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

J. D. Hadfield

June 22, 2022

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. I assume a basic familiarity with R. All example code can be extracted by using the `Stangle` function on the file `Tutorial.Rnw` in the `MasterBayes/inst/doc/Figures` folder.

```
> library("MasterBayes")
```

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 3GHz 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. There have been many updates and bug fixes since the original version of MasterBayes, and these used to be documented as footnotes in this manual. I have now updated the manual and absorbed them into the body of the text.

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 give 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 introductory texts on the subject; my favourite is Bayesian Data Analysis [Gelman et al., 2004].

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 β_1 and β_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).

For some very simple models the posterior distribution can be derived analytically. Let's imagine that the equation describing the joint posterior distribution

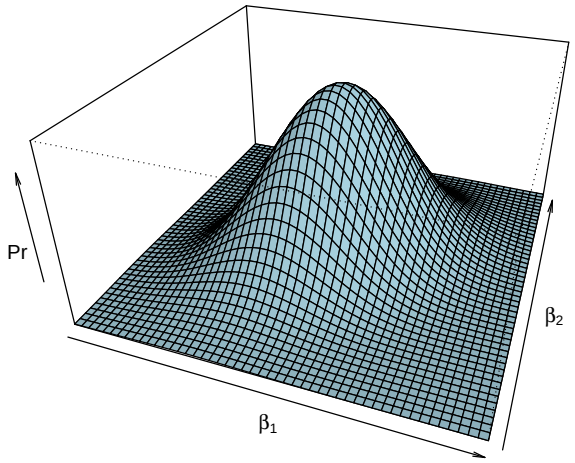

Figure 1: Bivariate posterior distribution of $\boldsymbol{\beta}$

of β_1 and β_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 β_1 and β_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 β_1 axis as a measure of the precision with which we have estimated β_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 β_1 and β_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 β_1 and β_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 coordinates 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 determine the place from which the chain is launched, the coordinates 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 coordinates are picked from a multivariate normal distribution that is centered on the initial coordinates $\boldsymbol{\beta}^{t=0}$, and has a user specified variance. We will denote this new set of coordinates as $\boldsymbol{\beta}^{new}$. The question then remains whether to move to this new set of coordinates or remain at our current coordinates $\boldsymbol{\beta}^{t=0} = \boldsymbol{\beta}^{old}$. If the height of the distribution at the new set of coordinates 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 lower 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 relative difference between the heights of the posterior distribution at the two coordinates. 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 approximation of the 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 coordinates under the peak of the distribution in Figure 1, 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 coordinates would lie outside of Figure 1 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 iterations before we could generate a histogram that was an adequate approximation of 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 co-

ordintates $\boldsymbol{\beta}^{new}$ and $\boldsymbol{\beta}^{old}$ would be very close, and the chain would move almost every iteration since the difference in heights of the posterior at the two coordinates would be tiny. Although the chain is continuously moving it is travelling such small distances each iteration that the chain would have to pass through many iterations before it adequately covered parameter space.

`tunePed` objects control the variance of the proposal distributions for parameters updated by Metropolis-Hastings updates. 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 assessed by specifying `verbose=TRUE` in `MCMCped`. The Metropolis-Hastings acceptance rates are then printed to the screen during model fitting.

1.2 Gibbs Sampling

Gibbs sampling is a special case of Metropolis-Hastings updating, 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 simultaneously. An equally valid approach would have been to set up two Metropolis-Hastings schemes where the chain was first allowed to move along the β_1 axis, and then along the β_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 β_1 given that $\beta_2 = 0$:

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

In some cases, the equation that describes this conditional distribution can be derived despite the equation for the complete joint distribution of Figure 1 remaining unknown. When the conditional distribution of β_1 is known we can use Gibbs sampling. Lets say the chain at a particular iteration is located at zero for β_2. If we updated β_1 using a Metropolis-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 β_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 β_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 specify the variance of a proposal distribution.

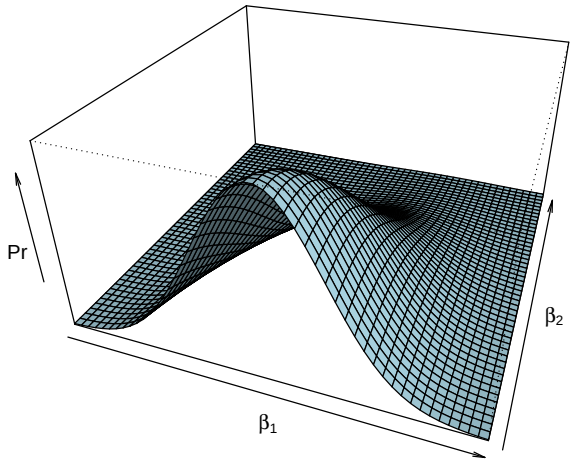

Figure 2: Bivariate posterior distribution of $\boldsymbol{\beta}$, for postive values of β_2

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 successive iterations are usually not independent and care needs to be taken regarding the validity of the approximation. `MCMCped` returns the Markov chain for continuous parameters as `mcmc` objects, which can be analysed using the `coda` package. The function `autocorr` estimates the level of non-independence between succes-

sive samples in the chain for continuous parameters. For discrete parameters, assessing auto-correlation is harder but the function `autocorrP` will calculate an auto-correlation metric for parentage assignments. When auto-correlation 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 were used to initialise the chain. In many cases the first iterations show a strong dependence on the starting parameterisation, 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 iterations 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 achieve. If however, the posterior was multimodal convergence would be harder to achieve and diagnose.

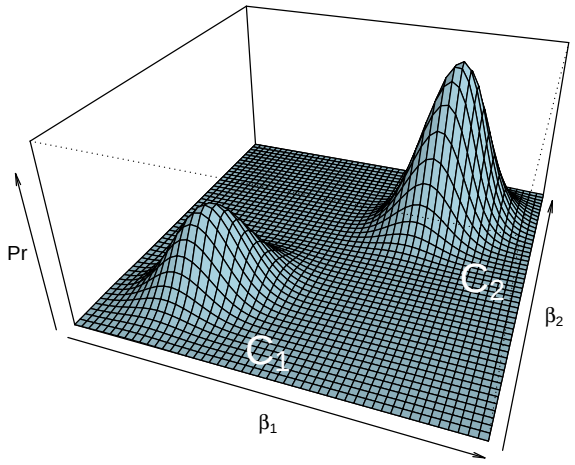

Figure 3: A bimodal bivariate posterior distribution of $\boldsymbol{\beta}$.

For example, imagine a chain was initailised at C_1 in Figure 3. The chain may provide valid approximates for the region of high probability at small values of β_1 and β_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 parameterisations (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 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, which have a vague Dirichlet prior. 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 would be compromised. For some models the data will contain enough information to make the posterior proper despite an improper prior specification. However, for chains in which a sampled pedigree has little or no structure (i.e. all parents are unsampled), or no sampled parents exist for some level of a categorical predictor variable, then problems can occur unless informative priors are used. This last problem is often encountered in real data-sets and has been referred to as the extreme category problem. For example, the categorical variable `'recorded in that year/only recorded in the previous year'` may be fitted to account for individuals that were not recorded but may have been alive. If these individuals really were dead then the estimated probability of them having offspring tends to zero which is $-\infty$ on the logit scale. This problem can usually be identified by the chain making long excursions into extreme values of the parameter space, and can be solved by placing a prior on the parameter. I often use a prior with mean zero and variance $\pi^2/3$, as this is approximately uniform between 0 and 1 on the probability scale. In the extreme case, models in which parameters are confounded, a proper prior distribution is required to make the parameters identifiable. Imagine that all territorial males were old, and all non-territorial males were young. If we tried to fit the two parameters, β_1 and β_2, in a single model we would run in to difficulties. The parameters are not identifiable because we cannot distinguish between the two alternatives; whether territorial males gain more paternity because they hold a territory, or because they are older. The likelihood surface for $\boldsymbol{\beta}$ would look something like Figure 4, with a

ridge of high probability extending to infinity.

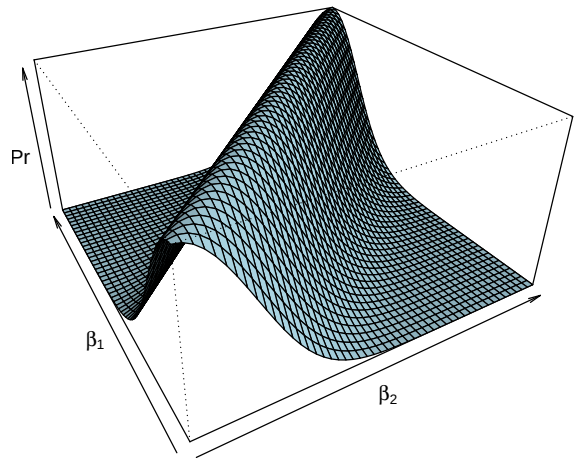

Figure 4: The likelihood of $\boldsymbol{\beta}$, when the two parameters are not identifiable from the data, $\boldsymbol{y}$

In this instance 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 identifiable.

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 threatened Seychelles warbler [Richardson et al., 2001].

```
> 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 [Richardson et al., 2000].

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 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 modeling 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 (β) is the exponential rate parameter, and we are interested in its probability distribution given the data we have collected:

$$Pr(\beta|\boldsymbol{G}_{obs}, \texttt{lat}, \texttt{long}, \texttt{terr}). \tag{3}$$

$\boldsymbol{G}$ represents genotypes, and we use the subscript $_{obs}$ to indicate observed rather than actual genotypes. This distinction is necessary when genotyping error is present (see Section 2.6). Evaluating the problem in this form is intractable, but the problem can be simplified by augmenting with the pedigree, $\boldsymbol{P}$:

$$\int_{\boldsymbol{P}} Pr(\beta, \boldsymbol{P}|\boldsymbol{G}_{obs}, \texttt{lat}, \texttt{long}, \texttt{terr}) d\boldsymbol{P}. \tag{4}$$

Equations 3 and 4 are equivalent. In the latter case we estimate the pedigree, but we also integrate over any uncertainty that may remain regarding its structure, leaving us with the marginal distribution of β. 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 evaluated as part of the model formula in a `PdataPed` object

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

`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 specifies 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, although if V is appended to the relational argument (e.g.relational="OFFSPRINGV") then the actual difference between the two phenotypes is used rather than the absolute difference. When x is a factor, the transformed variable is a logical vector with TRUE indicating that x for the two individuals 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="==" to retain mothers that have the same territory as the offspring for the transformed variable. Females that are on a different territory to the offspring are excluded as mothers. This is essentially a strong prior on the parameter associated with within territory/extra territory maternity, but is computationally faster because excluded mothers can be discarded for particular offspring.

We also need to exclude individuals 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 β, 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 coordinates of each individual 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 indicating 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` specifications can be found in Appendix A. $p_{i,j}^{(o)}$ 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 variable indicating 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 (see Sections 3.9 and 3.10). 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 for co-dominant markers (see the package genetics), or `genotypeD` objects for dominant markers. A list of `genotype` or `genotypeD` 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. Individuals 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`. The argument `marker.type` specifies whether the genotypes are dominant (`"AFLP"`) or co-dominant with genotyping errors occurring according to the model of Wang [2004] (`"MSW"`) or CERVUS [Marshall et al., 1998] (`"MSC"`). 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 estimated. Maximum likelihood or heuristic starting parameterisations are used by default, and all estimable parameters are estimated, unless otherwise specified. These parameters may include the pedigree (`P`), genotypes (`G`), base population allele frequencies (`A`), genotyping error rates (`E1` and `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 computationally intensive, and preliminary analyses can usually be carried out using approximate methods. A commonly used approximation for dealing with genotyping error is to integrate out any uncertainty prior to the parentage analysis (See Section 3.1). We can use the approximation by specifying `estG=FALSE`; if we were to use the exact solution we would be required to estimate the actual genotypes of all individuals. When `estG=FALSE`, error rates and allele frequencies cannot be estimated and need to be specified. Allele frequencies, if not explicitly specified, are taken directly from the genotype data using the function `extractA`.

2.3 Categorical Estimation

The most popular software for parentage analysis is CERVUS [Marshall et al., 1998]. MasterBayes can fit CERVUS type models as a special case, although a major differences exist between the MasterBayes implementation and the CERVUS implementation. In MasterBayes confidence in parentage assignments is assessed at the level of individual assignments and the measure of confidence uses all the information provided by potential parents. CERVUS on the other hand assesses confidence at the level of the population, and only uses the information provided by the two most likely parents [Nielsen et al., 2001]. 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 to be used for genotyping error, and we need to provide point estimates for the allele frequencies and the genotyping error rate (see Section 2.2). By default MasterBayes will not use CERVUS's model of genotyping error but uses the model devloped in Wang [2004]. CERVUS's model can be fitted by passing the argument `marker.type="MSC"` but we will stick with Wang's model and leave the error rates at their default values of 0.005 (see Section 2.6). We'll also specify that the number of unsampled males is not to be estimated (see Section 2.5), and use a point estimate of 10 instead.

```
> sP<-startPed(estG=FALSE, E1=0.005, E2=0.005, A=extractA(WarblerG),
+ estUSsire=FALSE, USsire = 10)
```

CERVUS breaks the problem down into two stages. In the first 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, β is estimated. We start by estimating the pedigree using the function `MCMCped` which returns samples of the posterior distribution of all unknowns.

```
> 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. Because the only unknown 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$P, threshold=0)
> ped1$P[1:10,]

      [,1]      [,2]      [,3]
 [1,] "N748835" "K278102" "N021989"
 [2,] "Z993558" "K278102" "K278027"
 [3,] "N748825" "K278115" "N021716"
 [4,] "N748841" "N021991" "VE80200"
 [5,] "N748842" "J368410" "J368403"
 [6,] "N748848" "J854303" "K278101"
 [7,] "Z993553" "K278011" "J368403"
 [8,] "N748838" "N021965" "N021760"
 [9,] "N748827" "K278018" "N021839"
[10,] "Z993562" "K278018" "N021707"
```

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 column is designated as `USsire`. For such a simple model, Markov chain Monte Carlo is redundant as the posterior distribution of the pedigree can be calculated analytically.

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

      [,1]      [,2]      [,3]
 [1,] "N748835" "K278102" "N021989"
 [2,] "Z993558" "K278102" "K278027"
 [3,] "N748825" "K278115" "N021716"
 [4,] "N748841" "N021991" "VE80200"
 [5,] "N748842" "J368410" "J368403"
 [6,] "N748848" "J854303" "K278101"
 [7,] "Z993553" "K278011" "J368403"
 [8,] "N748838" "N021965" "N021760"
 [9,] "N748827" "K278018" "N021839"
[10,] "Z993562" "K278018" "N021707"
```

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. All code is ran afresh each time this manual is updated, so for some readers there may be some discrepancies between the MCMC and ML output due to Monte Carlo error, when two or more fathers have almost identical probabilities of being the true father. We can extract the posterior probabilities for these assignments estimated by the different methods to assure ourselves that this is the case:

```
> discrepancies<-which(ped1$P[,3][1:10]!=ped2$P[,3][3:12])
> ped1$prob[1:10][discrepancies]

numeric(0)

> ped2$prob[3:12][discrepancies]

numeric(0)
```

We will return to the function `getXlist` later, and proceed to estimate β assuming the MLE/posterior mode of the pedigree is the true pedigree. We do this by passing the ML pedigree to a `startPed` object (`ped=ped2$P`)), and by specifying that the pedigree should not be estimated (`estP=FALSE`), but should be fixed.

```
> sP<-startPed(estP=FALSE, ped=ped2$P, estUSsire=FALSE)
> model2<-MCMCped(PdP=PdP, sP=sP, verbose=FALSE)
> plot(model2$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
    -0.0542810      0.0081881      0.0002589      0.0002719

2. Quantiles for each variable:

    2.5%      25%      50%      75%    97.5%
-0.07054 -0.05983 -0.05411 -0.04861 -0.03899
```

The genetic data are not required since the pedigree is treated as known, and only the `PdataPed` object (`PdP`) 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.

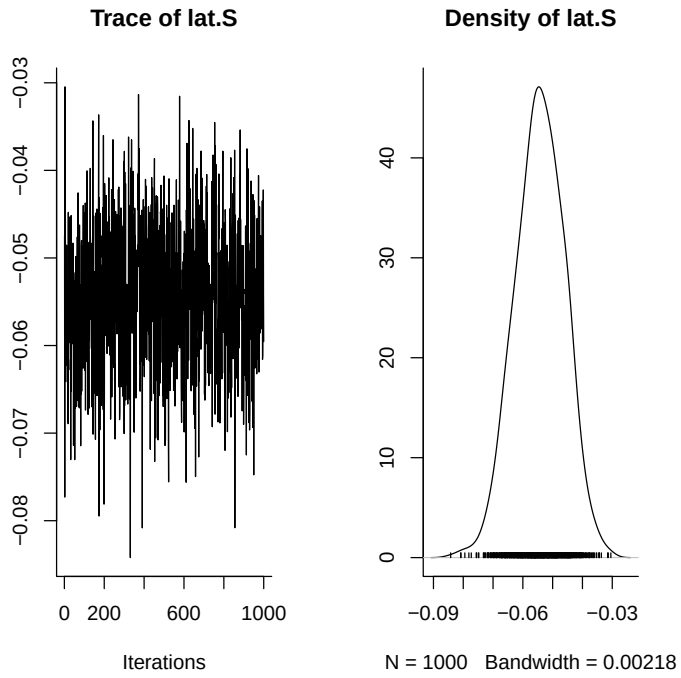

Figure 5: Posterior distribution of $\boldsymbol{\beta}$ from `model2`. The posterior is conditional on the categorical assignment of the pedigree being true. These plots are a way of summarising a Markov Chain using the coda package. The left plot is a trace of the sampled posterior, and can be thought 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.

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

            [,1]
[1,] -0.05351072

> # ML estimate
> sqrt(MLElat$C)

            [,1]
[1,] 0.00809248

> # SE
```

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

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

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

      [,1]       [,2]       [,3]
 [1,] "N748835" NA         NA
 [2,] "Z993558" NA         NA
 [3,] "N748825" NA         NA
 [4,] "N748841" NA         NA
 [5,] "N748842" NA         NA
 [6,] "N748848" NA         NA
 [7,] "Z993553" "K278011" "J368403"
 [8,] "N748838" "N021965" "N021760"
 [9,] "N748827" NA         NA
[10,] "Z993562" NA         NA

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

           [,1]
[1,] -0.2315504

> # ML estimate
> sqrt(MLElat3$C)

           [,1]
[1,] 0.04647355

> # SE estimate
```

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 β is more than 5 times larger than that in the previous model. However, the ML estimate of β 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 because we have failed to use the information contained in the phenotypic data to help us estimate the pedigree. Equation 4 can be rewritten

$$\frac{Pr(\beta, \boldsymbol{P}|\boldsymbol{G}_{obs}, \texttt{lat}, \texttt{long}, \texttt{terr})}{Pr(\boldsymbol{P}|\beta\boldsymbol{G}_{obs}, \texttt{lat}, \texttt{long}, \texttt{terr})}. \tag{6}$$

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

$$\frac{Pr(\boldsymbol{G}_{obs}|\boldsymbol{P})Pr(\boldsymbol{P}|\beta, \texttt{lat}, \texttt{long}, \texttt{terr})}{Pr(\boldsymbol{P}|\beta, \boldsymbol{G}_{obs}, \texttt{lat}, \texttt{long}, \texttt{terr})} Pr(\boldsymbol{P}, \beta) \tag{7}$$

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 β 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 Hadfield et al., 2006].

We can fit a model where the pedigree and β are estimated simultaneously, although for now we will retain the approximation for genotyping error and leave the number of unsampled males fixed at 10.

```
> sP<-startPed(estG=FALSE, A=extractA(WarblerG), E1=0.005, 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
    -0.2335404      0.0298062      0.0009426      0.0017287

2. Quantiles for each variable:

   2.5%     25%     50%     75%   97.5%
-0.2927 -0.2534 -0.2325 -0.2131 -0.1782
```

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

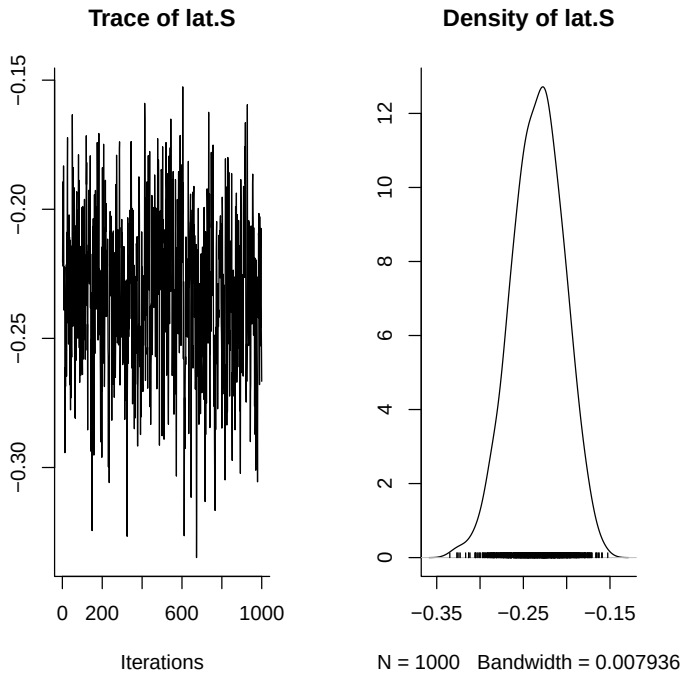

Figure 6: Posterior distribution of $\boldsymbol{\beta}$ from `model3`. The pedigree and β have been simultaneously estimated, but an approximation for genotyping error has still been used. It was also assumed that the size of the unsampled population and the genotyping error rate were known.

back of the categorical approach; once the pedigree is estimated you then have to treat it as if it was known with complete certainty, rather than a random variable.

2.5 Unsampled Parents

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

```
> sP<-startPed(estG=FALSE, A=extractA(WarblerG), E1=0.005, 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)
```

```
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
       34.0521        15.3947          0.4868         0.4867

2. Quantiles for each variable:

 2.5%   25%   50%   75% 97.5%
12.73 23.11 30.98 41.64 72.35
```

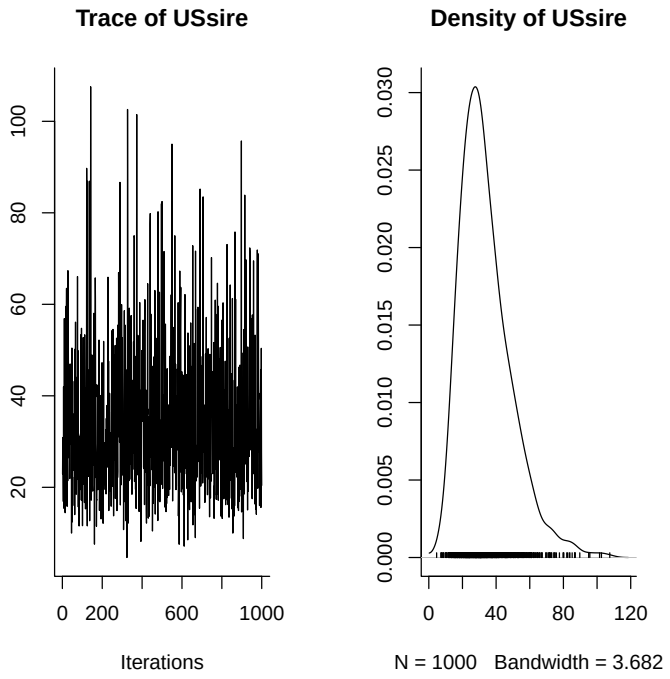

Figure 7: Posterior distribution of the number of unsampled males from `model4`. The number of unsampled males, the pedigree and β have all been simultaneously estimated, but we have still used an approximation for genotyping error.

The population of Seychelles Warbler on Cousin Island is small and essen-

tially closed. Given that the island has been extensively sampled we have reasonable prior belief that the number of unsampled males is somewhere between 2 and 15, and an estimate of 30.98 unsampled males seems unreasonably large. We could set up a prior specification that reflected this, but before doing this it will be useful to discuss some possible reasons why the model has failed to estimate the number of unsampled males accurately.

The problem of accommodating unsampled parents is a difficult one because the genotypes and phenotypes of unsampled individuals 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 possess an allele that is not present in the sampled parental population. If genotyping error was low enough we could be reasonably confident in stating that an unsampled male was present in the north of the island and that his genotype contained 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.

Nielsen et al. [2001] gives the likelihood 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. $\boldsymbol{O}$, $\boldsymbol{M}$ and $\boldsymbol{F}$ are the genotypes of offspring, mothers, and sampled father's respectively, and $\boldsymbol{A}$ are the allele frequencies in the unsampled population. 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 inferred from allele frequencies under the assumption of Hardy-Weinberg equilibrium; b) 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 e) $\boldsymbol{O}$ and $\boldsymbol{M}$ provide no information regarding the genotypes of unsampled males after conditioning on $\boldsymbol{A}$. Generally, $\boldsymbol{A}$ is estimated from $\boldsymbol{G}$ and an additional assumption must be made that f) unsampled males and sampled individuals come from the same statistical population with respect to allele frequencies. MasterBayes makes all these assumptions except b) and c), and I return to assumption e) in Section 3.8.

We can fit Nielsen's [2001] model using the function `MLE.popsize` by specifying the genotype error rate to be effectively zero

```
> MLEUSsire<-MLE.popsize(getXlist(PdP, GdP, E1=1e-10, E2=1e-10),
+ USsire=TRUE, USdam=FALSE)
> MLEUSsire$nUS

        [,1]
[1,] 102.828

> # ML estimate
> sqrt(MLEUSsire$C)

         [,1]
[1,] 32.71612

> # SE estimate
```

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 stochastic genotyping error rate set to 0.01 per allele

```
> MLEUSsire2<-MLE.popsize(getXlist(PdP, GdP, E1=1e-10, E2=0.01),
+ USsire=TRUE, USdam=FALSE)
> MLEUSsire2$nUS

         [,1]
[1,] 34.87078

> # ML estimate
> sqrt(MLEUSsire2$C)

         [,1]
[1,] 19.66623

> # SE estimate
```

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 systematically 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 4):

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

Here we have replaced the genetic likelihoods in Nielsen's original equation with the indicator variable δ_i^{obs}. δ_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:

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

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 expectation of the ratio in Equation 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 in this particular example, representing the pairing of each female with an unsampled male. The genetic likelihoods for these categories can be derived following 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 population, 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

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

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 Gelman et al., 2004, Chapter 7] I should emphasise at this point that the approximated linear predictors for missing individuals do not enter into the likelihoood for β, they are only there to simplify the estimation of the pedigree. However, if the phenotypes of unsampled individuals are known, then the linear predictors of unsampled individuals do not need to be estimated and can enter into the likelihood for β. In this instance 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 30.98 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 useful 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 erroneous 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 ($\boldsymbol{\varepsilon}$) and allele frequencies ($\boldsymbol{A}$). The approximate method defines the conditional probability of $\boldsymbol{G}$ as

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

whereas the exact solution defines the probability as

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

The distinction is subtle. The approximation implicitly assumes that the genotypes are collected from unrelated individuals, whereas the exact solution acknowledges the fact that the individuals 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 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 certainty.

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 iteration of the Markov chain. We are using the default model of genotyping error which is a two parameter model developed by Wang [2004]. In Wang's model an allele is assumed to be miss-scored as any other allele with equal probability (ε_2) and allele's of heterozgote genotypes can also dropout with probability ε_1. The model implemented in CERVUS can also be used by specifying `marker.type="MSC"` in the `GdataPed` object. However, although the CERVUS model is mathematically convenient it may be less biologically realistic. The model only includes stochastic errors and these errors are not independent for the two alleles within a genotype; if a genotyping error has occurred at one allele then a genotyping error occurs at the other allele with probability one. Accordingly, `E2`(2-`E2`) is the per-genotype rate defined in CERVUS and so a default value of `E1=0.005` is close to the CERVUS's default of 0.01 defined at the genotypic level. Perhaps a less realistic assumption of the CERVUS model is that an erroneous genotype is scored as another genotype (including itself) with a probability equal to that genotypes frequency in the population. Consequently, rare genotypes are believed to be miss-scored much less often than common genotypes.

```
> model5<-MCMCped(PdP=PdP, GdP=GdP, verbose=FALSE)
> plot(model5$E1)
> plot(model5$E2)
> plot(model5$USsire)
```

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  0.42385734
Lag 5  0.18403024
```

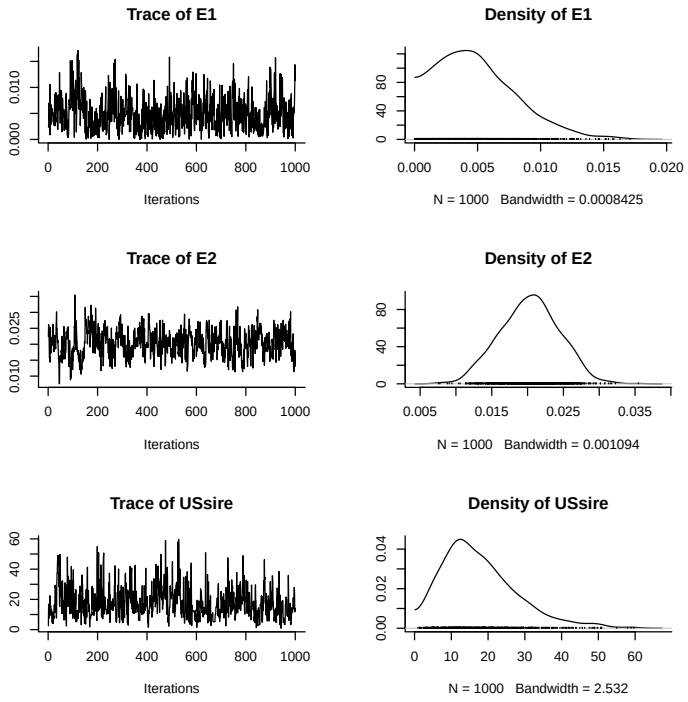

Figure 8: The marginal posterior distributions of the error rates and the number of unsampled males from `model5`. All unknowns have been simultaneously estimated.

```
Lag 10 0.11792381
Lag 50 0.02344455
```

The chain needs to be run for longer to ensure that we gather independent samples from the posterior, but nevertheless it is clear that we had underestimated the amount 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 a lot of uncertainty and a judicious use of prior information seems reasonable. The log normal is used as the prior specification for the number of unsampled males (see Figure 9)

```
> 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)
```

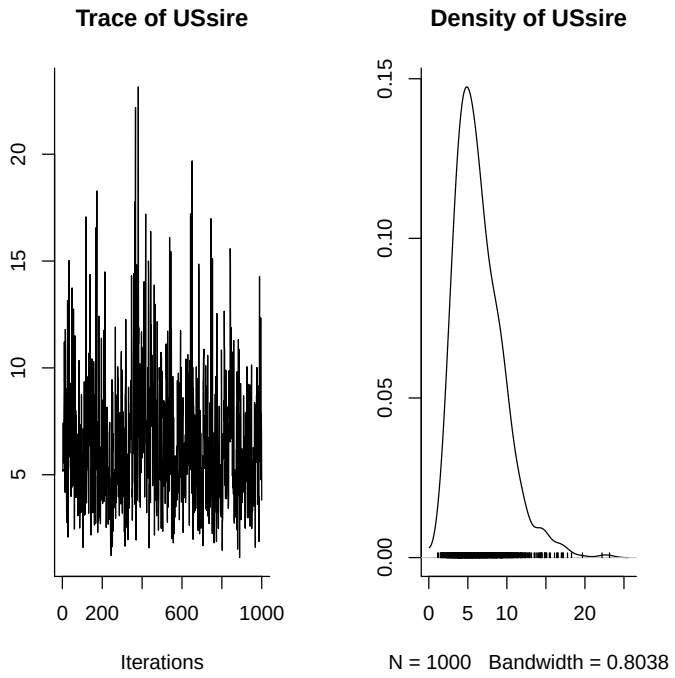

Figure 9: The marginal posterior distribution of the number of unsampled males from `model6`. In this model an informative prior was used.

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)
> # list off allele frequencies
```

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 the pedigree. We will simulate a large dropout rate (10% of alleles in a heterozygous state) but a low level of of stochastic error ($\approx 1\%$ of single locus genotypes).

```
> ped<-matrix(NA,50,3)
> ped[,1]<-1:50
> # 50 unrelated indivdiuals typed twice
>
> G<-simgenotypes(A=A,E1=0.1,E2=0.005,ped=ped,no_dup=2)
> # each individual is typed twice with high dropout
>
> tP<-tunePed(E1=15)
> #MH tuning paramater needs to be larger
>
> 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
     0.1007901      0.0095240      0.0003012      0.0003462

2. Quantiles for each variable:

   2.5%     25%     50%     75%   97.5%
0.08245 0.09416 0.10061 0.10688 0.11993

> 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
     2.254e-03      1.480e-03      4.680e-05      4.965e-05

2. Quantiles for each variable:

     2.5%       25%       50%       75%     97.5%
0.0003684 0.0011276 0.0019818 0.0030552 0.0056924
```

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, for the next example)[1]. 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
> # 50 dams and their offspring
>
> G<-simgenotypes(A=A, E1=0.1, E2=0.005, ped=ped, no_dup=1)
> # only typed once
>
> sP<-startPed(ped=ped)
> #specify that the dams are known,
> #males are treated as base by default
>
> 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
     0.0804682      0.0116173      0.0003674      0.0004445

2. Quantiles for each variable:

   2.5%     25%     50%     75%   97.5%
0.05955 0.07258 0.07965 0.08805 0.10481

> summary(model.pedE$E2)

Iterations = 1:1000
Thinning interval = 1
```

[1] Johnson and Haydon [2007] find that MasterBayes overestimates error rates by 31% when error rates are low (compared to a downward bias of 18% for Pedant). However, large scale simulation shows that MasterBayes is unbiased (if such a thing exists in a Bayesian analysis) if the posterior mode is used rather than the posterior mean.

```
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
     0.0177657      0.0082365      0.0002605      0.0007445

2. Quantiles for each variable:

    2.5%      25%      50%      75%    97.5%
0.002719 0.012021 0.017301 0.022807 0.035231
```

> plot(model.pedE$E2)

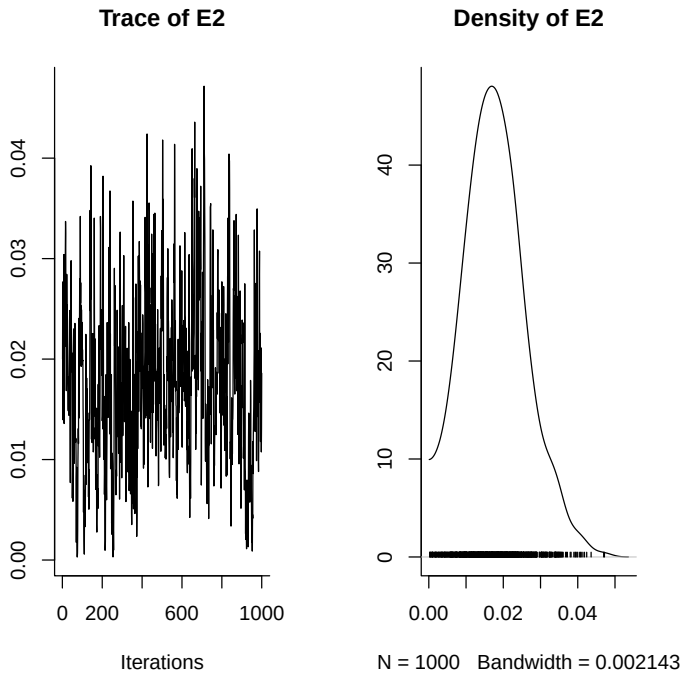

Figure 10: The marginal posterior distribution of stochastic error rate from `model.pedE`. The mode is probably a better summary of central tendency as the expectation of `E2` may not coincide with the most likely value it could take.

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 individual'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 separated 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(ped=ped, G=stG$Gobs, A=A)
> # A new set of random genotypes with positive probability
> # are generated and used as the starting configuration
> model.config<-MCMCped(GdP=GdP,sP=sP,tP=tP,verbose=FALSE)
> plot(model.config$E2)
```

When only dams are known the chain still mixes well and the chain converges very rapidly despite the absurd starting configuration (see Figure 11). However, we can simulate genotypes down a pedigree where individuals are grouped into large full-sib families. We will analyse the data using two chains, one initialised at the true genotype configuration, and one initialised at a random but legal configuration.

```
> ped<-matrix(NA,120,3)
> ped[,1]<-1:120
> ped[,2][21:120]<-rep(1:10, each=10)
> ped[,3][21:120]<-rep(11:20, each=10)
> # 10 full-sib families of size 10
>
> G<-simgenotypes(A=A,E1=0.1,E2=0.005,ped=ped,no_dup=1)
> sP<-startPed(ped=ped, G=G$G, A=A)
> #specify that the dams and sires are known,
> #and use the true G as the starting configuration
>
> GdP<-GdataPed(G=G$Gobs, id=G$id)
> model.DS1<-MCMCped(GdP=GdP,sP=sP,tP=tP,verbose=FALSE)
> stG<-simgenotypes(A=A,E1=0.1,E2=0.005,ped=ped,no_dup=1)
> sP<-startPed(ped=ped, G=stG$G,A=A)
> #use a random G as the starting configuration
>
> model.DS2<-MCMCped(GdP=GdP,sP=sP,tP=tP,verbose=FALSE)
> plot(mcmc.list(model.DS1$E2, model.DS2$E2))
```

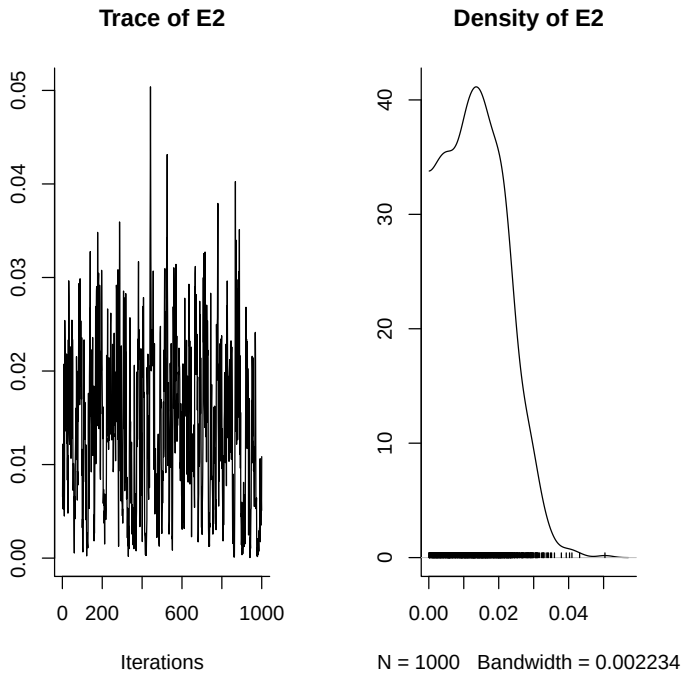

Figure 11: The marginal posterior distribution of stochastic error rate from `model.config`. This posterior should be identical to the posterior in Figure 10, although the chain was initialised in a very low probability region of genotype parameter space.

The chains are currently sampling from different regions of the posterior (Figure 12). Methods exist for traversing these regions [Sheehan, 2000, Lange, 2002] more efficiently 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 configurations have a positive probability under some pedigree configuration. Nevertheless, there may be regions of very low probability connecting high probability regions, and the chain may not mix well.

3.2 Mismatch Tolerance 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 iteration, and for each offspring 2500 Mendelian likelihoods

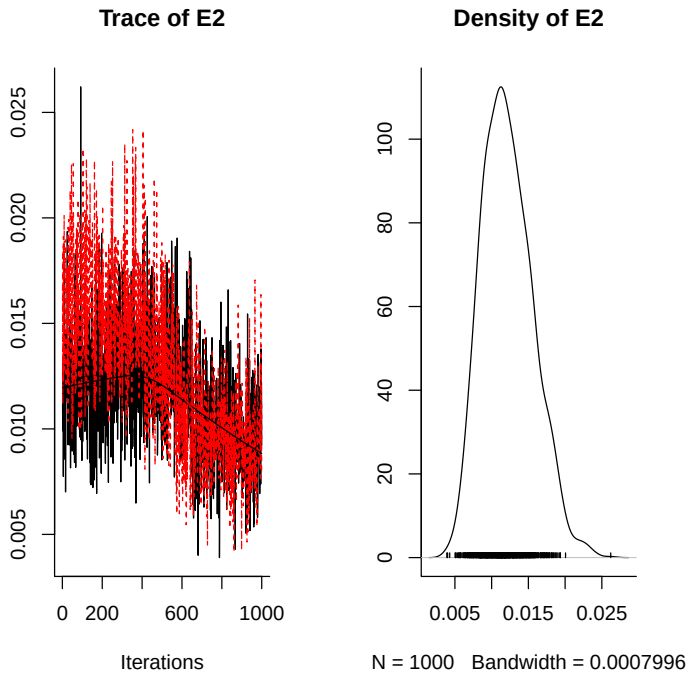

Figure 12: Samples from the posterior distribution of stochastic error rate estimated from the data set `Gobs`. The two chains were initialised at the true genotype configuration (black), and a random configuration (red).

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

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

A pedigree is simulated with β_1 and β_2 set to -1 and 1, respectively.

```
> 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$id)
> model_pedMM999<-MCMCped(PdP=PdP, GdP=GdP, verbose=FALSE)
> model_pedMM2<-MCMCped(PdP=PdP, GdP=GdP, mm.tol=2, verbose=FALSE)
> model_pedMM1<-MCMCped(PdP=PdP, GdP=GdP, mm.tol=1, verbose=FALSE)
> # identical models but the 2nd and 3rd models
> # exclude parents with 3 or more, or 2 or more,
> # mismatches, repectively
>
> summary(model_pedMM999$beta)[[1]][,1:2][1:2,]

             Mean        SD
size.D  1.1086914 0.1372865
size.S -0.9925665 0.1626351

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

             Mean        SD
size.D  1.1125573 0.1372343
size.S -0.9970467 0.1629536

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

            Mean        SD
size.D  1.111342 0.1341388
size.S -1.026845 0.1604380
```

The posterior distributions of β are almost identical, despite excluding parents with several mismatches. However, the saving in computer time was large: the first model took almost 7.7 minutes to fit where the last model took 1.9.

An alternative method for speeding things up is to specify `jointP=FALSE` in `MCMCped`. In this case mothers are sampled followed by fathers, rather than both parents at the same time. The chain may mix slower (per number of iterations) than the default `jointP=TRUE`, but the chain may iterate much faster if $n_i n_j >> n_i + n_j$, where n_i and n_j are the number of potential dams and sires:

```
> model_pedJPF<-MCMCped(PdP=PdP, GdP=GdP, verbose=FALSE, jointP=FALSE)
> summary(model_pedJPF$beta)[[1]][,1:2][1:2,]
```

```
            Mean        SD
size.D  1.099699 0.1353455
size.S -1.005569 0.1620252

> effectiveSize(model_pedMM999$beta)

  size.D   size.S
738.8332 745.7840

> effectiveSize(model_pedJPF$beta)

  size.D   size.S
813.4879 744.9783
```

The posterior distributions are identical up to Monte Carlo error since we have only changed the sampling scheme, not the posterior distribution from which we are trying to sample. However, the alternative sampling scheme iterates much faster (1.2 minutes compared to compared to 7.7) although the amount of information we gain per iteration is slightly less (745 effective samples compared to 745.8 for the effect of size on paternity). Models that include a variable with the argument `relational="MATE"` are particularly slow when the number of parental combinations is large and I discuss faster approximate methods in Section 3.7.

3.3 Equivalence with Poisson Models

If we knew the pedigree we may be inclined to fit a generalised linear model with a Poisson 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
> # number of offspring per mother
> 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
-1.6206  -0.8772  -0.6222   0.5838   2.0379

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
```

```
(Intercept) -11.8035     1.4977  -7.881 3.25e-15 ***
size[1:50]    1.1106     0.1338   8.302  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 99.188  on 49  degrees of freedom
Residual deviance: 41.641  on 48  degrees of freedom
AIC: 105.67

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 uncertainty 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)$P[,2:3])

FALSE  TRUE
    5    95
```

The modal parentage assignments are close to the true pedigree, with 95% 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 Poisson 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 because it can be derived directly from the number of parental combinations and the number of offspring which we have conditioned on [McCullagh and Nelder, 1989].

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 include 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 a list containing 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 δ 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 are only 3 parameters to be estimated because the variables are assumed to have the same 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 may not mix well when the posterior distribution of the parameters are not independent. This can be alleviated to some degree by sampling correlated parameters together, although when Metropolis-Hastings updates are used gains in efficiency are only possible when the proposal distribution is similar in form to the conditional posterior distribution of the parameters. MasterBayes attempts to find such a proposal distribution using approximate maximum likelihood methods, although this distribution tends to underestimate the strength of correlation between the parameters. In this example the posterior distribution of the three parameters is highly correlated with the correlation between age and the age by size interaction approaching -1. The efficiency of the chain is not necessarily compromised by this but we can see that the autocorrelation between the parameters is reasonably high despite a thinning interval of 10 (See Figure 13).

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

              size    age.young size.age.young
Lag 0  -0.73229292 -0.995353801    1.000000000
Lag 1  -0.24331016 -0.316794496    0.317773935
Lag 5  -0.01660023 -0.003279173    0.002917106
Lag 10 -0.04480474 -0.043269940    0.044938289
Lag 50  0.02226050  0.021094381   -0.018000246
```

The dependency between slopes and intercepts 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)
```

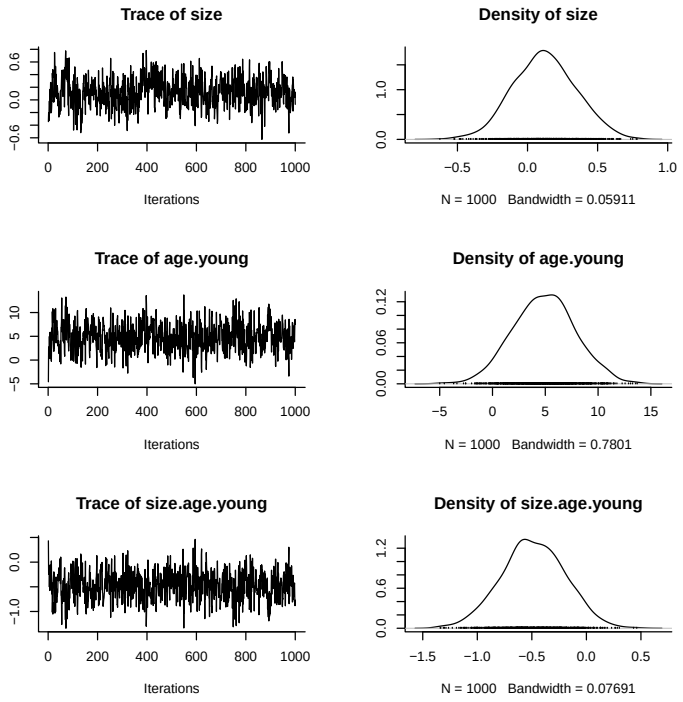

Figure 13: The marginal posterior distributions for the three elements of `beta` from `model.INT`. The intercept for young individuals `age.young` is evaluated at a `size` of 0, which lies well outside the distribution of `size` observed in the population.

```
> 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  -0.729478279 -0.113150855     1.00000000
Lag 1  -0.179349485 -0.014859268     0.26109208
Lag 5   0.003967927  0.047545476     0.05501976
Lag 10  0.004850002  0.072179126     0.03441719
Lag 50 -0.065998770 -0.005993626     0.03736367
```

Despite having similar Metropolis-Hastings acceptance rates the second chain mixes 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 (Figure 14). However, as old individuals get larger their ability to reproduce

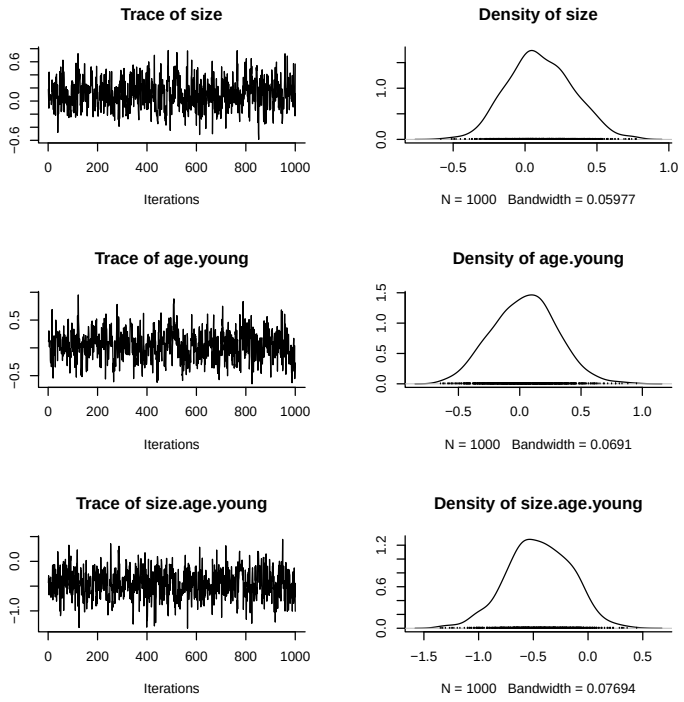

Figure 14: The marginal posterior distributions for the three elements of `beta` from `model.INT`, but with the intercept for young individuals `age.young` evaluated at the average `size` of 10.

increases (`size` is positive - the true underlying slope is 0.5), but young individuals 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 difference between the slopes of young and old individuals. 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 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 (Figure 15). 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]))
```

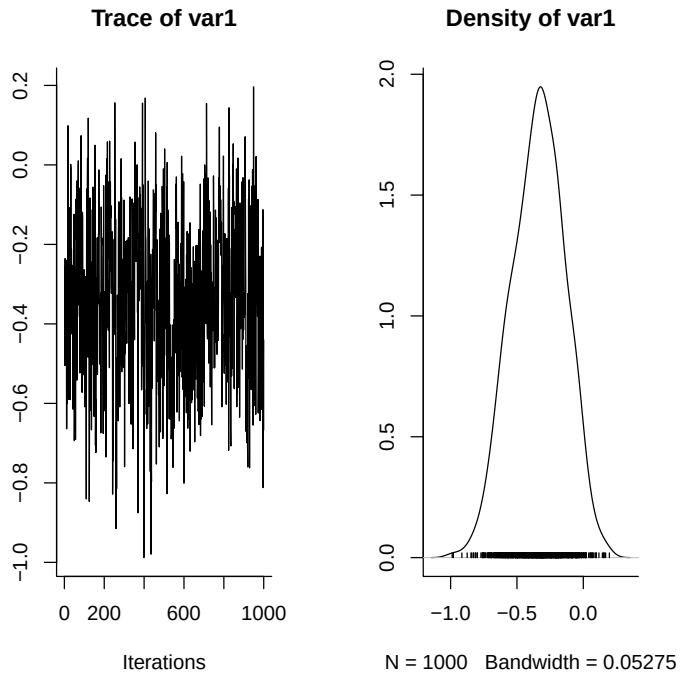

Figure 15: The marginal posterior distribution for the slope of young individuals from model `model.INT.cntr`.

```
> counts[as.numeric(names(noff))]<-counts[as.numeric(names(noff))]+noff
> par_age<-age[1:100]
> 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
-1.6569  -1.2287  -0.1118   0.4781   2.7727

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)          -0.09746    0.15665  -0.622  0.53384
par_ageyoung          0.15230    0.21023   0.724  0.46881
par_size              0.31936    0.19167   1.666  0.09567
```

```
par_ageyoung:par_size -0.68096    0.25142  -2.709  0.00676

(Intercept)
par_ageyoung
par_size              .
par_ageyoung:par_size **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 111.78  on 99  degrees of freedom
Residual deviance: 103.64  on 96  degrees of freedom
AIC: 260.05

Number of Fisher Scoring iterations: 5
```

A Poisson GLM indicates the same thing: young and old individuals 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 of old individuals. 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 (β) 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 θ 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 β to make valid inferences about the posterior of θ.

When the number of treated (N_E) and untreated (N_C) 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 (θ_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 β, although β_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 β over β_o since they are functionally equivalent. However, there are cases when β and β_o are not equivalent and care needs to be taken when choosing between the two models. A classic example of when β_o may be favoured over β 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="=="))
> 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$id)
> # remove unsampled males
>
> rm_males<-1:25
> data_EPP_miss<-data_EPP[-rm_males,]
> GdP_miss<-GdataPed(G=lapply(G$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 β, not the log odds ratio of the within-pair paternity rate which in this case is $\beta_o = logit^{-1}(0.8)$. 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. In some of the following analyses problems may be encountered when all fathers are assigned to the unsampled class during some iterations. At this point the posterior distribution is no longer proper without prior information because a) the most likely estimate of the unsampled population is infinity and b) there is no information to estimate β. Therefore, we'll also place a prior on the number of unsampled males which has a mode equal to the true value, and we'll place a prior on β which is approximately uniform on the probability scale θ.

```
> PdP_miss<-PdataPed(formula=list(res1, var1, res2),
+  data=data_EPP_miss, USsire=TRUE)
> pP <- priorPed(USsire = list(mu = log(25), sigma = 1), beta=list(mu = as.matrix(0), sigma
> model.miss<-MCMCped(PdP=PdP_miss, GdP=GdP_miss, pP=pP, verbose=FALSE)
> plot(model.miss$beta)
> plot(model.miss$USsire)
```

The estimate of β seems reasonable given that `logit(EPP)=5.28` (Figure 16), but the number of unsampled males is much smaller than we anticipated despite the prior having a mode equal to the true number (Figure 17). This is because 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 β, and so β is valid. However, our primary interest was in the within-pair paternity rate, θ_o, and we can use equation 18 to obtain the posterior

```
> theta<-inv.logit(model.miss$beta)
> NW<-1
> NE<-24
> # unsampled males are presumed to be present in
> # the same ratios so do not enter into the equation
>
> theta_o<-(NW*theta)/(NW*theta+NE*(1-theta))
> plot(mcmc(theta_o))
```

Given the mode of the posterior of β 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).

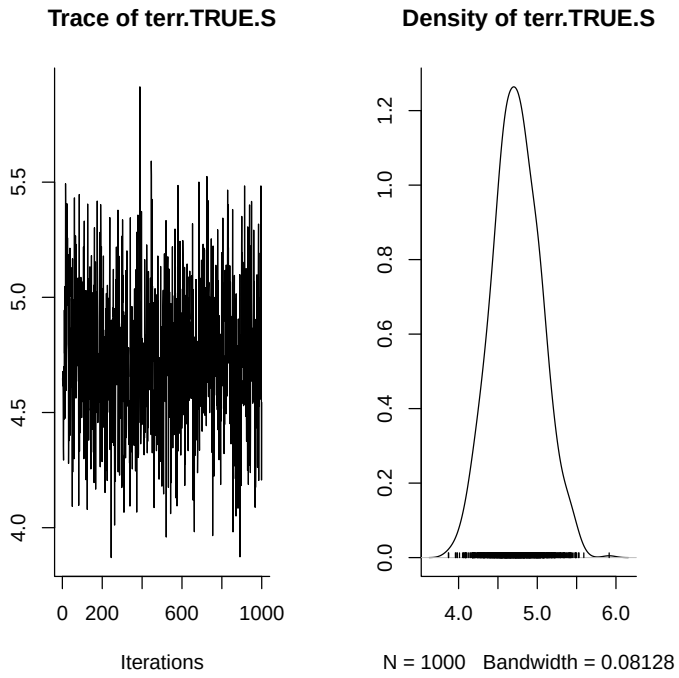

Figure 16: The marginal posterior distribution of β from `model.miss`. This is the log odds ratio of a male scoring paternity if it had been on the offspring's territory compared to if it had been on a different territory. The pedigree is sampled assuming that the phenotypes of unsampled males are unknown.

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

Once again β 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.

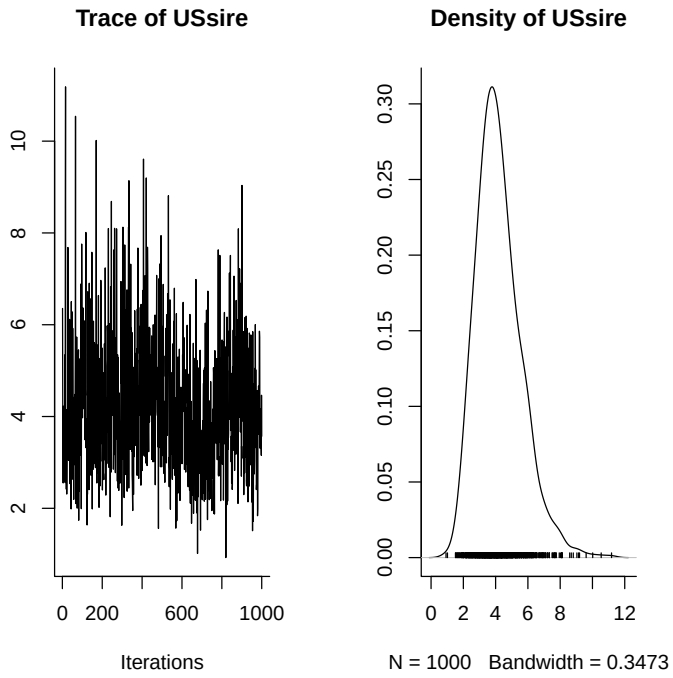

Figure 17: The marginal posterior distribution of the number of unsampled males from `model.miss`, when the pedigree is sampled assuming that the phenotypes of unsampled males are unknown. The actual number is 25!

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

```
> 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))
```

The estimate of within-pair paternity from `model.miss2` is plotted in Figure 21. Alternatively we can fit within-pair paternity explicitly using the argument `merge=TRUE`.

```
> var3<-expression(varPed(x="terr", gender="Male",
+ relational="OFFSPRING", USvar=FALSE, merge=TRUE))
> PdP_miss3<-PdataPed(formula=list(res1, var3, res2),
```

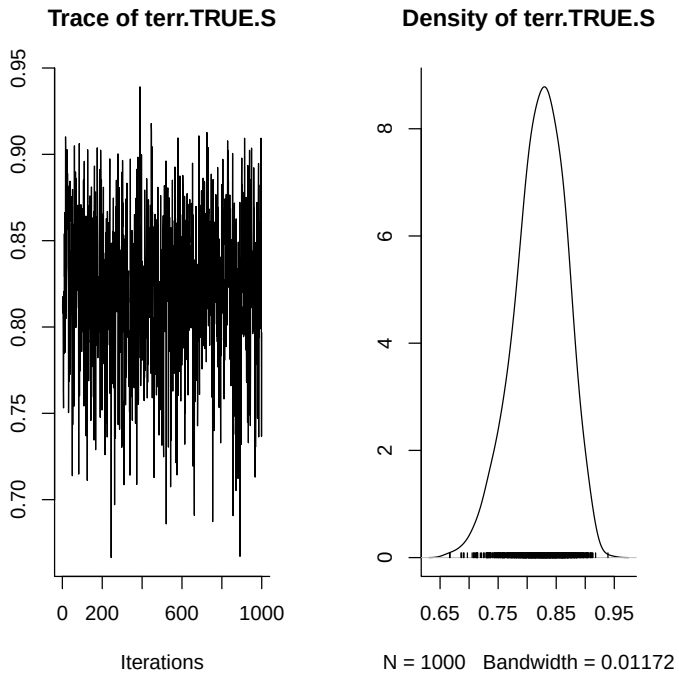

Figure 18: The marginal posterior distribution of θ_o derived from `model.miss`. θ_o should be the level of within pair paternity in the population (80%) but we can see that this value has a small posterior probability. This may be down to sampling error alone, but in fact the estimate is consistently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

```
+ data=data_EPP_miss, USsire=TRUE)
> model.miss3<-MCMCped(PdP=PdP_miss3, GdP=GdP_miss,
+ pP=pP, verbose=FALSE)
> theta_o<-inv.logit(model.miss3$beta)
> plot(mcmc(theta_o))
```

In this example the estimate of within-pair paternity in `model.miss2` (Figure 21) and `model.miss3` (Figure 22) 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 β and the number of unsampled males would not yield a valid posterior for β_o as it did in the simple example above (Figure 21).

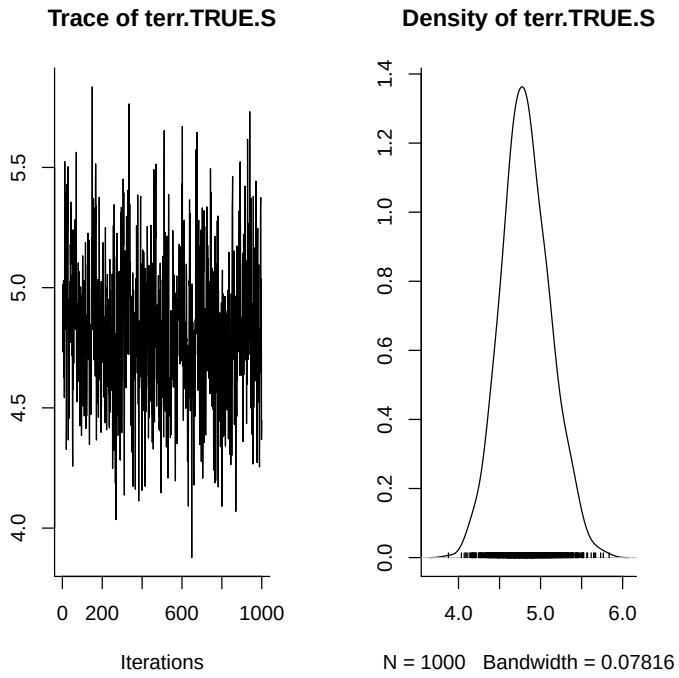

Figure 19: The marginal posterior distribution of β derived from `model.miss2`. As in Figure 16 this is the log odds ratio of a male scoring paternity if it had been on the offspring's territory compared to if it had been on a different territory. However, in this model the pedigree is sampled assuming that the unsampled males have to be extra-pair, and these males contribute to the likelihood of β.

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 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.

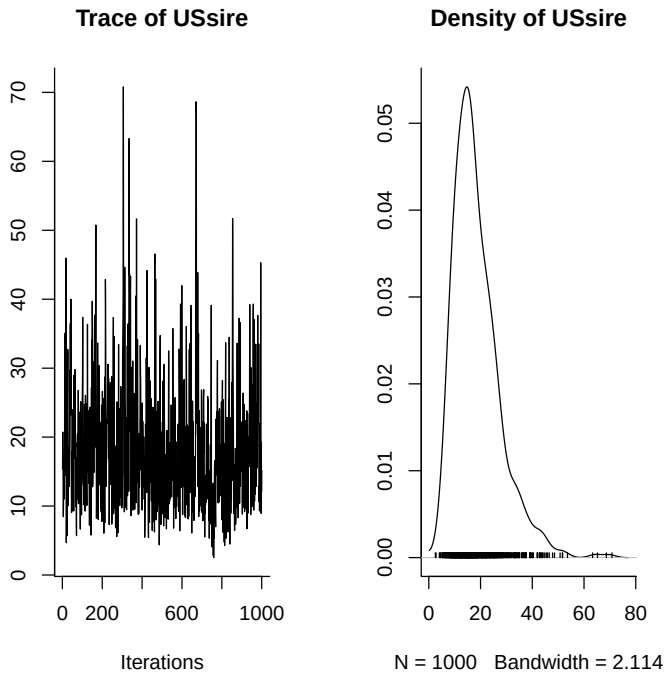

Figure 20: The marginal posterior distribution of the number of unsampled males from `model.miss2`. The true number is 25.

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

$\delta_{i,j}$ takes on the value one when the mating type of female i matches the mating type of male j. In this example β_1 is set to `log(3)=1.10`, and the results of the analysis shown in Figure 23.

```
> 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)
> 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])])
```

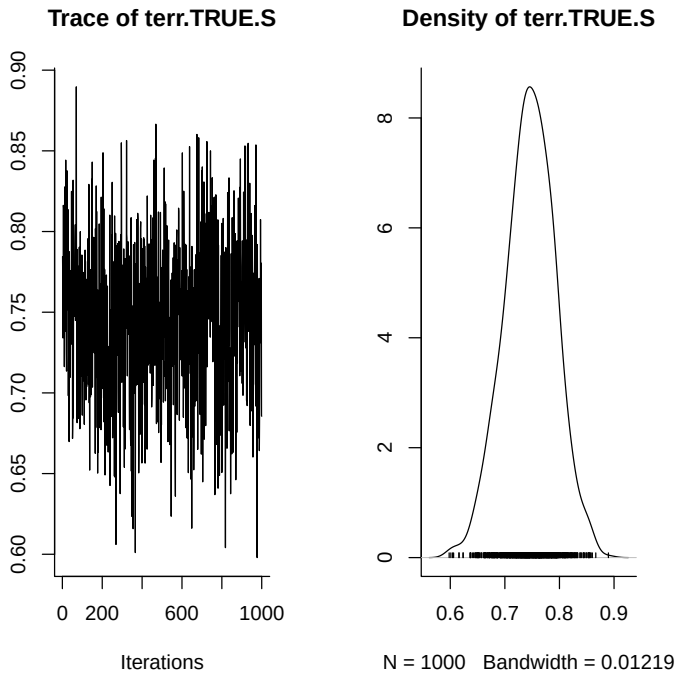

Figure 21: The marginal posterior distribution of within-pair paternity (θ_o) estimated from the posterior distributions of β and the number of unsampled males from `model.miss2`.

```
FALSE  TRUE
    7    23

> # offspring produced by +/- unions should make up (on average) a
> # quarter of the total offspring, and offspring produced by +/+
> # and -/- unions should make up the remaining 75%.  Note this is
> # only because the frequency of + and - mating types are equal
>
> 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 offspring and parental phenotypes using the argument `relational="OFFSPRING"`. A cautionary note should be made about using `relational="OFFSPRING"` to

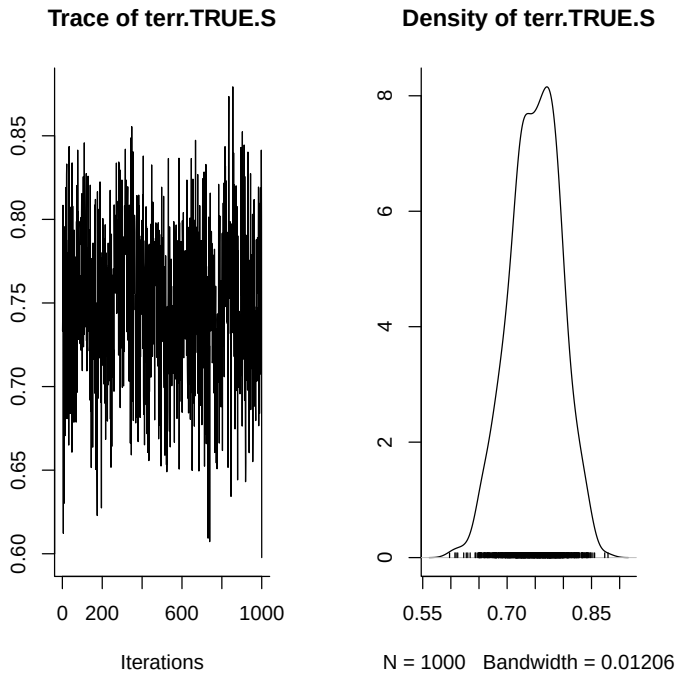

Figure 22: The marginal posterior distribution of within-pair paternity (θ_o) estimated directly in `model.miss3`. The posterior distribution is exactly equivalent to that shown in Figure 21.

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 `relational="OFFSPRING"` may capture the genetic process quite well. On the other hand, 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 heritability in the quantitative genetic sense, although they will be related. Future work is planned in this direction.

As noted in Section 3.2, when the number of potential mothers and fathers is large, models in which the distance between mates is included may be very slow. This is because for each offspring at each iteration the multinomial denominator must be calculated by exponentiating and summing over the linear predictors of all parental combinations. The likelihood of the data given β is

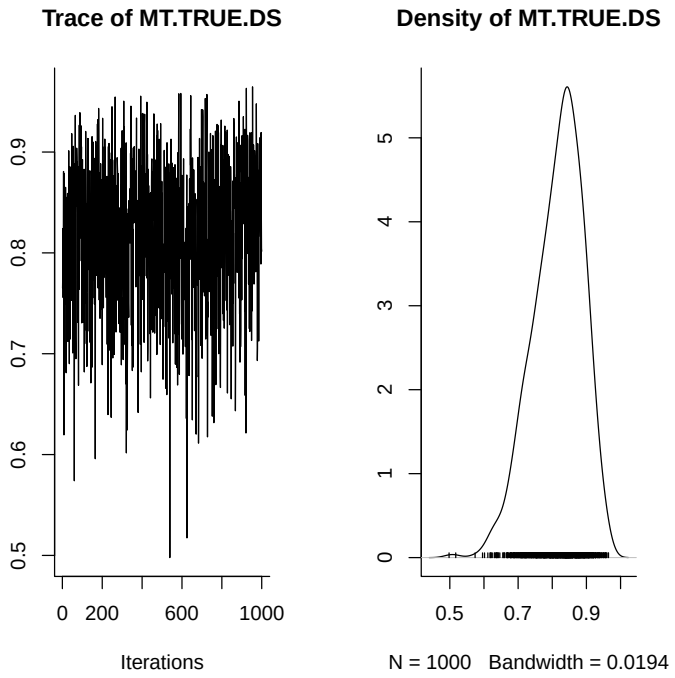

Figure 23: The posterior distribution of $\text{logit}^{-1}(\beta)$. In this instance the frequencies of the two mating types are exactly 0.5 both within sexes, and between the sexes, and this is the expected proportion of offspring resulting from the union of +/+ and -/- parents. The data were simulated with $\text{logit}^{-1}(\beta) = 0.75$

$$\prod_{o}^{n_o} \left[\frac{\exp(\eta_{d,s}^{(o)})}{\sum_{i=1}^{n_i^{(o)}} \sum_{j=1}^{n_j^{(o)}} \exp(\eta_{i,j}^{(o)})} \right] \tag{21}$$

where $\eta^{(o)}$ are linear predictors for offspring o. d and s are the dam and sire assigned as parents, and $n_i^{(o)}$ and $n_j^{(o)}$ are the total number of potential dams and sires for offspring o. The argument `DSapprox=TRUE` can be passed to `MCMCped` which uses an approximation for this likelihood which can be orders of magnitude faster for large problems. The approximate likelihood has the form:

$$\prod_{o}^{n_o} \left[\frac{\exp(\eta_{d,s}^{(o)})}{\sum_{i=1}^{n_i^{(o)}} \exp(\eta_{i,s}^{(o)}) + \sum_{j=1}^{n_j^{(o)}} \exp(\eta_{d,j}^{(o)}) - \exp(\eta_{d,s}^{(o)})} \right] \tag{22}$$

How well this approximation works is not entirely known, but for such a

small problem as this we can easily do it both ways and see:

```
> model.ass.mat.approx<-MCMCped(PdP=PdP, GdP=GdP, verbose=FALSE, DSapprox=TRUE)
> plot(mcmc.list(model.ass.mat$beta,model.ass.mat.approx$beta))
```

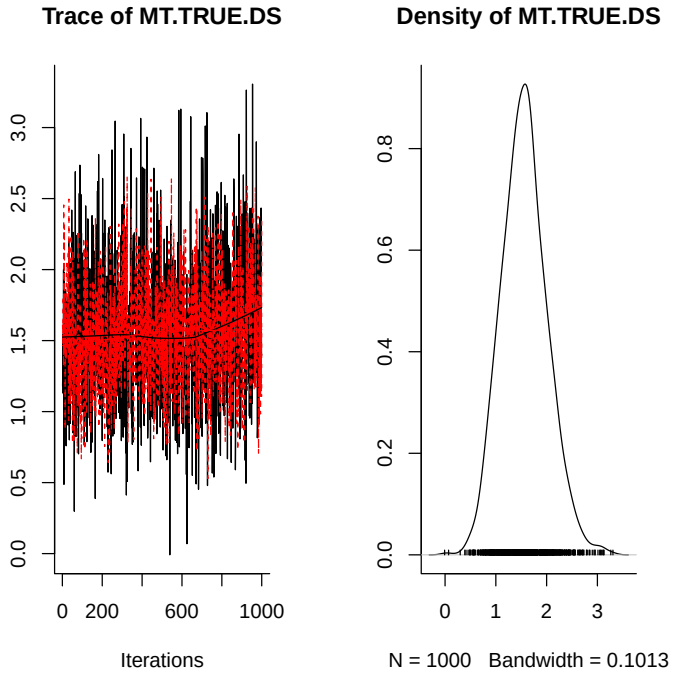

Figure 24: The posterior distribution of $\text{logit}^{-1}(\beta)$ using the exact likelihood (black) and an approximate likelihood (red).

The saving in time is marginal for such a small model (18 seconds versus 21) although for very large models using the approximation together with `jointP=FALSE` may be the only feasible way of performing the analysis in real time. At least for this example the approximation seems to work reasonably well (See Figure 24 and also Section 3.9).

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 important to ensure that

individuals with offspring records do not appear as potential parents of offspring with records in the same cohort, or a previous cohort When appropriate cohort or age data exist the argument `restrict` can be passed to `varPed` to ensure this does not happen. In the absence of age or cohort data the argument `checkP=TRUE` can be passed to `MCMCped`. This relaxes the restriction that potential parents do not appear in the set of potential offspring, but in such cases the probability that individual i is the offspring of individual j often has a similar posterior probability to the reverse scenario where individual j is the offspring of individual i. The function `post.pairs` will calculate posterior probabilities for 2 individuals falling into a certain relationship (e.g. parent-offspring or offspring-parent) when the joint posterior distribution of pedigrees are stored (i.e. `write_postP="TRUE"` is passed to `MCMCped`). However, in this example we will assume that appropriate cohort data exists.

```
> 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)
+ }
> # A founding population of 50 individuals is set up.
> # Each year 25 adults pass through to the next year
> # adn 25 die.  The 25 deaths are replaced by 25
> # offspring
>
> 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="=="))
> #parents must be recorded in the year the offspring was born
> #but must not be born in that year
>
> var1<-expression(varPed(x=c("lat", "long"), lag=c(0,0), relational="OFFSPRING"))
> # both maternity and paternity is a function of distance beween parents
> # and offspring in the year the offspring was born.
>
> 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)
```

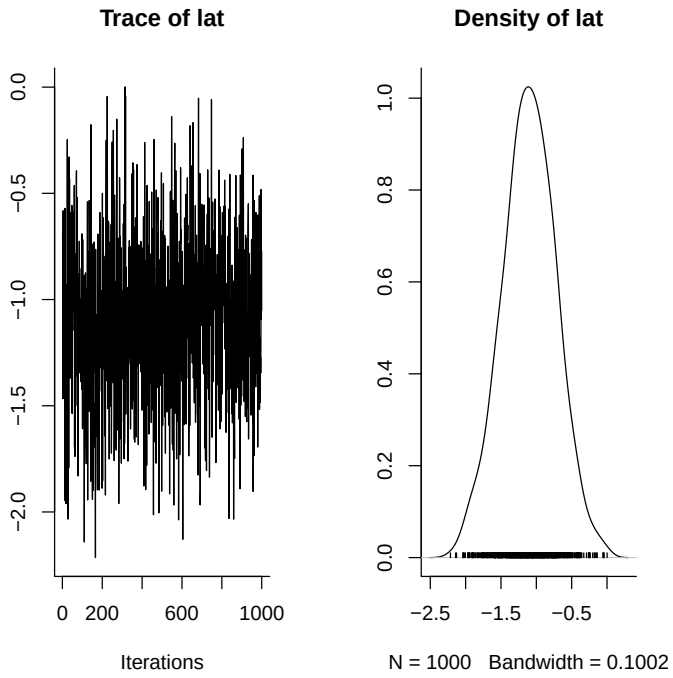

Figure 25: The posterior distribution of `beta`, the rate at which the probability of parentage drops with distance from the offspring. This parameter was estimated from a multigenerational pedigree where longitudinal data had been collected.

Several problems currently exist with reconstructing multigenerational pedigrees using MasterBayes. In reality, these problems exist for all MasterBayes analyses although the problem only becomes apparent when the notion of an arbitrary pedigree structure is entertained. 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 [Thompson, 1976]. Modelling the possibility that sampled individuals may be related through unsampled individuals is a challenging problem with a long history. A partial solution if potential parents are known but ungenotyped is to include them even though they have completely missing genotype information. If `estG=TRUE` MasterBayes will update that individuals genotype much like the algorithm proposed by Emery et al. [2001] although a lot of care needs to be taken that the chain converges and mixes well . A second problem with the reconstruction of multigenerational pedigrees,

which has received 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 [Elston and Stewart, 1971], which is based on the product of the Mendelian transition probabilities across offspring. Most parentage and sib analyses are special cases or approximations of this algorithm. However, a pedigree has both marriages and births, and an implicit assumption when calculating the likelihood of genotypes in this way is that marriages are independent of genotype. When inbreeding avoidance, or selfing is practiced, this assumption breaks down, and could compromise pedigree reconstruction.

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 inefficient (the following example took 4 minutes).

```
> 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$Gid)
> model.herm<-MCMCped(PdP,GdP,verbose=FALSE)
> plot(inv.logit(model.herm$beta))
```

The model is essentially 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 (Figure 26). 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 consideration 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 also those offspring with one or more unsampled parents. These genotype frequencies are calculated from the allele frequencies under the assumption of Hardy-Weinberg equilibrium. For sampled individuals, the probability distribution of the true genotypes is largely dominated by the observed genotype data rather than the genotype frequencies, and violations of Hardy-Weinberg equilibrium may not be that important. However, when unsampled parents are present an approximation for the genetic likelihood is used (see Section 2.5) that is more reliant on the information imparted by genotype frequencies, and care needs to be taken.

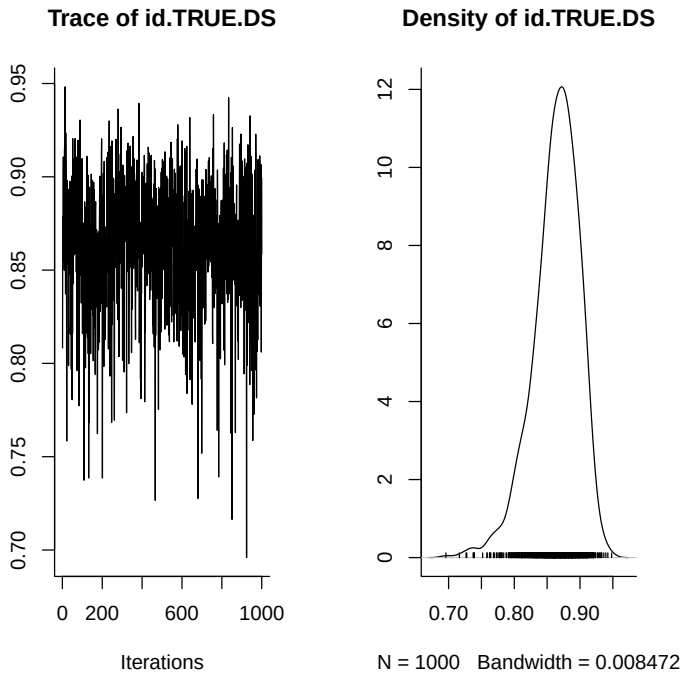

Figure 26: The posterior distribution of $\text{logit}^{-1}\beta$: the probability that an individual mates with itself compared to another random individual in a hermaphrodite system.

Although more efficient methods could be developed for modelling inbreeding, alternative sampling schemes and approximations that are already implemented can increase efficiency considerably. For example specifying `jointP=FALSE` and `DSapprox=TRUE` reduce the computation time from 4 minutes to 50 seconds:

```
> model.herm.approx<-MCMCped(PdP,GdP,verbose=FALSE, jointP=FALSE, DSapprox=TRUE)
> plot(mcmc.list(model.herm$beta, model.herm.approx$beta))
```

and the posterior distributions from the the two analyses are in good agreement (See Figure 27)

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 fruit or organelles have been typed) provide no information as to whether an offspring is produced from the male or female flowers of an individual. The

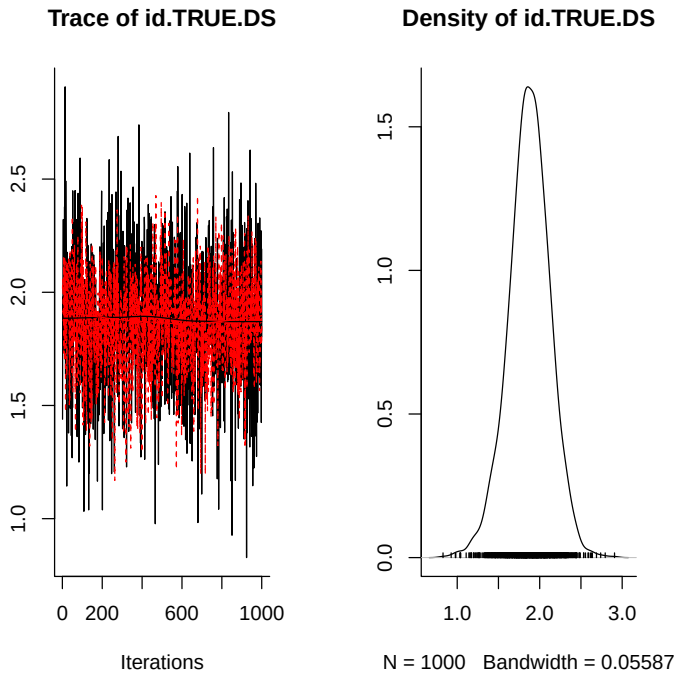

Figure 27: Posterior distribution of β: the log odds ratio that an individual mates with itself compared to another random individual in a hermaphrodite system. The black trace is the posterior using the exact likelihood, and the red trace is the posterior using an alternative sampling scheme and an approximation to the true likelihood.

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 and this measure positively covaries with pollen production but negatively covaries with seed production. Alternatively, we could have measured something less suggestive like the distance between 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)
```

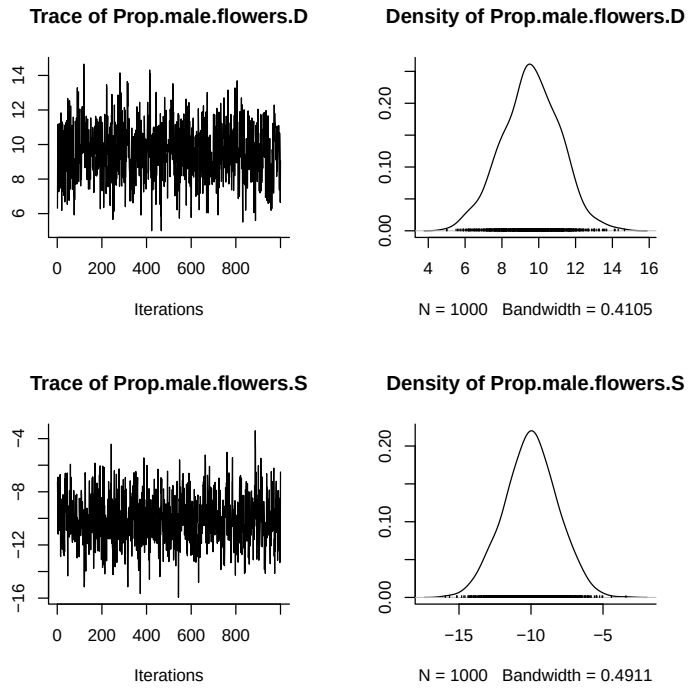

Figure 28: The posterior distribution of β: the parameters of the log-linear relationship between the number of male flowers on a monoecious plant and the probability of producing offspring from female and male flowers, respectively. The relationships are very strong and the chain gets stuck in a region of parameter space of high probability. A mirror image of this distribution exists outside of the parameter space sampled.

Figure 28 looks reasonable but what happens if we simulate some data where the proportion of male flowers is not such a good predictor of 'paternity/maternity'?

```
> P<-simpedigree(PdP, beta=c(-4,4))$ped
> G<-simgenotypes(A=A, ped=P)
> GdP<-GdataPed(G=G$Gobs, id=G$Gid)
> model.mon2<-MCMCped(PdP,GdP,mm.tol=1,verbose=FALSE)
> plot(model.mon2$beta)
```

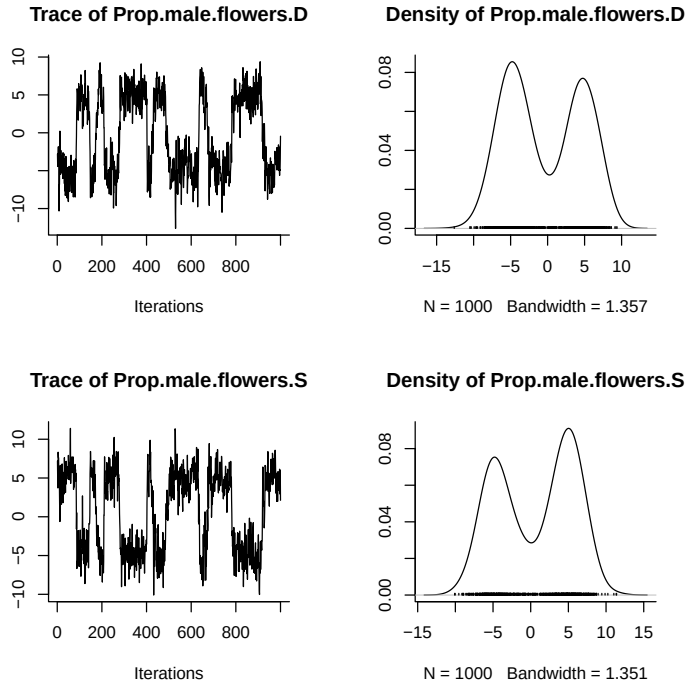

Figure 29: The posterior distribution of β: the parameters of the log-linear relationship between the number of male flowers on a monoecious plant and the probability of producing offspring from female and male flowers, respectively. The relationships are not so strong as in Figure 28, and the chain is able to sample from the full posterior, albeit with a lot of autocorrelation.

We see that the posterior distribution is actually bimodal (Figure 29)! We can be fairly confident that the proportion of male flowers indicates the probability of pollination 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 ambiguous variables 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 Lightning 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 β 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")`

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

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

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

`varPed(x)`

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

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

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (|\mathtt{x}_i - \mathtt{x}_o|) ...) \tag{26}$$

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

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (\mathtt{x}_i - \mathtt{x}_o) ...) \tag{27}$$

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

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (|\mathtt{x}_i - \mathtt{x}_j|) ...) \tag{28}$$

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

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (\mathtt{x}_i - \mathtt{x}_j) ...) \tag{29}$$

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

$$p_{i,j}^{(o)} \propto \exp(\beta_1 \mathtt{x}_{i,t-1} ...) \tag{30}$$

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

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (|\mathtt{x}_{i,t-1} - \mathtt{x}_{o,t}|) ...) \tag{31}$$

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

$$p_{i,j}^{(o)} \propto \exp(\beta_1 (|\mathtt{x}_{i,t} - \mathtt{x}_{j,t-1}|) ...) \tag{32}$$

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

$$p_{i,j}^{(o)} \propto \exp(\beta_1(|\mathtt{x}_{j,t} - \mathtt{x}_{o,t-1}|)...) \tag{33}$$

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

$$p_{i,j}^{(o)} \propto \exp(\beta_1 \delta_i ...) \tag{34}$$

where δ_i is an indicator variable taking the value 1 if $\mathtt{x}_i$ is equal to the first level of `x` and zero otherwise. β_1 is then the log odds ratio of the two levels of `x` with respect to maternity. If `merge=TRUE` is specified then β_1 may vary across offspring, and β_o is estimated. β_o is related to β_1:

$$\beta_o = \text{logit}\left[\frac{\theta N_A}{\theta N_A + (1-\theta)N_B}\right] \tag{35}$$

where θ is the inverse logit transformation of β_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) [Smouse et al., 1999]. Designating the first set of parents as baseline, the contrast for each set of parents is simply:

$$\eta_{i,j}^{(o)} = \log\left[\frac{p_{i,j}^{(o)}}{p_{1,1}^{(o)}}\right] \tag{36}$$

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

$$Pr(x|\boldsymbol{\beta}) = \prod_{o}^{n_o}\left[\frac{\exp(\eta_{d,s}^{(o)})}{\sum_{i=1}^{n_i^{(o)}}\sum_{j=1}^{n_j^{(o)}}\exp(\eta_{i,j}^{(o)})}\right] \tag{37}$$

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 $s \neq j$.

`DSapprox=TRUE` can be passed to `MCMCped` which approximates the likelihood when a variable specifies the distance between mates (i.e `relational="MATE"`). This approximation reduces the computational burden by fixing $i = d$ or $i = s$ in the denominator of the multinomial likelihood. The parent defined as the

"MATE" is fixed, so that a `varPed` expression with `gender="Male"` has the approximated likelihood:

$$Pr(x, i = d|\boldsymbol{\beta}) = \prod_{o}^{n_o} \left[\frac{\exp(\eta_{d,s}^{(o)})}{\sum_{j=1}^{n_j^{(o)}} \exp(\eta_{d,j}^{(o)})} \right] \tag{38}$$

References

R. C. Elston and J. Stewart. General model for genetic analysis of pedigree data. *Human Heredity*, 21:523–542, 1971.

A. M. Emery, I. J. Wilson, S. Craig, P. R. Boyle, and L. R. Noble. Assignment of paternity groups without access to parental genotypes: multiple mating and developmental plasticity in squid. *Molecular Ecology*, 10:1265–1278, 2001.

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman & Hall, 2^{nd} edition, 2004.

J. D. Hadfield, D. S. Richardson, and T. Burke. Towards unbiased parentage assignment: combining genetic, behavioural and spatial data in a Bayesian framework. *Molecular Ecology*, 15:3715–31, 2006.

P. C. D. Johnson and D. T. Haydon. Software for quantifying and simulating microsatellite genotyping error. *Bioinformatics & Biology Insights*, 1:71–75, 2007.

K. Lange. *Mathematical and Statistical Methods for Genetic Analyses*. Springer - Verlag, New York, 2^{nd} edition, 2002.

T. C. Marshall, J. Slate, L. E. B. Kruuk, and J. M. Pemberton. Statistical confidence for likelihood-based paternity inference in natural populations. *Molecular Ecology*, 7:639–655, 1998.

P. McCullagh and J. A. Nelder. *Generalised Linear Models*. Chapman & Hall, Cambridge, 2^{nd} edition, 1989.

R. Nielsen, D. K. Mattila, P. J. Clapham, and P. J. Palsbøll. Statistical approaches to paternity analysis in natural populations and applications to the North Atlantic humpback whale. *Genetics*, 157:1673–1682, 2001.

D. S. Richardson, F. L. Jury, D. A. Dawson, P. Salgueiro, J. Komdeur, and T. Burke. Fifty Seychelles warbler (*Acrocephalus sechellensis*) microsatellite loci polymorphic in Sylviidae species and their cross-species amplification in other passerine birds. *Molecular Ecology*, 9:2226–2231, 2000.

D. S. Richardson, F. L. Jury, K. Blaakmeer, J. Komdeur, and T. Burke. Parentage assignment and extra-group paternity in a cooperative breeder: the Seychelles warbler (*Acrocephalus sechellensis*). *Molecular Ecology*, 10:2263–2273, 2001.

N. A. Sheehan. On the application of Markov chain Monte Carlo methods to genetic analyses on complex pedigrees. *International Statistical Review*, 68: 83–110, 2000.

P. E. Smouse, T. R. Meagher, and C. J. Kobak. Parentage analysis in *Chamaelirium luteum* (L.) Gray (Liliaceae): why do some males have higher reproductive contributions? *Journal of Evolutionary Biology*, 12:1069–1077, 1999.

E. A. Thompson. Paradox of genealogical inference. *Advances in Applied Probability*, 8:648–650, 1976.

J. L. Wang. Sibship reconstruction from genetic data with typing errors. *Genetics*, 166:1963–1979, 2004.