) genotype frequencies can be infered from allele frequencies under the asumption of Hardy-Weinberg equilibrium; b) that the allele frequencies in the base population are known with certainty; c) genotyping error does not exist; d) the size of the unsampled population is so large that the allele frequencies in the unsampled population do not differ from $\boldsymbol{A}$; and finally the assumption stated in the previous paragraph e) $\boldsymbol{O}$ and $\boldsymbol{M}$ and provide no information regarding the genotypes of unsampled males after conditioning on $\boldsymbol{A}$. Generally, $\boldsymbol{A}$ is estimated from $\boldsymbol{G}$ and the additional assumption must be made f) that unsampled males and sampled individuals come from the same statistical poulation with respect to allele frequencies. MasterBayes makes all these assumptions except b) and c), and I return to assumption e) in Section **??**.

We can fit **?**, 's model using the function `MLE.popsize`, and specifying the genotype error rate to be effectively zero

```
> MLEUSsire <- MLE.popsize(getXlist(PdP, GdP, E = 1e-10), USsire = TRUE,
+     USdam = FALSE)
> MLEUSsire$nUS

         [,1]
[1,] 95.24434

> sqrt(MLEUSsire$C)

         [,1]
[1,] 30.53817
```

Even larger! The estimates of the unsampled population size are sensitive to even low levels of genotyping error, and this is obvious when we fit the same model but with the genotyping error rate set to 0.01 per allele

```
> MLEUSsire2 <- MLE.popsize(getXlist(PdP, GdP, E = 0.01), USsire = TRUE,
+     USdam = FALSE)
> MLEUSsire2$nUS

         [,1]
[1,] 40.36191

> sqrt(MLEUSsire2$C)

         [,1]
[1,] 19.97729
```

Perhaps we had underestimated the level of genotyping error in our original models, and this possibility will be returned to in Section 2.6. Another possible reason why our estimate of the unsampled population may be to high, is because Equation 8 makes the assumption that unsampled males and sampled males do not systematicly differ in their ability to sire the sampled offspring. When we estimate the number of unsampled males using `MCMCped` Equation 8 has a simpler form because the parameter space is augmented with the pedigree (See Equation **??**):

$$Pr(\boldsymbol{P}|N) = \prod_{i}^{n_o} \left[ \frac{N-n}{N}(1-\delta_i^{obs}) + \frac{1}{N}\delta_i^{obs} \right] \tag{9}$$

Here we have replaced the genetic likelihoods in Nielen's original equation with the indicator variable $\delta_i^{obs}$. $\delta_i^{obs}$ takes on the value 1 when offspring $i$'s father has been sampled, and 0 otherwise. If we knew the pedigree exactly then the Maximum Likelihood Estimate for the number of unsampled males $(N-n)$ is simply the number of offspring with unsampled fathers divided by the average number of offspring per sampled father:

$$\text{MLE}(N-n) = \frac{n_o^{us} n_o^{s}}{n} \tag{10}$$

where $n_o^{us}$ and $n_o^{s}$ are the number of offspring with sampled and unsampled parents parents, respectively.

A more general solution than Equation 9 is

$$Pr(\boldsymbol{P}|N) = \prod_{i}^{n_o} \left[ \frac{N-n}{N} E\left[ \frac{\hat{p}_i^{obs}}{\hat{p}_i^{miss}} \right] (1-\delta_i^{obs}) + \frac{1}{N}\delta_i^{obs} \right] \tag{11}$$

where $\hat{p}_i^{obs}$ and $\hat{p}_i^{miss}$ are the linear predictors of paternity for sampled and unsampled males, respectively. Nielsen's original formulation of the problem assumes that the expectaion of the ratio in 11 is exactly one, and the estimate of the male population size has to be interpreted as an effective population size assuming all males to be equal.

The default in MasterBayes is to assume that this ratio has an expectation of 1. However, we still need to estimate the distribution of the mean linear predictor for unsampled individuals ($\hat{p}_i^{miss}$) so that we can sample the pedigree correctly.

The parentage of each offspring is sampled from a multinomial distribution with each category representing a unique parental combination. The probability that the offspring falls into one of these categories is based on the Mendelian transition probability and the linear predictors of potential parents. The number of categories is equal to $(n_d^o)(n_s^o + 1)$ where $n_d^o$ and $n_s^o$ are the number of sampled females and males that could be the parents of offspring $o$. An additional $n_d^o$ categories are set up that represent the pairing of each female with an unsampled males. The genetic likelihoods for these categories can be derived folowing Equation 8 and the linear predictor for this category can be derived following a similar logic.

Under the assumption that sampled and unsampled males come from the same statistical poulation I use an approximation based around the central limit theorem for the distribution of the linear predictor for the unsampled category. Under the central limit theorem this distribution will be normally distributed irrespective of the underlying distribution of linear predictors when the sample size is large (equivalent to assumption a) for the genetic likelihoods). The distribution of the summed linear predictors of unsampled males can then be obtained using the following

$$p(\sum^{N-n} \hat{\boldsymbol{p}}^{(miss)} | \hat{\boldsymbol{p}}^{(obs)}) \approx N(\frac{(N-n)}{n} \sum^{n} \hat{\boldsymbol{p}}^{(obs)}, n(N-n)S^2_{obs}) \qquad (12)$$

which takes into account the sampling variance in calculating the expectation from the sampled population (avoiding assumption c)), and the sampling variance that arises because the size of the unsampled population is finite assumption d)). $S^2_{obs}$ is the sample variance of the observed linear predictors [see **?**, ,Chapter 7] I should emaphasise at this point that the approximated linear predictors for missing individuals do not enter into the likelihoood for $\beta$, they are only there to simplify the estimation of the pedigree. However, if the phenotypes of unsampled inidviuals are known, then the linear predictors of unsampled inidviduals do not need to be estimated and can enter into the likelihood for $\beta$. In this insatnce the ratio in Equation 11 may not be unity and assumption f) is relaxed (See Section 3.6 for an example).

We have no information regarding the phenotypes of unsampled males in the Seychelles Warbler, and therefore the data do not allow us to distinguish between a large unsampled population and a smaller population with a reduced chance of paternity. However, as noted earlier, the discrepancy between our estimate of 50 unsampled males and our prior belief that the number of males should be less than 15 may be a consequence of underestimating the level of

genotyping error.

## 2.6 Genotyping error

At this point it will be usefull to distinguish between exact and approximate methods for dealing with genotyping error. A useful notation will be $\boldsymbol{G}$ for true but unobserved genotypes and $\boldsymbol{G_{obs}}$ for observed but possibly erroneus genotypes. Both methods implicitly or explicitly use this concept, and the aim is to derive the probability distribution of $\boldsymbol{G}$ from $\boldsymbol{G_{obs}}$ and what we know about error rates ($\varepsilon$) and allele frequencies ($\boldsymbol{A}$). The approximate method defines the conditional probability of $\boldsymbol{G}$ as

$$Pr(\boldsymbol{G}|\boldsymbol{G}_{obs}, \varepsilon, \boldsymbol{A}) \tag{13}$$

whereas the exact solution defines the probability as

$$Pr(\boldsymbol{G}|\boldsymbol{G}_{obs}, \boldsymbol{P}, \varepsilon, \boldsymbol{A}). \tag{14}$$

The distinction is subtle. The approximation implicitlty assumes that the genotypes are collected from unrelated indivdiuals, whereas the exact solution acknowledges the fact that the indivdiuals are related. By acknowledging this fact we become aware that the approximation actually throws some information away; if individuals are related then the observed genotypes of some individuals actually provide information regarding the actual genotypes of others. The exact solution comes at a computational cost. Because the pedigree $\boldsymbol{P}$ is estimated, $\boldsymbol{P}$ may change with every iteration of the Markov chain and the probability in Equation 14 has to be updated each cycle. Beacase the probability distribution of $\boldsymbol{G}$ does not depend on $\boldsymbol{P}$ in the approximation, Equation 13 only needs to be evaluated once, prior to pedigree estimation. This is only true, however, if error rates and allele frequencies (in the base population) are known with complete certainity.

At this point we can drop the approximation for genotyping error and work with the exact solution, allowing us to estimate genotyping error rates and the allele frequencies in the parental generation. This becomes computationally demanding, as the probability distribution for each individual's genotype has to be calculated and sampled from each itteration of the Markov chain. When the exact solution is used we switch from the CERVUUS model of genotyping error to one proposed by [**?**]. The CERVUS model is mathematically convenient but is unlikely to be biologically realistic. The main assumption of the CERVUS model is that the probability of a genotype being misscored is proprtional to that genotypes frequency in the population. In Wang's model an allele is assumed to be misscored as any other allele with equal probability ($\varepsilon_2$). In addition, the process of allelic dropout is also modelled with the probability of an allele droping out being denoted as $\varepsilon_1$

```
> model5 <- MCMCped(PdP = PdP, GdP = GdP, verbose = FALSE)
> plot(model5$E1)
```

```
> plot(model5$E2)
> plot(model5$USsire)
```

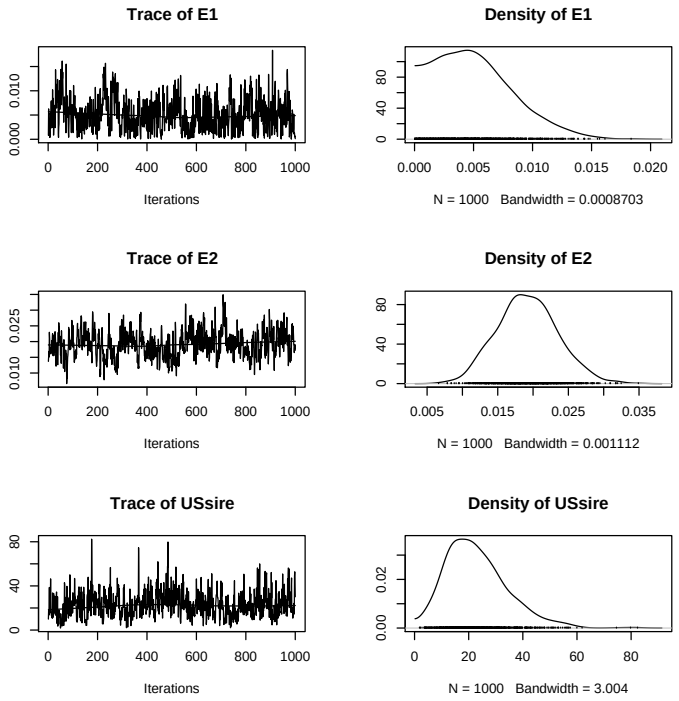


Figure 8: The marginal posterior distributions of the number of unsampled males and error rates from `model5`. All unknowns have been simultaneously estimated.

The chain is slow to mix when genotypes are estimated and successive samples from the posterior show autocorrelation

```
> autocorr(model5$E1)

, , E1

              E1
Lag 0  1.00000000
Lag 1  0.45968661
Lag 5  0.17609684
Lag 10 0.14347243
Lag 50 0.05374619
```

The chain needs to be run for longer to ensure that we gather independent sample s from the posterior, but nethertheless it is clear that we had underestimated the aamount of genotyping error (see Figure 8). Allelic drop out appears to occur for about 1 in 100 genotypes, but the stochastic error rate appears higher at around 1 in 25. The posterior for the number of unsampled males now looks more reasonable but there is still alot of uncertainity and a judicious use of prior information seems reasonable. The log normal is used as the prior specification for the number of unsampled males

```
> pP <- priorPed(USsire = list(mu = log(5), sigma = 0.5))
> model6 <- MCMCped(PdP = PdP, GdP = GdP, pP = pP, tP = tunePed(USsire = 0.1),
+     verbose = FALSE)
> plot(model6$USsire)
```

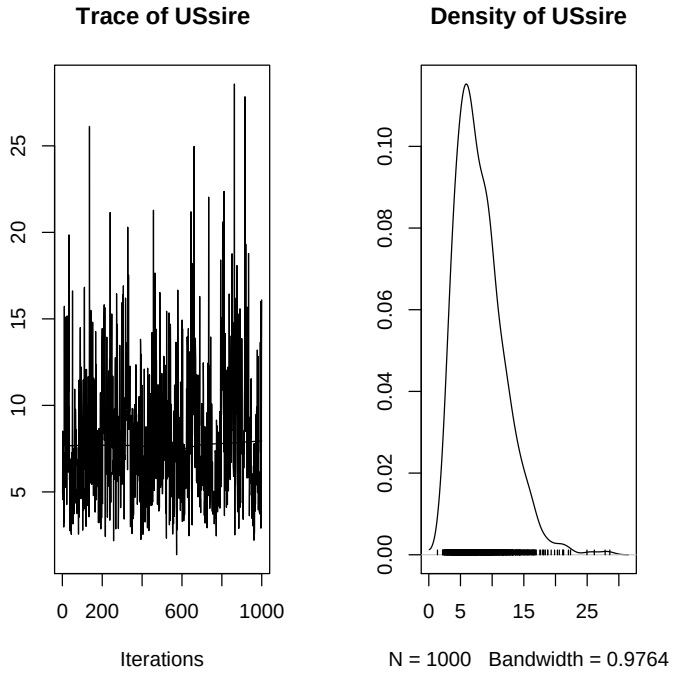


Figure 9: The marginal posterior distribution of the number of unsampled males from `model6`. In this model an informative prior was used.

# 3 Further Examples using Simulated Data

To illustrate some further functionality we will use the functions `simpedigree` and `simgenotypes` to simulate pedigrees and genotypes according to certain models and then analyse them. We will use an estimate of the Seychelles Warbler allele frequencies to sample genotypes from

```
> A <- extractA(WarblerG)
```

## 3.1 Error Rate Estimation with and without a Pedigree

To start we will simply estimate genotyping error for two data sets, one in which the information comes from individuals being sampled multiple times and one in which the information comes from pedigree inforamtion

```
> ped <- matrix(NA, 50, 3)
> ped[, 1] <- 1:50
> G <- simgenotypes(A = A, E1 = 0.1, E2 = 0.005, ped = ped, no_dup = 2)
> tP <- tunePed(E1 = 15)
> GdP <- GdataPed(G = G$Gobs, id = G$id)
> model.dupE <- MCMCped(GdP = GdP, tP = tP, verbose = FALSE)
> summary(model.dupE$E1)

Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
     0.1014073      0.0107350      0.0003395      0.0003116

2. Quantiles for each variable:

   2.5%     25%     50%     75%   97.5%
0.08144 0.09378 0.10093 0.10877 0.12312

> summary(model.dupE$E2)

Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
```

```
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
     6.670e-03      2.360e-03      7.463e-05      6.701e-05

2. Quantiles for each variable:

    2.5%      25%      50%      75%    97.5%
0.002696 0.005020 0.006355 0.008071 0.012199
```

The posterior summaries are consistent with the data we simulated, but notice that the mean may not be a very good summary of the central tendency of the posterior distribution for stochastic error (see Figure 10). We can simulate a similar set of data but with pedigree information

```
> ped <- matrix(NA, 100, 3)
> ped[, 1] <- 1:100
> ped[, 2][51:100] <- 1:50
> G <- simgenotypes(A = A, E1 = 0.1, E2 = 0.005, ped = ped, no_dup = 1)
> sP <- startPed(dam = ped[, 2])
> GdP <- GdataPed(G = G$Gobs, id = G$id)
> model.pedE <- MCMCped(GdP = GdP, sP = sP, tP = tP, verbose = FALSE)
> summary(model.pedE$E1)

Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
     0.0799451      0.0111415      0.0003523      0.0003775

2. Quantiles for each variable:

   2.5%     25%     50%     75%   97.5%
0.05956 0.07222 0.07973 0.08699 0.10236

> summary(model.pedE$E2)

Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000
```

```
1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
     0.0049145      0.0041599      0.0001315      0.0003299

2. Quantiles for each variable:

     2.5%       25%       50%       75%     97.5%
0.0002448 0.0016404 0.0038097 0.0070162 0.0152851
```

```
> plot(model.pedE$E2)
```

The posterior from the second data set has greater variance. With respect to genotyping error, multiple samples from the same individual are more informative than genotypes typed once in mother-offspring pairs. If individuals have been typed more than once then this information should be included in pedigree reconstruction as it improves error rate estimation and increases the precision with which an indivdiual's genotype is estimated.

In Section 1.3 I mentioned the fact that MCMC may be sensitive to the starting parameterisation, and that this is particularly so when the posterior distribution is high-dimensional and multimodal. The posterior distribution of genotypes has these properties, and high probability genotype configurations may seperated by regions of zero probability when pedigree data exist. As an example we will specify a random (but legal) starting configuration for the previous model:

```
> stG <- simgenotypes(A = A, E1 = 0, E2 = 0, ped = ped, no_dup = 1)
> sP <- startPed(dam = ped[, 2], G = stG$Gobs)
> model.config <- MCMCped(GdP = GdP, sP = sP, tP = tP, verbose = FALSE)
> plot(model.config$E2)
```

When only dams are known the chain still mixes well and the chain converges very rapidly despite the absurd starting configuration (see Figure, **??**). However, we can simulate genotypes down a pedigree when both dams and sire are known. We will analyse the data using two chains, one initialised at the true unobserved genotype configuration, and one initialised at a random but legal configuration.

```
> ped <- matrix(NA, 150, 3)
> ped[, 1] <- 1:150
> ped[, 2][101:150] <- 1:50
> ped[, 3][101:150] <- 51:100
> G <- simgenotypes(A = A, E1 = 0.1, E2 = 0.005, ped = ped, no_dup = 1)
> sP <- startPed(dam = ped[, 2], sire = ped[, 3], G = G$G)
> GdP <- GdataPed(G = G$Gobs, id = G$id)
```

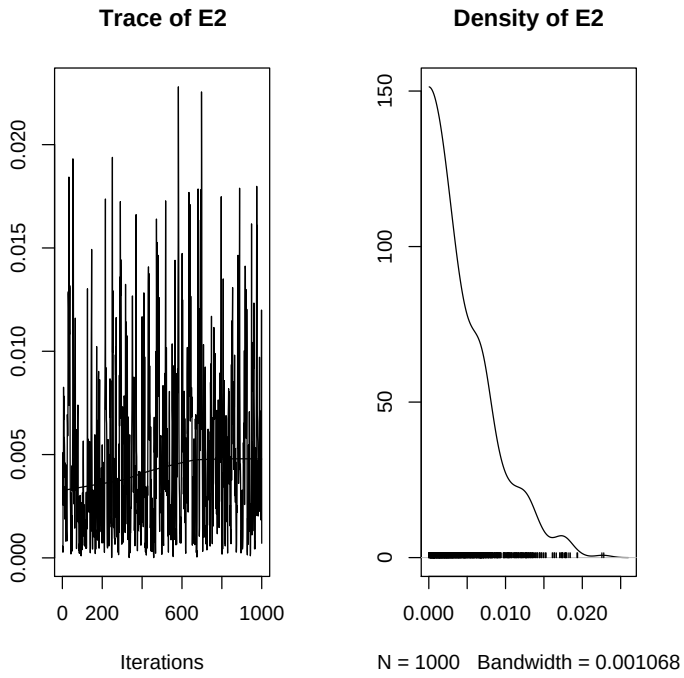


Figure 10: The marginal posterior distribution of stochastic error rate from `model.pedE`. The mode is probably a better summary of central tendency as the expectation of `E1` may not coincide with the most likely value it could take.

```
> model.DS1 <- MCMCped(GdP = GdP, sP = sP, tP = tP, verbose = FALSE)
> stG <- simgenotypes(A = A, E1 = 0.1, E2 = 0.005, ped = ped, no_dup = 1)
> sP <- startPed(dam = ped[, 2], sire = ped[, 3], G = stG$G)
> model.DS2 <- MCMCped(GdP = GdP, sP = sP, tP = tP, verbose = FALSE)
> post.max <- max(c(model.DS1$E2, model.DS2$E2))
> plot(c(model.DS1$E2), type = "l", col = "blue", ylim = c(0, post.max))
> lines(c(model.DS2$E2), type = "l", col = "red")
```

Clearly, the chains are sampling from different regions of the posterior 12. Methods exist for traversing these regions [**??**] but these are hard to implement on fixed pedigrees let alone pedigrees that are being constantly updated. However, when the pedigree is not fixed, zero probability regions do not exist because all genotype congigurations have a positive probability under some pedigree configuration. Nethertheless, there may be regions of very low probability connecting high probability regions, and the chain may not mix well.

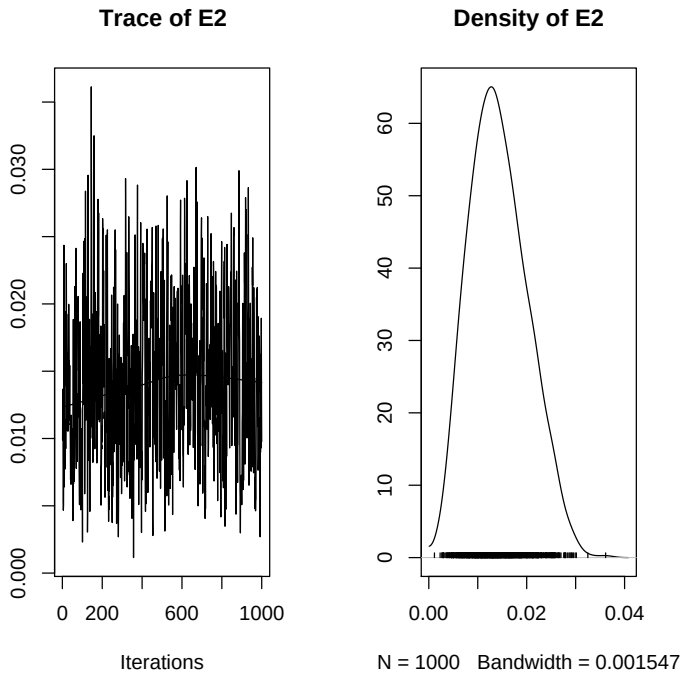


Figure 11: The marginal posterior distribution of stochastic error rate from `model.config`. This posterior should be identical to the posterior in Figure 10, although the chain was initilised in a very low probability region of genotype parameter space.

## 3.2 Mismatch tolerence and computational efficency

The number of potential parental combinations is not linear in the number of potential mothers and fathers. With 50 candidate mothers and 50 candidate fathers the number of parental combinations is 2500. This can slow the chain down because each itteration, and for each offspring 2500 Mendelian likelihoods have to be recalculated. Also, we have to sample parents from a multinomial distribution with as many categories. When genotyping error is low, we can safely assume that potential parents that have many mismatches with an offspring have a probability close to zero of being the true parents. The argument `mm.tol` can be passed to `MCMCped` specifying the number of mismatches that will be tolerated for a potential parent.

To illustrate I have simulated a pedigree where body size has a strong effect on both paternity and maternity. However, the effect of body size is not the

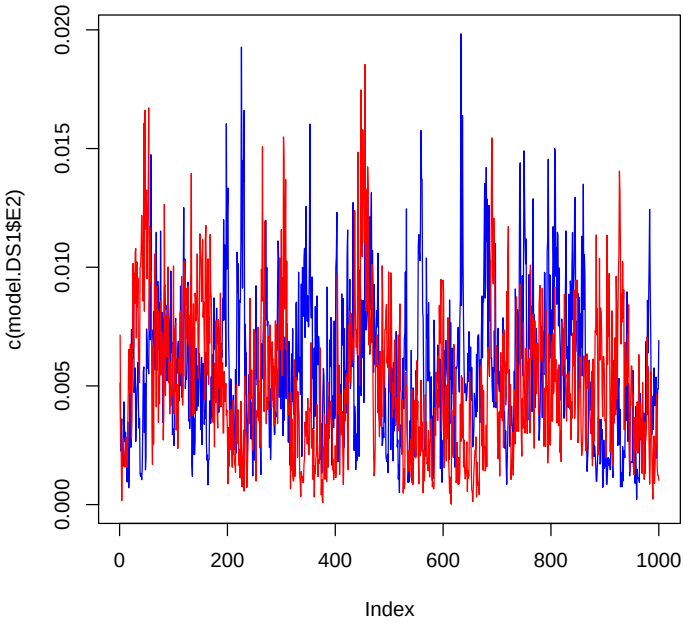


Figure 12: Samples from the posterior distribution of stochastic error rate estimated from the data set `Gobs`. The two chains were initialised at the true genotype configuration (blue), and a random congiguartion (red).

same for the sexes: large males have an increased chance of gaining paternity, but small females are more fecund. Follwoing the notation of Equation 5:

$$p_{i,j}^{(o)} \propto \exp(\beta_1 \texttt{size}_i + \beta_2 \texttt{size}_j) \tag{15}$$

```
> sex <- c(rep("Female", 50), rep("Male", 100))
> offspring <- c(rep(0, 100), rep(1, 50))
> size <- rnorm(150, 10)
> data.MM <- as.data.frame(list(id = 1:150, sex = sex, offspring = offspring,
+     size = size))
> res1 <- expression(varPed(x = "offspring", restrict = 0))
> var1 <- expression(varPed(x = "size", gender = "Male"))
> var2 <- expression(varPed(x = "size", gender = "Female"))
> PdP <- PdataPed(formula = list(res1, var1, var2), data = data.MM)
> simped <- simpedigree(PdP, beta = c(1, -1))
> G <- simgenotypes(A = A, E1 = 0.005, E2 = 0.005, ped = simped$ped,
+     no_dup = 1)
```

```
> GdP <- GdataPed(G = G$Gobs, id = G$id)
> model_pedMM999 <- MCMCped(PdP = PdP, GdP = GdP, verbose = FALSE)
> model_pedMM2 <- MCMCped(PdP = PdP, GdP = GdP, mm.tol = 2, verbose = FALSE)
> model_pedMM1 <- MCMCped(PdP = PdP, GdP = GdP, mm.tol = 1, verbose = FALSE)
> summary(model_pedMM999$beta)[[1]][, 1:2][1:2, ]

             Mean        SD
size.D  1.0438977 0.1945208
size.S -0.7591874 0.1637831

> summary(model_pedMM2$beta)[[1]][, 1:2][1:2, ]

            Mean        SD
size.D  1.057421 0.2004407
size.S -0.780508 0.1532398

> summary(model_pedMM1$beta)[[1]][, 1:2][1:2, ]

             Mean        SD
size.D  1.0292386 0.1976253
size.S -0.7667134 0.1558759
```

The posterior distributions of $\beta$ are almost identical, despite excluding parents with several mismatches. However, the saving in computer time was large: the first model took almost 15 minutes to fit where the last model took under 3.

### 3.3 Equivalence with Poisson Models

If we knew the pedigree we may be inclined to fit a generalised linear model with a Possion error distribution and log link to the data set `data.MM`. We can analyse the relationship between female body size and fecundity by counting the number of offspring per female, and using the standard `glm` function in R

```
> counts <- rep(0, 50)
> noff <- table(simped$ped[, 2])
> counts[as.numeric(names(noff))] <- counts[as.numeric(names(noff))] +
+     noff
> model.Pd <- glm(counts ~ size[1:50], family = "poisson")
> summary(model.Pd)

Call:
glm(formula = counts ~ size[1:50], family = "poisson")

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6006  -0.9984  -0.5605   0.6946   2.4119
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -10.4111     1.8959  -5.491 3.99e-08 ***
size[1:50]    1.0161     0.1784   5.696 1.23e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 88.549  on 49  degrees of freedom
Residual deviance: 52.737  on 48  degrees of freedom
AIC: 118.31

Number of Fisher Scoring iterations: 5
```

The slopes are virtually the same, but the standard errors are a little larger for the `MCMCped` model, reflecting uncertainity in the pedigree. However, because size is such a good predictor of parentage, and the genotype data are relatively informative, the pedigree is resolved quite well.

```
> table(simped$ped[, 2:3][101:150, ] == modeP(model_pedMM999$P)[,
+     2:3])

FALSE  TRUE
   12    88
```

The modal parenatge assignments are close to the true pedigree, with 92% of assignments correct. One important difference arises because `MCMCped` uses a mutinomial log-linear model rather than the Poisson log-linear model used above. The models are very similar except the multinomial model conditions on the number of counts (offspring) whereas the Possion model does not. Consequently the number of parameters in a multinomial model (excluding the count total) is one less than the equivalent Poisson model. In the above example an intercept term is not calculated becuase it can be derived directly from the number of parental combinations and the number of offspring which we have conditioned on [**?**].

## 3.4 Interactions and Reparameterisation

Next we will consider a model similar to that above, but we will have a constant effect of size for both sexes. However, we will will incude a second variable `age` which has two levels: `old` and `young`. We will fit both main effects together with an interaction. Interactions can fitted by adding an extra element to the `formula` argument that is list of the two `varPed` expressions to be combined.

$$p_{i,j}^{(o)} \propto \exp(\beta_1\delta_i + \beta_1\delta_j + \beta_2\texttt{size}_i\delta_i + \beta_2\texttt{size}_j\delta_j + \beta_3\texttt{size}_i(1-\delta_i) + \beta_3\texttt{size}_j(1-\delta_j)) \tag{16}$$

The $\delta$ variables take on the values 1 if the individual is young, and zero if not. Notice that although there are six terms in Equation 16 there only 3 parameters to be estimated because the variables are assumed to have consistent effects on both maternity and paternity.

```
> age <- gl(2, 1, 150, label = c("old", "young"))
> data_INT <- cbind(data.MM, age)
> res1 <- expression(varPed(x = "offspring", restrict = 0))
> var1 <- expression(varPed(x = "size"))
> var2 <- expression(varPed(x = "age"))
> PdP <- PdataPed(formula = list(res1, var1, var2, list(var1, var2)),
+     data = data_INT)
> simped <- simpedigree(PdP, beta = c(0.5, 10, -1))
> G <- simgenotypes(A = A, E1 = 0.005, E2 = 0.005, ped = simped$ped,
+     no_dup = 1)
> GdP <- GdataPed(G = G$Gobs, id = G$id)
> model.INT <- MCMCped(PdP = PdP, GdP = GdP, mm.tol = 1, verbose = FALSE)
```

Markov chains do not mix well when the posterior distribution of the parameters are not independent. In this instance plotting the Markov chain indicates high autocorrelation between the slope and intercept for young individuals. Indeed the autocorrelation approaches 1 despite a thinning interval of 10.

```
> plot(model.INT$beta)
> autocorr(model.INT$beta)[, , 3]

               size   age.young size.age.young
Lag 0  -0.796625575 -0.99401869     1.00000000
Lag 1  -0.243809676 -0.27797947     0.27747599
Lag 5  -0.003681770 -0.01257271     0.01918920
Lag 10  0.039798676  0.02944716    -0.02916027
Lag 50  0.029900082  0.01333722    -0.01408458
```

The dependency between slopes and intercpets in regression models is well know and we can alleviate the problem by simply centering the covariate, `size`. The intercept is now evaluated at the population mean for size (10) rather than at a size of zero

```
> data_INT[, "size"] <- size - 10
> PdP <- PdataPed(formula = list(res1, var1, var2, list(var1, var2)),
+     data = data_INT)
> model.INT.cntr <- MCMCped(PdP = PdP, GdP = GdP, mm.tol = 1, verbose = FALSE)
> plot(model.INT.cntr$beta)
> autocorr(model.INT.cntr$beta)[, , 3]

              size   age.young size.age.young
Lag 0  -0.79560221  0.11290401     1.00000000
```

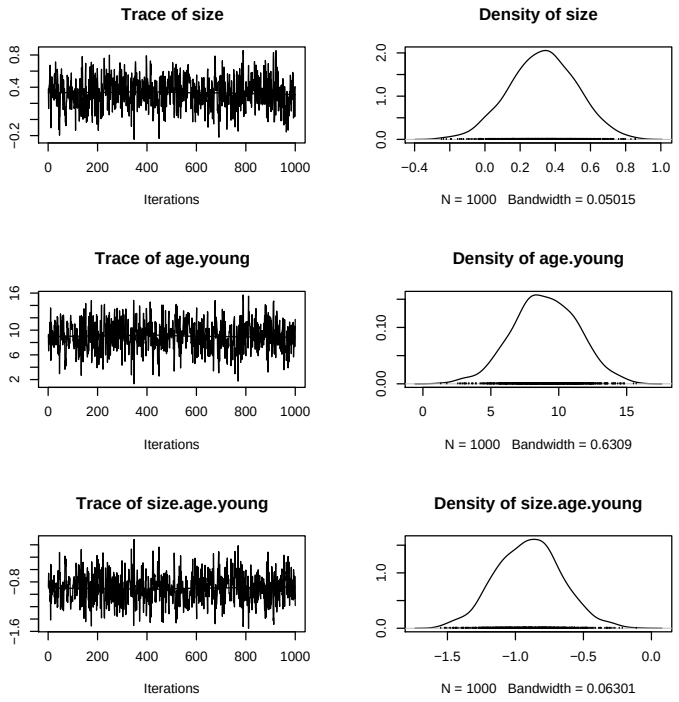


Figure 13: The marginal posterior distributions for the three elements of `beta` from `model.INT`. The intercpet for young individuals `age.young` is evaluated at a `size` of 0, which lies well outside the distribution of `size` observed in the population.

```
Lag 1  -0.27266200 -0.01891453     0.31984918
Lag 5  -0.04588246 -0.01983185     0.06783559
Lag 10 -0.01817254 -0.02650195     0.02035480
Lag 50  0.05272709 -0.01938244    -0.07162828
```

Despite having similiar Metropolis-Hastings acceptance rates the second chain mixes much better, and the parameters have a better biological interpretation: at the average size no differences in the ability to reproduce exists between young and old individuals; the posterior distribution of `age.young` is centered around zero. However, as old individuals get larger their ability to reproduce increases (`size` is positive - the true underlying slope is 0.5), but young indivdiuals have a significantly shallower slope (`size.age.young` is less than zero). The contrasts are set up so that `size.age.young` is is the differnce between the slopes of young and old inidviduals. Inference from posterior samples is very flexible. If we wish to see whether the slope is so shallow it is in fact likely to be negative we can create the posterior distribution for the young

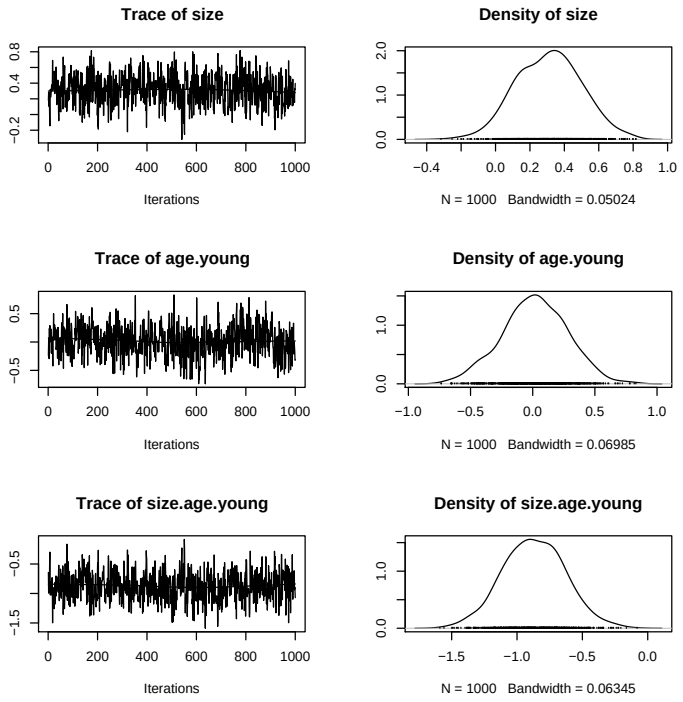


Figure 14: The marginal posterior distributions for the three elements of `beta` from `model.INT`, but with the intercpet for young individuals `age.young` evaluated at the average `size` of 10.

slope

```
> young.slope <- mcmc(model.INT.cntr$beta[, 1] + model.INT.cntr$beta[,
+     3])
> plot(young.slope)
```

There is good evidence that the slope for young individuals is negative. In fact the pedigree was simulated so that the slopes would be of the same magnitude (0.5), but with different signs. In this instance the sexes followed the same rules with regard to reproduction, and we can fit a single GLM across the sexes

```
> counts <- rep(0, 100)
> noff <- table(c(simped$ped[, 2], simped$ped[, 3]))
> counts[as.numeric(names(noff))] <- counts[as.numeric(names(noff))] +
+     noff
> par_age <- age[1:100]
```

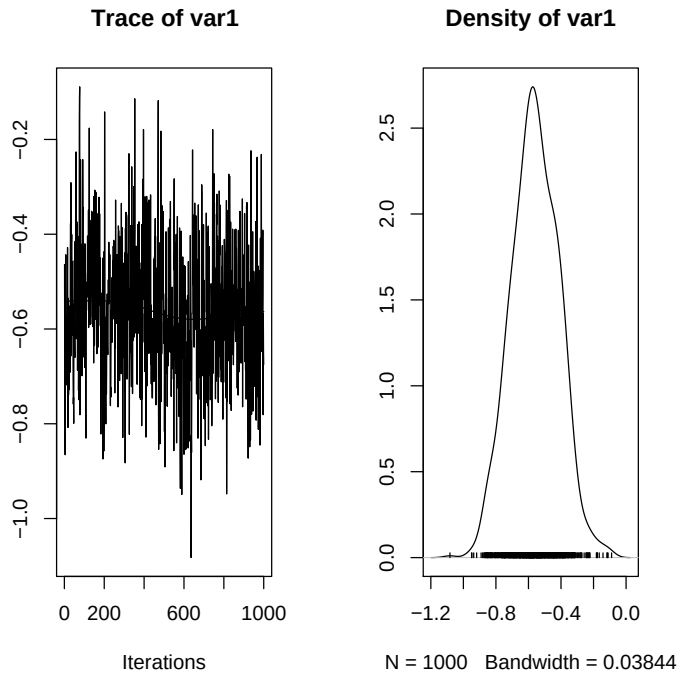


Figure 15: The marginal posterior distribution for the slope of young individuals from model `model.INT.cntr`.

```
> par_size <- size[1:100] - 10
> model.Pint <- glm(counts ~ par_age * par_size, family = "poisson")
> summary(model.Pint)

Call:
glm(formula = counts ~ par_age * par_size, family = "poisson")

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.84821  -1.22156  -0.09956   0.47921   2.06393

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          -0.04867    0.14614  -0.333 0.739104
par_ageyoung         -0.08885    0.21728  -0.409 0.682603
par_size              0.29275    0.16387   1.787 0.074016 .
par_ageyoung:par_size -0.81749    0.21167  -3.862 0.000112 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 118.91  on 99  degrees of freedom
Residual deviance: 100.81  on 96  degrees of freedom
AIC: 253.22

Number of Fisher Scoring iterations: 5
```

A posison GLM indicates the same thing: young and old indivdiuals do not appear to behave differently at the average body size. However, the slope for old individuals is significantly positive, and the slope for young individuals differs significantly from the slope in old indivdiuals. Testing whether young individuals have a negative slope would involve setting different contrasts.

## 3.5 Interpreting Parameters Associated with Categorical Variables

Imagine an experiment in which 30 males are randomly selected from a large population and treated, and we are interested in whether the treatment affects fecundity. We would like a statistic that does not depend on the number of treated and control males in the population. We are not interested in the probability that an offspring has a treated father compared to a control father, because this probability could be increased by simply treating a greater proportion of males. The default in MasterBayes is to estimate the logs odd ratio ($\beta$) of a male siring an offspring if that male had been treated compared to if it had been left untreated, where

$$\beta = log\left[\frac{\theta}{1-\theta}\right] = logit(\theta) \tag{17}$$

and $\theta$ is the probability if that male had been treated. A nice property of posterior distributions is that we can take the inverse logit transformation of the samples gathered for the posterior distribution of $\beta$ to make valid inferences about the posterior of $\theta$.

When the number of treated and untreated potential fathers is known, and does not vary between offspring we can derive the probability that an offspring will be sired by a treated male ($\theta_o$):

$$\theta_o = \frac{\theta N_E}{\theta N_E + (1-\theta)N_C}, \tag{18}$$

and also the logs odd ratio

$$\beta_o = logit(\theta_o) \tag{19}$$

By default MasterBayes estimates $\beta$, although $\beta_o$ can be estimated using the argument `merge=TRUE` in `varPed`. When the number of experimental and non-experimental males are known, and the sets of potential fathers for all offspring are equal, there is no statistical reason for choosing $\beta$ over $\beta_o$ since they are functionally equivalent. However, there are cases when $\beta$ and $\beta_o$ are not equivalent and care needs to be taken when choosing between the two models. A classic example of when $\beta_o$ would be favoured over $\beta$ is when estimating the level of extra-pair paternity.

## 3.6 Unsampled Parents with Known Phenotypes: Estimating Extra-pair Paternity

Consider a study site with 50 territories, each of which contains a pair of adults. We will assume that the mother of all offspring on a given territory is the territorial female, but some offspring may be sired by males from another territory. Let's imagine that we have sampled all the individuals on half the territories, and would like to know a) the extra-pair paternity rate, and b) the number of unsampled territories, which in this case is 25. We will start by simulating a pedigree with an extra-pair paternity rate of 20%.

```
> sex <- c(rep("Male", 50), rep("Female", 125))
> terr <- as.factor(c(1:50, 26:50, rep(26:50, each = 4)))
> offspring <- c(rep(0, 75), rep(1, 100))
> data_EPP <- as.data.frame(list(id = 1:175, sex = sex, offspring = offspring,
+     terr = terr))
> res1 <- expression(varPed(x = "terr", gender = "Female", relational = "OFFSPRING",
+     restrict = TRUE))
> var1 <- expression(varPed(x = "terr", gender = "Male", relational = "OFFSPRING"))
> res2 <- expression(varPed(x = "offspring", restrict = 0))
> PdP <- PdataPed(formula = list(res1, var1, res2), data = data_EPP)
> EPP <- 0.8/(0.8 + (0.2/49))
> simped <- simpedigree(PdP, beta = logit(EPP))
> G <- simgenotypes(A = A, E1 = 0.01, E2 = 0.01, ped = simped$ped,
+     no_dup = 1)
> GdP <- GdataPed(G = G$Gobs, id = G$id)
> rm_males <- 1:25
> data_EPP_miss <- data_EPP[-rm_males, ]
> GdP_miss <- GdataPed(G = lapply(G$Gobs, function(x) {
+     x[-rm_males]
+ }), id = G$id[-rm_males])
```

Notice the argument `merge=FALSE` is passed to `varPed` for `var1` indicating that we are estimating $\beta$, not the within-pair paternity rate which in this case is $\beta_o = 0.8$. Also, we are treating the unsampled males as if they came from the same statistical population as the sampled males (`USvar=NULL`), although we know that the logical variable indicating whether offspring and males are on the same territory should be `FALSE` for unsampled males.

```
> PdP_miss <- PdataPed(formula = list(res1, var1, res2), data = data_EPP_miss,
+     USsire = TRUE)
> model.miss <- MCMCped(PdP = PdP_miss, GdP = GdP_miss, verbose = FALSE)
> plot(model.miss$beta)
> plot(model.miss$USsire)
```

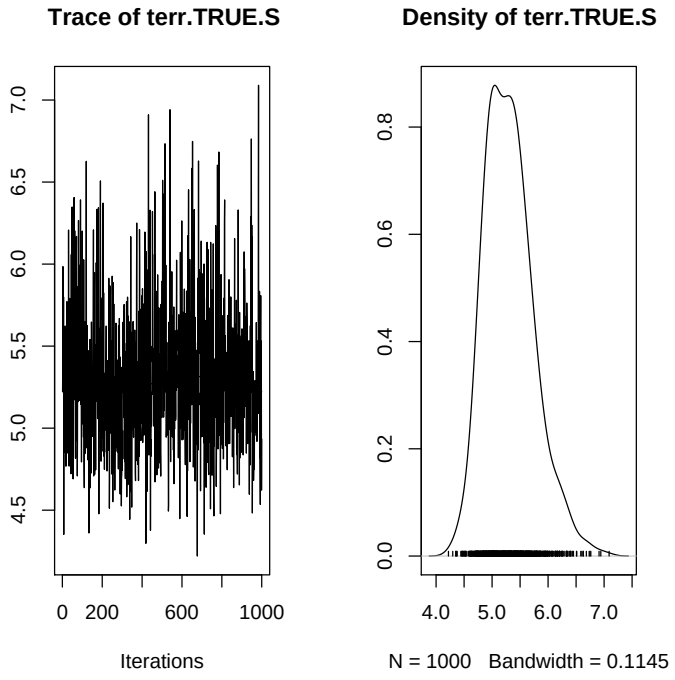


Figure 16: The marginal posterior distribution of $\beta$ from `model.miss`. This is the log odds ratio of a male scoring paternity if it had been on the offspring's territory compared to if it had been on a different territory. The pedigree is sampled assuming that the phenotypes of unsampled males are unknown.

The estimate of $\beta$ seems reasonable given that `logit(EPP)=5.28` (Figure 16), but the number of unsampled males is much smaller than we anticipated (Figure 17). This is beacuse we left `USvar=NULL`, and approximated the summed linear predictors of unsampled males from the linear predictors of sampled males (see Section 2.5). In fact the probability that an unsampled male gains paternity over a sampled male are not equal, since unsampled males are always extra-pair fathers but sampled males may be within-pair fathers.

When `USvar=NULL`, the records of unsampled fathers do not enter into the likelihood equation for $\beta$, and so $\beta$ is valid. However, our primary interest was

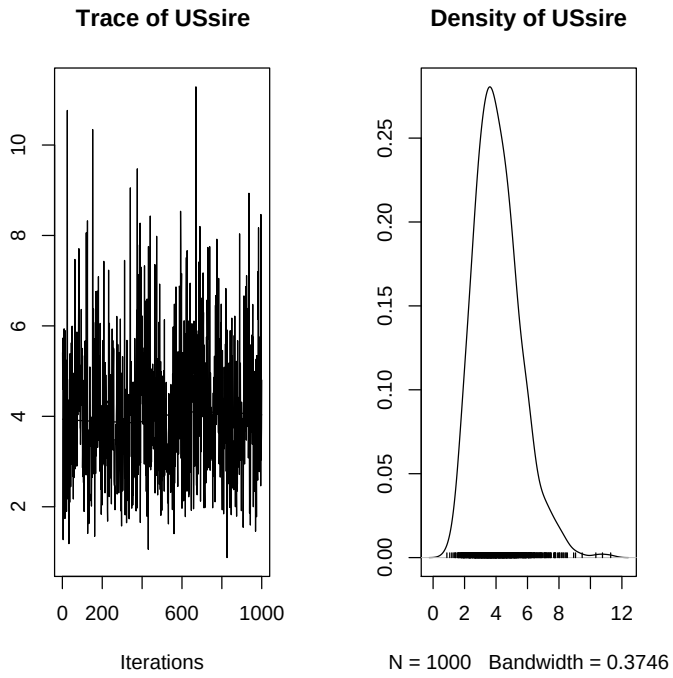


Figure 17: The marginal posterior distribution of the number of unsampled males from `model.miss`, when the pedigree is sampled assuming that the phenotypes of unsampled males are unknown. The actual number is 25!

in the within-pair paternity rate, $\beta_o$, and we can use equation 18 to obtain the posterior

```
> theta <- inv.logit(model.miss$beta)
> NW <- 1
> NE <- 24
> theta_o <- (NW * theta)/(NW * theta + NE * (1 - theta))
> plot(mcmc(theta_o))
```

Given the mode of the posterior of $\beta$ is very close to 5.28 (Figure 16), it seems surprising that the mode of the within-pair paternity rate is not 0.8 (Figure 18). We can go one step better and specify that the phenotypes of the unsampled males are known

```
> var2 <- expression(varPed(x = "terr", gender = "Male", relational = "OFFSPRING",
+     USvar = FALSE))
> PdP_miss2 <- PdataPed(formula = list(res1, var2, res2), data = data_EPP_miss,
```

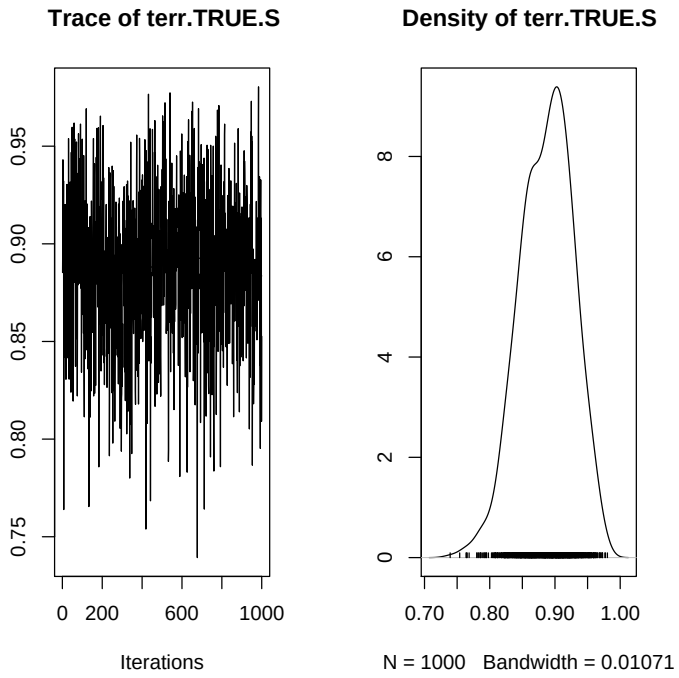


Figure 18: The marginal posterior distribution of $\beta_o$ derived from `model.miss`. $\beta_o$ should be the level of within pair paternity in the population (80%) but we can see that this value has a small posterior probability. This may be down to sampling error alone, but in fact the estimate is consistently biased upwards because the pedigree is sampled assuming that the phenotypes of unsampled males are unknown.

```
+     USsire = TRUE)
> model.miss2 <- MCMCped(PdP = PdP_miss2, GdP = GdP_miss, verbose = FALSE)
> plot(model.miss2$beta)
> plot(model.miss2$USsire)
```

Once again $\beta$ is estimated correctly (Figure 19), but this time the number of unsampled males is estimated correctly, although considerable uncertainty remains (Figure 20). In part this is due to the moderate levels of genotyping error.

We can derive the posterior distribution for within pair paternity by combining the posterior distribution of /*beta* with the number of unsampled males, which we have assumed are extra-territorial males.

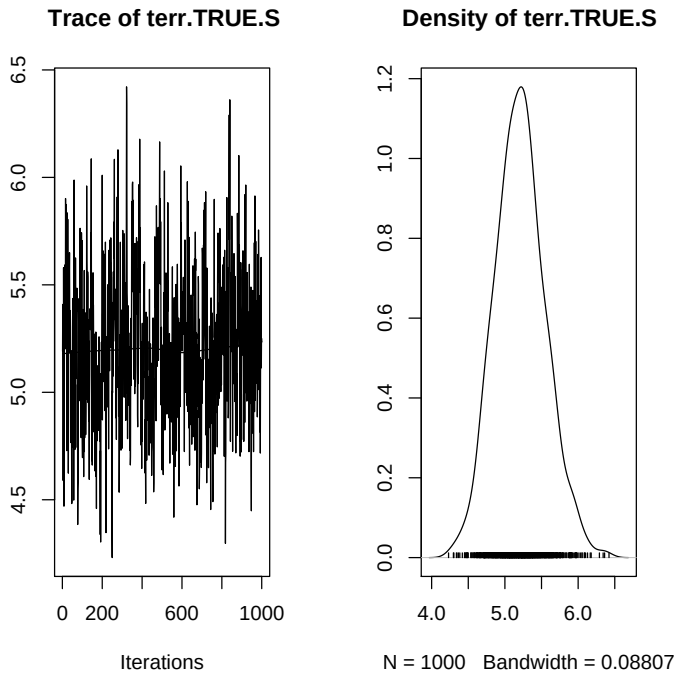


Figure 19: The marginal posterior distribution of $\beta$ derived from `model.miss2`. As in Figure **??** this is the log odds ratio of a male scoring paternity if it had been on the offspring's territory compared to if it had been on a different territory. However, in this model the pedigree is sampled assuming that the unsampled males have to be extra-pair, and these males contribute to the likelihood of $\beta$.

```
> theta <- inv.logit(model.miss2$beta)
> NW <- 1
> NE <- 24 + model.miss2$USsire
> theta_o <- (NW * theta)/(NW * theta + NE * (1 - theta))
> plot(mcmc(theta_o))
```

Alternativley we can fit within-pair paternity explicitly using the argument `merge`.

```
> var3 <- expression(varPed(x = "terr", gender = "Male", relational = "OFFSPRING",
+     USvar = FALSE, merge = TRUE))
> PdP_miss3 <- PdataPed(formula = list(res1, var3, res2), data = data_EPP_miss,
+     USsire = TRUE)
> model.miss3 <- MCMCped(PdP = PdP_miss3, GdP = GdP_miss, verbose = FALSE)
> theta_o <- inv.logit(model.miss3$beta)
```

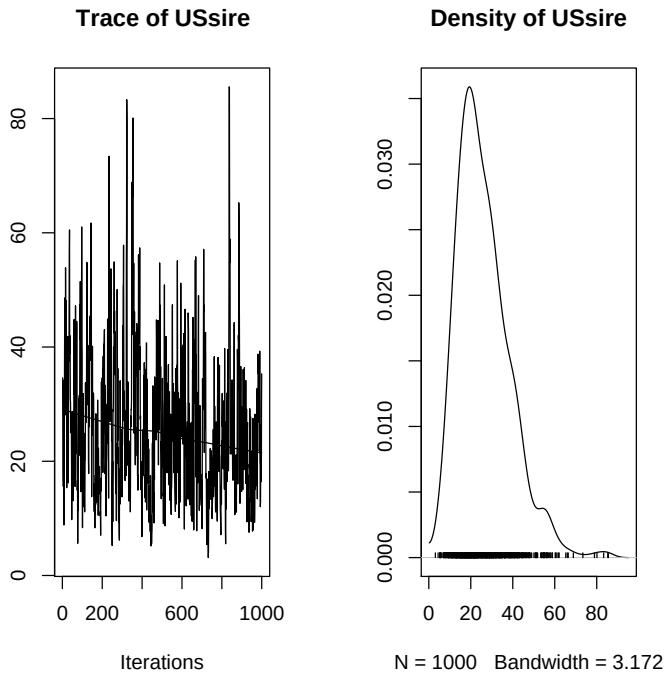


Figure 20: The marginal posterior distribution of the number of unsampled males from `model.miss2`. the true number is 25.

```
> plot(mcmc(theta_o))
```

In this example the estimate of within-pair paternity in `model.miss2` and `model.miss3` are equivalent, but for analyses where the set of males varies between offspring the two models would not be equivalent. For example, if the aim was to model a constant extra-pair paternity rate across years but the male population size fluctuated between years then the argument `merge=TRUE` would have to be specified. In this instance, a *post-hoc* transformation of $\beta$ and the number of unsampled males would not yield a valid posterior for $\beta_o$ as it did in the simple example above (Figure 21).

## 3.7 Assortative Mating and Heritability

Assortative and disassortative mating can modelled using the argument `relational="MATE"` in `varPed`. This sets up a variable for the distance between male and female phenotypes. If the variable is numeric then this distance is Euclidean, if the variable is categorical then the distance is simply one or zero, depending on whether the sexes belong to the same category or not. Below we

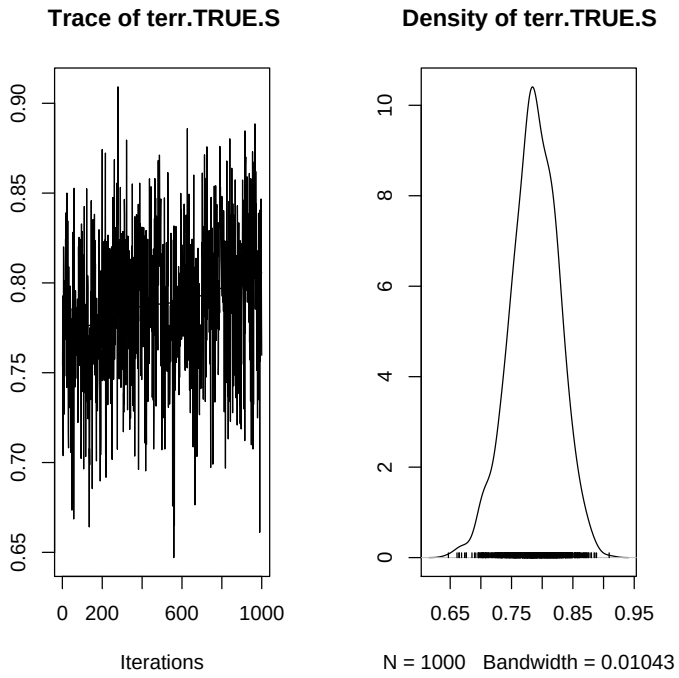


Figure 21: The marginal posterior distribution of within-pair paternity estimated from the posterior distributions of $\beta$ and the number of unsampled males from `model.miss2`.

will consider a population of 10 males, 10 females, and 30 offspring. There are 2 mating types, + and -, and they are distributed evenly across and within the sexes. The aim is to test whether unions such +/+ and -/- occur more often than +/- unions, than would be expected by chance. We will simulate data where assortative unions are 3 times more likely than disassortative unions than would be expected under random mating.

$$p_{i,j}^{(o)} \propto \exp(\beta_1 \delta_{i,j}) \tag{20}$$

$\delta_{i,j}$ takes on the value one when the mating type of female $i$ matches the mating type of male $j$, and $\beta_1$ in this example is log(3).

```
> id <- 1:50
> sex <- rep(c("Male", "Female"), each = 10, length = 50)
> offspring <- c(rep(0, 20), rep(1, 30))
> MT <- gl(2, 1, 50, labels = c("+", "-"))
> test.data <- data.frame(id, offspring, MT, sex)
```

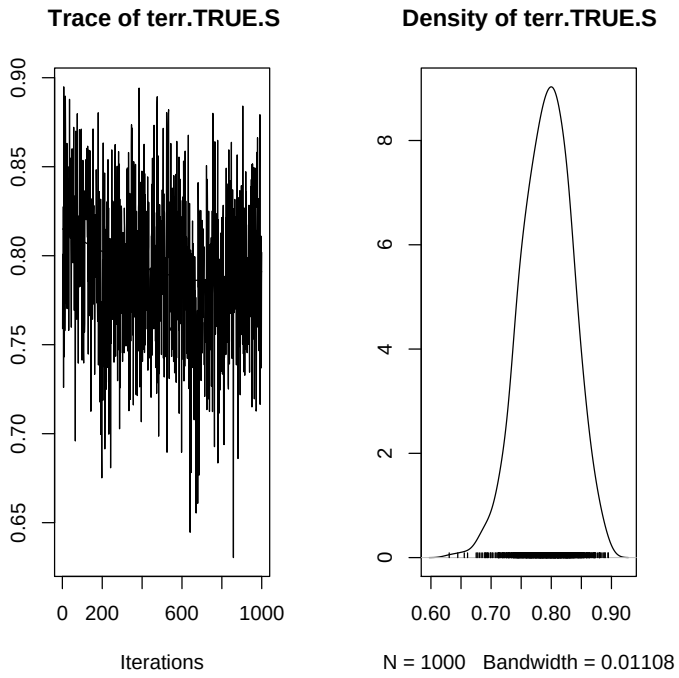


Figure 22: The marginal posterior distribution of within-pair paternity estimated directly in `model.miss3`. The posterior distribution is exactly equivalent to that shown in Figure 21.

```
> res1 <- expression(varPed("offspring", restrict = 0))
> var2 <- expression(varPed(c("MT"), gender = "Female", relational = "MATE"))
> PdP <- PdataPed(formula = list(res1, var2), data = test.data)
> simped <- simpedigree(PdP, beta = logit(0.75))
> table(MT[as.numeric(simped$ped[, 2])] == MT[as.numeric(simped$ped[,
+     3])])

FALSE  TRUE
    7    23

> G <- simgenotypes(A = A, E1 = 0.005, E2 = 0.005, ped = simped$ped,
+     no_dup = 1)
> GdP <- GdataPed(G = G$Gobs, id = G$id)
> model.ass.mat <- MCMCped(PdP = PdP, GdP = GdP, verbose = FALSE)
> plot(mcmc(inv.logit(model.ass.mat$beta)))
```

Mating types may well be inherited, and if we had data on the mating types of offspring we may be inclined to also model the distance between off-

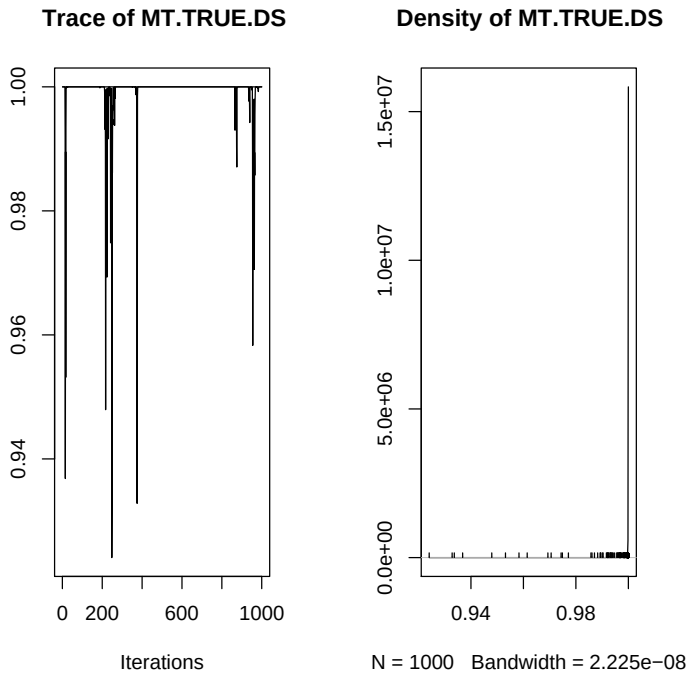


Figure 23: The posterior distribution of $\text{logit}^{-1}(\beta)$. In this instance the frequencies of the two mating types are exactly 0.5 both within sexes, and between the sexes, and this is the expected proportion of offspring resulting from the union of +/+ and -/- parents.

spring and parental phenotypes using the argument `realtional="OFFSPRING"`. A cautionary note should be made about using `realtional="OFFSPRING"` to model the transmission of phenotypes between parents and offspring. If the individuals in the above example were haploid, and mating type was under the control of a single locus then the function `realtional="OFFSPRING"` may capture the genetic process quite well. On the otherhand, if the organisms were diploid then the model would certainly not be consistent with a genetic process. Likewise, parameters associated with continuous variables derived using `relational="OFFSPRING"` should not be interpreted as a measure of heritibility in the quantitative genetic sense, although they will be related. Future work is planned in this direction.

## 3.8 Longitudinal Data and Multigenerational Pedigrees

MasterBayes is able to work with longitudinal data, allowing the possibility of reconstructing multigenerational pedigrees. To work with longitudinal data a time variable needs to be passed to `PdataPed`. It is VERY important that inidviduals with offspring records do not appear as potential parents of offspring with records in the same cohort, or a previous cohort. The argument `restrict` can be passed to `varPed` to ensure this does not happen. By not doing this you are allowing for the possibility that an individual can be both the parent and an offspring of another.

```
> id <- 1:50
> year <- rep(1:4, each = 50)
> for (yr in 1:3) {
+     id <- c(id, sample(id[year == yr], 25))
+     id <- c(id, max(id) + 1:25)
+ }
> sex <- rep(c("Male", "Female"), max(id))[id]
> off.within.cohort <- c(rep(0, 25), rep(1, 25))
> offspring <- c(rep(0, 50), rep(off.within.cohort, 3))
> lat <- runif(200)
> long <- runif(200)
> data.ped <- data.frame(id, sex, offspring, lat, long, year)
> res1 <- expression(varPed(x = "offspring", restrict = 0))
> res2 <- expression(varPed(x = "year", relational = "OFFSPRING",
+     restrict = TRUE))
> var1 <- expression(varPed(x = c("lat", "long"), lag = c(0, 0),
+     relational = "OFFSPRING"))
> PdP <- PdataPed(formula = list(res1, res2, var1), data = data.ped,
+     timevar = year)
> P <- simpedigree(PdP, beta = -1)$ped
> G <- simgenotypes(A = A, ped = P)
> GdP <- GdataPed(G = G$Gobs, id = G$Gid)
> model.ped <- MCMCped(PdP, GdP, mm.tol = 1, verbose = FALSE)
> plot(model.ped$beta)
```

Several problems currently exist with reconstructing multigenerational pedigrees using MasterBayes. The first, and perhaps the biggest problem is the assumption that those individuals classed as coming from the base population (including unsampled individuals) are actually unrelated. For example, if the mother of two sisters (A and B) was not sampled, but sister B was in the set of potential mothers for sister A, then there is a high chance that sister B would be picked as sister A's mother [**?**]. Modelling the possibility that sampled individuals may be related through unsampled individuals is a challenging problem with a long history. In the future I may incorporate the reversible jump MCMC methods outlined in **?** to try and alleviate the problem that full-siblings without

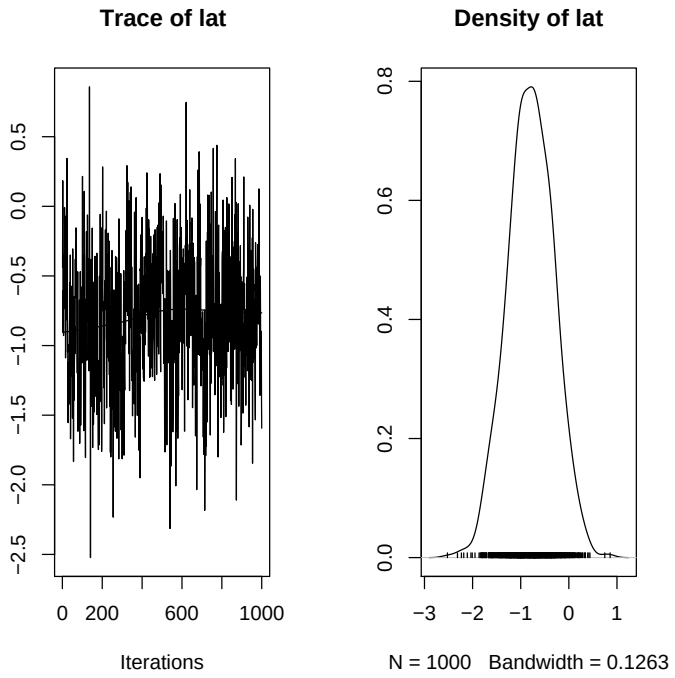


Figure 24: The posterior distribution of `beta`, the rate at which the probability of parentage drops with distance from the offspring. This parameter was estimated from a multigenerational pedigree where longitudinal data had been collected. In this toy example, the model was consistent with the data, but for multigenerational problems appropriate models are hard to construct and caution needs to be exercised.

sampled parents cause. A second problem with the reconstruction of multigenerational pedigrees, which has recieved much less attention is the problem of inbreeding avoidance. The likelihood of a genotype configuration given a pedigree can be calculated using the Elston-Stewart algorithm [**?**], which is based on the product of the Mendelian transition probabilities across offspring. Most parentage and sib analyses are special cases or approxiamtions of this algorithm. However, a pedigree has both marriages and births, and an implicit assumption when calculating the likelihood of genotypes given a pedigree is that marriages are independent of genotype. When inbreeding avoidance, or obligatory selfing is practised, this assumption breaks down, and could presumably compromise pedigree reconstruction. The next version of MasterBayes will incorporate a method for modelling inbreeding avoidance.

## 3.9 Hermaphrodites and Selfing Rates

An extreme form of inbreeding is selfing in hermaphrodites. MasterBayes is able to work with hermaphrodite systems by not passing a `sex` vector to `PdataPed`. Currently, MasterBayes can be forced to model selfing in hermaphrodites although it is inefficent (the following example took 5 minutes depsite a mismatch tolerance of 1). More efficient methods will be developed in the future, but a small example will highlight the logic behind the approach, and some of its pitfalls.

```
> id <- as.factor(1:100)
> offspring <- c(rep(0, 25), rep(1, 75))
> Herm <- data.frame(id, offspring)
> res1 <- expression(varPed(x = "offspring", restrict = 0))
> var1 <- expression(varPed(x = "id", relational = "MATE"))
> PdP <- PdataPed(formula = list(res1, var1), data = Herm)
> P <- simpedigree(PdP, beta = logit(0.9))$ped
> G <- simgenotypes(A = A, ped = P)
> GdP <- GdataPed(G = G$Gobs, id = G$Gid)
> model.herm <- MCMCped(PdP, GdP, mm.tol = 1, verbose = FALSE)
> plot(inv.logit(model.herm$beta))
```

The model is essentialy one of assortative mating for `id` although the genotype updating algorithm becomes a little more complex. `logit(beta)` should be interpreted as the probability that an individual mates with itself compared to another random individual. As in Section 3.6, Equation 18 can be used to get the posterior distribution for the expected proportion of offspring produced by selfing. An important conisderation when the system does not mate randomly is the assumption of Hardy-Weinberg equilibrium. When evaluating Equation 14, genotype frequencies enter into the probability for the true genotypes of base individuals and offspring with one or more unsampled parents. These genotype frequencies are calculated from the allele frequencies under the assumption of Hardy-Weinberg equilibrium. For those individuals that have been typed, however, the probability is largely dominated by the observed genotype data rather than the genotype frequencies. However, when unsampled parents are present an approximation for the genetic likelihood is used 2.5 that is more reliant on the information imparted by genotype frequencies, and care needs to be taken.

## 3.10 Schrodinger's Hermaphrodite Cat

Monoecy in plants is a common phenomenon; each flower is unisexual but flowers of both sex can be found on the same plant. The genotype data (unless the endosperm or organelles have been typed) provide no information whether an offspring is produced from the male or female flowers of an individual, the only information the genotype data provide is whether the individual is a parent. However, let's say we measure the proportion of male flowers on the adult plants

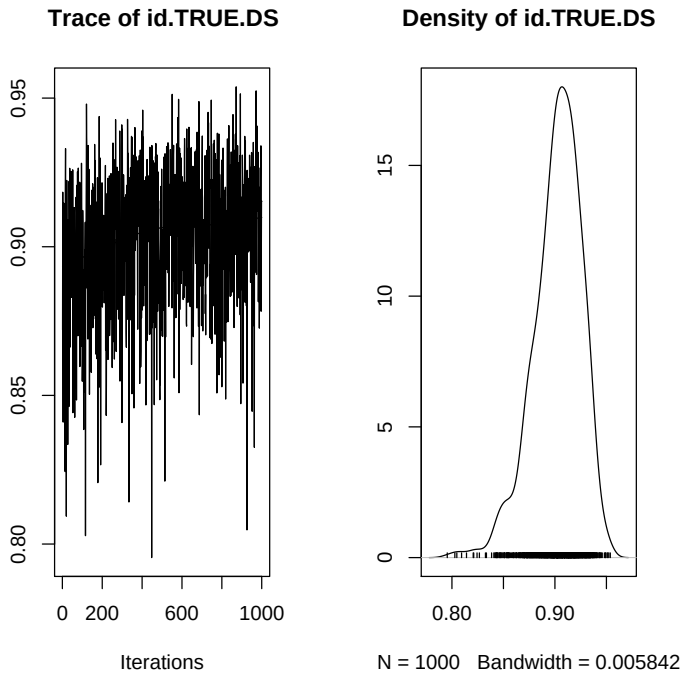


Figure 25: The posterior distribution of $\text{logit}^{-1}\beta$: the probability that an individual mates with itself compared to another random individual in a hermaphrodite system.

and this measure positively covaries with pollen production but negatively covaries with seed production. Alternatively, we could have measured something less sugestive like the distance beween offspring and parents. MasterBayes can fit gender specific variables to hermaphrodite data but the resulting posteriors are ambiguous with respect to any notion of gender. To illustrate we will simulate data from a monoecious population in which the proportion of male flowers on each individual varies substantially. The number of offspring produced by each individual is independent of the proportion of male flowers, but individuals with proportionally more male flowers produce more offspring through pollination. We will start with a very strong relationship between 'paternity' and the number of male flowers.

```
> id <- as.factor(1:100)
> offspring <- c(rep(0, 25), rep(1, 75))
> Prop.male.flowers <- rbeta(100, 10, 10)
> SeedPollen <- data.frame(id, offspring, Prop.male.flowers)
> res1 <- expression(varPed(x = "offspring", restrict = 0))
```

```
> var1 <- expression(varPed(x = "Prop.male.flowers", gender = "Male"))
> var2 <- expression(varPed(x = "Prop.male.flowers", gender = "Female"))
> PdP <- PdataPed(formula = list(res1, var1, var2), data = SeedPollen)
> P <- simpedigree(PdP, beta = c(-7.5, 7.5))$ped
> G <- simgenotypes(A = A, ped = P)
> GdP <- GdataPed(G = G$Gobs, id = G$Gid)
> model.mon <- MCMCped(PdP, GdP, mm.tol = 1, verbose = FALSE)
> plot(model.mon$beta)
```

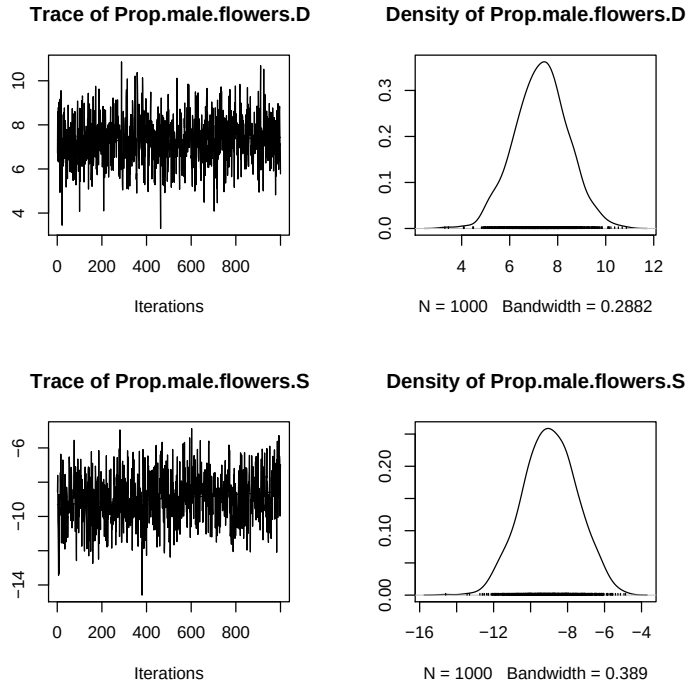


Figure 26: The posterior distribution of $\beta$: the parameters of the log-linear relationship between the number of male flowers on a monoecious plant and the probability of producing offspring from female and male flowers, respectively. The relationships are very strong and the chain gets stuck in a region of parameter space of high probability. A mirror image of this distribution exists outside of the parameter space sampled.

Figure 26 looks reasonable but what happens if we simulate some data where the proportion of male flowers is not such a good predictor.

```
> P <- simpedigree(PdP, beta = c(-5, 5))$ped
> G <- simgenotypes(A = A, ped = P)
```

```
> GdP <- GdataPed(G = G$Gobs, id = G$Gid)
> model.mon2 <- MCMCped(PdP, GdP, mm.tol = 1, verbose = FALSE)
> plot(model.mon2$beta)
```

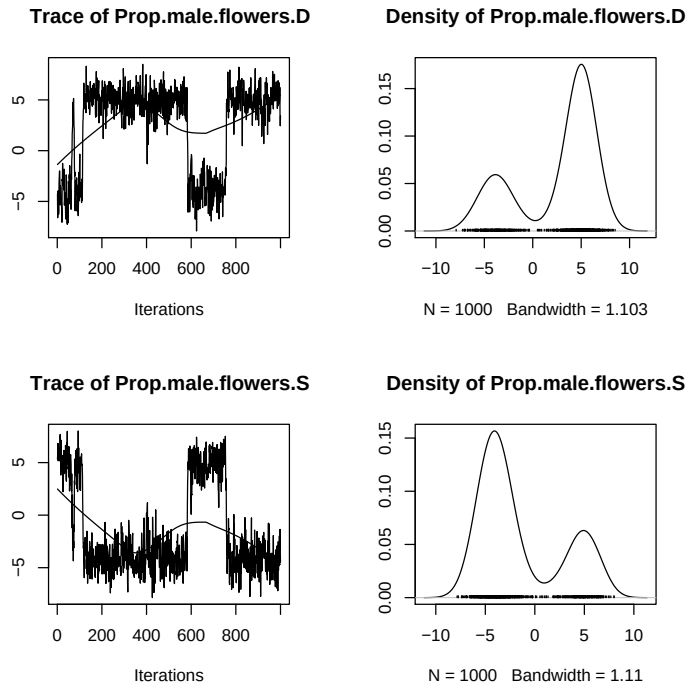


Figure 27: The posterior distribution of $\beta$: the parameters of the log-linear relationship between the number of male flowers on a monoecious plant and the probability of producing offspring from female and male flowers, respectively. The relationships are not so strong as in Figure 26, and the chain is able to sample from the full posterior, albeit with a lot of autocorrelation.

We see that the posterior distribution is actually similar bimodal! We can be fairly confident that the proportion of male flowers indicates the probability of polination versus seed production, but we have no way of saying what the sign of the relationship is: whether the proportion of male flowers is positively correlated with pollination and negatively correlated with seed production or vica versa. We could probably make an informed guess, but with more amiguous varables it may not be possible to do so. If any one has any idea on how to set up a sensible prior for this type of model, please email me.

# A A lightening tour of model specification

$p_{i,j}^{(o)}$ is the probability that female $i$ and male $j$ are the parents of offspring $o$. `x` are explanatory variable(s) and $\beta$ the vector of associated parameter(s). $t$ indicates the time (`timevar` in a `PdataPed` object) to which the offspring record belongs. For continous variables...

`varPed(x, gender="Female")`

$$p_{i,j}^{(o)} \propto \exp(\beta_1 \mathtt{x}_i ...) \tag{21}$$

`varPed(x, gender="Male")`

$$p_{i,j}^{(o)} \propto \exp(\beta_1 \mathtt{x}_j ...) \tag{22}$$

`varPed(x)`

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (\mathtt{x}_i + \mathtt{x}_j) ...) \tag{23}$$

`varPed(x, gender="Female", relational="OFFSPRING")`

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (|\mathtt{x}_i - \mathtt{x}_o|) ...) \tag{24}$$

`varPed(x, gender="Female", relational="MATE")`

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (|\mathtt{x}_i - \mathtt{x}_j|) ...) \tag{25}$$

`varPed(x, gender="Female", lag=c(-1,-1))`

$$p_{i,j}^{(o)} \propto \exp(\beta_1 \mathtt{x}_{i,t-1} ...) \tag{26}$$

`varPed(x, gender="Female", lag=c(-1,-1), relational="OFFSPRING")`

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (|\mathtt{x}_{i,t-1} - \mathtt{x}_{o,t}|) ...) \tag{27}$$

`varPed(x, gender="Female", lag=c(0,0), relational="MATE", lag_relational=c(-1,-1))`

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (\mathtt{x}_{i,t} + \mathtt{x}_{j,t-1}) ...) \tag{28}$$

For a categorical variable with two levels (`A` and `B`) the model specified by `varPed(x, gender="Female")` takes on the form

$$p_{i,j}^{(o)} \propto \exp(\beta_1 \delta_i ...) \tag{29}$$

where $\delta_i$ is an indicator variable taking the value 1 if $\mathtt{x}_i$ is equal to the first level of `x` and zero otherwise. $\beta_1$ is then the log odds ratio of the two levels of `x` with respect to maternity. If `merge=TRUE` is specified then $\beta_1$ may vary across offspring, and $\beta_o$ is estimated. $\beta_o$ is related to $\beta_1$:

$$\beta_o = \text{logit}\left[\frac{\theta N_A}{\theta N_A + (1-\theta) N_B}\right] \tag{30}$$

where $\theta$ is the inverse logit transformation of $\beta_1$, and $N_A$ and $N_B$ are the number of potential mothers that have level `A` and `B` for `x`. If $N_A$ and $N_B$ are invariant over offspring the models are functionally equivalent.

The denominator of the multinomial likelihood is the summed linear predictors of all possible parents (after setting up a contrast with the baseline parents) [**?**]. Designating the first set of parents as baseline, the contrast for each set of parents is simply:

$$\eta_{i,j}^{(o)} = \log\left[\frac{p_{i,j}^{(o)}}{p_{1,1}^{(o)}}\right] \tag{31}$$

and the likelihood of $\boldsymbol{\beta}$

$$Pr(x|\boldsymbol{\beta}) = \prod_{o}^{n_o}\left[\frac{\exp(\eta_{d,s}^{(o)})}{\sum_{i=1}^{n_i^{(o)}}\sum_{j=1}^{n_j^{(o)}}\exp(\eta_{i,j}^{(o)})}\right] \tag{32}$$

where $n_o$, $n_i^{(o)}$ and $n_j^{(o)}$ are the number of offspring, the number of potential mothers for offspring $o$, and the number of potential fathers for offspring $o$, respectively. $d$ and $s$ are the actual parents of offspring $o$. The set of possible parents in the denominator of the multinomial likelihood are those that are not excluded using the argument `restrict`. However, if the argument `keep=TRUE` is used then the denominator of the likelihood will include excluded parents depsite the fact that $d \neq i$ and $s \neq j$.