

# How to fit an animal model

An ecologist guide

Julien Martin

21-05-2021



# Table des Matières

<b>Preface</b>	<b>5</b>
<b>1 Introduction</b>	<b>7</b>
1.1 Data . . . . .	7
1.2 R . . . . .	8
<b>2 Univariate animal model</b>	<b>11</b>
2.1 Scenario and data . . . . .	11
2.2 Asreml-R . . . . .	12
2.3 gremlin . . . . .	23
2.4 MCMCglmm . . . . .	23
2.5 brms . . . . .	35
2.6 stan . . . . .	43
<b>3 Multivariate animal model</b>	<b>45</b>
3.1 Scenario and data . . . . .	45
3.2 Asreml-R . . . . .	46
3.3 gremlin . . . . .	60
3.4 MCMCglmm . . . . .	60
3.5 brms . . . . .	100
3.6 stan . . . . .	103
<b>4 A repeated measures animal model</b>	<b>105</b>
4.1 Scenario and data . . . . .	105
4.2 Asreml-R . . . . .	106
4.3 gremlin . . . . .	111
4.4 MCMCglmm . . . . .	111
4.5 brms . . . . .	116
4.6 stan . . . . .	116
<b>5 Quick comparison of codes</b>	<b>117</b>
5.1 Univariate model with repeated measures . . . . .	117
5.2 bivariate model . . . . .	117



# Preface

This book is a collection of tutorial from the excellent paper by ([Wilson et al., 2010](#)). Instead of just copy pasting the tutorial in a bookdown format, the tutorials have been updated to work with the newest version of the softwares and extended to present other softwares. **However, this is still a work in progress.**



Do not take anything in this manual as gospel.

## Contributors

List of people who contributed to update and extend tutorials:

- Eric Postma
- Julien Martin
- Mathieu Videlier



# Chapitre 1

## Introduction

The book provides a series of tutorials (and accompanying data files) to fit animal models in R using different packages (`ASReml-R`, `gremlin`, `MCMCglmm` and `brms/stan`). You will need to carefully follow the instructions below to first download the data files and second install the R packages. Before beginning the tutorial, we assume the reader has successfully installed the chosen R package on their computer and has saved the required data files to an appropriate directory from which they will be read. Full instructions for how to do this are provided with software distributions.

To work through the different tutorial I would recommend to create a folder where you will save your different R scripts for the tutorials.

### 1.1 Data

#### 1.1.1 Data files

You will need to download 3 data files for the tutorial in R:

- `gryphon.csv`: data on gryphon birth weight
- `gryphonRM.csv`: data
- `gryphonped.csv`

In addition, some models presented in the tutorials can take a while to run (sometimes > 1 hour), thus we are also providing model outputs to allow you continue the tutorial without waiting for the model to run.

The files are available [here](#)

I recommend to save the data and Rdata files in a subfolder `data` in the folder you will use as your working directory for R and where you will save your R scripts. It should be noted that the tutorial are using this structure to read or save data.

#### 1.1.2 Notes on data and pedigree

It is always important to take time to think carefully about the strengths and potential limitations of your pedigree information before embarking on quantitative genetic analyses. Pedigree Viewer,

written by Brian Kinghorn, is an extremely useful application for visualizing pedigrees, and can be downloaded from: <http://www-personal.une.edu.au/~bkinghor/pedigree.htm>. Pedantics an R package written by Michael Morrissey and distributed through CRAN (<http://cran.r-project.org/>) can also be used for this and offers some nice additional features for visualizing pedigree structures and generating associated statistics. Before you begin running through the tutorials, we advise taking a moment to look at the pedigree files provided with them using Pedigree Viewer or Pedantics.

## 1.2 R

You should check that you have the most current version of R and R packages. You can check the number of the current version on CRAN. If you need to update (or install) R packages, use `install.packages()` and follow the prompted instructions.

### 1.2.1 R packages

#### 1.2.1.1 asreml-r

ASReml-R is commercial software published by VSN international (<http://www.vsnl.co.uk/software/asreml/>). This package is not free and requires a key access () Additional information and guide can be find in the Asreml-R manual: (<https://asreml.kb.vsnl.co.uk/wp-content/uploads/sites/3/2018/02/ASReml-R-Reference-Manual-4.pdf>)

#### 1.2.1.2 gremlin

`gremlin` is a little monster appearing if you feed a mugwai after midnight. It is also a great and promising software to fit mixed models using a frequentist approach.

#### 1.2.1.3 MCMCglmm

MCMCglmm is an R package for Bayesian mixed model analysis written by Jarrod Hadfield. It is a freeware distributed through CRAN (<http://cran.r-project.org/>). Information and guide about the package can be find in the user manual and vignettes (<http://cran.r-project.org/web/packages/MCMCglmm/index.html>). Reference: (Hadfield, 2010, Hadfield (2021)).

This module provides some information that applies to MCMCglmm-based analyses in general, but that will not be included in other tutorials. Most importantly, this applies to some of the simplest ways of determining the performance of a run using MCMCglmm, i.e., verification of the validity of of the posterior distribution. This tutorial is not a substitute for working through the MCMCglmm course notes, which is available from CRAN (the Comprehensive R ArchiveNetwork, <http://cran.r-project.org/>, or can be accessed in R using the command `vignette("CourseNotes", "MCMCglmm")`). These tutorials do not introduce one of the main advantages of using MCMCglmm for analyses of data from natural populations -the ability to properly model non-normal responses. These capabilities are introduced in the documentation that is distributed with MCMCglmm, and available from CRAN.



#### 1.2.1.4 brms

**brms** provides an interface to fit Bayesian generalized multivariate (non-)linear multilevel models using **Stan**, which is a C++ package for obtaining full Bayesian inference (see <https://mc-stan.org/>). The formula syntax is an extended version of the syntax applied in the ‘lme4’ package to provide a familiar and simple interface for performing regression analyses.

It should be noted that if **brms** is able to fit animal model the parametrization used is not the most efficient and can take quite longer than using a different parametrization directly in **stan**.



# Chapitre 2

## Univariate animal model

This tutorial will demonstrate how to run a univariate animal model to estimate genetic variance in birth weight in the mighty gryphons.

### 2.1 Scenario and data

#### 2.1.1 Scenario

In a population of gryphons there is strong positive selection on birth weight with heavier born individuals having, on average higher fitness. To find out whether increased birth weight will evolve in response to the selection, and if so how quickly, we want to estimate the heritability of birth weight.

#### 2.1.2 Data files

Open `gryphonped.csv` and `gryphon.csv` in your text editor. The structure and contents of these files is fairly self-explanatory. The pedigree file `gryphonped.csv` contains three columns containing unique IDs that correspond to each animal, its father, and its mother. Note that this is a multigenerational pedigree, with the earliest generation (for which parentage information is necessarily missing) at the beginning of the file. For later-born individuals maternal identities are all known but paternity information is incomplete (a common situation in real world applications). The phenotype data, as well as additional factors and covariates that we may wish to include in our model are contained in `gryphon.csv`. Columns correspond to individual identity (`animal`), maternal identity (`mother`), year of birth (`byear`), sex (`sex`, where 1 is female and 2 is male), birth weight (`bwt`), and tarsus length (`tarsus`). Each row of the data file contains a record for a different offspring individual. Note that all individuals included in the data file must be included as offspring in the pedigree file.

We can read the data file, using `read.csv()` which consider by default that `NA` is the symbol for missing values and that the first line of the file contains the column headers.

It is a good idea to make sure that all variables are correctly assigned as numeric or factors:

```

gryphon$animal <- as.factor(gryphon$animal)
gryphon$mother <- as.factor(gryphon$mother)
gryphon$byear <- as.factor(gryphon$byear)
gryphon$sex <- as.factor(gryphon$sex)
gryphon$bwt <- as.numeric(gryphon$bwt)
gryphon$tarsus <- as.numeric(gryphon$tarsus)

```

Similarly we can read in the pedigree file, using `read.csv()` which consider by default that NA is the symbol for missing values and that the first line of the file contains the column headers.

```

## 'data.frame':   1309 obs. of  3 variables:
## $ id      : int  1306 1304 1298 1293 1290 1288 1284 1283 1282 1278 ...
## $ father: int   NA NA NA NA NA NA NA NA NA NA ...
## $ mother: int   NA NA NA NA NA NA NA NA NA NA ...

```

```

gryphonped$id <- as.factor(gryphonped$id)
gryphonped$father <- as.factor(gryphonped$father)
gryphonped$mother <- as.factor(gryphonped$mother)

```

Now that we have imported the data and the pedigree file, we are ready to fit an animal model.

## 2.2 Asreml-R

### 2.2.1 Running the model

First we need to load the `asreml` library:

```
library(asreml)
```

To be able to fit an animal model, `Asreml-r` needs (the inverse of) the relationship matrix using the `ainverse` function:

```
ainv <- ainverse(gryphonped)
```

We are now ready to specify our first model:

```

model1 <- asreml(
  fixed = bwt ~ 1, random = ~ vm(animal, ainv),
  residual = ~ idv(units),
  data = gryphon,
  na.action = na.method(x = "omit", y = "omit")
)

```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:47:39 2021
##           LogLik           Sigma2      DF      wall      cpu
##  1      -4128.454           1.0      853 01:47:39      0.0
##  2      -3284.272           1.0      853 01:47:39      0.0
##  3      -2354.992           1.0      853 01:47:39      0.0
##  4      -1710.357           1.0      853 01:47:39      0.0
##  5      -1363.555           1.0      853 01:47:39      0.0
##  6      -1263.516           1.0      853 01:47:39      0.0
##  7      -1247.854           1.0      853 01:47:39      0.0
##  8      -1247.185           1.0      853 01:47:39      0.0
##  9      -1247.183           1.0      853 01:47:39      0.0
```

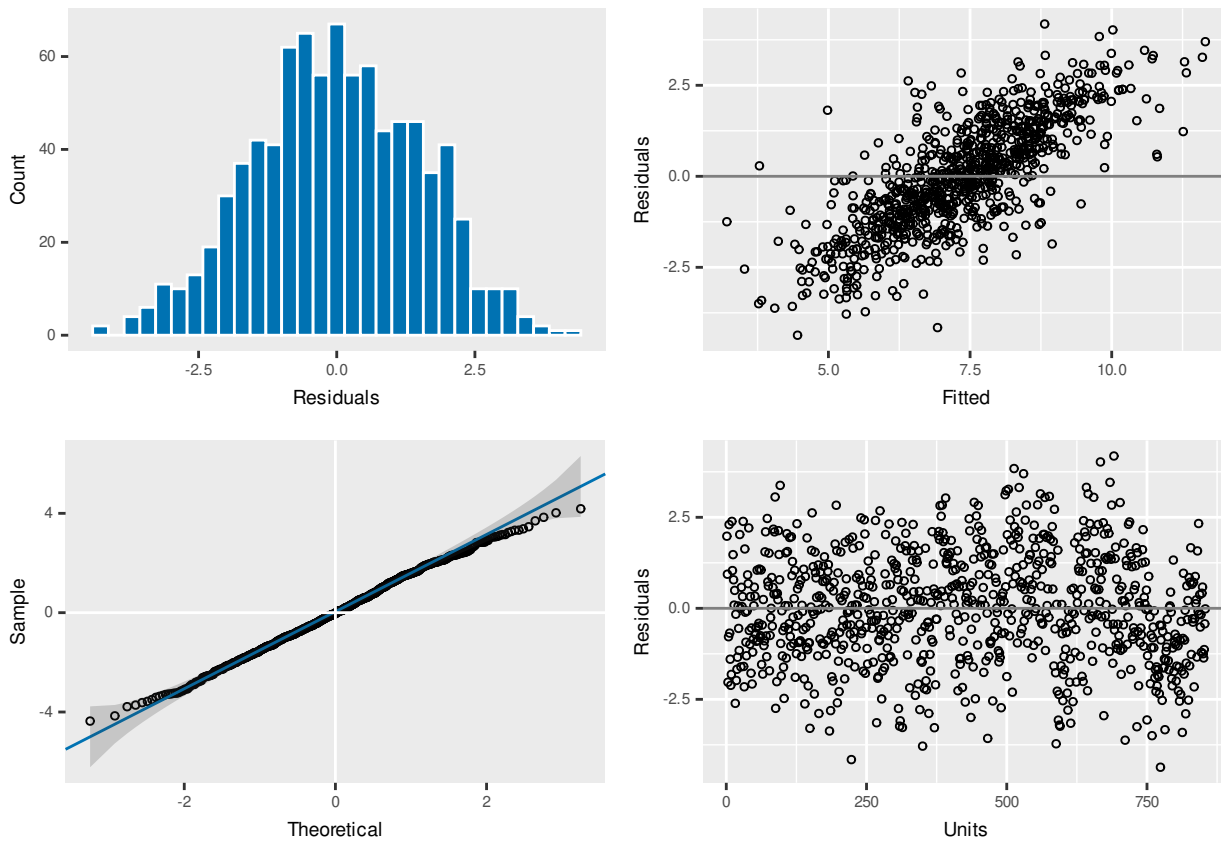
In this model, **bwt** is the response variable and the only fixed effect is the mean (the intercept, denoted as 1). The only random effect we have fitted is **animal**, which will provide an estimate of  $V_A$ . Our random **animal** effect is connected to the inverse related matrix **ainv** which integrate the relativeness or pedigree information.

**data=** specifies the name of the dataframe that contains our variables. Finally, we tell **asreml()** what to when it encounters NAs in either the dependent or predictor variables (in this case we choose to remove the records). If you use the argument “include” instead of “omit”, model will keep the NA. With **x=“include”**, the model will exchange NA with 0. Be careful you need to standardize your trait so the mean will be equal to 0, if not estimates (including covariance in multivariate models) could be strongly biased due to the the missing values. **y=“include”** will exchange NA with a factor labeled **mv** which will be included in the sparse equation. For more details see Asreml-R manual.

A note of the specification of the structure of the residuals: This simple univariate model will run fine without **residual=~idv(units)**. However, if you are going to use **vpredict()** to calculate the heritability (see below), not specifying the residuals in this way will result in a standard error for the heritability that is incorrect.

Any model has assumption which need to be checked. The model can be plot which help visualizing the distribution of the model residual and check the different assumptions.

```
plot(model1)
```



To see the estimates for the variance components, we run:

```
summary(model1)$varcomp
```

```
##               component std.error  z.ratio bound %ch
## vm(animal, ainv)  3.395398 0.6349915  5.347154    P    0
## units!units      3.828602 0.5185919  7.382687    P    0
## units!R          1.000000      NA      NA      F    0
```

We fitted a single random effect so we partitioned the phenotypic variance into two components. The `vm(animal, ainv)` variance component is  $V_A$  and is estimated as 3.4. Given that the ratio of  $V_A$  to its standard error (`z.ratio`) is considerably larger than 2 (*i.e.* the parameter estimate is more than 2 SEs from zero), this looks likely to be significant. The `units!units` component refers to the residual variance  $V_R$ , and `units$R` should be ignored. If you don't include `residual=~idv(units)` in your model specification, `units$R` will provide you with the residual variance.

## 2.2.2 Estimating heritability

We can calculate the  $h^2$  of birth weight from the components above since  $h^2 = V_A/V_P = V_A/(V_A + V_R)$ . Thus according to this model,  $h^2 = 3.4 / (3.4 + 3.83) = 0.47$ .

Alternatively we can use the `vpredict()` function to calculate  $h^2$  and its standard error. `vpredict()` function has two structures, first the model used (here `model1`) and then the estimate name with its associated equation. The equation used different V and their associated numbers

depend of the order of the different random and residual effects included in the model.

```
vpredict(model1, h2.bwt ~ V1 / (V1 + V2))
```

```
##           Estimate          SE
## h2.bwt 0.4700163 0.07650881
```

### 2.2.3 Adding fixed effects

To add fixed effects to a univariate model, we simply modify the model statement. For example, we might know (or suspect) that birth weight is a sexually dimorphic trait and therefore fit in the model.

```
model2 <- asreml(
  fixed = bwt ~ 1 + sex,
  random = ~ vm(animal, ainv),
  residual = ~ idv(units),
  data = gryphon,
  na.action = na.method(x = "omit", y = "omit")
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:47:41 2021
##           LogLik          Sigma2      DF      wall      cpu
## 1      -3364.126           1.0      852 01:47:41      0.0
## 2      -2702.117           1.0      852 01:47:41      0.0
## 3      -1978.916           1.0      852 01:47:41      0.0
## 4      -1487.834           1.0      852 01:47:41      0.0
## 5      -1236.350           1.0      852 01:47:41      0.0
## 6      -1172.771           1.0      852 01:47:41      0.0
## 7      -1165.270           1.0      852 01:47:41      0.0
## 8      -1165.093           1.0      852 01:47:41      0.0
## 9      -1165.093           1.0      852 01:47:41      0.0
```

Now we can look at the fixed effects parameters and assess their significance with a conditional Wald F-test:

```
summary(model2, coef = TRUE)$coef.fixed
wald.asreml(model2, ssType = "conditional", denDF = "numeric")
```

```
##           solution std error  z.ratio
## sex_1      0.000000      NA      NA
## sex_2      2.206996 0.1619974 13.62365
## (Intercept) 6.058669 0.1718244 35.26082
## Model fitted using the sigma parameterization.
```

```
## ASReml 4.1.0 Fri May 21 01:47:41 2021
##           LogLik           Sigma2      DF      wall      cpu
##  1      -1165.093             1.0     852 01:47:41     0.0
##  2      -1165.093             1.0     852 01:47:41     0.0
## Calculating denominator DF
```

The very small probability (Pr) in the Wald test above shows that **sex** is a highly significant fixed effect, and from the parameter estimates (`summary(model2,coef=T)$coef.fixed`) we can see that the average male (sex 2) is 2.2 kg ( $\pm 0.16$  SE) heavier than the average female (sex 1). However, when we look at the variance components in the model including **sex** as a fixed effect, we see that they have changed slightly from the previous model:

```
summary(model2)$varcomp
```

```
##           component std.error  z.ratio bound %ch
## vm(animal, ainv)  3.060441 0.5243571 5.836558      P   0
## units!units      2.938412 0.4161473 7.060991      P   0
## units!R          1.000000      NA      NA      F   0
```

In fact since **sex** effects were previously contributing to the residual variance of the model, our estimate of  $V_R$  (denoted **units!R** in the output) is now slightly lower than before. This has an important consequence for estimating heritability since if we calculate  $V_P$  as  $V_A + V_R$  then as we include fixed effects we will soak up more residual variance driving  $V_P$ . Assuming that  $V_A$  is more or less unaffected by the fixed effects fitted then as  $V_P$  goes down we expect our estimate of  $h^2$  will go up:

```
vpredict(model2, h2.bwt ~ V1 / (V1 + V2))
```

```
##           Estimate      SE
## h2.bwt 0.510171 0.07432388

##           Estimate      SE
## h2.bwt 0.4700163 0.07650881

##           Estimate      SE
## h2.bwt 0.510171 0.07432388
```

Here  $h^2$  has increased slightly from 0.47 to 0.51. Which is the better estimate? It depends on what your question is. The first is an estimate of the proportion of variance in birth weight explained by additive effects, the latter is an estimate of the proportion of variance in birth weight *after conditioning on sex* that is explained by additive effects.

An important piece of advice, each researcher should be consistent in how they name their estimates and always correctly describe which estimates they are using conditional or not (to avoid any confusion).

## 2.2.4 Adding random effects

This is done by simply modifying the model statement in the same way. For instance fitting



```
model3 <- asreml(
  fixed = bwt ~ 1 + sex,
  random = ~ vm(animal, ainv) + byear,
  residual = ~ idv(units),
  data = gryphon,
  na.action = na.method(x = "omit", y = "omit")
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:47:41 2021
##           LogLik      Sigma2      DF      wall      cpu
##  1      -2742.658         1.0      852 01:47:41      0.0
##  2      -2237.268         1.0      852 01:47:41      0.0
##  3      -1690.453         1.0      852 01:47:41      0.0
##  4      -1328.910         1.0      852 01:47:41      0.0
##  5      -1154.597         1.0      852 01:47:41      0.0
##  6      -1116.992         1.0      852 01:47:41      0.0
##  7      -1113.809         1.0      852 01:47:41      0.0
##  8      -1113.772         1.0      852 01:47:41      0.0
##  9      -1113.772         1.0      852 01:47:41      0.0
```

```
summary(model3)$varcomp
```

```
##           component std.error  z.ratio bound %ch
## byear           0.8862604 0.2695918 3.287416      P  0
## vm(animal, ainv) 2.7068665 0.4422140 6.121169      P  0
## units!units      2.3092415 0.3451025 6.691466      P  0
## units!R          1.0000000      NA      NA      F  0
```

```
(h2.3 <- vpredict(model3, h2.bwt ~ V1 / (V1 + V2 + V3)))
```

```
##           Estimate      SE
## h2.bwt 0.1501533 0.03960814
```

Here the variance in `bwt` explained by `byear` is 0.886 and, based on the `z.ratio`, appears to be significant ( $>2$ ). Thus we would conclude that year-to-year variation (*e.g.*, in weather, resource abundance) contributes to  $V_P$ . Note that although  $V_A$  has changed somewhat, as most of what is now partitioned as a birth year effect was previously partitioned as  $V_R$ . Thus what we have really done here is to partition environmental effects into those arising from year-to-year differences versus everything else, and we do not really expect much change in  $h^2$  (since now  $h^2 = V_A / (V_A + V_{BY} + V_R)$ ).

However, we get a somewhat different result if we also add a random effect of `mother` to test for maternal effects:

```
model4 <- asreml(
  fixed = bwt ~ 1 + sex,
  random = ~ vm(animal, ainv) + byear + mother,
  residual = ~ idv(units),
  data = gryphon,
  na.action = na.method(x = "omit", y = "omit")
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:47:41 2021
##          LogLik      Sigma2      DF      wall      cpu
##  1      -2033.178         1.0      852 01:47:41      0.0
##  2      -1723.734         1.0      852 01:47:41      0.0
##  3      -1396.354         1.0      852 01:47:41      0.0
##  4      -1193.012         1.0      852 01:47:41      0.0
##  5      -1107.946         1.0      852 01:47:41      0.0
##  6      -1095.327         1.0      852 01:47:41      0.0
##  7      -1094.816         1.0      852 01:47:41      0.0
##  8      -1094.815         1.0      852 01:47:41      0.0
```

```
summary(model4)$varcomp
```

```
##          component std.error  z.ratio bound %ch
## byear          0.8820313 0.2632455 3.350604    P    0
## mother          1.1184698 0.2386239 4.687167    P    0
## vm(animal, ainv) 2.2985320 0.4962496 4.631806    P    0
## units!units      1.6290034 0.3714154 4.385934    P    0
## units!R           1.0000000      NA      NA    F    0
```

```
(h2.4 <- vpredict(model4, h2.bwt ~ V1 / (V1 + V2 + V3 + V4)))
```

```
##          Estimate      SE
## h2.bwt 0.1487898 0.03861552
```

Here partitioning of significant maternal variance has resulted in a further decrease in  $V_R$  but also a decrease in  $V_A$ . The latter is because maternal effects of the sort we simulated (fixed differences between mothers) will have the consequence of increasing similarity among maternal siblings. Consequently they can look very much like additive genetic effects and if present, but unmodelled, represent a type of “common environment effect” that can - and will - cause upward bias in  $V_A$  and so  $h^2$ . The “common environment” can be conceived as the indissociable sum of the maternal additive genetic effect (such as loci affecting milk production) and the maternal environment or permanent environment (such as litter or nest environment created or modified by the mother).

### 2.2.5 Testing significance of random effects

An important point to note in this tutorial is that while the `z.ratio` (`component/std.error`) reported is a good indicator of likely statistical significance ( $>1.96?$ ), the standard errors are approximate and are not recommended for formal hypothesis testing. A better approach is to use likelihood-ratio tests (LRT).

For example, to test the significance of maternal effects we could compare models with and without the inclusion of maternal identity as a random effect and compare the final log-likelihoods of these models.

```
model4$loglik
```

```
## [1] -1094.815
```

shows that the model including maternal identity has a log-likelihood of -1094.815, and

```
model3$loglik
```

```
## [1] -1113.772
```

shows that the model excluding maternal identity has a log-likelihood of -1113.772.

A test statistic equal to twice the absolute difference in these log-likelihoods is assumed to be distributed as Chi square with **one** degree of freedom (one term of difference between the two models). In this case we would conclude that the maternal effects are highly significant since:  $2 \times (-1094.8145793 - -1113.7719147)$  equals 37.9146708, and the p-value that comes with this is:

```
1 - pchisq(2 * (model4$loglik - model3$loglik), 1)
```

```
## [1] 7.390738e-10
```

As  $P < 0.0001$  we would therefore conclude that the additional of maternal identity as a random effect significantly improves the fit of the model, given an increase in log-likelihood of approximately 19.

### 2.2.6 Further partitioning the variance

A population can be further fragmented into different groups or categories (such as females and males, juveniles and adults or treated and untreated). Some scientific questions require further and deeper analysis of the variance. To avoid multiple model (one for each group), we can directly partition the variance between groups in a unique model. In addition, by doing so, we can also test if the variance are different between groups.

As example, we decide to take the model4 and partition its additive genetic variance and residual variance by sex. It is impossible to further partition the other variances but complexify an animal model requires sufficient sample size.

First, the dataset needs to be order by group (here sex).

```

gryphon <- gryphon[order(gryphon$sex), ]

model_SEX <- asreml(
  fixed = bwt ~ 1 + sex,
  random = ~ at(sex):vm(animal, ainv) + byear + mother,
  residual = ~ dsum(~ units | sex),
  data = gryphon,
  na.action = na.method(x = "omit", y = "omit")
)

```

```

## Multi-section model using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:47:41 2021
##           LogLik      Sigma2      DF      wall      cpu
##  1      -1142.164        1.0      852 01:47:41    0.0
##  2      -1126.308        1.0      852 01:47:41    0.0
##  3      -1111.536        1.0      852 01:47:41    0.0
##  4      -1105.383        1.0      852 01:47:41    0.0
##  5      -1104.375        1.0      852 01:47:41    0.0
##  6      -1104.364        1.0      852 01:47:41    0.0

```

To partition variance, two distinct function are requires `at()` at the random level, and `dsum()` at the residual level. Both are specify structure.

```
summary(model_SEX)$varcomp
```

```

##           component std.error  z.ratio bound %ch
## byear           0.9001595 0.2690012 3.346303    P 0.0
## mother          1.3396184 0.2663118 5.030263    P 0.0
## at(sex, 1):vm(animal, ainv) 1.4372390 0.6514306 2.206281    P 0.1
## at(sex, 2):vm(animal, ainv) 1.9861434 0.9974302 1.991261    P 0.3
## sex_1!R          2.1706213 0.5542492 3.916327    P 0.0
## sex_2!R          1.7112948 0.8246188 2.075256    P 0.3

```

```
(h2.F <- vpredict(model_SEX, h2.bwt ~ V3 / (V1 + V2 + V3 + V5)))
```

```

##           Estimate      SE
## h2.bwt 0.2457811 0.1070794

```

```
(h2.M <- vpredict(model_SEX, h2.bwt ~ V4 / (V1 + V2 + V4 + V6)))
```

```

##           Estimate      SE
## h2.bwt 0.3345244 0.1619218

```

By partitioning the additive genetic variance and the residual variance, the model estimates the  $V_A$  and  $V_R$  for each group (sex). Doing so, we can calculate the  $h^2$  for each group (sex).

To test if the variance are different between groups, we can compare the model partitioned `model_SEX` and the previous model without the partitioning `model4` in a likelihood ratio test (LRT) with 2 degrees of freedom since models have two components of variance of difference.

```
model_SEX$loglik
```

```
## [1] -1104.364
```

```
model4$loglik
```

```
## [1] -1094.815
```

```
1 - pchisq(2 * (model_SEX$loglik - model4$loglik), 2)
```

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

Here, we can see the point estimates of  $h^2$  seems to differ between sexes (0.25 and 0.33), but their SE overlaps. LRT give more information and showed that partitioning the variance and the residual between group (sex) did not improved the fit of the model and so their variance are not significantly different.

### 2.2.7 Modification of model parameter

Variance represents the deviation of the distribution and it expected to be a positive values. Due to a lack of power, a structural problem in the dataset or a very low variance, Asreml-r often fixes the variance to a boundary B instead of a positive value P. When it is happen, it is generally a good idea to examine it.

To examine the boundary effect, we can exploit an alternative model where the model allowed a unstructured parameter for variance of interest or the entire variance matrix. For this example: we allowed the model to estimate any values (so allowing possible negative values of estimates) for the random and residual matrix. First, we create a temporary model `model.temp` with the exact structure to modify.

```
model.temp <- asreml(
  fixed = bwt ~ 1,
  random = ~ vm(animal, ainv) + byear + mother,
  residual = ~ idv(units),
  data = gryphon,
  na.action = na.method(x = "omit", y = "omit"),
  start.values = T
)
G.temp <- model.temp$vparameters[(1:3), ]
G.temp$Constraint <- "U"
R.temp <- model.temp$vparameters[-(1:3), ]
R.temp$Constraint[2] <- "U"
```

The argument `start.values=T` allowed the `model.temp` to change its random parameters. We can create the two different matrices and specify which parameters will be modified. For this example we modified the G and the R matrix to fit all variance to be U unstructured. it is important to not for the R matrix the lines `units!R` has to be fix to 1, so it will never change.

The object `G.temp` and `R.temp` can be implemented in the following model as new parameters using the argument `R.param` and `G.param`.

```
model5 <- asreml(
  fixed = bwt ~ 1 + sex,
  random = ~ vm(animal, ainv) + byear + mother,
  residual = ~ idv(units),
  data = gryphon,
  na.action = na.method(x = "omit", y = "omit"),
  R.param = R.temp, G.param = G.temp
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:47:42 2021
##           LogLik          Sigma2      DF      wall      cpu
##  1      -2033.178           1.0      852 01:47:42     0.0
##  2      -1723.734           1.0      852 01:47:42     0.0
##  3      -1396.354           1.0      852 01:47:42     0.0
##  4      -1193.012           1.0      852 01:47:42     0.0
##  5      -1107.946           1.0      852 01:47:42     0.0
##  6      -1095.327           1.0      852 01:47:42     0.0
##  7      -1094.816           1.0      852 01:47:42     0.0
##  8      -1094.815           1.0      852 01:47:42     0.0
```

```
summary(model5)$varcomp
```

```
##           component std.error  z.ratio bound %ch
## byear           0.8820313 0.2632455 3.350604    U    0
## mother          1.1184698 0.2386239 4.687167    U    0
## vm(animal, ainv) 2.2985320 0.4962496 4.631806    U    0
## units!units      1.6290034 0.3714154 4.385934    U    0
## units!R          1.0000000          NA          NA    F    0
```

Since `model4` did not showed boundary, the `model5` is very similar.

## 2.2.8 Covariance between two random effects

Sometimes, but changing the parameters of the model and allowing unstructured variance instead of boundary, significant negative variance can appeared. A negative variance is counter-intuitive

because statistically the mean within the random effect is less similar than expected by chance. However a possible biological reason can be hypothesized, such as a sibling competition within the nest creating a negative among-individual covariance within the nest. To test this hypotheses, we need to estimate the covariance between two random effect in a univariate model. For example, we can estimate the covariance between the additive genetic variance and the mother variance is negative using the argument `str`. This argument has two components, first the equation term `~vm(animal,Ainv)+mother` and second the structural term `~us(2):id(4500)`. Here within the structural term, we fit a 2x2 unstructured matrix `us(2)` which estimated the variance and the covariance between the random effects in the equation term. To successfully work, the structural term also requires the number of level identified within `id()`

## 2.3 gremlin

TODO (maybe just bother Matthew to do it)

Meanwhile



Figure 2.1: Keep it dry and do no feed after midnight.

## 2.4 MCMCglmm

### 2.4.1 Running the model

First load MCMCglmm:



```
library(MCMCglmm)
```

The first model we will fit is a simple animal model with no fixed effects, and only an ‘animal’ random effect relating individuals to their additive genetic values through the pedigree. First we are going to define priors. In a way we might want to avoid using priors, because we would like all of the information in our analysis to come from our data. By default MCMCglmm uses improper priors, but this can cause inferential and numerical problems. We will specify priors for the animal effect and the residual variance using the following code:

```
prior1.1 <- list(
  G = list(G1 = list(V = 1, nu = 0.002)),
  R = list(V = 1, nu = 0.002)
)
```

A prior allowed the model to fit different variance structures. With the unique random effect “animal”, we partitioned the phenotypic variance into two distinct variances matrices **G** (additive genetic) and **R** (residual). This prior specification is the simplistic one and often used because it was believed to be relatively uninformative, and is equivalent to an inverse-gamma prior with shape and scale equal to 0.001. In many cases it is relatively uninformative but when the posterior distribution for the variances has support close to zero it can behave poorly. Parameter expanded priors (See Chapter 8 of the MCMCglmm CourseNotes, available from CRAN) are gaining in popularity due to their better behaviour but for the purposes of this tutorial we will stick with the inverse-gamma prior.

We have told MCMCglmm to pay little heed to our prior expectation (**V**) by specifying a small degree of belief parameter (**nu**) of 0.002. Since this is a univariate analysis, the priors are matrix of order 1 and thus  $\text{nu} > 0$  is the smallest degree of belief that provides what is known as a ‘proper’ prior, avoiding numerical problems. In fact, there is a lot of information in the data regarding the marginal distributions of the parameters, and MCMCglmm will run most of the models that we suggest in these tutorials without priors. However, this is poor practice, but we will therefore use this simple priors throughout these tutorials. We can now fit an animal model. The model to decompose variation in birth weight into genetic and residual effects is as follows:

The lower case “animal” is a can be a **special** word for MCMCglmm. If a **pedigree** argument is provided then MCMCglmm will recognize the term **animal** as the term to use to estimate additive genetic variance. When the argument **pedigree** is not provided then the word **animal** is not different than any other variable. However, instead of providing a pedigree argument to the call to MCMCglmm function, it is much more flexible to use the **ginv** argument to specify the random effect that must be linked to the pedigree (with the inverse relatedness matrix). We thus first estimate the inverse relatedness matrix using **inverseA()** then fit the animal model.

```
Ainv <- inverseA(gryphonped)$Ainv
model1.1 <- MCMCglmm(bwt ~ 1,
  random = ~ animal, ginv = list(animal = Ainv),
  data = gryphon, prior = prior1.1)
```



```
)
```

```
##
##           MCMC iteration = 0
##
##           MCMC iteration = 1000
##
##           MCMC iteration = 2000
##
##           MCMC iteration = 3000
##
##           MCMC iteration = 4000
##
##           MCMC iteration = 5000
##
##           MCMC iteration = 6000
##
##           MCMC iteration = 7000
##
##           MCMC iteration = 8000
##
##           MCMC iteration = 9000
##
##           MCMC iteration = 10000
##
##           MCMC iteration = 11000
##
##           MCMC iteration = 12000
##
##           MCMC iteration = 13000
```

After typing this code, MCMCglmm will run, taking about 20 seconds on a modern desktop computer. The progress of the run will be printed to the screen. Also, note the warning message will be printed at the end of the run. This is natural too. In order for the MCMC algorithm to work, MCMCglmm must keep track of effects associated with unmeasured individuals appearing in the pedigree. This will not affect the answers, but when many unmeasured individuals exist, it can hinder the ability of the algorithm to explore the parameter space (more on this, and a solution, later). Lets have a look at the MCMCglmm outputs. First we will evaluate how confident we can be that MCMCglmm found good answers. By entering

```
plot(model1.1$Sol)
```

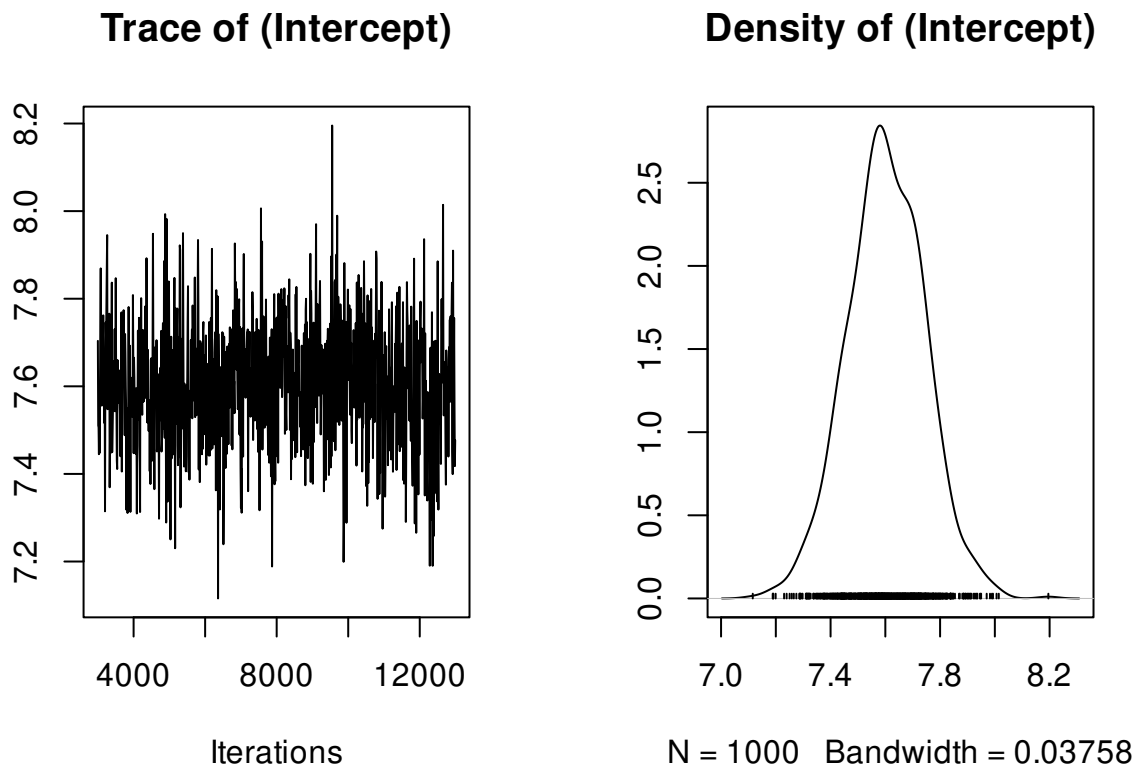


Figure 2.2: The posterior distribution of the fixed effect (the intercept, or mean) in model 1.1

in the console, we get Figure 1 (p. 5). The plot on the left shows a time series of the values of 1000 samples of the posterior distribution of the the model intercept (mean birthweight). The plot on the right shows the same data as a distribution. Complicated statistical methods for estimating population means are of course of little interest; rather, we are examining these outputs to check that MCMCglmm's algorithms worked well for our data and for this model. The important point here is that a consistent amount of variation around a largely unchanging mean value of the intercept was obtained (which give this fluctuating trace concentrated around the mean), and the posterior distribution of the intercept appears to be valid. More rigorous means of evaluation the independence of the samples in the posterior distribution (evaluating autocorrelation) are discussed in the MCMCglmm CourseNotes, available from CRAN. Note that your output for `model 1.1` may not be identical to this due to Monte Carlo (random number) error. So every times, you run the model, you will get similar but slightly different results.

The posterior distributions of the the variance components are generally of more interest to animal model users. We can view plots of the posterior distribution for the variance components for model 1.1 by

```
plot(model1.1$VCV)
```

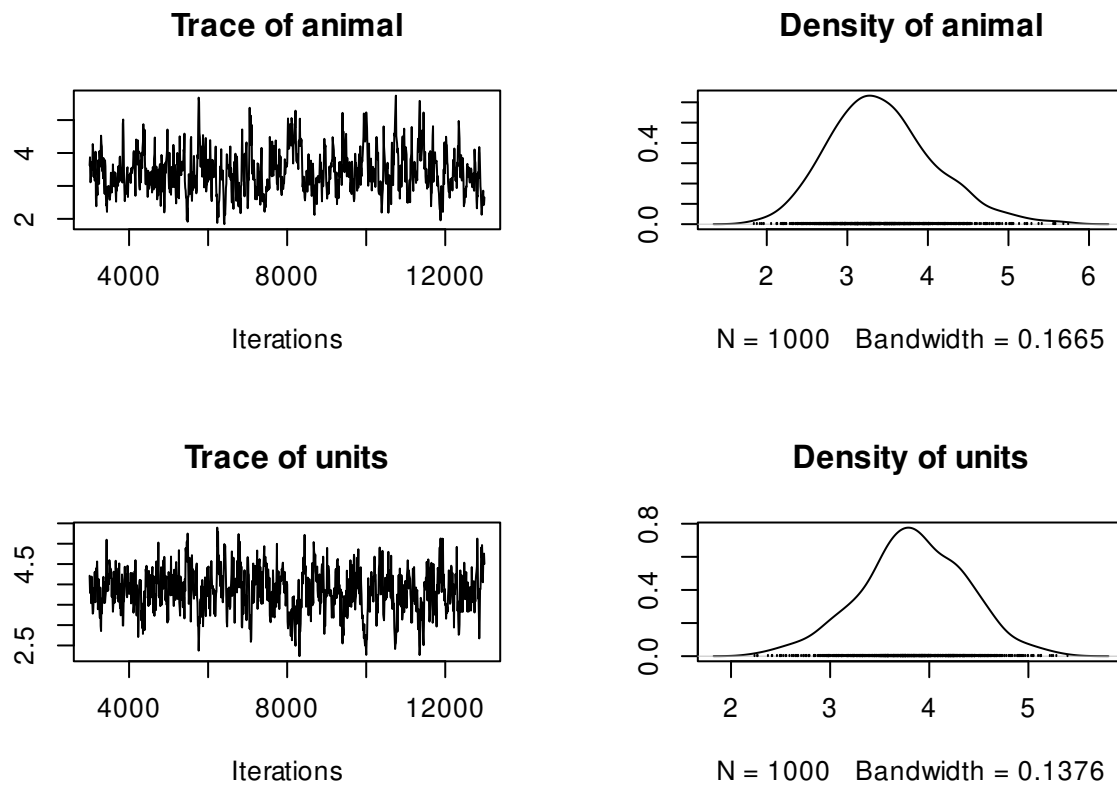


Figure 2.3: The posterior distributions of the variance components of model 1.1, based on an analysis with the default values for nitt, burnin, and thin in MCMCglmm

which generates Figure 2 (p. 6). Here we see distributions of the estimates of the additive genetic (animal) and residual (units) effects. These samples contain some autocorrelation, i.e., trends are apparent in the left-hand plot. We can deal with this easily.

### 2.4.2 Change in iteration and sampling

We will simply re-run the model for a longer number of iterations, and sample the chain less frequently. So far we have been running MCMCglmm with its default values. These defaults are a total run length of 13000 iterations, the first 3000 of which are discarded as a ‘burn-in’ period to make sure that the converges to the part of the parameter space where the maximum likelihood exists. The remaining 10000 iterations are sampled (estimates retained) every 10 iterations (the thinning interval). Because the values in the left-hand plots in figure 2 to appear to have different values at the beginning of the run, we might suspect that a longer burn-in period might be required. We can reduce the autocorrelation by lengthening the rest of the run and sampling the chain less frequently. The following code runs the same model 1.1, but is likely to produce better samples of the posterior distributions. This model should take about two minutes to analyze.

```
model1.1 <- MCMCglmm(bwt ~ 1,
  random = ~animal, ginv = list(animal = Ainv),
  data = gryphon, nitt = 65000, thin = 50, burnin = 15000,
  prior = prior1.1, verbose = FALSE
)
```

Notes that we have now included the argument `verbose=FALSE` in the `MCMCglmm` call. We will continue this throughout the tutorial so that more complete screen outputs can be included in this document without using too much space. Now produce the plots of the samples of the fixed and random effects (they have not been included in this document). Note that the autocorrelation is much reduced. A more compact way to evaluate the validity of the posterior distributions is to calculate autocorrelation among samples, as follows:

```
autocorr.diag(model1.1$VCV)
```

```
##           animal      units
## Lag 0      1.000000000  1.000000000
## Lag 50     0.206132870  0.152649149
## Lag 250    -0.003396339 -0.001107505
## Lag 500     0.011277620  0.014167911
## Lag 2500    0.046030492  0.037726623
```

We will consider these levels of autocorrelation acceptable, at least for the purposes of this tutorial. Ideally, all samples of the posterior distribution should be independent, and the autocorrelation for all lag values greater than zero should be near zero. However, in practice this will not strictly be achievable for all analytic scenarios. Certainly the levels of autocorrelation observed here should not be tolerated in any formal analysis. Note that the validity of posterior distributions of any analysis should always be checked; however, for brevity we will not continue to be so consistently diligent throughout the rest of these tutorials. We can now proceed with confidence to recover some more information from these samples. We can obtain estimates of the additive genetic and residual variance by calculating the modes of the posterior distributions:

```
posterior.mode(model1.1$VCV)
```

```
##   animal    units
## 3.211534 3.739673
```

We can obtain the Bayesian equivalent of confidence intervals by calculating the the values of the estimates that bound 95% (or any other proportion) of the posterior distributions:

```
HPDinterval(model1.1$VCV)
```

```
##           lower    upper
## animal 2.153336 4.655585
## units  2.930350 4.888197
## attr(,"Probability")
## [1] 0.95
```

### 2.4.3 Change priors parameters

We specified weak priors in this analyses. Now we will check whether or not proper priors would have influenced the results that we obtained. The simplest way to do this is to rerun the model with different priors. Here we construct priors with a larger degree of belief parameter ( $\nu$ ), and we will specify that a large proportion (50%) of the variation is under genetic control ( $V$ ):

```
p.var <- var(gryphon$bwt, na.rm = TRUE)
prior1.1.2 <- list(
  G = list(G1 = list(V = matrix(p.var * 0.05), nu = 1)),
  R = list(V = matrix(p.var * 0.95), nu = 1)
)

model1.1.2 <- MCMCglmm(bwt ~ 1,
  random = ~animal, ginv = list(animal = Ainv),
  data = gryphon, prior = prior1.1.2, nitt = 65000, thin = 50,
  burnin = 15000, verbose = FALSE
)
posterior.mode(model1.1$VCV)
```

```
## animal units
## 3.211534 3.739673
```

```
posterior.mode(model1.1.2$VCV)
```

```
## animal units
## 2.901751 3.972179
```

and we can therefore conclude that the difference in the priors has little effect on the outcome of the analysis. This is typical for an analysis where lots of data are available relative to the complexity of the model, but is often not the case. In all cases, it is important to check the effect of priors on conclusions drawn from a model. In addition, you can also specify the prior with previous knowledge or expectation for the variance.

### 2.4.4 Estimating heritability

A useful property of Bayesian posterior distributions is that we can apply almost any transformation to these distributions and they will remain valid. This applies to the calculation of heritabilities. We can obtain an estimate of the heritability by applying the basic formula  $h^2 = V_A / V_P$  to each sample of the posterior distribution:

```
posterior.heritability1.1 <- model1.1$VCV[, "animal"] /
  (model1.1$VCV[, "animal"] + model1.1$VCV[, "units"])
HPDinterval(posterior.heritability1.1, 0.95)
```

```
## lower upper
```

```
## var1 0.3031621 0.5995078
## attr("Probability")
## [1] 0.95
```

```
posterior.mode(posterior.heritability1.1)
```

```
##      var1
## 0.4546461
```

Generate a plot of the posterior distribution of this heritability estimate:

```
plot(posterior.heritability1.1)
```

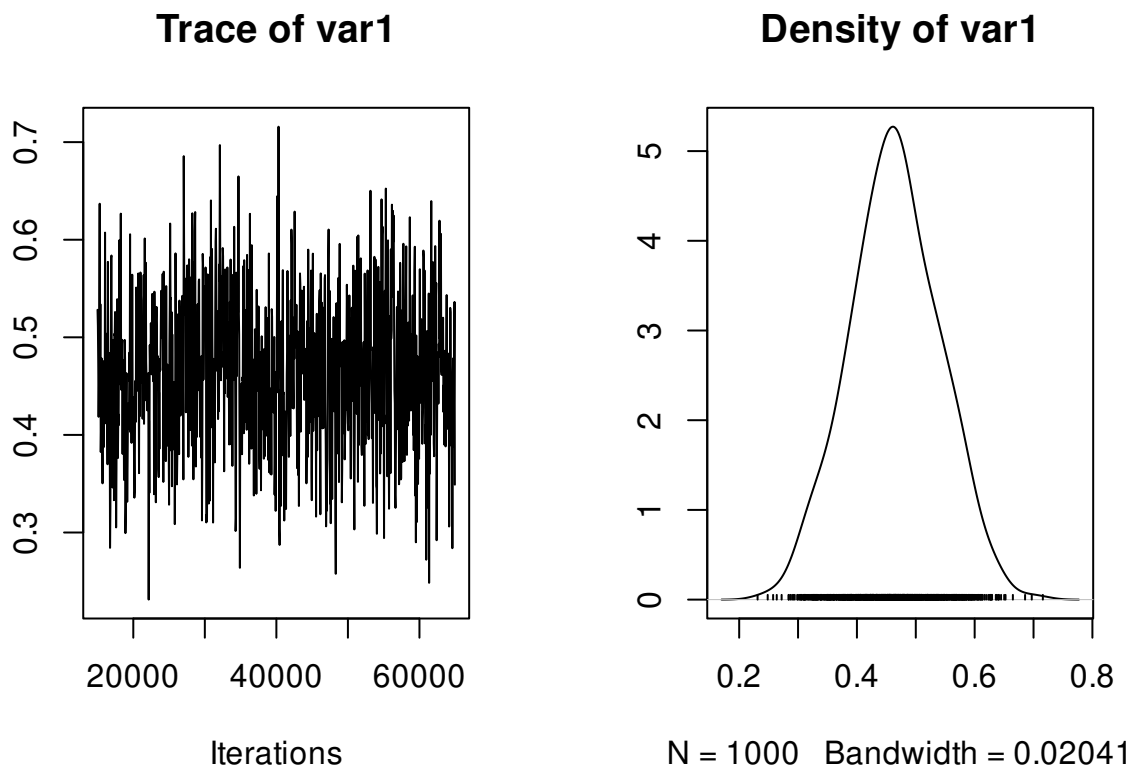


Figure 2.4: The posterior distributions the heritability from model 1.1

### 2.4.5 Adding fixed effects

To add effects to a univariate model, we simply modify the fixed effect part of the model specification:

```
model1.2 <- MCMCglmm(bwt ~ sex,
  random = ~animal, ginv = list(animal = Ainv),
  data = gryphon, prior = prior1.1,
```

```
nitt = 65000, thin = 50, burnin = 15000, verbose = FALSE
)
```

We can assess the significance of `sex` as a fixed effect by examining its posterior distribution.

```
posterior.mode(model1.2$Sol[, "sex2"])
```

```
##      var1
## 2.220736
```

```
HPDinterval(model1.2$Sol[, "sex2"], 0.95)
```

```
##      lower      upper
## var1 1.867391 2.508573
## attr("Probability")
## [1] 0.95
```

The posterior distribution of the `sex2` term does not overlap zero. Thus, we can infer that sex has an effect on birthweight (presence of a sexual dimorphism) in this model and is a useful addition to the model, for most purposes. It is also worth noting that the variance components have changed slightly:

```
posterior.mode(model1.2$VCV)
```

```
##      animal      units
## 3.096504 2.787671
```

In fact since sex effects were previously contributing to the residual variance of the model our estimate of  $V_R$  (denoted 'units' in the output) is now slightly lower than before. This has an important consequence for estimating heritability since if we calculate  $V_P$  as  $V_A + V_R$  then as we include fixed effects we will soak up more residual variance driving  $V_P$ . Assuming that  $V_A$  is more or less unaffected by the fixed effects fitted then as  $V_P$  goes down we expect our estimate of  $h^2$  will go up.

```
posterior.heritability1.2 <- model1.2$VCV[, "animal"] /
  (model1.2$VCV[, "animal"] + model1.2$VCV[, "units"])
posterior.mode(posterior.heritability1.2)
```

```
##      var1
## 0.5179268
```

```
HPDinterval(posterior.heritability1.2, 0.95)
```

```
##      lower      upper
```

```
## var1 0.3686665 0.6418853
## attr("Probability")
## [1] 0.95
```

Here  $h^2$  has increased slightly from 0.4829 to 0.5079 (again, your values may differ slightly due to Monte Carlo error). Which is the better estimate? It depends on what your question is. The first is an estimate of the proportion of variance in birth weight explained by additive effects, the latter is an estimate of the proportion of variance in birth weight after conditioning on sex that is explained by additive effects. An important piece of advice, each researcher should be consistent in how they name their estimates and always correctly describe which estimates they are using conditional or not (to avoid any confusion).

## 2.4.6 Adding random effects

This is done by simply modifying the model statement in the same way, but requires addition of a prior for the new random effect. For instance, we can fit an effect of birth year:

```
prior1.3 <- list(
  G = list(G1 = list(V = 1, nu = 0.002), G2 = list(V = 1, nu = 0.002)),
  R = list(V = 1, nu = 0.002))
model1.3 <- MCMCglmm(bwt ~ sex,
  random = ~ animal + byear, ginv = list(animal = Ainv),
  data = gryphon,
  nitt = 65000, thin = 50, burnin = 15000,
  prior = prior1.3, verbose = FALSE
)
posterior.mode(model1.3$VCV)
```

```
##      animal      byear      units
## 2.6462756 0.8360178 2.2879627
```

Here the variance in birth weight explained by birth year is 0.7887. Note that although  $V_A$  has changed somewhat, most of what is now partitioned as a **birth year** effect was previously partitioned as  $V_R$ . Thus what we have really done here is to partition environmental effects into those arising from year to year differences versus everything else, and we do not really expect much change in  $h^2$  (since now  $h^2 = V_A / (V_A + V_{BY} + V_R)$ ). However, we get a somewhat different result if we also add a random effect of **mother** to test for maternal effects:

```
p.var <- var(gryphon$bwt, na.rm = TRUE)
prior1.4 <- list(
  G = list(
    G1 = list(V = 1, nu = 0.002),
    G2 = list(V = 1, nu = 0.002),
    G3 = list(V = 1, nu = 0.002)),
  R = list(V = 1, nu = 0.002)
)
```



```

model1.4 <- MCMCglmm(bwt ~ sex,
  random = ~ animal + byear + mother,
  ginv = list(animal = Ainv), data = gryphon,
  nitt = 65000, thin = 50, burnin = 15000,
  prior = prior1.4, verbose = FALSE
)
posterior.mode(model1.4$VCV)

```

```

##      animal      byear      mother      units
## 2.4964379 0.8993490 0.9651266 1.3814133

```

Here partitioning of significant maternal variance has resulted in a further decrease in  $V_R$  but also a decrease in  $V_A$ . The latter is because maternal effects of the sort we simulated (fixed differences between mothers) will have the consequence of increasing similarity among maternal siblings. Consequently they can look very much like additive genetic effects and if present, but unmodelled, represent a type of ‘common environment effect’ that can - and will- cause upward bias in  $V_A$  and so  $h^2$ . Let’s compare the estimates of heritability from each of models 1.2, 1.3 and 1.4:

```

posterior.heritability1.3 <- model1.3$VCV[, "animal"] /
  (model1.3$VCV[, "animal"] + model1.3$VCV[, "byear"] + model1.3$VCV[, "units"])
posterior.heritability1.4 <- model1.4$VCV[, "animal"] /
  (model1.4$VCV[, "animal"] + model1.4$VCV[, "byear"] + model1.4$VCV[, "mother"] + model1.4$VCV[, "units"])
posterior.mode(posterior.heritability1.2)

```

```

##      var1
## 0.5179268

```

```

posterior.mode(posterior.heritability1.3)

```

```

##      var1
## 0.4389851

```

```

posterior.mode(posterior.heritability1.4)

```

```

##      var1
## 0.4409629

```

### 2.4.7 Testing significance of variance components

While testing the significance of fixed effects by evaluating whether or not their posterior distributions overlap zero was simple and valid, this approach does not work for variance components. Variance components are bounded to be positive (given a proper prior), and thus even when a random effect is not meaningful, its posterior distribution will never overlap zero. Model comparisons can be performed using the deviance information criterion (DIC), although it should be noted

that the properties of DIC are not well understood and that the DIC may be focused at the wrong level for most people's intended level of inference - particularly with non-Gaussian responses. The implementation of DIC in MCMCglmm is further described in the reference manual. DIC values are calculated by MCMCglmm by default. Briefly, DIC like other information criteria balance model fit and model complexity simultaneously, and small values of DIC are preferred. We can compare models 1.4 and 1.3, i.e., models with and without the mother term:

```
model1.3$DIC
```

```
## [1] 3549.694
```

```
model1.4$DIC
```

```
## [1] 3316.696
```

model 1.4 has a much lower DIC value. Since the maternal effect term is the only difference between the models, we can consider the inclusion of this term statistically justifiable. We should note however that DIC has a large sampling variance and should probably only be calculated based on much longer MCMC runs.

### 2.4.8 Further partitioning variance

A population can be further fragmented into different groups or categories (such as females and males, juveniles and adults or treated and untreated). Some scientific questions require further and deeper analysis of the variance. To avoid multiple model (one for each group), we can directly partition the variance between groups in a unique model. In addition, by doing so, we can also test if the variance are different between groups.

As example, we can partition the additive genetic variance and residual variance by sex. It is impossible to further partition the other variances but complexify an animal model requires sufficient sample size.

```
prior1.4.SEX <- list(
  G = list(G1 = list(V = diag(2), nu = 1.002), G2 = list(V = 1, nu = 0.002), G3 = list(V =
  R = list(V = diag(2), nu = 1.002)
)

model1.4.SEX <- MCMCglmm(bwt ~ sex,
  random = ~ idh(sex):animal + byear + mother,
  rcov = ~ idh(sex):units,
  ginv = list(animal = Ainv), data = gryphon, nitt = 65000, thin = 50, burnin = 15000,
  prior = prior1.4.SEX, verbose = FALSE
)
posterior.mode(model1.4.SEX$VCV)
```

```
## sex1.animal sex2.animal      byear      mother sex1.units sex2.units
##   0.8704608   1.3035599   0.7596410   1.2952954   2.2853356   1.5491127
```

```
posterior.heritability1.4.FEM <- model1.4.SEX$VCV[, "sex1.animal"] /
  (model1.4.SEX$VCV[, "sex1.animal"] + model1.4.SEX$VCV[, "byear"] +
    model1.4.SEX$VCV[, "mother"] + model1.4.SEX$VCV[, "sex1.units"])
posterior.heritability1.4.MAL <- model1.4.SEX$VCV[, "sex2.animal"] /
  (model1.4.SEX$VCV[, "sex2.animal"] + model1.4.SEX$VCV[, "byear"] +
    model1.4.SEX$VCV[, "mother"] + model1.4.SEX$VCV[, "sex2.units"])

posterior.mode(posterior.heritability1.4.FEM)
```

```
##      var1
## 0.1399386
```

```
HPDinterval(posterior.heritability1.4.FEM, 0.95)
```

```
##           lower      upper
## var1 0.04297738 0.4022488
## attr("Probability")
## [1] 0.95
```

```
posterior.mode(posterior.heritability1.4.MAL)
```

```
##      var1
## 0.221888
```

```
HPDinterval(posterior.heritability1.4.MAL, 0.95)
```

```
##           lower      upper
## var1 0.03165791 0.5510488
## attr("Probability")
## [1] 0.95
```

Here, we can estimate the heritability for each sex. Both doesn't overlap with zero, so we can conclude both sexes have significant. However due to their overlaps CIs, we can not conclude the heritability is not significantly different between sexes. An important quote to remember is "A difference in significance is not a significant difference"

## 2.4.9 Modification of model parameter

### 2.4.10 Covariance between two random effects

## 2.5 brms

### 2.5.1 Running the model

First we need to load the `brms` library:

```
library(brms)
```

```
()
```

To be able to fit an animal model, brms needs (the inverse of) the relationship matrix using the `nadiv` package `()`

```
Amat <- as.matrix(nadiv::makeA(gryphonped))
```

We are now ready to specify our first model: The structure of a brms model is similar to `lme4`, the random effect is added to the model with the term `(1 | gr(animal, cov = Amat))`. `()`

```
brms_m1.1 <- brm(
  bwt ~ 1 + (1 | gr(animal, cov = Amat)),
  data = gryphon,
  data2 = list(Amat = Amat),
  family = gaussian(),
  chains = 2, cores = 2, iter = 1000
)
save(brms_m1.1, file = "data/brms_m1_1.rda")
```

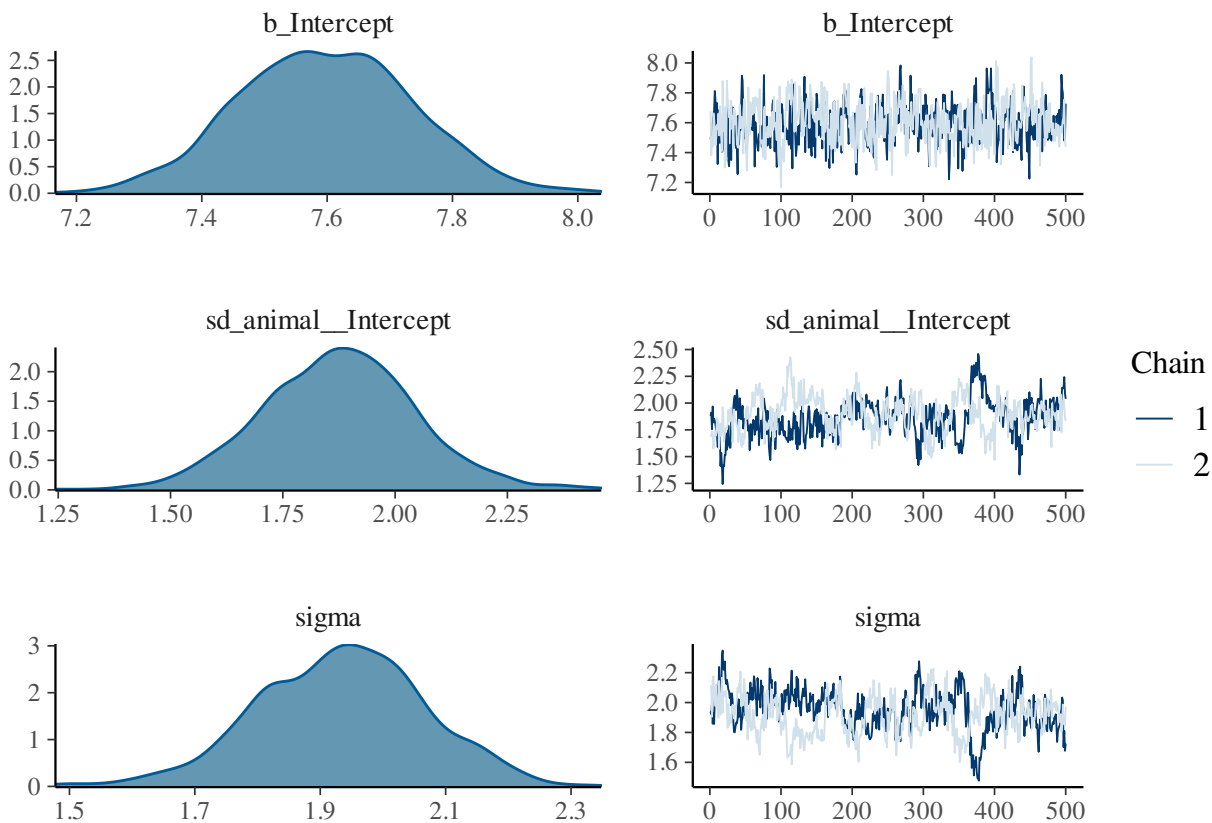
The result of the long model calculation is save in a spare file `brms_m1_1.rda`. reloading it, we can examine (or directly using the model) the variance estimate and their distributions It is also possible to calcule the heritability using `xxx`

```
load("data/brms_m1_1.rda")
summary(brms_m1.1)
```

```
## Family: gaussian
## Links: mu = identity; sigma = identity
## Formula: bwt ~ 1 + (1 | gr(animal, cov = Amat))
## Data: gryphon (Number of observations: 854)
## Samples: 2 chains, each with iter = 1000; warmup = 500; thin = 1;
##          total post-warmup samples = 1000
##
## Group-Level Effects:
## ~animal (Number of levels: 854)
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)    1.88     0.17    1.54    2.23 1.03      74      99
##
## Population-Level Effects:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept      7.60     0.14    7.33    7.86 1.01     428     727
##
## Family Specific Parameters:
```

```
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma      1.93      0.13    1.66    2.18 1.04      71      112
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
plot(brms_m1.1)
```



```
v_animal <- (VarCorr(brms_m1.1, summary = FALSE)$animal$sd)^2
v_r <- (VarCorr(brms_m1.1, summary = FALSE)$residual$sd)^2
h.bwt <- as.mcmc(v_animal / (v_animal + v_r))
summary(h.bwt)
```

```
##
## 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
```

```
##      0.484221      0.074533      0.002357      0.009275
##
## 2. Quantiles for each variable:
##
##   2.5%   25%   50%   75%  97.5%
## 0.3433 0.4338 0.4841 0.5350 0.6369
```

## 2.5.2 Adding fixed effects

To add effects to a univariate model we simply modify the fixed effect portion of the model specification:

```
brms_m1.2 <- brm(
  bwt ~ 1 + sex + (1 | gr(animal, cov = Amat)),
  data = gryphon,
  data2 = list(Amat = Amat),
  family = gaussian(),
  chains = 2, cores = 2, iter = 1000
)

save(brms_m1.2, file = "data/brms_m1_2.rda")
```

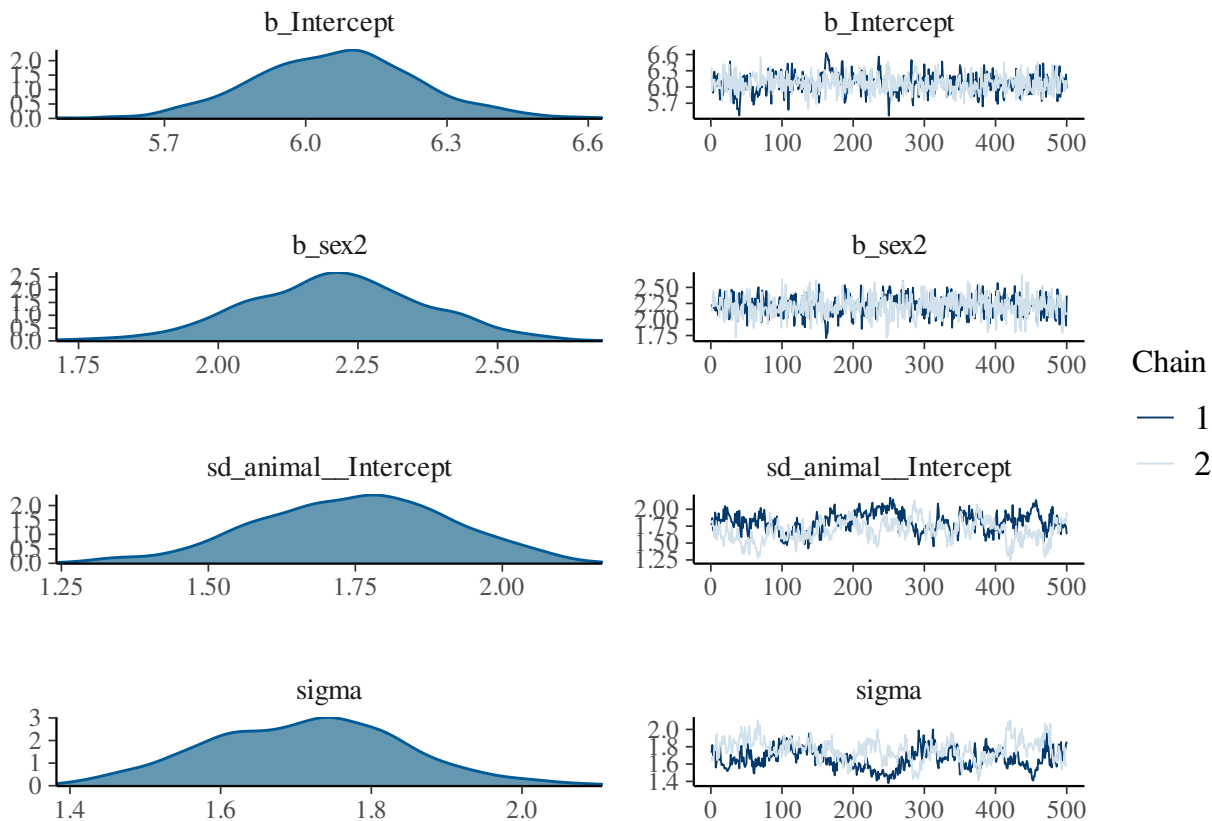
```
load("data/brms_m1_2.rda")
summary(brms_m1.2)
```

```
## Warning: Parts of the model have not converged (some Rhats are > 1.05). Be careful when

## Family: gaussian
## Links: mu = identity; sigma = identity
## Formula: bwt ~ 1 + sex + (1 | gr(animal, cov = Amat))
## Data: gryphon (Number of observations: 854)
## Samples: 2 chains, each with iter = 1000; warmup = 500; thin = 1;
##          total post-warmup samples = 1000
##
## Group-Level Effects:
## ~animal (Number of levels: 854)
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)    1.75     0.16    1.42    2.05 1.13      13      113
##
## Population-Level Effects:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept      6.06     0.17    5.74    6.41 1.00     357     574
## sex2           2.21     0.16    1.90    2.52 1.00     722     657
##
## Family Specific Parameters:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
```

```
## sigma      1.71      0.13      1.47      1.98 1.12      14      97
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
plot(brms_m1.2)
```



```
summary(brms_m1.2)$fixed
```

```
## Warning: Parts of the model have not converged (some Rhats are > 1.05). Be careful when
```

```
##           Estimate Est.Error 1-95% CI u-95% CI      Rhat Bulk_ESS Tail_ESS
## Intercept 6.064853 0.1726459 5.735170 6.410117 1.003015      357      574
## sex2      2.210675 0.1574542 1.898645 2.520026 1.002318      722      657
```

```
summary(brms_m1.2)$random
```

```
## Warning: Parts of the model have not converged (some Rhats are > 1.05). Be careful when
```

```
## $animal
##           Estimate Est.Error 1-95% CI u-95% CI      Rhat Bulk_ESS Tail_ESS
## sd(Intercept) 1.747083 0.1632548 1.419377 2.050884 1.126556      13      113
```

### 2.5.3 Adding random effects

This is done by simply modifying the model statement in the same way, but requires addition of a prior for the new random effect. For instance, we can fit an effect of birth year:

```
brms_m1.3 <- brm(
  bwt ~ 1 + sex + (1 | gr(animal, cov = Amat)) + (1 | byear) + (1 | mother),
  data = gryphon,
  data2 = list(Amat = Amat),
  family = gaussian(),
  chains = 2, cores = 2, iter = 1000
)
save(brms_m1.3, file = "data/brms_m1_3.rda")
```

```
load("data/brms_m1_3.rda")
summary(brms_m1.3)
```

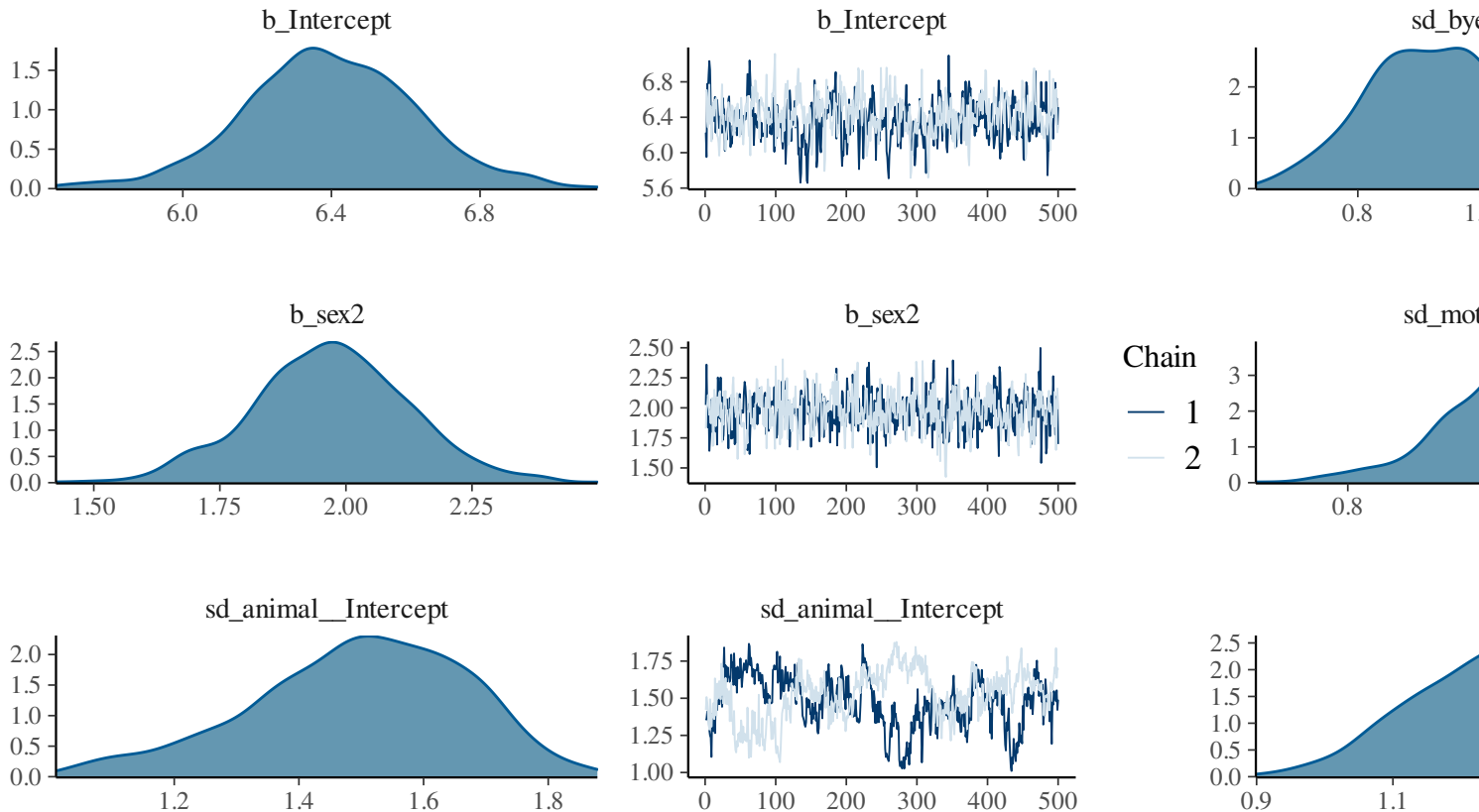
```
## Warning: Parts of the model have not converged (some Rhats are > 1.05). Be careful when
```

```
## Family: gaussian
## Links: mu = identity; sigma = identity
## Formula: bwt ~ 1 + sex + (1 | gr(animal, cov = Amat)) + (1 | byear) + (1 | mother)
## Data: gryphon (Number of observations: 854)
## Samples: 2 chains, each with iter = 1000; warmup = 500; thin = 1;
##           total post-warmup samples = 1000
##
## Group-Level Effects:
## ~animal (Number of levels: 854)
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)      1.50      0.17    1.11    1.78 1.16          9        51
##
## ~byear (Number of levels: 34)
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)      0.96      0.14    0.71    1.28 1.00        400       642
##
## ~mother (Number of levels: 394)
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)      1.05      0.11    0.83    1.26 1.01        193       379
##
## Population-Level Effects:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept          6.39      0.23    5.93    6.87 1.02        251       504
## sex2                1.97      0.15    1.67    2.28 1.00        644       785
##
## Family Specific Parameters:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
```



```
## sigma      1.30      0.14      1.03      1.56 1.17      9      55
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
plot(brms_m1.3, ask = FALSE, N = 3)
```



## 2.5.4 Testing significance of variance components

## 2.5.5 Further partitioning of the variance

Depending of the research question and the presence of different group within the dataset, MCMCglmm allowed to partition the variance at different level. For the example, we can partition the additive genetic and residual variance between SEXE (male and female) to estimate the sex-specific heritability

```
brms_m1.4 <- brm(
  bwt ~ 1 + sex + (1 | gr(animal, cov = Amat)) + (1 | byear) + (1 | mother),
  data = gryphon,
  data2 = list(Amat = Amat),
  family = gaussian(),
  chains = 2, cores = 2, iter = 1000
```

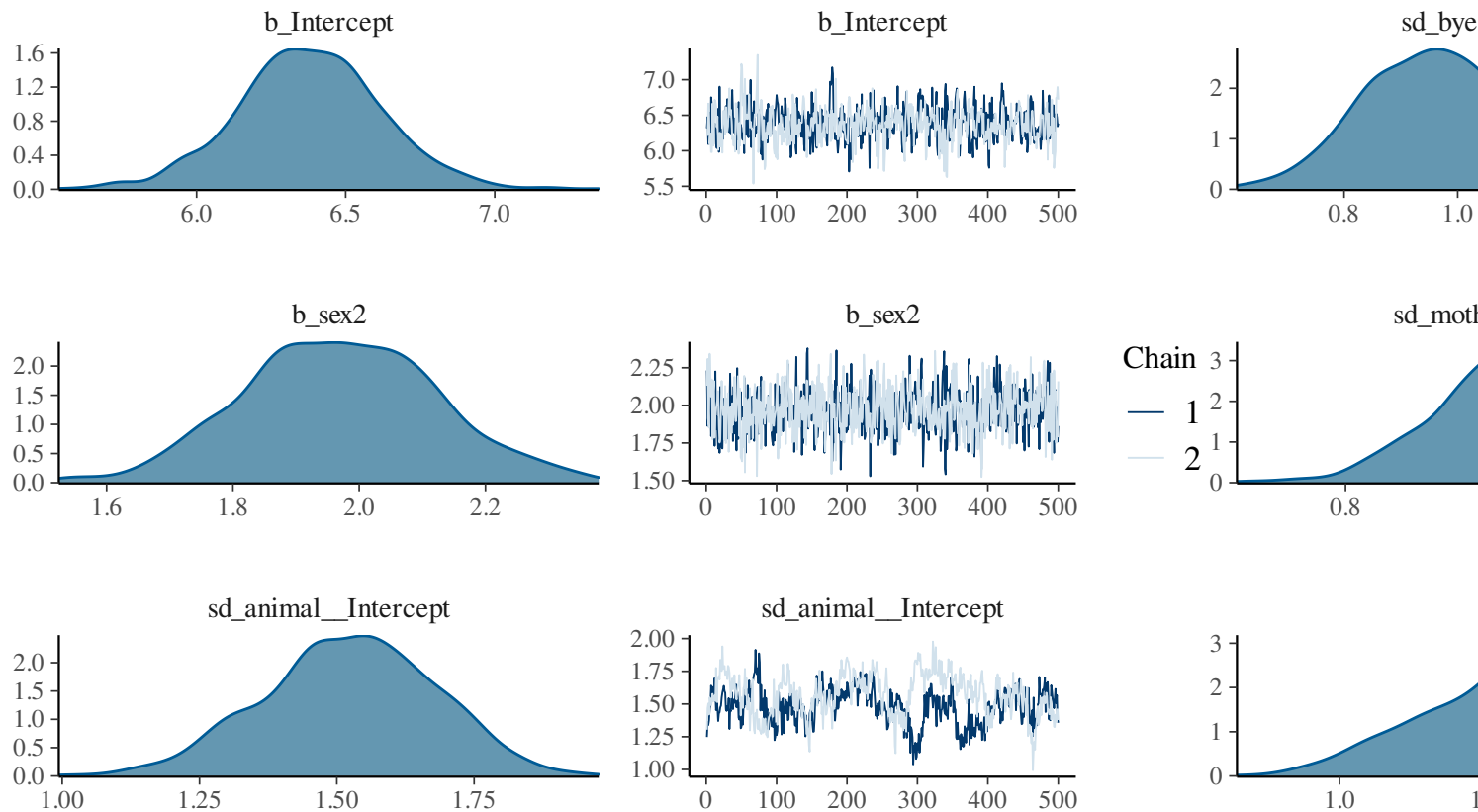
```
)
save(brms_m1.4, file = "data/brms_m1_4.rda")
```

```
load("data/brms_m1_4.rda")
summary(brms_m1.4)
```

```
## Warning: Parts of the model have not converged (some Rhats are > 1.05). Be careful when
```

```
## Family: gaussian
## Links: mu = identity; sigma = identity
## Formula: bwt ~ 1 + sex + (1 | gr(animal, cov = Amat)) + (1 | byear) + (1 | mother)
## Data: gryphon (Number of observations: 854)
## Samples: 2 chains, each with iter = 1000; warmup = 500; thin = 1;
##           total post-warmup samples = 1000
##
## Group-Level Effects:
## ~animal (Number of levels: 854)
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)      1.52      0.16    1.21    1.82 1.09        22        103
##
## ~byear (Number of levels: 34)
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)      0.98      0.15    0.73    1.32 1.00        538        677
##
## ~mother (Number of levels: 394)
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)      1.05      0.11    0.83    1.28 1.00        219        428
##
## Population-Level Effects:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept          6.38      0.24    5.91    6.86 1.00        516        579
## sex2                1.97      0.15    1.68    2.28 1.00       1044        766
##
## Family Specific Parameters:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma            1.27      0.13    1.01    1.50 1.11         16         48
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
plot(brms_m1.4, ask = FALSE, N = 3)
```



### 2.5.6 Modification of model parameter

### 2.5.7 Covariance between two random effects

## 2.6 stan

to do



# Chapitre 3

## Multivariate animal model

This tutorial will demonstrate how to run a multivariate animal model looking at birth weight and tarsus length of the phenomenal gryphons.

### 3.1 Scenario and data

#### 3.1.1 Scenario

Since natural selection rarely acts on single traits, to understand how birth weight might evolve in our population of gryphons, we may also want to think about possible covariance with other traits. If tarsus length at fledging is also under positive selection, what implications does it have for birth weight and vice versa? If the two traits are positively genetically correlated then this will facilitate evolution of larger size (since response of one trait will induce a positively correlated response in the other). If there is negative genetic covariance then this could act as an evolutionary constraint.

Using multivariate models allows the estimation of parameters relating to each trait alone (*i.e.*  $V_A$ ,  $h^2$ , etc), but also yields estimates of covariance components between traits. These include the (additive) genetic covariance  $COV_A$  which is often rescaled to give the additive genetic correlation  $r_A$ . However, covariance can also arise through other random effects (*e.g.* maternal covariance) and these sources can also be explicitly modelled in a bivariate analysis.

#### 3.1.2 gryphon files

`gryphonpedigree` and phenotypic data files are the same as those used in tutorial 1 (*i.e.*, `gryphonped.csv` and `gryphon.csv` respectively).

Reading the data

```
gryphon <- read.csv("data/gryphon.csv")
gryphon$animal <- as.factor(gryphon$animal)
gryphon$mother <- as.factor(gryphon$mother)
gryphon$byear <- as.factor(gryphon$byear)
gryphon$sex <- as.factor(gryphon$sex)
```

```
gryphon$bwt <- as.numeric(gryphon$bwt)
gryphon$tarsus <- as.numeric(gryphon$tarsus)
```

Reading the pedigree

```
gryphonped <- read.csv("data/gryphonped.csv")
gryphonped$id <- as.factor(gryphonped$id)
gryphonped$father <- as.factor(gryphonped$father)
gryphonped$mother <- as.factor(gryphonped$mother)
```

## 3.2 Asreml-R

### 3.2.1 Running the model

First we need to load the `asreml` library:

```
library(asreml)
```

For running multivariate analyses in ASReml-R, the code is slightly more complex than for the univariate case. This is because ASReml-R allows us to make different assumptions about the way in which traits might be related. So we need to explicitly code a model of the (co)variance structure we want to fit by specifying some starting values. These can be very approximate *guessimates*, but having reasonable starting values can aid convergence. Finally, we have increased the default maximum number of iterations (`maxiter`) which can help to achieve convergence for more complicated models. Another way to increase the number of iterations will be to use the `update` function. Notes that if the `LogLik` is not stabilized after several iterations, it is good indication of the model requiring more iteration.

It is also possible to let the model run without any specifying starting values but usually univariate model will allow to get some *guessimates* in the additive genetic variances.

```
ainv <- ainverse(gryphonped)
modela <- asreml(
  fixed = cbind(bwt, tarsus) ~ trait,
  random = ~ us(trait):vm(animal, ainv), init = c(1, 0.1, 1),
  residual = ~ id(units):us(trait, init = c(1, 0.1, 1)),
  data = gryphon,
  na.action = na.method(x = "include", y = "include"),
  maxit = 20
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:53 2021
##           LogLik           Sigma2           DF           wall           cpu
```

```
## 1      -5118.122      1.0  1535 01:55:53  0.0
## 2      -4358.769      1.0  1535 01:55:53  0.0
## 3      -3540.792      1.0  1535 01:55:53  0.0
## 4      -3004.970      1.0  1535 01:55:53  0.0
## 5      -2747.831      1.0  1535 01:55:53  0.0
## 6      -2687.807      1.0  1535 01:55:53  0.0
## 7      -2680.057      1.0  1535 01:55:53  0.0
## 8      -2679.743      1.0  1535 01:55:53  0.0
## 9      -2679.741      1.0  1535 01:55:53  0.0
```

```
modela <- update(modela)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:53 2021
##          LogLik      Sigma2      DF      wall      cpu
## 1      -2679.741      1.0    1535 01:55:53  0.0
## 2      -2679.741      1.0    1535 01:55:53  0.0
```

modela has fitted a bivariate model of `bwt` and `tarsus`, with the mean for each of the traits as a fixed effect (`trait`). The additive genetic variance-covariance matrix ( $\mathbf{G}$ ) is unstructured (`us`; *i.e.* all elements are free to vary) and the starting values for  $V_A$  for `bwt`,  $COV_A$  between `bwt` and `tarsus`, and  $V_A$  for `tarsus` are set to 1, 0.1 and 1, respectively. Similarly, the residual matrix is unstructured and uses the same starting values.

Note that the argument `na.action = na.method(x = "include", y = "include")` can be added to the model. In a bivariate model, it will help calculate the covariance between two traits with different missing information NA and so help imbalance phenotype and save sample size. However, it is important to scale (mean = 0, var = 1) the two traits to correctly adjust the model (see ASReml-R manual for more information). For the example, no missing information are present in the dataset.

Let's have a look at the variance components, and notice that there are now seven (co)variance components reported in the table:

```
summary(modela)$varcomp
```

##	component	std.error	z.ratio	bound	%ch
## trait:vm(animal, ainv)!trait_bwt:bwt	3.368405	0.6348356	5.305948	P	0
## trait:vm(animal, ainv)!trait_tarsus:bwt	2.459827	1.0732809	2.291876	P	0
## trait:vm(animal, ainv)!trait_tarsus:tarsus	12.345849	3.0744787	4.015591	P	0
## units:trait!R	1.000000	NA	NA	F	0
## units:trait!trait_bwt:bwt	3.849910	0.5200095	7.403539	P	0
## units:trait!trait_tarsus:bwt	3.313269	0.9129222	3.629300	P	0
## units:trait!trait_tarsus:tarsus	17.646386	2.6670308	6.616491	P	0

The first three terms are related to the genetic matrix and, in order are  $V_{A,bwt}$ ,  $COV_A$ ,  $V_{A,tarsus}$ . Below is again a line where the `units:trait!R` component equals to 1, which again can be ignored. The final three terms relate to the residual matrix and correspond to  $V_{R,bwt}$ ,  $COV_R$ ,

$V_{R,tarsus}$ . Based on our quick and dirty check (is `z.ratio` > 1.96?) all components look to be statistically significant.

We can calculate the genetic correlation as  $COV_A / \sqrt{V_{A,bwt} \cdot V_{A,tarsus}}$ . Thus this model gives an estimate of  $r_A = 0.38$ . It is also possible to estimate the residual correlation  $r_{res} = 0.4$ .

Although we can calculate this by hand, we can also use `vpredict()`, which also provides an (approximate) standard error:

```
vpredict(modela, r_A ~ V2 / sqrt(V1 * V3))
```

```
##      Estimate      SE
## r_A 0.381445 0.1299765
```

```
vpredict(modela, r_res ~ V6 / sqrt(V5 * V7))
```

```
##      Estimate      SE
## r_res 0.4019791 0.08607119
```

Of course we can also calculate the heritability of `bwt` and `tarsus` from this model:

```
vpredict(modela, h2.bwt ~ V1 / (V1 + V5))
```

```
##      Estimate      SE
## h2.bwt 0.4666469 0.07671563
```

```
vpredict(modela, h2.tarsus ~ V3 / (V3 + V7))
```

```
##      Estimate      SE
## h2.tarsus 0.4116348 0.09305947
```

### 3.2.2 Adding fixed and random effects

Fixed and random effects can be added just as for the univariate case. Given that our full model of `bwt` from tutorial 1 had `sex` as a fixed effect as well as birth year and mother as random effects, we could specify a bivariate formulation with the same complexity:

```
modelb <- asreml(
  fixed = cbind(bwt, tarsus) ~ trait + at(trait):sex,
  random = ~ us(trait, init = c(1, 0.1, 1)):vm(animal, ainv) +
    us(trait, init = c(1, 0.1, 1)):byear +
    us(trait, init = c(1, 0.1, 1)):mother,
  residual = ~ id(units):us(trait, init = c(1, 0.1, 1)),
  data = gryphon,
  na.action = na.method(x = "include", y = "include"),
```



```
maxit = 20
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:53 2021
##           LogLik           Sigma2      DF      wall      cpu
##  1      -4672.301             1.0    1533 01:55:53    0.0
##  2      -4005.615             1.0    1533 01:55:53    0.0
##  3      -3271.483             1.0    1533 01:55:53    0.0 (1 restrained)
##  4      -2761.414             1.0    1533 01:55:53    0.0 (1 restrained)
##  5      -2481.357             1.0    1533 01:55:53    0.0
##  6      -2395.858             1.0    1533 01:55:53    0.0
##  7      -2381.050             1.0    1533 01:55:53    0.0
##  8      -2380.251             1.0    1533 01:55:53    0.0
##  9      -2380.246             1.0    1533 01:55:53    0.0
```

```
modelb <- update(modelb)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:54 2021
##           LogLik           Sigma2      DF      wall      cpu
##  1      -2380.246             1.0    1533 01:55:54    0.0
##  2      -2380.246             1.0    1533 01:55:54    0.0
```

Note that we have specified a covariance structure for each random effect and an estimate of the effect of sex on both birth weight and tarsus length.

There will now be thirteen (co)variance components reported after running the code:

```
summary(modelb)$varcomp
```

```
##           component std.error    z.ratio bound %ch
## trait:byear!trait_bwt:bwt      0.9746385 0.2825727  3.4491602    P    0
## trait:byear!trait_tarsus:bwt    0.1624076 0.4185079  0.3880635    P    0
## trait:byear!trait_tarsus:tarsus  3.7383721 1.2065992  3.0982716    P    0
## trait:mother!trait_bwt:bwt      1.1445184 0.2302182  4.9714512    P    0
## trait:mother!trait_tarsus:bwt   -1.5567306 0.4051848 -3.8420260    P    0
## trait:mother!trait_tarsus:tarsus  4.8206132 1.3201300  3.6516202    P    0
## trait:vm(animal, ainv)!trait_bwt:bwt  1.9893546 0.4410246  4.5107569    P    0
## trait:vm(animal, ainv)!trait_tarsus:bwt  3.3170404 0.9032323  3.6724110    P    0
## trait:vm(animal, ainv)!trait_tarsus:tarsus 10.2294887 2.8077066  3.6433610    P    0
## units:trait!R                    1.0000000      NA      NA    F    0
## units:trait!trait_bwt:bwt      1.8443110 0.3443178  5.3564203    P    0
## units:trait!trait_tarsus:bwt    4.0142841 0.7412540  5.4155308    P    0
## units:trait!trait_tarsus:tarsus 12.4845955 2.2893363  5.4533690    P    0
```

we can estimate the different correlations using `vpredict`:

```
vpredict(modelb, r_byear ~ V2 / sqrt(V1 * V3))
```

```
##           Estimate      SE
## r_byear 0.08508312 0.2134209
```

```
vpredict(modelb, r_M ~ V5 / sqrt(V4 * V6))
```

```
##           Estimate      SE
## r_M -0.6627518 0.2487963
```

```
vpredict(modelb, r_A ~ V8 / sqrt(V7 * V9))
```

```
##           Estimate      SE
## r_A 0.7353053 0.1094747
```

```
vpredict(modelb, r_res ~ V12 / sqrt(V11 * V13))
```

```
##           Estimate      SE
## r_res 0.8365729 0.07366762
```

Now we can look at the fixed effects parameters and assess their significance with a conditional Wald F-test:

```
summary(modelb, coef = TRUE)$coef.fi
wald.asreml(modelb, denDF = "default", ssType = "conditional")$Wald
```

```
##           solution std error    z.ratio
## at(trait, tarsus):sex_1  0.0000000      NA         NA
## at(trait, tarsus):sex_2 -0.0684413 0.3823448 -0.1790041
## at(trait, bwt):sex_1    0.0000000      NA         NA
## at(trait, bwt):sex_2    1.9502053 0.1480467 13.1729086
## trait_bwt              6.3844483 0.2328210 27.4221324
## trait_tarsus           20.5936436 0.5098944 40.3880569
```

```
## Model fitted using the sigma parameterization.
```

```
## ASReml 4.1.0 Fri May 21 01:55:54 2021
```

```
##           LogLik      Sigma2      DF      wall      cpu
## 1      -2380.246          1.0    1533 01:55:54    0.1
## 2      -2380.246          1.0    1533 01:55:54    0.0
```

```
## Calculating denominator DF
```

```
##
##           Df denDF    F.inc    F.con Margin      Pr
## trait      2  52.6 1396.00 1396.00      0.00000
```

```
## at(trait, bwt):sex      1 812.8  298.40  173.50      B 0.00000
## at(trait, tarsus):sex  1 747.9    0.03    0.03      B 0.85798
```

Note that it is possible to specify a fixed effect to a specific trait by adding the number of order within `cbind` inside the argument `at(trait,x)`. For example, here we apply the fixed effect `sex` only to the response variable `tarsus`.

```
modelb_2 <- asreml(
  fixed = cbind(bwt, tarsus) ~ trait + at(trait, 2):sex,
  random = ~ us(trait, init = c(1, 0.1, 1)):vm(animal, ainv) +
    us(trait, init = c(1, 0.1, 1)):byear +
    us(trait, init = c(1, 0.1, 1)):mother,
  residual = ~ id(units):us(trait, init = c(1, 0.1, 1)),
  data = gryphon,
  na.action = na.method(x = "include", y = "include"),
  maxit = 20
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:54 2021
##           LogLik      Sigma2      DF      wall      cpu
##  1      -4810.563          1.0    1534 01:55:54     0.0
##  2      -4129.799          1.0    1534 01:55:54     0.0
##  3      -3382.529          1.0    1534 01:55:54     0.0 (1 restrained)
##  4      -2864.076          1.0    1534 01:55:54     0.0
##  5      -2574.891          1.0    1534 01:55:54     0.0
##  6      -2478.879          1.0    1534 01:55:55     0.0
##  7      -2458.305          1.0    1534 01:55:55     0.1
##  8      -2456.425          1.0    1534 01:55:55     0.0
##  9      -2456.377          1.0    1534 01:55:55     0.0
## 10      -2456.376          1.0    1534 01:55:55     0.0
```

```
summary(modelb_2, coef = TRUE)$coef.fi
wald.asreml(modelb_2, denDF = "default", ssType = "conditional")$Wald
```

```
##           solution std error  z.ratio
## at(trait, tarsus):sex_1  0.000000      NA      NA
## at(trait, tarsus):sex_2 -3.267042 0.2953279 -11.06242
## trait_bwt              7.636226 0.2389515  31.95722
## trait_tarsus           22.703658 0.4827348  47.03133
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:55 2021
##           LogLik      Sigma2      DF      wall      cpu
##  1      -2456.376          1.0    1534 01:55:55     0.0
##  2      -2456.376          1.0    1534 01:55:55     0.0
## Calculating denominator DF
```

```
##
##              Df denDF  F.inc  F.con Margin      Pr
## trait              2  50.7 1233.0 1233.0      0.00000e+00
## at(trait, tarsus):sex 1 522.9 122.4 122.4      B 1.02886e-25
```

### 3.2.3 Significance testing

Under the model above  $r_M$  is estimated as -0.66 and the `z.ratio` associated with the corresponding covariance ( $COV_M$ ) is  $>2$  (in absolute terms). We might therefore infer that there is evidence for a strong negative correlation between the traits with respect to the mother and that while maternal identity explains variance in both traits those mothers that tend to produce heavier offspring actually tend to produce offspring with shorter tarsus lengths.

To formally test if  $COV_M$  is significantly different from zero, we can compare the log-likelihood for this model:

```
modelb$loglik
```

```
## [1] -2380.246
```

to a model in which we specify that  $COV_M=0$ . Since this constraint reduces the number of parameters to be estimated by one, we can use a likelihood ratio test (LRT) with one degree of freedom. To run the constrained model, we modify the G structure defined for the `mother` random effect to diagonal (`diag`), which means we only estimate the variances (the diagonal of the matrix) but not the covariance (the covariance are fixed to 0):

```
modelc <- asreml(
  fixed = cbind(bwt, tarsus) ~ trait + at(trait):sex,
  random = ~ us(trait, init = c(1, 0.1, 1)):vm(animal, ainv) +
    us(trait, init = c(1, 0.1, 1)):byear +
    diag(trait, init = c(1, 1)):mother,
  residual = ~ id(units):us(trait, init = c(1, 0.1, 1)),
  data = gryphon,
  na.action = na.method(x = "include", y = "include"),
  maxit = 20
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:55 2021
##           LogLik      Sigma2      DF      wall      cpu
## 1      -4677.820          1.0    1533 01:55:55      0.0
## 2      -4010.442          1.0    1533 01:55:55      0.0
## 3      -3275.409          1.0    1533 01:55:55      0.0
## 4      -2763.519          1.0    1533 01:55:55      0.0
## 5      -2483.732          1.0    1533 01:55:55      0.0
## 6      -2400.242          1.0    1533 01:55:55      0.0
## 7      -2386.663          1.0    1533 01:55:55      0.0
```

```
## 8      -2386.049          1.0   1533 01:55:55    0.0
## 9      -2386.045          1.0   1533 01:55:56    0.0
```

You can run `summary(modelc)$varcomp` to confirm this worked. We can now obtain the log-likelihood of this model and compare this to that of `modelb` using a likelihood ratio test:

```
modelc$loglik
```

```
## [1] -2386.045
```

We can see that the model log-likelihood is now -2386.05. And comparing the models using a likelihood ratio test:

```
2 * (modelb$loglik - modelc$loglik)
```

```
## [1] 11.59835
```

So our chi-square test statistic is  $\chi_1^2 = 11.6$ . The p-value that goes with this is obtained by:

```
1 - pchisq(2 * (modelb$loglik - modelc$loglik), 1)
```

```
## [1] 0.0006601037
```

We would therefore conclude that the maternal covariance is significantly different from zero.

We could apply the same procedure to show that the residual (environmental) covariance and the genetic covariance estimates are significantly greater than zero (*i.e.*, heavier individuals tend to have longer tarsus lengths). In contrast, we should find that the byear covariance between the two traits is non-significant.

```
modeld <- asreml(
  fixed = cbind(bwt, tarsus) ~ trait + at(trait):sex,
  random = ~ us(trait, init = c(1, 0.1, 1)):vm(animal, ainv) +
    diag(trait, init = c(1, 1)):byear +
    us(trait, init = c(1, 0.1, 1)):mother,
  residual = ~ id(units):us(trait, init = c(1, 0.1, 1)),
  data = gryphon,
  na.action = na.method(x = "include", y = "include"),
  maxit = 20
)
```

```
## Model fitted using the sigma parameterization.
```

```
## ASReml 4.1.0 Fri May 21 01:55:56 2021
```

```
##          LogLik          Sigma2      DF      wall      cpu
## 1      -4672.708           1.0    1533 01:55:56    0.1
## 2      -4005.953           1.0    1533 01:55:56    0.0
## 3      -3271.737           1.0    1533 01:55:56    0.0 (1 restrained)
```

```
## 4      -2761.626      1.0  1533 01:55:56      0.0 (1 restrained)
## 5      -2481.649      1.0  1533 01:55:56      0.0
## 6      -2395.992      1.0  1533 01:55:56      0.0
## 7      -2381.136      1.0  1533 01:55:56      0.0
## 8      -2380.331      1.0  1533 01:55:56      0.0
## 9      -2380.326      1.0  1533 01:55:56      0.0
```

```
2 * (modelb$loglik - modeld$loglik)
```

```
## [1] 0.1600641
```

```
1 - pchisq(2 * (modelb$loglik - modeld$loglik), 1)
```

```
## [1] 0.6890975
```

### 3.2.4 Estimate directly the genetic correlation within the model

Within ASreml-r, different matrix structure can be specify such as `us`, `corg`, `diag`, etc (cf see the ASreml-r guide). Instead of the fitting an unstructured matrix with the argument `us` or a reduced model with no covariance with the argument `diag`, we can also directly estimate the genetic correlation between the `bwt` and `tarsus` with `corgh`. Here we decide to estimate directly the additive genetic correlation.

```
modele <- asreml(
  fixed = cbind(bwt, tarsus) ~ trait + at(trait):sex,
  random = ~ corgh(trait):vm(animal, ainv) +
    us(trait, init = c(1, 0.1, 1)):byear +
    us(trait, init = c(1, 0.1, 1)):mother,
  residual = ~ id(units):us(trait, init = c(1, 0.1, 1)),
  data = gryphon,
  na.action = na.method(x = "include", y = "include"),
  maxit = 20
)
```

```
## Model fitted using the sigma parameterization.
```

```
## ASReML 4.1.0 Fri May 21 01:55:56 2021
```

```
##      LogLik      Sigma2      DF      wall      cpu
## 1      -3544.356      1.0  1533 01:55:56      0.0
## 2      -3177.908      1.0  1533 01:55:56      0.0 (1 restrained)
## 3      -2784.992      1.0  1533 01:55:56      0.0
## 4      -2527.352      1.0  1533 01:55:56      0.0
## 5      -2406.422      1.0  1533 01:55:56      0.0
## 6      -2382.062      1.0  1533 01:55:56      0.0
## 7      -2380.263      1.0  1533 01:55:56      0.0
## 8      -2380.246      1.0  1533 01:55:56      0.0
```

```
## 9      -2380.246          1.0    1533 01:55:56    0.0
```

```
modele <- update(modele)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:56 2021
##          LogLik      Sigma2      DF      wall      cpu
## 1      -2380.246          1.0    1533 01:55:57    0.1
## 2      -2380.246          1.0    1533 01:55:57    0.0
```

```
summary(modele)$varcomp
```

	component	std.error	z.ratio	bound
## trait:byear!trait_bwt:bwt	0.9746376	0.2825690	3.449203	P
## trait:byear!trait_tarsus:bwt	0.1624052	0.4184989	0.388066	P
## trait:byear!trait_tarsus:tarsus	3.7383599	1.2065684	3.098341	P
## trait:mother!trait_bwt:bwt	1.1445196	0.2302189	4.971441	P
## trait:mother!trait_tarsus:bwt	-1.5567326	0.4051847	-3.842032	P
## trait:mother!trait_tarsus:tarsus	4.8206100	1.3201239	3.651634	P
## trait:vm(animal, ainv)!trait!tarsus:!trait!bwt.cor	0.7353034	0.1094744	6.716668	U
## trait:vm(animal, ainv)!trait_bwt	1.9893544	0.4410242	4.510760	P
## trait:vm(animal, ainv)!trait_tarsus	10.2296667	2.8078190	3.643279	P
## units:trait!R	1.0000000	NA	NA	F
## units:trait!trait_bwt:bwt	1.8443104	0.3443173	5.356427	P
## units:trait!trait_tarsus:bwt	4.0142731	0.7412587	5.415482	P
## units:trait!trait_tarsus:tarsus	12.4844745	2.2893621	5.453255	P

It is important to note that using `corgh` change the order of the estimate (co)variance. All different calculation need to be adjust in consequence. It is also important to check the difference between the model with `us` and `corgh` to make sure any mistake are made.

```
summary(modelb)$loglik
```

```
## [1] -2380.246
```

```
summary(modele)$loglik
```

```
## [1] -2380.246
```

There two main advantages to use `corgh`: first, a direct estimation of correlation within the G matrix can avoid mistake in the `vpredict` calculation; second, it is possible to test if the correlation is significantly different than 0 (similar result as LRT with the covariance) but also to -1 and 1 which correspond of the correlation boundaries. The following code showed how to create a reduced model with the correlation close to 1 and compared to the initial model. Since we compared the correlation to its boundary, the degree of freedom is only half as a one tail LTR.

```

MODEL_MODIF <- update.asreml(modele, start.values = T)
G_MOD <- MODEL_MODIF$vpparameters.table[(1:9), ]
G_MOD[1, 2] <- 0.99999
G_MOD[1, 3] <- "F"
modele.red <- asreml(
  fixed = cbind(bwt, tarsus) ~ trait + at(trait):sex,
  random = ~ corgh(trait):vm(animal, ainv) +
    us(trait, init = c(1, 0.1, 1)):byear +
    us(trait, init = c(1, 0.1, 1)):mother,
  residual = ~ id(units):us(trait, init = c(1, 0.1, 1)),
  data = gryphon,
  na.action = na.method(x = "include", y = "include"),
  maxit = 20,
  G.param = G_MOD
)

```

```

## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:57 2021
##           LogLik      Sigma2      DF      wall      cpu
##  1      -2545.227         1.0    1533 01:55:57    0.0
##  2      -2483.880         1.0    1533 01:55:57    0.0
##  3      -2423.502         1.0    1533 01:55:57    0.0
##  4      -2392.508         1.0    1533 01:55:57    0.0
##  5      -2383.661         1.0    1533 01:55:57    0.0
##  6      -2383.084         1.0    1533 01:55:57    0.0
##  7      -2383.033         1.0    1533 01:55:57    0.0
##  8      -2383.022         1.0    1533 01:55:57    0.0
##  9      -2383.019         1.0    1533 01:55:57    0.0
## 10      -2383.019         1.0    1533 01:55:57    0.0

```

```

2 * (modele$loglik - modele.red$loglik)

```

```

## [1] 5.544679

```

```

1 - pchisq(2 * (modele$loglik - modele.red$loglik), df = 0.5)

```

```

## [1] 0.006598675

```

### 3.2.5 Partitionning (co)variance between groups

Similar to the univariate model, it is possible to partition the variance and also the covariance between different groups within the dataset. Here, we can estimate sex-specific genetic correlation. Note, to partition a correlation, it is require to have important sample size within each group. For this example, we simplify the model !



```

gryphon <- gryphon[order(gryphon$sex), ]
model_sex <- asreml(
  fixed = cbind(bwt, tarsus) ~ trait + at(trait):sex,
  random = ~ at(sex):us(trait):vm(animal, ainv) +
    us(trait, init = c(1, 0.1, 1)):byear +
    us(trait, init = c(1, 0.1, 1)):mother,
  residual = ~ dsum(~ id(units):us(trait) | sex),
  data = gryphon,
  na.action = na.method(x = "include", y = "include"),
  maxit = 20
)

```

```

## Multi-section model using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:57 2021
##           LogLik      Sigma2      DF      wall      cpu
##  1      -2495.853         1.0    1807 01:55:57    0.1 (1 restrained)
##  2      -2444.497         1.0    1807 01:55:58    0.1
##  3      -2401.367         1.0    1807 01:55:58    0.1
##  4      -2390.943         1.0    1807 01:55:58    0.1
##  5      -2388.819         1.0    1807 01:55:58    0.1
##  6      -2388.738         1.0    1807 01:55:58    0.1
##  7      -2388.736         1.0    1807 01:55:58    0.1

```

```

model_sex <- update(model_sex)

```

```

## Multi-section model using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:58 2021
##           LogLik      Sigma2      DF      wall      cpu
##  1      -2388.736         1.0    1807 01:55:58    0.1
##  2      -2388.736         1.0    1807 01:55:58    0.1

```

```

summary(model_sex)$varcomp

```

```

##                                     component std.error   z.ratio b
## trait:byear!trait_bwt:bwt          0.9858502 0.2863930  3.4422982
## trait:byear!trait_tarsus:bwt       0.1525073 0.4334219  0.3518681
## trait:byear!trait_tarsus:tarsus    3.9981750 1.2798231  3.1240060
## trait:mother!trait_bwt:bwt         1.3312802 0.2484496  5.3583516
## trait:mother!trait_tarsus:bwt      -1.6173911 0.4283985 -3.7754356
## trait:mother!trait_tarsus:tarsus    4.7543015 1.3546795  3.5095396
## at(sex, 1):trait:vm(animal, ainv)!trait_bwt:bwt 1.3402726 0.5670807  2.3634603
## at(sex, 1):trait:vm(animal, ainv)!trait_tarsus:bwt 2.3607838 1.1348534  2.0802544
## at(sex, 1):trait:vm(animal, ainv)!trait_tarsus:tarsus 6.0624925 3.1304679  1.9366091
## at(sex, 2):trait:vm(animal, ainv)!trait_bwt:bwt 1.8645331 0.8887843  2.0978466
## at(sex, 2):trait:vm(animal, ainv)!trait_tarsus:bwt 5.0952433 2.0683128  2.4634781

```

```
## at(sex, 2):trait:vm(animal, ainv)!trait_tarsus:tarsus 14.9762227 6.4472652 2.3228799
## sex_1!R 1.0000000 NA NA
## sex_1!trait_bwt:bwt 2.3079924 0.5015700 4.6015360
## sex_1!trait_tarsus:bwt 4.4288323 1.0376540 4.2681204
## sex_1!trait_tarsus:tarsus 13.4858721 2.9285483 4.6049683
## sex_2!R 1.0000000 NA NA
## sex_2!trait_bwt:bwt 1.7957128 0.7549623 2.3785463
## sex_2!trait_tarsus:bwt 2.6342177 1.7685005 1.4895205
## sex_2!trait_tarsus:tarsus 9.6101885 5.4914087 1.7500407
```

we can estimate the different correlations using `vpredict`:

```
vpredict(model_sex, r_byear ~ V2 / sqrt(V1 * V3))
```

```
##           Estimate      SE
## r_byear 0.07681647 0.213139
```

```
vpredict(model_sex, r_M ~ V5 / sqrt(V4 * V6))
```

```
##           Estimate      SE
## r_M -0.6428904 0.2489498
```

```
vpredict(model_sex, r_A.1 ~ V8 / sqrt(V7 * V9))
```

```
##           Estimate      SE
## r_A.1 0.8281977 0.1723661
```

```
vpredict(model_sex, r_A.2 ~ V11 / sqrt(V10 * V12))
```

```
##           Estimate      SE
## r_A.2 0.9642258 0.1241699
```

```
vpredict(model_sex, r_res.1 ~ V15 / sqrt(V14 * V16))
```

```
##           Estimate      SE
## r_res.1 0.7938392 0.0789263
```

```
vpredict(model_sex, r_res.2 ~ V19 / sqrt(V18 * V20))
```

```
##           Estimate      SE
## r_res.2 0.6341139 0.1894661
```

and the heritability too:

```
vpredict(model_sex, h2.bwt.1 ~ V7 / (V1 + V4 + V7 + V14))
```

```
##           Estimate      SE
## h2.bwt.1 0.2246746 0.09176899
```

```
vpredict(model_sex, h2.bwt.2 ~ V10 / (V1 + V4 + V10 + V18))
```

```
##           Estimate      SE
## h2.bwt.2 0.3119317 0.1442501
```

```
vpredict(model_sex, h2.tarsus.1 ~ V9 / (V3 + V6 + V9 + V16))
```

```
##           Estimate      SE
## h2.tarsus.1 0.214216 0.1070477
```

```
vpredict(model_sex, h2.tarsus.2 ~ V12 / (V3 + V6 + V12 + V20))
```

```
##           Estimate      SE
## h2.tarsus.2 0.4492118 0.1833703
```

Now we can look at the fixed effects parameters and assess their significance with a conditional Wald F-test:

```
summary(model_sex, coef = TRUE)$coef.fi
wald.asreml(model_sex, denDF = "default", ssType = "conditional")$Wald
```

```
##           solution std error    z.ratio
## at(trait, tarsus):sex_1  0.00000000      NA      NA
## at(trait, tarsus):sex_2 -0.05549717 0.4758546 -0.1166263
## at(trait, bwt):sex_1    0.00000000      NA      NA
## at(trait, bwt):sex_2    1.93936200 0.1903213 10.1899393
## trait_bwt              6.37791872 0.2311775 27.5888427
## trait_tarsus           20.58389296 0.4942600 41.6458836

## Multi-section model using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:55:59 2021
##           LogLik      Sigma2      DF      wall      cpu
## 1      -2388.736        1.0    1807 01:55:59    0.1
## 2      -2388.736        1.0    1807 01:55:59    0.1
## Calculating denominator DF

##
##           Df denDF    F.inc    F.con Margin      Pr
## trait           2  44.8 1522.00 1522.00      0.00000
## at(trait, bwt):sex 1 137.5  220.90  103.80      B 0.00000
```

```
## at(trait, tarsus):sex 1 138.6 0.01 0.01 B 0.90737
```

### 3.3 gremlin

Might not available yet

Meanwhile



Figure 3.1: Keep it dry and do no feed after midnight.

### 3.4 MCMCglmm

All others software will remove automatically lines with missing data from the dataset use to fit the model. `MCMCglmm`, however, will not do it and will try to fit the model use latent variables for missing data. For comparison, we will remove the missing values from the data before fitting the model.

```
gryphon2 <- subset(gryphon, !is.na(bwt) & !is.na(tarsus))
```

First load `MCMCglmm`:

```
library(MCMCglmm)
Ainv <- inverseA(gryphonped)$Ainv
```

### 3.4.1 Fitting the model

Fitting a multivariate model in MCMCglmm involves several new consideration above those for fitting univariate models. First, we have to fit multivariate priors; second, we have to specify the ways in which effects on different traits may covary, including the nature of residual (co)variation; and third, we will have to be a little more specific when specifying to MCMCglmm what type of distributions from which we assume our data are drawn. Our most basic model can be specified as:

```
prior2.1 <- list(
  G = list(G1 = list(V = diag(2), nu = 1.002)),
  R = list(V = diag(2), nu = 1.002)
)

model2.1 <- MCMCglmm(cbind(bwt, tarsus) ~ trait - 1,
  random = ~ us(trait):animal,
  rcov = ~ us(trait):units,
  family = c("gaussian", "gaussian"),
  ginv = list(animal = Ainv),
  data = gryphon, prior = prior2.1, verbose = FALSE
)
summary(model2.1)
```

```
##
## Iterations = 3001:12991
## Thinning interval = 10
## Sample size = 1000
##
## DIC: 7914.014
##
## G-structure: ~us(trait):animal
##
##               post.mean l-95% CI u-95% CI eff.samp
## traitbwt:traitbwt.animal      3.459   2.3201   4.892   150.32
## traittarsus:traitbwt.animal    2.649   0.4966   5.014    82.60
## traitbwt:traittarsus.animal    2.649   0.4966   5.014    82.60
## traittarsus:traittarsus.animal 12.015   5.4475  18.350    49.79
##
## R-structure: ~us(trait):units
##
##               post.mean l-95% CI u-95% CI eff.samp
## traitbwt:traitbwt.units        3.829    2.791    4.850   169.08
## traittarsus:traitbwt.units      3.211    1.269    5.185   113.39
## traitbwt:traittarsus.units      3.211    1.269    5.185   113.39
## traittarsus:traittarsus.units  18.133   12.153   23.967    64.55
##
## Location effects: cbind(bwt, tarsus) ~ trait - 1
```

```
##
##               post.mean l-95% CI u-95% CI eff.samp  pMCMC
## traitbwt       7.595    7.291    7.874    839.2 <0.001 ***
## traittarsus    20.543   19.916   21.126    702.3 <0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(model2.1$VCV[, "traittarsus:traittarsus.animal"])
```

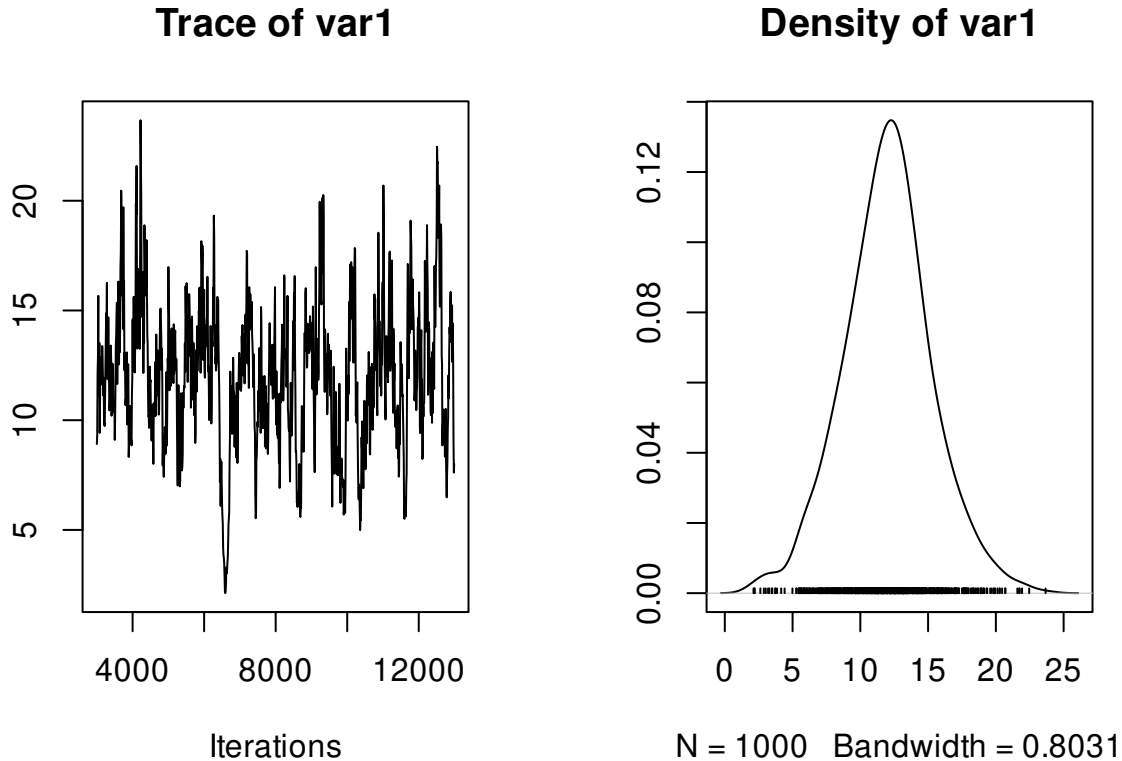


Figure 3.2: The posterior distribution of the additive genetic effect for tarsus length in a MCMCglmm run with default values

```
autocorr.diag(model2.1$VCV)[, "traittarsus:traittarsus.animal"][2]
```

```
##    Lag 10
## 0.8466367
```

We have constructed the prior similarly to the those in the univariate models in tutorial 1, only we are specifying a 2x2 covariance matrix rather than a single variance. In order to provide proper priors, we have set the degree of belief parameter to greater than 1 (1.002). Those priors are not necessarily weak or uninformative in all circumstances. We will consider them adequate nonetheless for this tutorial. Please the vignette of the MCMCglmm packages ([Hadfield, 2021](#)) for more information on priors. In tutorial 1, we used full autocorrelation tables to evaluate the validity of the posterior distribution. Note that we have not done this here. For a bivariate model

this table can become very complex. Nonetheless, it is worth evaluating, rather it is simply too large to include here. It can be viewed in the console as before. Here we have displayed only the autocorrelation for estimates of additive genetic effects for tarsus length with a lag of one samples (10 iterations given this MCMCglmm run with default values). This lag of 0.8466367 is clearly unacceptable. The posterior distribution of the additive genetic effect on tarsus length is shown in Figure 4 (p. 15), note the autocorrelation evident in the left-hand plot.

We will opt to run the analysis for longer. This longer run could be run using the following code (including a line to save the output):

```
model2.1 <- MCMCglmm(cbind(bwt, tarsus) ~ trait - 1,
  random = ~ us(trait):animal,
  rcov = ~ us(trait):units,
  family = c("gaussian", "gaussian"),
  ginv = list(animal = Ainv),
  data = gryphon,
  nitt = 130000, thin = 100, burnin = 30000,
  prior = prior2.1, verbose = FALSE
)
save(model2.1, file = "data/MCMCglmm_model2_1_LongRun.rda")
```

However, this run might take as long as an hour. For the purpose of this tutorial we have provided an output for such a run. It can be obtained and manipulated as follows, assuming that the file MCMCglmm\_model2\_1\_LongRun.rda is available at the specified location:

```
load(file = "data/MCMCglmm_model2_1_LongRun.rda")
autocorr.diag(model2.1$VCV)[, "traittarsus:traittarsus.animal"][2]
```

```
## Lag 100
## 0.2608752
```

This level of autocorrelation is more acceptable, at least for the purpose of demonstration in this tutorial. We can recover variance components, heritabilities, and genetic correlations from the posterior distribution of this model:

```
posterior.mode(model2.1$VCV)
```

```
##      traitbwt:traitbwt.animal      traittarsus:traitbwt.animal      traitbwt:traittarsus.a
##                                3.370616                        2.581839                        2.5
##      traitbwt:traittarsus.units  traittarsus:traittarsus.units
##                                2.982413                        19.556443
```

```
heritability.bwt2.1 <- model2.1$VCV[, "traitbwt:traitbwt.animal"] / (model2.1$VCV[, "traitbwt:traitbwt.animal"] + model2.1$VCV[, "traittarsus:traittarsus.animal"])
posterior.mode(heritability.bwt2.1)
```

```
##      var1
```

```
## 0.4999336
```

```
heritability.tarsus2.1 <- model2.1$VCV[, "traittarsus:traittarsus.animal"] / (model2.1$VCV[
posterior.mode(heritability.tarsus2.1)
```

```
##      var1
## 0.4038754
```

```
genetic.correlation2.1 <- model2.1$VCV[, "traitbwt:traittarsus.animal"] / sqrt(model2.1$VCV[
posterior.mode(genetic.correlation2.1)
```

```
##      var1
## 0.3691503
```

### 3.4.2 Adding fixed and random effects

Fixed and random effects can be added just as for the univariate case. Given that our full model of bwt from tutorial 1 had sex as a fixed effect as well as random effects of byear and mother, we could specify a bivariate formulation of this using the following code (including a line to save the output):

```
prior2.2 <- list(
  G = list(
    G1 = list(V = diag(2), nu = 1.002),
    G2 = list(V = diag(2), nu = 1.002),
    G3 = list(V = diag(2), nu = 1.002)
  ),
  R = list(V = diag(2), nu = 1.002)
)
model2.2 <- MCMCglmm(cbind(bwt, tarsus) ~ trait - 1 + trait:sex,
  random = ~ us(trait):animal + us(trait):byear + us(trait):mother,
  rcov = ~ us(trait):units,
  family = c("gaussian", "gaussian"),
  ginv = list(animal = Ainv), data = gryphon,
  nitt = 130000, thin = 100, burnin = 30000,
  prior = prior2.2, verbose = FALSE
)
save(model2.2, file = "data/MCMCglmm_model2_2_LongRun.rda")
```

Again we have provided the data from one such run. It can be accessed using the code:

```
load(file = "data/MCMCglmm_model2_2_LongRun.rda")
summary(model2.2)
```

```
##
```



```

## Iterations = 30001:529501
## Thinning interval = 500
## Sample size = 1000
##
## DIC: 6222.288
##
## G-structure: ~us(trait):animal
##
##               post.mean 1-95% CI u-95% CI eff.samp
## traitbwt:traitbwt.animal    1.237  0.010802    2.183    71.22
## traittarsus:traitbwt.animal  2.138 -0.067106    4.268    49.85
## traitbwt:traittarsus.animal  2.138 -0.067106    4.268    49.85
## traittarsus:traittarsus.animal 5.265  0.005733   11.751    49.24
##
##               ~us(trait):byear
##
##               post.mean 1-95% CI u-95% CI eff.samp
## traitbwt:traitbwt.byear      0.85157  0.3924    1.3839    1000
## traittarsus:traitbwt.byear   -0.01322 -0.7624    0.8352    1000
## traitbwt:traittarsus.byear   -0.01322 -0.7624    0.8352    1000
## traittarsus:traittarsus.byear 3.30411  1.2574    5.6918    1000
##
##               ~us(trait):mother
##
##               post.mean 1-95% CI u-95% CI eff.samp
## traitbwt:traitbwt.mother      1.213  0.7414    1.688    164.7
## traittarsus:traitbwt.mother   -1.969 -2.3860   -1.533    422.4
## traitbwt:traittarsus.mother   -1.969 -2.3860   -1.533    422.4
## traittarsus:traittarsus.mother 3.484  1.7016    5.451    215.8
##
## R-structure: ~us(trait):units
##
##               post.mean 1-95% CI u-95% CI eff.samp
## traitbwt:traitbwt.units       2.531  1.618    3.485    133.27
## traittarsus:traitbwt.units     5.360  3.508    7.703    76.07
## traitbwt:traittarsus.units     5.360  3.508    7.703    76.07
## traittarsus:traittarsus.units 17.744 11.386   23.397    61.97
##
## Location effects: cbind(bwt, tarsus) ~ trait - 1 + trait:sex
##
##               post.mean 1-95% CI u-95% CI eff.samp pMCMC
## traitbwt      6.2643  5.8046  6.7063    1027 <0.001 ***
## traittarsus   20.3901 19.4377 21.2534    1000 <0.001 ***
## traitbwt:sex2  2.0315  1.7096  2.3452    1000 <0.001 ***
## traittarsus:sex2 0.1141 -0.6095  0.8959    1000  0.746
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```
autocorr(model2.2$VCV)[, , "traittarsus:traittarsus.animal"][3, 4]
```

```
## [1] 0.5231926
```

We can evaluate the fixed effect, their Ci evaluate their significance.

```
posterior.mode(model2.2$Sol)
```

```
##          traitbwt      traittarsus      traitbwt:sex2      traittarsus:sex2
##          6.37708530      20.33434582          1.97881662          0.00709564
```

```
HPDinterval(model2.2$Sol, 0.95)
```

```
##          lower      upper
## traitbwt      5.8045704  6.706339
## traittarsus    19.4377228 21.253397
## traitbwt:sex2   1.7095714  2.345200
## traittarsus:sex2 -0.6094733 0.895885
## attr(,"Probability")
## [1] 0.95
```

```
plot(model2.2$Sol)
```

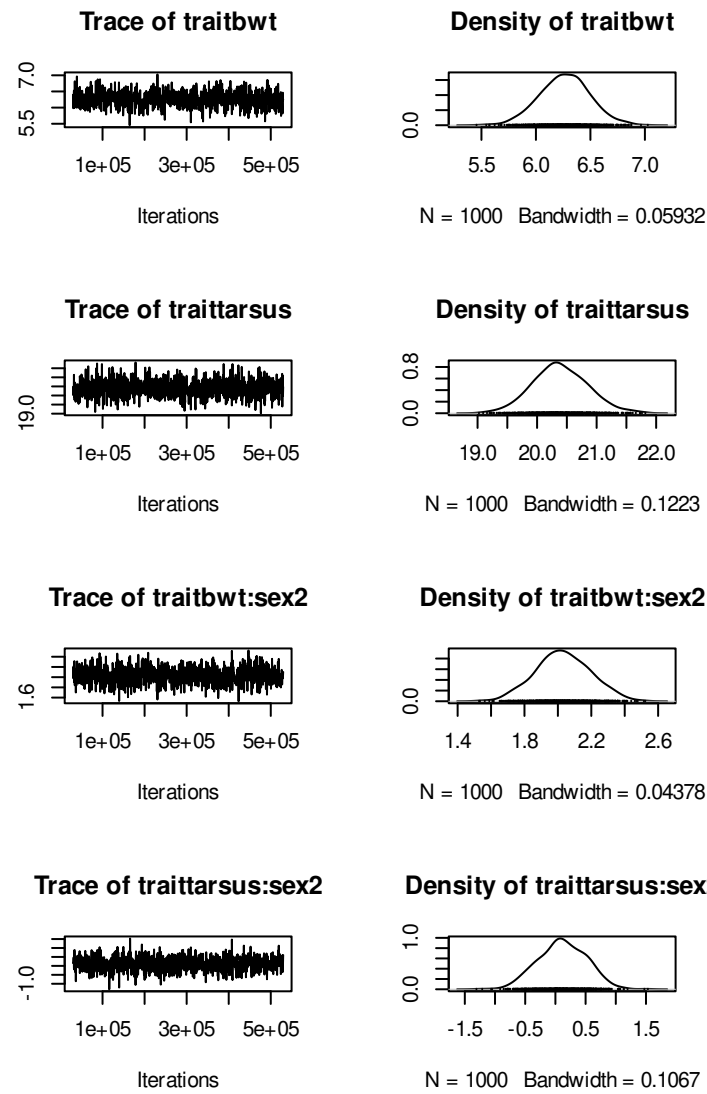


Figure 3.3: Posterior trace and distribution for the fixed effects in model 2.2

As before we can obtain the raw variance component estimates and genetic correlations for the random effects:

```
posterior.mode(model2.2$VCV)
```

```
##      traitbwt:traitbwt.animal      traittarsus:traitbwt.animal      traitbwt:traittarsus.a
##              1.32761860              1.98076467              1.980
##      traitbwt:traittarsus.byear      traittarsus:traittarsus.byear      traitbwt:traitbwt.m
##              -0.14691430              2.79242863              1.292
##      traitbwt:traitbwt.units      traittarsus:traitbwt.units      traitbwt:traittarsus.
##              2.53632072              5.22764927              5.227
```

```
genetic.correlation2.2 <- model2.2$VCV[, "traitbwt:traittarsus.animal"] / sqrt(model2.2$VCV[
maternal.correlation2.2 <- model2.2$VCV[, "traitbwt:traittarsus.mother"] / sqrt(model2.2$VCV
```

```
posterior.mode(genetic.correlation2.2)
```

```
##      var1
## 0.9942065
```

```
posterior.mode(maternal.correlation2.2)
```

```
##      var1
## -0.9920879
```

Evaluation of the statistical support for these genetic and maternal correlations is straightforward. Because we imposed no constraint on their estimation, we can evaluate the extent to which the posterior distributions overlap zero:

```
HPDinterval(genetic.correlation2.2, 0.95)
```

```
##      lower      upper
## var1 0.3935369 0.9990187
## attr("Probability")
## [1] 0.95
```

```
HPDinterval(maternal.correlation2.2, 0.95)
```

```
##      lower      upper
## var1 -0.9980476 -0.9443838
## attr("Probability")
## [1] 0.95
```

Neither of these posterior distributions overlaps zero, so we can consider them both statistically supported.

### 3.4.3 Partitioning (co)variances

As in the tutorial 1, it is possible to partition the variance-covariance matrix between groups (here sex)

```
prior2.3 <- list(
  G = list(
    G1 = list(V = diag(2), nu = 1.002),
    G2 = list(V = diag(2), nu = 1.002),
    G3 = list(V = diag(2), nu = 1.002),
    G4 = list(V = diag(2), nu = 1.002)
  ),
  R = list(
```

```

    V1 = list(V = diag(2), nu = 1.002),
    V2 = list(V = diag(2), nu = 1.002)
  )
)

model2.3 <- MCMCglmm(cbind(bwt, tarsus) ~ trait - 1 + trait:sex,
  random = ~ us(at.level(sex, "1"):trait):animal + us(at.level(sex, "2"):trait):animal + u
  rcov = ~ us(at.level(sex, "1"):trait):units + us(at.level(sex, "2"):trait):units,
  family = c("gaussian", "gaussian"),
  ginv = list(animal = Ainv), data = gryphon,
  nitt = 130000, thin = 100, burnin = 30000,
  prior = prior2.3, verbose = FALSE
)
save(model2.3, file = "data/MCMCglmm_model2_3_LongRun.rda")

```

Again we have provided the data from one such run. It can be accessed using the code:

```

load(file = "data/MCMCglmm_model2_3_LongRun.rda")
summary(model2.3)

##
## Iterations = 30001:129901
## Thinning interval = 100
## Sample size = 1000
##
## DIC: 5641.752
##
## G-structure: ~us(at.level(sex, "1"):trait):animal
##
##
##                                     post.mean 1-95% CI
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal      1.290   0.2723
## at.level(sex, "1"):traittarsus:at.level(sex, "1"):traitbwt.animal    1.695  -0.3272
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal    1.695  -0.3272
## at.level(sex, "1"):traittarsus:at.level(sex, "1"):traittarsus.animal  4.764   0.2145
##
##               ~us(at.level(sex, "2"):trait):animal
##
##
##                                     post.mean 1-95% CI
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal      2.631   1.054
## at.level(sex, "2"):traittarsus:at.level(sex, "2"):traitbwt.animal    6.797   2.275
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal    6.797   2.275
## at.level(sex, "2"):traittarsus:at.level(sex, "2"):traittarsus.animal 21.582  9.301
##
##               ~us(trait):byear
##

```

```

##                                post.mean l-95% CI u-95% CI eff.samp
## traitbwt:traitbwt.byear         1.0808    0.533    1.704    1000.0
## traittarsus:traitbwt.byear      0.1822   -0.678    1.181    692.8
## traitbwt:traittarsus.byear      0.1822   -0.678    1.181    692.8
## traittarsus:traittarsus.byear   4.3624    1.916    7.713    1256.3
##
##                                ~us(trait):mother
##
##                                post.mean l-95% CI u-95% CI eff.samp
## traitbwt:traitbwt.mother         1.351    0.9377    1.8114    737.8
## traittarsus:traitbwt.mother      -1.583   -2.1887   -0.8234    359.6
## traitbwt:traittarsus.mother      -1.583   -2.1887   -0.8234    359.6
## traittarsus:traittarsus.mother    4.488    2.4913    6.9268    526.6
##
## R-structure: ~us(at.level(sex, "1"):trait):units
##
##                                post.mean l-95% CI u-95% CI
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.units      2.425    1.499
## at.level(sex, "1"):traittarsus:at.level(sex, "1"):traitbwt.units    5.105    3.200
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units    5.105    3.200
## at.level(sex, "1"):traittarsus:at.level(sex, "1"):traittarsus.units 15.083    8.819
##
##                                ~us(at.level(sex, "2"):trait):units
##
##                                post.mean l-95% CI u-95% CI
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units      1.174    0.1719
## at.level(sex, "2"):traittarsus:at.level(sex, "2"):traitbwt.units    1.202   -0.7302
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units    1.202   -0.7302
## at.level(sex, "2"):traittarsus:at.level(sex, "2"):traittarsus.units 4.300    0.1394
##
## Location effects: cbind(bwt, tarsus) ~ trait - 1 + trait:sex
##
##                                post.mean l-95% CI u-95% CI eff.samp pMCMC
## traitbwt          6.37718  5.88327  6.81113    1000 <0.001 ***
## traittarsus       20.57067 19.54250 21.46986    1000 <0.001 ***
## traitbwt:sex2      1.95075  1.56879  2.34728    1000 <0.001 ***
## traittarsus:sex2  -0.04345 -0.93488  0.97908    1000  0.904
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

autocorr(model2.3$VCV)

## , , at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal
##
##                                at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "
## Lag 0                                1.00000000

```

```

## Lag 100 0.55397334
## Lag 500 0.17022926
## Lag 1000 0.06337109
## Lag 5000 -0.02509190
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0 0.78900626
## Lag 100 0.48327050
## Lag 500 0.19390026
## Lag 1000 0.06793138
## Lag 5000 -0.05286532
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):
## Lag 0 0.069605265
## Lag 100 0.094995154
## Lag 500 0.058573181
## Lag 1000 -0.003320081
## Lag 5000 -0.037248927
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0 0.07767707
## Lag 100 0.08733352
## Lag 500 0.03149552
## Lag 1000 0.01343395
## Lag 5000 -0.04572885
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.020803022 -0.020803022 -0.062421
## Lag 100 -0.020819486 -0.020819486 -0.016131
## Lag 500 -0.003372663 -0.003372663 0.000501
## Lag 1000 -0.049049620 -0.049049620 0.006171
## Lag 5000 -0.003451790 -0.003451790 -0.067031
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 -0.07667827
## Lag 100 -0.11462894
## Lag 500 -0.04945437
## Lag 1000 0.01279152
## Lag 5000 0.01063678
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 -0.65709502
## Lag 100 -0.42587753
## Lag 500 -0.17449911
## Lag 1000 -0.01887227
## Lag 5000 0.05356917
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0 -0.04704752
## Lag 100 -0.07487166
## Lag 500 -0.03174021
## Lag 1000 0.01461737
## Lag 5000 0.03438299
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):

```

```

## Lag 0 -0.05664210
## Lag 100 -0.06795349
## Lag 500 -0.01045678
## Lag 1000 -0.00544654
## Lag 5000 0.04865394
##
## , , at.level(sex, "1"):traittarsus:at.level(sex, "1"):traitbwt.animal
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):traittarsus:at.level(sex, "1"):traitbwt.animal
## Lag 0 0.78900626
## Lag 100 0.48940534
## Lag 500 0.20162861
## Lag 1000 0.08548021
## Lag 5000 -0.03210448
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):traittarsus:at.level(sex, "1"):traittarsus.animal
## Lag 0 1.00000000
## Lag 100 0.67035954
## Lag 500 0.29214704
## Lag 1000 0.09634396
## Lag 5000 -0.04654686
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):traittarsus:at.level(sex, "2"):traitbwt.animal
## Lag 0 0.03891144
## Lag 100 0.07541352
## Lag 500 0.08719705
## Lag 1000 0.02631996
## Lag 5000 0.03707974
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):traittarsus:at.level(sex, "2"):traittarsus.animal
## Lag 0 0.04885100
## Lag 100 0.06901723
## Lag 500 0.05108866
## Lag 1000 0.03771859
## Lag 5000 0.04393141
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.03155374 -0.03155374 -0.03851144
## Lag 100 -0.01235856 -0.01235856 -0.01541144
## Lag 500 -0.03654693 -0.03654693 0.00101144
## Lag 1000 -0.06974536 -0.06974536 -0.03881144
## Lag 5000 0.02136226 0.02136226 -0.04861144
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.mother
## Lag 0 -0.11748729
## Lag 100 -0.13156978
## Lag 500 -0.10579913
## Lag 1000 -0.02884682
## Lag 5000 -0.02931723
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):traittarsus:at.level(sex, "1"):traittarsus.units
## Lag 0 -0.81241841
## Lag 100 -0.58511641

```



```

## Lag 500 -0.23385312
## Lag 1000 -0.04261848
## Lag 5000 0.05109275
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2")
## Lag 0 -0.006983857
## Lag 100 -0.045096027
## Lag 500 -0.058550089
## Lag 1000 0.011350413
## Lag 5000 -0.044368431
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2")
## Lag 0 -0.008328493
## Lag 100 -0.030043649
## Lag 500 -0.021340607
## Lag 1000 -0.014582657
## Lag 5000 -0.042776639
##
## , , at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1")
## Lag 0 0.78900626
## Lag 100 0.48940534
## Lag 500 0.20162861
## Lag 1000 0.08548021
## Lag 5000 -0.03210448
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1")
## Lag 0 1.00000000
## Lag 100 0.67035954
## Lag 500 0.29214704
## Lag 1000 0.09634396
## Lag 5000 -0.04654686
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2")
## Lag 0 0.03891144
## Lag 100 0.07541352
## Lag 500 0.08719705
## Lag 1000 0.02631996
## Lag 5000 0.03707974
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2")
## Lag 0 0.04885100
## Lag 100 0.06901723
## Lag 500 0.05108866
## Lag 1000 0.03771859
## Lag 5000 0.04393141
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.03155374 -0.03155374 -0.0385
## Lag 100 -0.01235856 -0.01235856 -0.0154
## Lag 500 -0.03654693 -0.03654693 0.0010
## Lag 1000 -0.06974536 -0.06974536 -0.0388

```

```

## Lag 5000          0.02136226          0.02136226          -0.0486
##      traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0             -0.11748729
## Lag 100            -0.13156978
## Lag 500            -0.10579913
## Lag 1000           -0.02884682
## Lag 5000           -0.02931723
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex,
## Lag 0              -0.81241841
## Lag 100            -0.58511641
## Lag 500            -0.23385312
## Lag 1000           -0.04261848
## Lag 5000           0.05109275
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2
## Lag 0              -0.006983857
## Lag 100            -0.045096027
## Lag 500            -0.058550089
## Lag 1000           0.011350413
## Lag 5000           -0.044368431
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex,
## Lag 0              -0.008328493
## Lag 100            -0.030043649
## Lag 500            -0.021340607
## Lag 1000           -0.014582657
## Lag 5000           -0.042776639
##
## , , at.level(sex, "1"):traittarsus:at.level(sex, "1"):traittarsus.animal
##
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "
## Lag 0              0.50907405
## Lag 100            0.33371522
## Lag 500            0.16786851
## Lag 1000           0.09508980
## Lag 5000           -0.01160465
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex
## Lag 0              0.85973347
## Lag 100            0.60647833
## Lag 500            0.30364438
## Lag 1000           0.11809885
## Lag 5000           -0.01621713
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "
## Lag 0              0.04172858
## Lag 100            0.08374547
## Lag 500            0.12694344
## Lag 1000           0.07590539
## Lag 5000           0.06760669
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex

```

```

## Lag 0 0.04446314
## Lag 100 0.07584401
## Lag 500 0.09237150
## Lag 1000 0.08060798
## Lag 5000 0.08871292
##      traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.02521301 -0.02521301 -0.01771301
## Lag 100 -0.01064701 -0.01064701 -0.01234701
## Lag 500 -0.03224542 -0.03224542 -0.00764542
## Lag 1000 -0.07434829 -0.07434829 -0.04814829
## Lag 5000 0.04361589 0.04361589 -0.03411589
##      traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 -0.16381629
## Lag 100 -0.14936492
## Lag 500 -0.14524964
## Lag 1000 -0.05284469
## Lag 5000 -0.03496960
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 -0.68658404
## Lag 100 -0.52807612
## Lag 500 -0.22374206
## Lag 1000 -0.06443430
## Lag 5000 0.01601359
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0 -0.01115803
## Lag 100 -0.05709045
## Lag 500 -0.10102388
## Lag 1000 -0.03082573
## Lag 5000 -0.07768457
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 -0.00224780
## Lag 100 -0.03128327
## Lag 500 -0.06483536
## Lag 1000 -0.05641989
## Lag 5000 -0.08909507
##
## , , at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal
##
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0 0.069605265
## Lag 100 0.072256284
## Lag 500 0.053797194
## Lag 1000 -0.000442817
## Lag 5000 0.004334379
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0 0.038911436
## Lag 100 0.046490324

```

```

## Lag 500 0.023982452
## Lag 1000 -0.022082138
## Lag 5000 0.002024369
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):traitbwt.animal
## Lag 0 1.00000000
## Lag 100 0.72687483
## Lag 500 0.29712402
## Lag 1000 0.09345869
## Lag 5000 -0.02661430
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):traittarsus.animal
## Lag 0 0.88935579
## Lag 100 0.68152951
## Lag 500 0.34189066
## Lag 1000 0.14877079
## Lag 5000 -0.05267066
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 0.042945762 0.042945762 0.06511111
## Lag 100 0.034889169 0.034889169 0.00751111
## Lag 500 0.012415428 0.012415428 0.02051111
## Lag 1000 -0.002266020 -0.002266020 0.01901111
## Lag 5000 0.009395412 0.009395412 -0.00681111
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.mother
## Lag 0 -0.038671695
## Lag 100 -0.032842556
## Lag 500 -0.038030546
## Lag 1000 -0.014755552
## Lag 5000 0.009608549
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):traittarsus.units
## Lag 0 -0.032836717
## Lag 100 -0.017837859
## Lag 500 -0.003168012
## Lag 1000 0.029353832
## Lag 5000 -0.033006249
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):traitbwt.units
## Lag 0 -0.91555116
## Lag 100 -0.74121973
## Lag 500 -0.31676022
## Lag 1000 -0.12984095
## Lag 5000 0.03999793
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):traittarsus.units
## Lag 0 -0.82737809
## Lag 100 -0.70500699
## Lag 500 -0.35753289
## Lag 1000 -0.19747432
## Lag 5000 0.07246841
##
## , , at.level(sex, "2"):traittarsus:at.level(sex, "2"):traitbwt.animal

```

```

##
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):traitbwt.animal
## Lag 0                                          0.077677066
## Lag 100                                       0.089976565
## Lag 500                                       0.094802795
## Lag 1000                                      0.054458884
## Lag 5000                                      -0.005097372
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):traittarsus.animal
## Lag 0                                          0.048851002
## Lag 100                                       0.056231551
## Lag 500                                       0.039570907
## Lag 1000                                      0.016174999
## Lag 5000                                      -0.007734496
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):traitbwt.animal
## Lag 0                                          0.88935579
## Lag 100                                       0.67781643
## Lag 500                                       0.28648920
## Lag 1000                                      0.09666293
## Lag 5000                                      -0.01204019
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):traittarsus.animal
## Lag 0                                          1.00000000
## Lag 100                                       0.77771494
## Lag 500                                       0.39167762
## Lag 1000                                      0.17883082
## Lag 5000                                      -0.07775854
##          traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0          0.029328800          0.029328800          0.047734496
## Lag 100         0.038845063          0.038845063          0.015113526
## Lag 500         0.018874626          0.018874626          0.029328800
## Lag 1000        -0.012224002         -0.012224002          0.013526416
## Lag 5000        -0.003721765         -0.003721765         -0.014018800
##          traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.mother
## Lag 0          -0.12765635
## Lag 100         -0.09362701
## Lag 500         -0.07507670
## Lag 1000        -0.03080525
## Lag 5000         0.02983728
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):traittarsus.units
## Lag 0          -0.030749722
## Lag 100         -0.016217311
## Lag 500         -0.020696084
## Lag 1000         0.004301752
## Lag 5000        -0.032243808
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):traitbwt.units
## Lag 0          -0.82656679
## Lag 100         -0.69378831
## Lag 500         -0.30703065

```

```

## Lag 1000 -0.12287954
## Lag 5000 0.02286923
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 -0.92799199
## Lag 100 -0.79777083
## Lag 500 -0.40858358
## Lag 1000 -0.21574629
## Lag 5000 0.09114505
##
## , , at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal
##
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0 0.077677066
## Lag 100 0.089976565
## Lag 500 0.094802795
## Lag 1000 0.054458884
## Lag 5000 -0.005097372
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0 0.048851002
## Lag 100 0.056231551
## Lag 500 0.039570907
## Lag 1000 0.016174999
## Lag 5000 -0.007734496
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):
## Lag 0 0.88935579
## Lag 100 0.67781643
## Lag 500 0.28648920
## Lag 1000 0.09666293
## Lag 5000 -0.01204019
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0 1.00000000
## Lag 100 0.77771494
## Lag 500 0.39167762
## Lag 1000 0.17883082
## Lag 5000 -0.07775854
##          traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 0.029328800 0.029328800 0.047771494
## Lag 100 0.038845063 0.038845063 0.015771494
## Lag 500 0.018874626 0.018874626 0.029328800
## Lag 1000 -0.012224002 -0.012224002 0.013771494
## Lag 5000 -0.003721765 -0.003721765 -0.014771494
##          traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 -0.12765635
## Lag 100 -0.09362701
## Lag 500 -0.07507670
## Lag 1000 -0.03080525
## Lag 5000 0.02983728

```

```

##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex,
## Lag 0                                          -0.030749722
## Lag 100                                       -0.016217311
## Lag 500                                       -0.020696084
## Lag 1000                                      0.004301752
## Lag 5000                                      -0.032243808
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2
## Lag 0                                          -0.82656679
## Lag 100                                       -0.69378831
## Lag 500                                       -0.30703065
## Lag 1000                                      -0.12287954
## Lag 5000                                      0.02286923
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex,
## Lag 0                                          -0.92799199
## Lag 100                                       -0.79777083
## Lag 500                                       -0.40858358
## Lag 1000                                      -0.21574629
## Lag 5000                                      0.09114505
##
## , , at.level(sex, "2"):traittarsus:at.level(sex, "2"):traittarsus.animal
##
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "
## Lag 0                                          0.09797335
## Lag 100                                       0.10351817
## Lag 500                                       0.13155871
## Lag 1000                                      0.10405306
## Lag 5000                                      -0.01867047
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex
## Lag 0                                          0.06916387
## Lag 100                                       0.06723792
## Lag 500                                       0.05661327
## Lag 1000                                      0.04949334
## Lag 5000                                      -0.02266138
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "
## Lag 0                                          0.62542960
## Lag 100                                       0.50127738
## Lag 500                                       0.22112585
## Lag 1000                                      0.08374181
## Lag 5000                                      0.01908858
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex
## Lag 0                                          0.88193914
## Lag 100                                       0.69427920
## Lag 500                                       0.35645051
## Lag 1000                                      0.16527850
## Lag 5000                                      -0.06295365
##          traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.
## Lag 0          0.01689248          0.01689248          0.0268

```

```

## Lag 100          0.04383334          0.04383334          0.0265
## Lag 500          0.02339834          0.02339834          0.0205
## Lag 1000         -0.01928577         -0.01928577          0.0028
## Lag 5000          0.00705730          0.00705730         -0.0068
##      traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0            -0.20886061
## Lag 100          -0.13591598
## Lag 500          -0.09687975
## Lag 1000         -0.02449457
## Lag 5000          0.03190234
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex,
## Lag 0            -0.05109208
## Lag 100          -0.03188022
## Lag 500          -0.04005001
## Lag 1000         -0.02780074
## Lag 5000         -0.01852278
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2
## Lag 0            -0.59303753
## Lag 100          -0.51940561
## Lag 500          -0.24130575
## Lag 1000         -0.10573107
## Lag 5000         -0.01338206
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex,
## Lag 0            -0.81645230
## Lag 100          -0.71433129
## Lag 500          -0.37519197
## Lag 1000         -0.19591092
## Lag 5000          0.06790739
##
## , , traitbwt:traitbwt.byear
##
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "
## Lag 0            0.024065323
## Lag 100          -0.007228001
## Lag 500          -0.030438771
## Lag 1000         -0.037319806
## Lag 5000          0.024016255
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex,
## Lag 0            0.0004695241
## Lag 100          0.0041014557
## Lag 500          -0.0373017266
## Lag 1000         -0.0290826060
## Lag 5000          0.0367761623
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "
## Lag 0           -0.0003564631
## Lag 100          0.0041477479
## Lag 500          0.0001227158

```



```

## Lag 1000 -0.0280572485
## Lag 5000 -0.0238953239
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0 0.018892747
## Lag 100 0.022147308
## Lag 500 0.008227267
## Lag 1000 -0.018154492
## Lag 5000 -0.028508237
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 0.2416288100 0.2416288100 -0.004075442
## Lag 100 0.0408026921 0.0408026921 0.0403076151
## Lag 500 -0.0005245788 -0.0005245788 -0.013618036
## Lag 1000 0.0234016930 0.0234016930 0.0071076151
## Lag 5000 -0.0205987850 -0.0205987850 -0.0247076151
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 -0.004075442
## Lag 100 0.027366151
## Lag 500 -0.012918036
## Lag 1000 0.002873670
## Lag 5000 0.004175319
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 0.02448148
## Lag 100 -0.02348874
## Lag 500 0.06170381
## Lag 1000 0.03904823
## Lag 5000 -0.06013763
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0 -0.01327914
## Lag 100 -0.02432572
## Lag 500 -0.01282050
## Lag 1000 0.03378831
## Lag 5000 0.03616682
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 -0.03186040
## Lag 100 -0.03568005
## Lag 500 -0.01732292
## Lag 1000 0.01976704
## Lag 5000 0.04292710
##
## , , traittarsus:traitbwt.byear
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0 -0.020803022
## Lag 100 -0.016047256
## Lag 500 0.008481995
## Lag 1000 -0.086888857
## Lag 5000 0.006010889

```

```

##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0                                          -0.03155374
## Lag 100                                       -0.02090225
## Lag 500                                       -0.01282744
## Lag 1000                                      -0.07269148
## Lag 5000                                      0.00906733
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):
## Lag 0                                          0.04294576
## Lag 100                                       0.01731935
## Lag 500                                       0.03962076
## Lag 1000                                      0.02125431
## Lag 5000                                      0.03200298
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0                                          0.029328800
## Lag 100                                       0.018729883
## Lag 500                                       0.031732275
## Lag 1000                                      0.027724247
## Lag 5000                                      0.005040522
##          traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0          1.0000000000          1.0000000000          0.3218
## Lag 100         0.0390208931          0.0390208931         -0.0018
## Lag 500        -0.0100929579        -0.0100929579        -0.0082
## Lag 1000       -0.0076623070       -0.0076623070       -0.0140
## Lag 5000      -0.0005155025      -0.0005155025         0.0160
##          traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0          0.016340654
## Lag 100         0.009692208
## Lag 500         0.003429620
## Lag 1000       -0.022193805
## Lag 5000      -0.027062081
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0          -0.001255739
## Lag 100         0.043187160
## Lag 500         0.048070249
## Lag 1000        0.063998723
## Lag 5000       -0.032351394
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0          -0.04867322
## Lag 100        -0.04918743
## Lag 500        -0.06575193
## Lag 1000       -0.03077705
## Lag 5000       -0.03431783
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0          -0.04503037
## Lag 100        -0.03923249
## Lag 500        -0.05136636
## Lag 1000       -0.03533135

```

```

## Lag 5000 -0.01007200
##
## , , traitbwt:traittarsus.byear
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "
## Lag 0 -0.020803022
## Lag 100 -0.016047256
## Lag 500 0.008481995
## Lag 1000 -0.086888857
## Lag 5000 0.006010889
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex
## Lag 0 -0.03155374
## Lag 100 -0.02090225
## Lag 500 -0.01282744
## Lag 1000 -0.07269148
## Lag 5000 0.00906733
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "
## Lag 0 0.04294576
## Lag 100 0.01731935
## Lag 500 0.03962076
## Lag 1000 0.02125431
## Lag 5000 0.03200298
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex
## Lag 0 0.029328800
## Lag 100 0.018729883
## Lag 500 0.031732275
## Lag 1000 0.027724247
## Lag 5000 0.005040522
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.
## Lag 0 1.0000000000 1.0000000000 0.3218
## Lag 100 0.0390208931 0.0390208931 -0.0018
## Lag 500 -0.0100929579 -0.0100929579 -0.0082
## Lag 1000 -0.0076623070 -0.0076623070 -0.0140
## Lag 5000 -0.0005155025 -0.0005155025 0.0160
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 0.016340654
## Lag 100 0.009692208
## Lag 500 0.003429620
## Lag 1000 -0.022193805
## Lag 5000 -0.027062081
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex,
## Lag 0 -0.001255739
## Lag 100 0.043187160
## Lag 500 0.048070249
## Lag 1000 0.063998723
## Lag 5000 -0.032351394
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2

```

```

## Lag 0 -0.04867322
## Lag 100 -0.04918743
## Lag 500 -0.06575193
## Lag 1000 -0.03077705
## Lag 5000 -0.03431783
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 -0.04503037
## Lag 100 -0.03923249
## Lag 500 -0.05136636
## Lag 1000 -0.03533135
## Lag 5000 -0.01007200
##
## , , traittarsus:traittarsus.byear
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0 -0.062422239
## Lag 100 0.011583874
## Lag 500 0.037441803
## Lag 1000 -0.009627398
## Lag 5000 -0.022238184
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0 -0.038505768
## Lag 100 0.001804516
## Lag 500 0.005261718
## Lag 1000 0.005983226
## Lag 5000 0.032748208
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):
## Lag 0 0.06519556
## Lag 100 0.02663892
## Lag 500 0.04808592
## Lag 1000 0.02428387
## Lag 5000 -0.02466630
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0 0.04789443
## Lag 100 0.03493449
## Lag 500 0.05614210
## Lag 1000 0.04153017
## Lag 5000 -0.01844607
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 0.321896680 0.321896680 1.000000000
## Lag 100 -0.025763034 -0.025763034 -0.054000000
## Lag 500 0.007271524 0.007271524 0.014000000
## Lag 1000 -0.010372408 -0.010372408 -0.016000000
## Lag 5000 0.008050074 0.008050074 -0.011000000
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 0.09836651
## Lag 100 0.02150474

```

```

## Lag 500 -0.03193224
## Lag 1000 -0.05678049
## Lag 5000 -0.01297283
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex,
## Lag 0 -0.009668697
## Lag 100 0.020499050
## Lag 500 0.017314825
## Lag 1000 -0.004472068
## Lag 5000 -0.035467618
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2
## Lag 0 -0.0715338265
## Lag 100 -0.0471815918
## Lag 500 -0.0565789706
## Lag 1000 -0.0181400233
## Lag 5000 0.0004730541
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex,
## Lag 0 -0.062503872
## Lag 100 -0.053534956
## Lag 500 -0.056343269
## Lag 1000 -0.036210432
## Lag 5000 0.001141946
##
## , , traitbwt:traitbwt.mother
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "
## Lag 0 -0.20570295
## Lag 100 -0.16090204
## Lag 500 -0.04215661
## Lag 1000 -0.08030891
## Lag 5000 -0.03677952
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex
## Lag 0 -0.15824356
## Lag 100 -0.14929872
## Lag 500 -0.08184054
## Lag 1000 -0.05556932
## Lag 5000 -0.04664653
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "
## Lag 0 -0.17381816
## Lag 100 -0.08982420
## Lag 500 -0.05302974
## Lag 1000 0.07059613
## Lag 5000 0.01444299
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex
## Lag 0 -0.12011810
## Lag 100 -0.05615154
## Lag 500 -0.04397641
## Lag 1000 0.04529630

```

```

## Lag 5000 0.01188481
##          traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.0264843891 -0.0264843891 -0.0264843891
## Lag 100 0.0007355686 0.0007355686 0.0007355686
## Lag 500 0.0725614701 0.0725614701 0.0725614701
## Lag 1000 0.0716302875 0.0716302875 0.0716302875
## Lag 5000 0.0218336784 0.0218336784 0.0218336784
##          traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 -0.15600691
## Lag 100 0.04752473
## Lag 500 -0.02095386
## Lag 1000 0.01549741
## Lag 5000 0.04374839
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 0.11597642
## Lag 100 0.10292257
## Lag 500 0.07497668
## Lag 1000 0.02331658
## Lag 5000 0.01885065
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0 0.07873715
## Lag 100 0.07663687
## Lag 500 0.03807065
## Lag 1000 -0.07032803
## Lag 5000 -0.02922942
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 0.07662601
## Lag 100 0.05392363
## Lag 500 0.02480332
## Lag 1000 -0.04252999
## Lag 5000 -0.01921695
##
## , , traittarsus:traitbwt.mother
##
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0 -0.176162265
## Lag 100 -0.201942319
## Lag 500 -0.098827162
## Lag 1000 -0.107446398
## Lag 5000 0.009469247
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0 -0.215691293
## Lag 100 -0.222444593
## Lag 500 -0.090633732
## Lag 1000 -0.081329922
## Lag 5000 -0.000908406
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):

```

```

## Lag 0 -0.206244301
## Lag 100 -0.130239112
## Lag 500 -0.073358914
## Lag 1000 -0.066630548
## Lag 5000 0.003242124
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0 -0.25319625
## Lag 100 -0.18667698
## Lag 500 -0.09190098
## Lag 1000 -0.08300273
## Lag 5000 0.02361537
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 0.03421338 0.03421338 -0.0053
## Lag 100 -0.02147513 -0.02147513 -0.0298
## Lag 500 -0.01764091 -0.01764091 -0.0465
## Lag 1000 0.05625276 0.05625276 0.0157
## Lag 5000 -0.01776929 -0.01776929 0.0103
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 0.16721552
## Lag 100 0.15522894
## Lag 500 0.05303655
## Lag 1000 0.04025560
## Lag 5000 -0.01089828
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 0.088466946
## Lag 100 0.172871353
## Lag 500 0.064930722
## Lag 1000 0.044248231
## Lag 5000 0.002955207
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0 0.146008840
## Lag 100 0.080717973
## Lag 500 0.080314209
## Lag 1000 0.079015021
## Lag 5000 -0.006204512
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 0.16532024
## Lag 100 0.13361759
## Lag 500 0.09526141
## Lag 1000 0.09645763
## Lag 5000 -0.02604643
##
## , , traitbwt:traittarsus.mother
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0 -0.176162265
## Lag 100 -0.201942319

```

```

## Lag 500 -0.098827162
## Lag 1000 -0.107446398
## Lag 5000 0.009469247
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0 -0.215691293
## Lag 100 -0.222444593
## Lag 500 -0.090633732
## Lag 1000 -0.081329922
## Lag 5000 -0.000908406
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):
## Lag 0 -0.206244301
## Lag 100 -0.130239112
## Lag 500 -0.073358914
## Lag 1000 -0.066630548
## Lag 5000 0.003242124
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0 -0.25319625
## Lag 100 -0.18667698
## Lag 500 -0.09190098
## Lag 1000 -0.08300273
## Lag 5000 0.02361537
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 0.03421338 0.03421338 -0.0053
## Lag 100 -0.02147513 -0.02147513 -0.0298
## Lag 500 -0.01764091 -0.01764091 -0.0465
## Lag 1000 0.05625276 0.05625276 0.0157
## Lag 5000 -0.01776929 -0.01776929 0.0103
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 0.16721552
## Lag 100 0.15522894
## Lag 500 0.05303655
## Lag 1000 0.04025560
## Lag 5000 -0.01089828
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 0.088466946
## Lag 100 0.172871353
## Lag 500 0.064930722
## Lag 1000 0.044248231
## Lag 5000 0.002955207
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0 0.146008840
## Lag 100 0.080717973
## Lag 500 0.080314209
## Lag 1000 0.079015021
## Lag 5000 -0.006204512
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 0.16532024

```



```

## Lag 100 0.13361759
## Lag 500 0.09526141
## Lag 1000 0.09645763
## Lag 5000 -0.02604643
##
## , , traittarsus:traittarsus.mother
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0 -0.07667827
## Lag 100 -0.06226975
## Lag 500 -0.08505831
## Lag 1000 -0.02905388
## Lag 5000 0.01421665
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0 -0.117487289
## Lag 100 -0.106945179
## Lag 500 -0.080885806
## Lag 1000 -0.005649094
## Lag 5000 0.022493960
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):
## Lag 0 -0.03867169
## Lag 100 -0.03628672
## Lag 500 -0.05826822
## Lag 1000 -0.09013252
## Lag 5000 0.02007633
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0 -0.12765635
## Lag 100 -0.10093720
## Lag 500 -0.07484634
## Lag 1000 -0.07105350
## Lag 5000 0.04241702
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 0.016340654 0.016340654 0.098361
## Lag 100 0.004193689 0.004193689 -0.04216
## Lag 500 0.003520238 0.003520238 -0.00018
## Lag 1000 0.036042597 0.036042597 0.08187
## Lag 5000 -0.026499253 -0.026499253 0.00621
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 1.00000000
## Lag 100 0.20522469
## Lag 500 0.06686362
## Lag 1000 0.02863809
## Lag 5000 -0.03434318
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 0.0243583650
## Lag 100 0.0878499058
## Lag 500 0.0669412439

```

```
## Lag 1000 0.0159489282
## Lag 5000 0.0006521949
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2")
## Lag 0 0.01708099
## Lag 100 0.02260996
## Lag 500 0.04166800
## Lag 1000 0.08719186
## Lag 5000 -0.03439141
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2")
## Lag 0 0.07362567
## Lag 100 0.06893873
## Lag 500 0.06870851
## Lag 1000 0.08644843
## Lag 5000 -0.05929514
##
## , , at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.units
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1")
## Lag 0 -0.801992699
## Lag 100 -0.466443587
## Lag 500 -0.150327965
## Lag 1000 -0.006651214
## Lag 5000 0.021076965
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1")
## Lag 0 -0.63791397
## Lag 100 -0.39655482
## Lag 500 -0.14069619
## Lag 1000 -0.01005484
## Lag 5000 0.05652797
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2")
## Lag 0 -0.07682782
## Lag 100 -0.08897986
## Lag 500 -0.02976621
## Lag 1000 0.01569476
## Lag 5000 0.05852136
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2")
## Lag 0 -0.076833431
## Lag 100 -0.075735828
## Lag 500 -0.001251349
## Lag 1000 0.001383869
## Lag 5000 0.054098508
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.004458633 -0.004458633 0.0094
## Lag 100 -0.024525504 -0.024525504 -0.0109
## Lag 500 -0.042750470 -0.042750470 -0.0156
## Lag 1000 0.033296719 0.033296719 -0.0493
## Lag 5000 -0.012060362 -0.012060362 0.0625
```

```

##          traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0          0.01599739
## Lag 100         0.07877892
## Lag 500         0.03430764
## Lag 1000        -0.05261433
## Lag 5000        0.02086879
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex,
## Lag 0          0.81033812
## Lag 100         0.35818951
## Lag 500         0.13916697
## Lag 1000        -0.03014218
## Lag 5000        -0.04609000
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2
## Lag 0          0.04586363
## Lag 100         0.08104246
## Lag 500         0.01149950
## Lag 1000        -0.02427903
## Lag 5000        -0.04131701
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex,
## Lag 0          0.056053162
## Lag 100         0.069215391
## Lag 500         -0.003748895
## Lag 1000        -0.004420742
## Lag 5000        -0.044430568
##
## , , at.level(sex, "1"):traittarsus:at.level(sex, "1"):traitbwt.units
##
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "
## Lag 0          -0.65709502
## Lag 100         -0.40847504
## Lag 500         -0.16003963
## Lag 1000        -0.03935069
## Lag 5000        0.02830878
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex,
## Lag 0          -0.81241841
## Lag 100         -0.54669565
## Lag 500         -0.21812611
## Lag 1000        -0.05053514
## Lag 5000        0.04040407
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "
## Lag 0          -0.032836717
## Lag 100         -0.052172861
## Lag 500         -0.065393159
## Lag 1000        0.002569735
## Lag 5000        -0.012092021
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex,
## Lag 0          -0.030749722

```

```

## Lag 100 -0.037358966
## Lag 500 -0.023624561
## Lag 1000 -0.006129874
## Lag 5000 -0.025591759
##          traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.001255739 -0.001255739 -0.00961759
## Lag 100 -0.003961776 -0.003961776 0.00571759
## Lag 500 -0.021931503 -0.021931503 -0.01131759
## Lag 1000 0.057231093 0.057231093 0.00521759
## Lag 5000 -0.043285219 -0.043285219 0.04061759
##          traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 0.02435836
## Lag 100 0.09460292
## Lag 500 0.07354709
## Lag 1000 -0.01282799
## Lag 5000 0.03529550
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 1.000000000
## Lag 100 0.476755183
## Lag 500 0.179577504
## Lag 1000 -0.003352382
## Lag 5000 -0.024575546
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0 -0.009637802
## Lag 100 0.035505918
## Lag 500 0.039910212
## Lag 1000 -0.028636833
## Lag 5000 0.033100911
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 -0.013857108
## Lag 100 0.014173844
## Lag 500 0.003568133
## Lag 1000 -0.007944374
## Lag 5000 0.039786826
##
## , , at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units
##
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0 -0.65709502
## Lag 100 -0.40847504
## Lag 500 -0.16003963
## Lag 1000 -0.03935069
## Lag 5000 0.02830878
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0 -0.81241841
## Lag 100 -0.54669565
## Lag 500 -0.21812611

```

```

## Lag 1000 -0.05053514
## Lag 5000 0.04040407
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):traitbwt.animal
## Lag 0 -0.032836717
## Lag 100 -0.052172861
## Lag 500 -0.065393159
## Lag 1000 0.002569735
## Lag 5000 -0.012092021
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):traittarsus.animal
## Lag 0 -0.030749722
## Lag 100 -0.037358966
## Lag 500 -0.023624561
## Lag 1000 -0.006129874
## Lag 5000 -0.025591759
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.001255739 -0.001255739 -0.009611111
## Lag 100 -0.003961776 -0.003961776 0.005714286
## Lag 500 -0.021931503 -0.021931503 -0.011311111
## Lag 1000 0.057231093 0.057231093 0.005238095
## Lag 5000 -0.043285219 -0.043285219 0.040611111
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.mother
## Lag 0 0.02435836
## Lag 100 0.09460292
## Lag 500 0.07354709
## Lag 1000 -0.01282799
## Lag 5000 0.03529550
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):traittarsus.units
## Lag 0 1.000000000
## Lag 100 0.476755183
## Lag 500 0.179577504
## Lag 1000 -0.003352382
## Lag 5000 -0.024575546
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):traitbwt.units
## Lag 0 -0.009637802
## Lag 100 0.035505918
## Lag 500 0.039910212
## Lag 1000 -0.028636833
## Lag 5000 0.033100911
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):traittarsus.units
## Lag 0 -0.013857108
## Lag 100 0.014173844
## Lag 500 0.003568133
## Lag 1000 -0.007944374
## Lag 5000 0.039786826
##
## , , at.level(sex, "1"):traittarsus:at.level(sex, "1"):traittarsus.units
##

```

```

##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0                                          -0.45087062
## Lag 100                                       -0.28739699
## Lag 500                                       -0.13251881
## Lag 1000                                      -0.07316564
## Lag 5000                                      0.02041486
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):
## Lag 0                                          -0.72502529
## Lag 100                                       -0.50401217
## Lag 500                                       -0.23102776
## Lag 1000                                      -0.09178023
## Lag 5000                                      0.02295265
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):
## Lag 0                                          -0.03763051
## Lag 100                                       -0.05789881
## Lag 500                                       -0.10643108
## Lag 1000                                      -0.02645220
## Lag 5000                                      -0.05174777
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0                                          -0.02000832
## Lag 100                                       -0.03719279
## Lag 500                                       -0.06000913
## Lag 1000                                      -0.03409990
## Lag 5000                                      -0.07553958
##          traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0          0.0098141346          0.0098141346          -0.0241141346
## Lag 100         0.0006238442          0.0006238442          0.0141141346
## Lag 500        -0.0029616561        -0.0029616561          0.0101141346
## Lag 1000        0.0609602246          0.0609602246          0.0221141346
## Lag 5000       -0.0551569939        -0.0551569939          0.0261141346
##          traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0          0.01763805
## Lag 100         0.09506980
## Lag 500         0.10621181
## Lag 1000        0.02274044
## Lag 5000        0.02807370
##          at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0          0.862148469
## Lag 100         0.435184658
## Lag 500         0.171023540
## Lag 1000        0.031505725
## Lag 5000        0.002557116
##          at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0          -0.00202713
## Lag 100         0.04482181
## Lag 500         0.07846680
## Lag 1000       -0.00584738

```

```

## Lag 5000                                0.07083952
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex,
## Lag 0                                    -0.026565121
## Lag 100                                0.008433569
## Lag 500                                0.034277377
## Lag 1000                               0.017663601
## Lag 5000                               0.086095462
##
## , , at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units
##
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "
## Lag 0                                    -0.04704752
## Lag 100                                -0.03379234
## Lag 500                                -0.03949223
## Lag 1000                               0.01961744
## Lag 5000                               -0.02397025
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex
## Lag 0                                    -0.006983857
## Lag 100                                -0.004436391
## Lag 500                                -0.013243079
## Lag 1000                               0.032487992
## Lag 5000                               -0.005502160
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "
## Lag 0                                    -0.9155511556
## Lag 100                                -0.7376972476
## Lag 500                                -0.2894597816
## Lag 1000                               -0.0869605591
## Lag 5000                               -0.0004460346
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex
## Lag 0                                    -0.82656679
## Lag 100                                -0.69942848
## Lag 500                                -0.34348383
## Lag 1000                               -0.15386769
## Lag 5000                               0.03083674
##      traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.
## Lag 0                 -0.0486732184          -0.0486732184          -0.0715
## Lag 100                -0.0422553353          -0.0422553353          -0.0160
## Lag 500                -0.0068642332          -0.0068642332           0.0014
## Lag 1000               -0.0075606736          -0.0075606736          -0.0174
## Lag 5000                0.0008585885           0.0008585885           0.0255
##      traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0                   0.0170809947
## Lag 100                 -0.0037653532
## Lag 500                 -0.0003468626
## Lag 1000                0.0273309637
## Lag 5000               -0.0135559407
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex,

```

```

## Lag 0 -0.0096378021
## Lag 100 -0.0125008943
## Lag 500 -0.0009416032
## Lag 1000 -0.0279523801
## Lag 5000 0.0306544784
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2")
## Lag 0 1.00000000
## Lag 100 0.75428810
## Lag 500 0.31054236
## Lag 1000 0.12371798
## Lag 5000 -0.01140046
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2")
## Lag 0 0.8861603
## Lag 100 0.7245493
## Lag 500 0.3612800
## Lag 1000 0.1966026
## Lag 5000 -0.0473486
##
## , , at.level(sex, "2"):traittarsus:at.level(sex, "2"):traitbwt.units
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1")
## Lag 0 -0.05664210
## Lag 100 -0.06189130
## Lag 500 -0.08037954
## Lag 1000 -0.03811311
## Lag 5000 -0.02556470
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1")
## Lag 0 -0.008328493
## Lag 100 -0.013077572
## Lag 500 -0.023186615
## Lag 1000 -0.003441329
## Lag 5000 -0.009315017
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2")
## Lag 0 -0.827378091
## Lag 100 -0.698680660
## Lag 500 -0.293915297
## Lag 1000 -0.075432320
## Lag 5000 -0.006566931
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2")
## Lag 0 -0.92799199
## Lag 100 -0.79641710
## Lag 500 -0.40507130
## Lag 1000 -0.17373081
## Lag 5000 0.05921217
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.0450303697 -0.0450303697 -0.062
## Lag 100 -0.0353347165 -0.0353347165 -0.015

```



```

## Lag 500          -0.0139632410          -0.0139632410          -0.024
## Lag 1000         0.0016932093          0.0016932093          -0.011
## Lag 5000        -0.0002862914        -0.0002862914          0.020
##      traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0            0.07362567
## Lag 100          0.05538368
## Lag 500          0.03584330
## Lag 1000         0.05064644
## Lag 5000        -0.02384110
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex,
## Lag 0            -0.013857108
## Lag 100          -0.007575433
## Lag 500          0.013108517
## Lag 1000         -0.007812007
## Lag 5000         0.036032541
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2
## Lag 0            0.886160329
## Lag 100          0.717772991
## Lag 500          0.321665315
## Lag 1000         0.103897403
## Lag 5000        -0.007716877
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex,
## Lag 0            1.00000000
## Lag 100          0.82518740
## Lag 500          0.42756493
## Lag 1000         0.20452354
## Lag 5000        -0.07400879
##
## , , at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units
##
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "
## Lag 0            -0.05664210
## Lag 100          -0.06189130
## Lag 500          -0.08037954
## Lag 1000         -0.03811311
## Lag 5000        -0.02556470
##      at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex
## Lag 0            -0.008328493
## Lag 100          -0.013077572
## Lag 500          -0.023186615
## Lag 1000         -0.003441329
## Lag 5000        -0.009315017
##      at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "
## Lag 0            -0.827378091
## Lag 100          -0.698680660
## Lag 500          -0.293915297
## Lag 1000         -0.075432320

```

```

## Lag 5000 -0.006566931
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0 -0.92799199
## Lag 100 -0.79641710
## Lag 500 -0.40507130
## Lag 1000 -0.17373081
## Lag 5000 0.05921217
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.0450303697 -0.0450303697 -0.0621111111
## Lag 100 -0.0353347165 -0.0353347165 -0.0151111111
## Lag 500 -0.0139632410 -0.0139632410 -0.0241111111
## Lag 1000 0.0016932093 0.0016932093 -0.0111111111
## Lag 5000 -0.0002862914 -0.0002862914 0.0201111111
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 0.07362567
## Lag 100 0.05538368
## Lag 500 0.03584330
## Lag 1000 0.05064644
## Lag 5000 -0.02384110
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 -0.013857108
## Lag 100 -0.007575433
## Lag 500 0.013108517
## Lag 1000 -0.007812007
## Lag 5000 0.036032541
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0 0.886160329
## Lag 100 0.717772991
## Lag 500 0.321665315
## Lag 1000 0.103897403
## Lag 5000 -0.007716877
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 1.00000000
## Lag 100 0.82518740
## Lag 500 0.42756493
## Lag 1000 0.20452354
## Lag 5000 -0.07400879
##
## , , at.level(sex, "2"):traittarsus:at.level(sex, "2"):traittarsus.units
##
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traitbwt.animal at.level(sex, "1"):
## Lag 0 -0.07483207
## Lag 100 -0.08775348
## Lag 500 -0.11285363
## Lag 1000 -0.08634561
## Lag 5000 -0.01371202
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.animal at.level(sex, "1"):

```

```

## Lag 0 -0.030362949
## Lag 100 -0.039754967
## Lag 500 -0.033011864
## Lag 1000 -0.029514426
## Lag 5000 -0.001985781
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.animal at.level(sex, "2"):
## Lag 0 -0.60748687
## Lag 100 -0.53691093
## Lag 500 -0.23923956
## Lag 1000 -0.05612519
## Lag 5000 -0.02437865
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.animal at.level(sex, "2"):
## Lag 0 -0.83302980
## Lag 100 -0.72884123
## Lag 500 -0.38274139
## Lag 1000 -0.15431830
## Lag 5000 0.05605301
## traittarsus:traitbwt.byear traitbwt:traittarsus.byear traittarsus:traittarsus.byear
## Lag 0 -0.02352674 -0.02352674 -0.04771348
## Lag 100 -0.03785607 -0.03785607 -0.02471348
## Lag 500 -0.02280501 -0.02280501 -0.02211348
## Lag 1000 0.01610589 0.01610589 -0.01071348
## Lag 5000 -0.01180713 -0.01180713 0.00821348
## traittarsus:traittarsus.mother at.level(sex, "1"):traitbwt:at.level(sex, "1"):
## Lag 0 0.12535823
## Lag 100 0.11202094
## Lag 500 0.06571377
## Lag 1000 0.03808944
## Lag 5000 -0.02117192
## at.level(sex, "1"):traitbwt:at.level(sex, "1"):traittarsus.units at.level(sex, "1"):
## Lag 0 0.007675143
## Lag 100 0.016491158
## Lag 500 0.025505457
## Lag 1000 0.010905754
## Lag 5000 0.026867691
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traitbwt.units at.level(sex, "2"):
## Lag 0 0.649582994
## Lag 100 0.557866006
## Lag 500 0.268804888
## Lag 1000 0.081158620
## Lag 5000 0.009810318
## at.level(sex, "2"):traitbwt:at.level(sex, "2"):traittarsus.units at.level(sex, "2"):
## Lag 0 0.89529429
## Lag 100 0.75772487
## Lag 500 0.40719270
## Lag 1000 0.18139226
## Lag 5000 -0.06899758

```

Evaluation of the statistical support for these sex-specific correlations is straightforward. Because we imposed no constraint on their estimation, we can evaluate the extent to which the posterior distributions overlap zero or overlap each other:

Neither of these posterior distributions overlaps zero, so we can consider them both statistically supported.

### 3.5 brms

```
library(brms)
Amat <- as.matrix(nadiv::makeA(gryphonped))
bf_bwt <- bf(bwt ~ 1 + (1 | p | gr(animal, cov = Amat)))
bf_tarsus <- bf(tarsus ~ 1 + (1 | p | gr(animal, cov = Amat)))
brms_m2.1 <- brm(
  bf_bwt + bf_tarsus + set_rescor(TRUE),
  data = gryphon,
  data2 = list(Amat = Amat),
  chains = 2, cores = 2, iter = 1000
)
save(brms_m2.1, file = "data/brms_m2_1.rda")
```

```
load("data/brms_m2_1.rda")
summary(brms_m2.1)
```

```
## Warning: Parts of the model have not converged (some Rhats are > 1.05). Be careful when

## Family: MV(gaussian, gaussian)
## Links: mu = identity; sigma = identity
## mu = identity; sigma = identity
## Formula: bwt ~ 1 + (1 | p | gr(animal, cov = Amat))
## tarsus ~ 1 + (1 | p | gr(animal, cov = Amat))
## Data: gryphon (Number of observations: 683)
## Samples: 2 chains, each with iter = 1000; warmup = 500; thin = 1;
## total post-warmup samples = 1000
##
## Group-Level Effects:
## ~animal (Number of levels: 683)
##
```

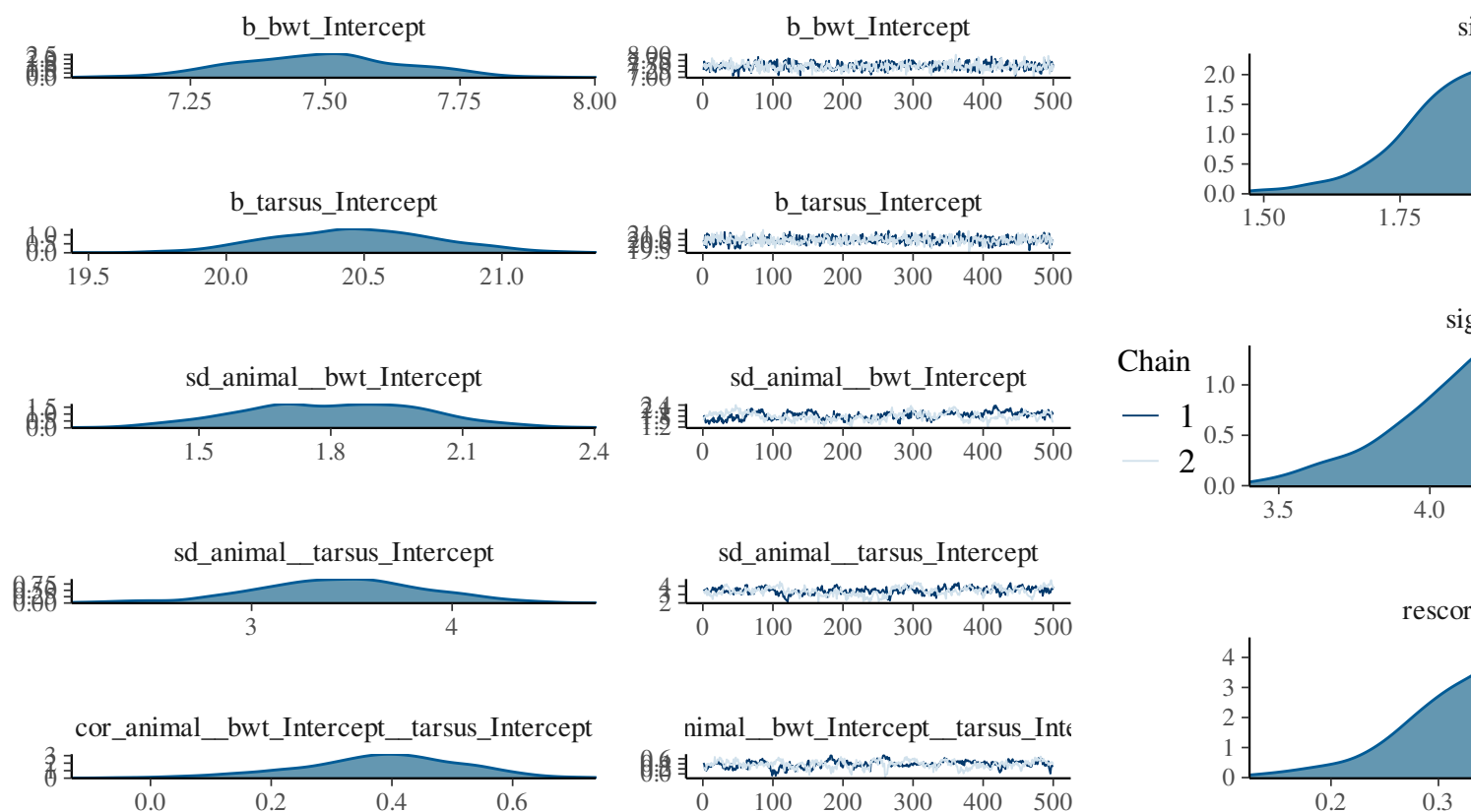
	Estimate	Est.Error	1-95% CI	u-95% CI	Rhat	Bulk_ESS
sd(bwt_Intercept)	1.81	0.21	1.41	2.20	1.06	32
sd(tarsus_Intercept)	3.44	0.43	2.49	4.25	1.05	61
cor(bwt_Intercept,tarsus_Intercept)	0.38	0.14	0.08	0.62	1.02	101

```
##
## Population-Level Effects:
##
```

	Estimate	Est.Error	1-95% CI	u-95% CI	Rhat	Bulk_ESS	Tail_ESS
bwt_Intercept	7.49	0.16	7.20	7.79	1.00	607	839

```
## tarsus_Intercept      20.47      0.30      19.92      21.03 1.00      868      803
##
## Family Specific Parameters:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma_bwt      1.97      0.16      1.66      2.28 1.06      27      172
## sigma_tarsus    4.24      0.30      3.63      4.82 1.04      71      162
##
## Residual Correlations:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## rescor(bwt,tarsus) 0.39      0.09      0.21      0.55 1.02      95      179
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
plot(brms_m2.1, ask = FALSE)
```



```
VarCorr(brms_m2.1)
```

```
## $animal
## $animal$sd
##           Estimate Est.Error      Q2.5      Q97.5
## bwt_Intercept  1.808171 0.2050233 1.412824 2.204805
## tarsus_Intercept 3.438368 0.4283612 2.491218 4.245264
```

```
##
## $animal$cor
## , , bwt_Intercept
##
##           Estimate Est.Error      Q2.5      Q97.5
## bwt_Intercept  1.0000000 0.0000000 1.0000000 1.0000000
## tarsus_Intercept 0.3814062 0.1380014 0.07581464 0.6209038
##
## , , tarsus_Intercept
##
##           Estimate Est.Error      Q2.5      Q97.5
## bwt_Intercept  0.3814062 0.1380014 0.07581464 0.6209038
## tarsus_Intercept 1.0000000 0.0000000 1.0000000 1.0000000
##
##
## $animal$cov
## , , bwt_Intercept
##
##           Estimate Est.Error      Q2.5      Q97.5
## bwt_Intercept  3.311473 0.7430185 1.9960721 4.861167
## tarsus_Intercept 2.440166 1.0901689 0.3870783 4.668720
##
## , , tarsus_Intercept
##
##           Estimate Est.Error      Q2.5      Q97.5
## bwt_Intercept  2.440166 1.090169 0.3870783 4.66872
## tarsus_Intercept 12.005688 2.918741 6.2061701 18.02226
##
##
## $residual__
## $residual__$sd
##           Estimate Est.Error      Q2.5      Q97.5
## bwt  1.970532 0.1597581 1.658782 2.276074
## tarsus 4.244704 0.2984518 3.632824 4.820109
##
## $residual__$cor
## , , bwt
##
##           Estimate Est.Error      Q2.5      Q97.5
## bwt  1.0000000 0.0000000 1.0000000 1.0000000
## tarsus 0.3888754 0.08510488 0.2127907 0.5526631
##
## , , tarsus
##
##           Estimate Est.Error      Q2.5      Q97.5
## bwt  0.3888754 0.08510488 0.2127907 0.5526631
```

```
## tarsus 1.0000000 0.0000000 1.0000000 1.0000000
##
##
## $residual__$cov
## , , bwt
##
##      Estimate Est.Error      Q2.5      Q97.5
## bwt      3.908493 0.6282892 2.751557 5.180511
## tarsus 3.289995 0.9305960 1.572647 5.147133
##
## , , tarsus
##
##      Estimate Est.Error      Q2.5      Q97.5
## bwt      3.289995 0.930596 1.572647 5.147133
## tarsus 18.106495 2.530138 13.197409 23.233452
```

## 3.6 stan

to do





# Chapitre 4

## A repeated measures animal model

This tutorial will demonstrate how to run a univariate animal model for a trait with repeated observations using different R packages and example data files provided.

### 4.1 Scenario and data

#### 4.1.1 scenario

Since gryphons are iteroparous, multiple observations of reproductive traits are available for some individuals. Here we have repeated measures of lay date (measured in days after January 1) for individual females varying in age from 2 (age of maturation) up until age 6. Not all females lay every year so the number of observations per female is variable. We want to know how repeatable the trait is, and (assuming it is repeatable) how heritable it is.

#### 4.1.2 Data files

The pedigree file `gryphonped.csv` is that used in the preceding tutorials but we now use a new data file `gryphonRM.csv`. Columns correspond to individual identity (`animal`), birth year (`byear`), age in years (`age`), year of measurement (`year`) and lay date (`laydate`). Each row of the data file corresponds to a single phenotypic observation. Here data are sorted by identity and then age so that the repeated observations on individuals are readily apparent. However this is not a requirement for analysis - data could equally be sorted by some other variable (*e.g.*, measurement year) or be in a random order.

```
str(gryphonRM)
```

```
## 'data.frame':   1607 obs. of  5 variables:
## $ animal : Factor w/ 469 levels "1","2","3","8",...: 1 1 1 1 1 2 2 2 3 3 ...
## $ byear  : Factor w/ 34 levels "968","970","971",...: 22 22 22 22 22 22 22 22 22 22 ...
## $ age    : Factor w/ 5 levels "2","3","4","5",...: 1 2 3 4 5 1 2 3 1 2 ...
## $ year   : Factor w/ 39 levels "970","971","972",...: 23 24 25 26 27 23 24 25 23 24 ...
## $ laydate: num  19 23 24 23 29 21 17 21 20 20 ...
```

```
head(gryphonRM)
```

```
##   animal byear age year laydate
## 1      1   990  2  992      19
## 2      1   990  3  993      23
## 3      1   990  4  994      24
## 4      1   990  5  995      23
## 5      1   990  6  996      29
## 6      2   990  2  992      21

## 'data.frame':   1309 obs. of  3 variables:
##  $ id      : int  1306 1304 1298 1293 1290 1288 1284 1283 1282 1278 ...
##  $ father: int   NA NA NA NA NA NA NA NA NA NA ...
##  $ mother: int   NA NA NA NA NA NA NA NA NA NA ...
```

```
gryphonped$id <- as.factor(gryphonped$id)
gryphonped$father <- as.factor(gryphonped$father)
gryphonped$mother <- as.factor(gryphonped$mother)
```

## 4.2 Asreml-R

### 4.2.1 Estimating repeatability

With repeated measures on individuals it is often of interest, prior to fitting a genetic model, to see how repeatable a trait is. We can estimate the repeatability of a trait as the proportion of phenotypic variance explained by individual identity.

```
modelv <- asreml(
  fixed = laydate ~ 1,
  random = ~animal,
  residual = ~ idv(units),
  data = gryphonRM,
  na.action = na.method(x = "omit", y = "omit")
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:56:35 2021
##           LogLik          Sigma2      DF      wall      cpu
## 1      -10182.83           1.0    1606 01:56:35    0.0
## 2      -8266.10           1.0    1606 01:56:35    0.0
## 3      -6145.01           1.0    1606 01:56:35    0.0
## 4      -4651.57           1.0    1606 01:56:35    0.0
## 5      -3819.31           1.0    1606 01:56:35    0.0
## 6      -3554.22           1.0    1606 01:56:35    0.0
```

```
## 7      -3501.56      1.0  1606 01:56:35  0.0
## 8      -3497.58      1.0  1606 01:56:35  0.0
## 9      -3497.54      1.0  1606 01:56:35  0.0
## 10     -3497.54      1.0  1606 01:56:35  0.0
```

Note that since we want to estimate the amount of variance explained by individual identity (rather than by additive effects), we fit `animal` as a normal random effect and we don't associate it with the pedigree.

This model partitions the phenotypic variance in `laydate` as follows:

```
summary(modelv)$varcomp
```

```
##          component std.error   z.ratio bound %ch
## animal      11.08634 1.1794319  9.399728    P    0
## units!units  21.29643 0.8896196 23.938798    P    0
## units!R      1.00000      NA        NA     F    0
```

Between-individual variance is given by the `animal` component, while the residual component (`units!units`) represents within-individual variance. Here then the repeatability of the trait can be determined by hand as 0.34 (*i.e.*, as  $11.086/(11.086 + 21.296)$ ).

Mean lay date might change with age, so we could ask what the repeatability of lay date is after conditioning on age. This would be done by adding `age` into the model as a fixed effect.

```
modelw <- asreml(
  fixed = laydate ~ age,
  random = ~animal,
  residual = ~ idv(units),
  data = gryphonRM,
  na.action = na.method(x = "omit", y = "omit")
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:56:36 2021
##          LogLik      Sigma2      DF      wall      cpu
## 1      -8402.968      1.0    1602 01:56:36  0.0
## 2      -6912.361      1.0    1602 01:56:36  0.0
## 3      -5274.379      1.0    1602 01:56:36  0.0
## 4      -4143.634      1.0    1602 01:56:36  0.0
## 5      -3541.895      1.0    1602 01:56:36  0.0
## 6      -3372.909      1.0    1602 01:56:36  0.0
## 7      -3347.670      1.0    1602 01:56:36  0.0
## 8      -3346.655      1.0    1602 01:56:36  0.0
## 9      -3346.652      1.0    1602 01:56:36  0.0
```

```
summary(modelw)$varcomp
```

```
##           component std.error  z.ratio bound %ch
## animal          12.28982  1.156115 10.63027    P    0
## units!units      16.37989  0.686619 23.85586    P    0
## units!R           1.00000         NA      NA    F    0
```

The repeatability of lay date, after accounting for age effects, is now estimated as 0.43 (*i.e.*, as  $12.29/(12.29 + 16.38)$ ). So, just as we saw when estimating  $h^2$  in Tutorial 1, the inclusion of fixed effects will alter the estimated effect size if we determine total phenotypic variance as the sum of the variance components. Thus, proper interpretation is vital.

Here age is modelled as a 5-level factor (specified using the function `as.factor()` at the beginning of the analysis). We could equally have fitted it as a continuous variable, in which case, given potential for a late life decline, we would probably also include a quadratic term.

## 4.2.2 Partitioning additive and permanent environment effects

Generally we expect that the repeatability will set the upper limit for heritability since, while additive genetic effects will cause among-individual variation, so will other types of effects. Non-additive contributions to fixed among-individual differences are normally referred to as *permanent environment effects*. If a trait has repeated measures then it is necessary to model permanent environment effects in an animal model to prevent upward bias in  $V_A$ .

To illustrate this fit the animal model:

```
modelx <- asreml(
  fixed = laydate ~ age,
  random = ~ vm(animal, ainv),
  residual = ~ idv(units),
  data = gryphonRM,
  na.action = na.method(x = "omit", y = "omit")
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:56:36 2021
##           LogLik      Sigma2      DF      wall      cpu
## 1      -8751.390          1.0    1602 01:56:36    0.0
## 2      -7169.205          1.0    1602 01:56:36    0.0
## 3      -5427.604          1.0    1602 01:56:36    0.0
## 4      -4219.598          1.0    1602 01:56:36    0.0
## 5      -3569.815          1.0    1602 01:56:36    0.0
## 6      -3382.341          1.0    1602 01:56:36    0.0
## 7      -3352.867          1.0    1602 01:56:36    0.0
## 8      -3351.565          1.0    1602 01:56:36    0.0
## 9      -3351.560          1.0    1602 01:56:36    0.0
```

Variance components are almost unchanged:

```
summary.asreml(modelx)$varcomp
```

```
##                component std.error   z.ratio bound %ch
## vm(animal, ainv)  13.91784  1.443968  9.638607     P    0
## units!units      16.84008  0.707365 23.806768     P    0
## units!R           1.00000         NA         NA     F    0
```

This suggests that all of the among-individual variance is – rightly or wrongly – being partitioned as  $V_A$  here. To instead obtain an unbiased estimate of  $V_A$  we need to allow for both additive genetic *and* non-genetic sources of individual variation. We do this by fitting `animal` twice, once with a pedigree, and once without a pedigree (using `ide()`).

```
modely <- asreml(
  fixed = laydate ~ age,
  random = ~ vm(animal, ainv) + ide(animal),
  residual = ~ idv(units),
  data = gryphonRM,
  na.action = na.method(x = "omit", y = "omit")
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:56:36 2021
##           LogLik      Sigma2      DF      wall      cpu
##  1      -7731.394          1.0    1602 01:56:36     0.0
##  2      -6426.548          1.0    1602 01:56:36     0.0
##  3      -4997.252          1.0    1602 01:56:36     0.0
##  4      -4018.486          1.0    1602 01:56:36     0.0
##  5      -3504.988          1.0    1602 01:56:36     0.0
##  6      -3363.160          1.0    1602 01:56:36     0.0
##  7      -3341.611          1.0    1602 01:56:36     0.0
##  8      -3340.682          1.0    1602 01:56:36     0.0
##  9      -3340.679          1.0    1602 01:56:36     0.0
```

```
summary(modely)$varcomp
```

```
##                component std.error   z.ratio bound %ch
## vm(animal, ainv)  4.876101  1.8087709  2.695809     P    0
## ide(animal)       7.400983  1.7280113  4.282948     P    0
## units!units      16.380188  0.6866189 23.856300     P    0
## units!R           1.000000         NA         NA     F    0
```

The estimate of  $V_A$  is now much lower since the additive and permanent environment effects are being properly separated. We can estimate  $h^2$  and the repeatability from this model:

```
vpredict(modely, h2 ~ V1 / (V1 + V2 + V3))
```

```
##      Estimate      SE
## h2 0.1701523 0.06073974
```

```
vpredict(modely, repeatability ~ (V1 + V2) / (V1 + V2 + V3))
```

```
##              Estimate      SE
## repeatability 0.4284108 0.02741602
```

### 4.2.3 Adding additional effects and testing significance

Models of repeated measures can be extended to include other fixed or random effects. For example try including year of measurement (`year`) and birth year (`byear`) as random effects.

```
modelz <- asreml(
  fixed = laydate ~ age,
  random = ~ vm(animal, ainv) + ide(animal) +
    year + byear,
  residual = ~ idv(units),
  data = gryphonRM,
  na.action = na.method(x = "omit", y = "omit")
)
```

```
## Model fitted using the sigma parameterization.
## ASReml 4.1.0 Fri May 21 01:56:36 2021
##      LogLik      Sigma2      DF      wall      cpu
## 1    -4650.748        1.0    1602 01:56:36    0.0
## 2    -4088.264        1.0    1602 01:56:36    0.0
## 3    -3494.147        1.0    1602 01:56:36    0.0
## 4    -3127.161        1.0    1602 01:56:36    0.0 (1 restrained)
## 5    -2976.449        1.0    1602 01:56:36    0.0 (1 restrained)
## 6    -2955.785        1.0    1602 01:56:36    0.0 (1 restrained)
## 7    -2955.097        1.0    1602 01:56:36    0.0 (1 restrained)
## 8    -2955.095        1.0    1602 01:56:36    0.0 (1 restrained)
## 9    -2955.095        1.0    1602 01:56:36    0.0
```

```
summary(modelz)$varcomp
```

```
##      component std.error  z.ratio bound %ch
## byear      1.650876e-07      NA      NA    B    0
## year       7.938576e+00 1.9344619 4.103765    P    0
## vm(animal, ainv) 4.815136e+00 1.6682351 2.886365    P    0
## ide(animal)    8.433325e+00 1.5495778 5.442337    P    0
```

```
## units!units      7.795560e+00 0.3324411 23.449443      P      0
## units!R          1.000000e+00      NA      NA      F      0
```

This model will return additional variance components corresponding to variation in lay dates between years of measurement and between birth cohorts of females.  $V_{byear}$  is very low and if you compare this model to a reduced model with byear excluded the log-likelihood remains unchanged.

year effects could alternatively be included as fixed effects (try this!). This will reduce  $V_R$  and increase the estimates of heritability and repeatability, which must now be interpreted as proportions of phenotypic variance after conditioning on both age and year of measurement effects.

## 4.3 gremlin

TODO (maybe just bother Matthew to do it)

Meanwhile



Figure 4.1: Keep it dry and do no feed after midnight.

## 4.4 MCMCglmm

### 4.4.1 Estimating repeatability

With repeated measures on individuals it is often of interest, prior to fitting a genetic model, to see how repeatable a trait is. We can estimate the repeatability of a trait as the proportion of phenotypic variance explained by individual identity using the commands below

```

p.var <- var(gryphonRM$laydate, na.rm = TRUE)
prior3.1 <- list(G = list(G1 = list(V = 1, nu = 0.002)), R = list(
  V = 1,
  nu = 0.002
))
model3.1 <- MCMCglmm(laydate ~ 1,
  random = ~animal, data = gryphonRM,
  prior = prior3.1, verbose = FALSE
)
posterior.mode(model3.1$VCV)

```

```

##   animal      units
## 11.00957 20.97417

```

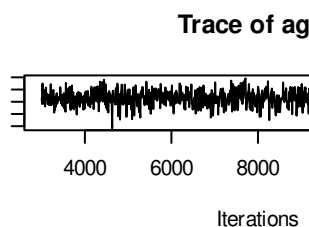
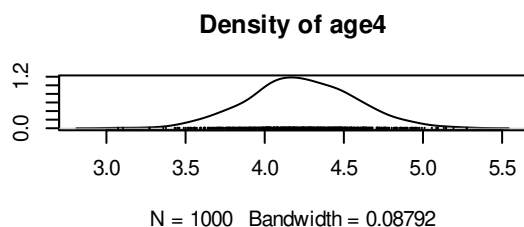
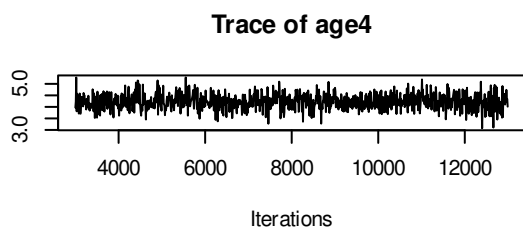
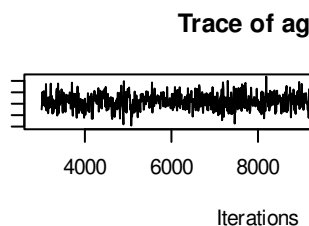
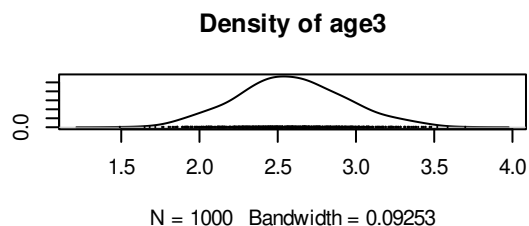
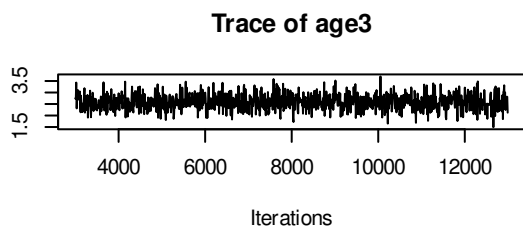
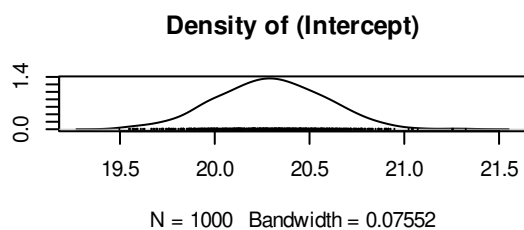
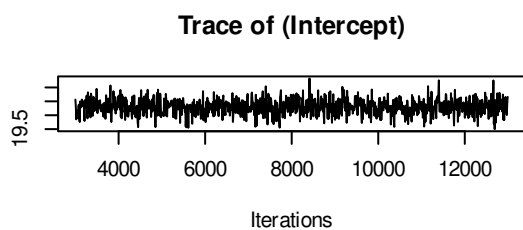
Note the use of the term `animal`, since neither the `pedigree` nor the `ginv` argument are provided, it represents the between-individual variance, while the residual component (Variance) therefore represents within-individual variance. Here then the repeatability of the trait can be determined by as 0.353 (i.e.,  $11.4532/(11.4532+21.1432)$ ). Given that we set up the simulation such that mean lay date changes with age (initially increasing to age 5 before a late life decline) we might ask what the repeatability of lay date is after conditioning on age effect. This would be done by adding age into the model as a fixed effect.

```

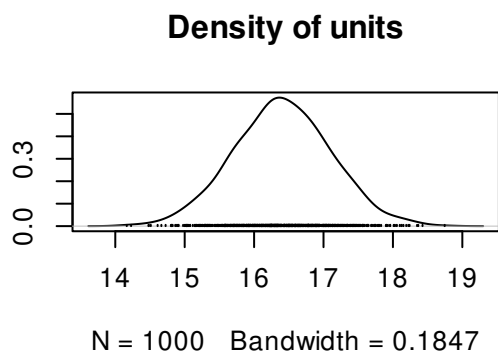
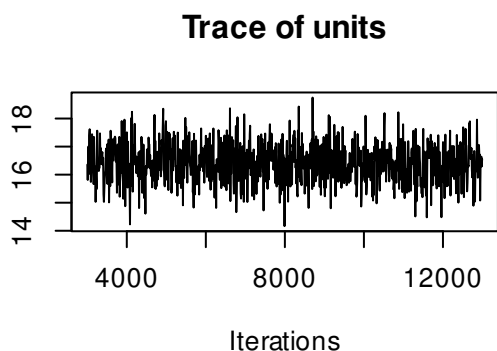
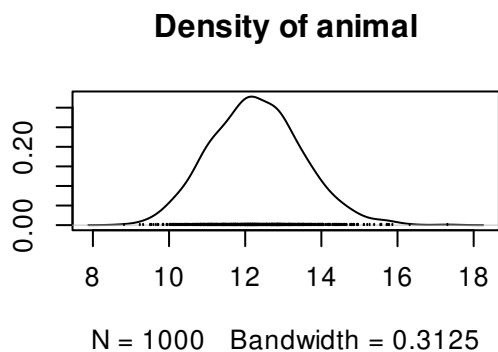
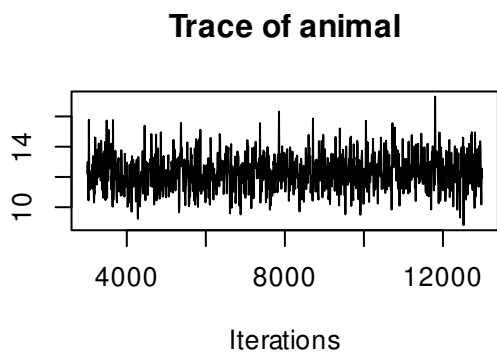
model3.2 <- MCMCglmm(laydate ~ age,
  random = ~animal, data = gryphonRM,
  prior = prior3.1, verbose = FALSE
)
plot(model3.2$Sol)

```





```
plot(model3.2$VCV)
```



```
posterior.mode(model3.2$VCV)
```

```
## animal units
## 12.85222 16.24052
```

Note that the random effect structure has remained unchanged, and so we have not modified the prior between models 3.1 and 3.2. So that the repeatability of laydate, after accounting for age effects, is now estimated as 0.445 (i.e.,  $12.6379/(12.6379+16.465)$ ). So, just as we saw when estimating  $h^2$  in tutorial 1, the inclusion of fixed effects will alter the estimated effect size if we determine total phenotypic variance as the sum of the variance components. Thus, proper interpretation is vital.

Here age is modelled as a 5 level factor (see the conversion of age to a factor in section 3.2). We could equally have fitted it as a continuous variable instead in which case, given the late life decline, we would probably also include a quadratic term.

#### 4.4.2 Partitioning additive and permanent environment effects

Generally we expect that the repeatability will set the upper limit for heritability since, while additive genetic effects will cause among-individual variation, so will other types of effect. Non-additive contributions to fixed among-individual differences are normally referred to as ‘permanent environment effects’, although ‘non-heritable effects’ that are consistent within individuals may be a better way to think of modelling this effect. If a trait has repeated measures then it is necessary to model permanent environment effects in an animal model to prevent upward bias in  $V_A$ . To illustrate this fit the animal model

```
Ainv <- inverseA(gryphonped)$Ainv
model3.3 <- MCMCglmm(laydate ~ 1 + age,
  random = ~animal, ginv = list(animal = Ainv),
  data = gryphonRM, prior = prior3.1, verbose = FALSE
)
posterior.mode(model3.3$VCV)
```

```
## animal units
## 13.87540 17.00617
```

This suggests that all of the among-individual variance is - rightly or wrongly - being partitioned as  $V_A$  here. In fact here the partition is wrong since the simulation included both additive genetic effects and additional fixed heterogeneity that was not associated with the pedigree structure (i.e. permanent environment effects). In order to fit both permanent environment and additive genetic effects, we need to fit the individual identity twice in the model: once linked to the pedigree (genetic effect) and once not linked to the pedigree (permanent environment effect). To do so, we need to duplicate the variable containing the individual identity and give it a new name. An more appropriate estimate of  $V_A$  is given by the model:

```
gryphonRM$animal_pe <- gryphonRM$animal
p.var <- var(gryphonRM$laydate, na.rm = TRUE)
```

```
prior3.4 <- list(G = list(G1 = list(V = 1, nu = 0.002), G2 = list(
  V = 1,
  nu = 0.002
)), R = list(V = 1, nu = 0.002))
model3.4 <- MCMCglmm(laydate ~ 1 + age,
  random = ~ animal + animal_pe,
  ginv = list(animal = Ainv), data = gryphonRM, prior = prior3.4, verbose = FALSE
)
posterior.mode(model3.4$VCV)
```

```
##      animal animal_pe      units
## 4.499468  7.407864 16.472613
```

The estimate of  $V_A$  is now much lower (reduced from 13.6735 to 5.1238) since the additive and permanent environment effects are being properly separated. We could obtain estimates of  $h^2$  and of the repeatability from this model using the following commands:

```
model3.4.VP <- model3.4$VCV[, "animal"] + model3.4$VCV[, "animal_pe"] + model3.4$VCV[, "units"]
model3.4.PE_VA <- model3.4$VCV[, "animal"] + model3.4$VCV[, "animal_pe"]
posterior.mode(model3.4.PE_VA / model3.4.VP)
```

```
##      var1
## 0.4236285
```

```
posterior.mode(model3.4$VCV[, "animal"] / model3.4.VP)
```

```
##      var1
## 0.1585904
```

### 4.4.3 Adding additional effects and testing significance

Models of repeated measures can be extended to include other fixed or random effects. For example try including year of measurement (year).

```
p.var <- var(gryphonRM$laydate, na.rm = TRUE)
prior3.5 <- list(G = list(G1 = list(V = 1, nu = 0.002), G2 = list(
  V = 1,
  nu = 0.002
)), G3 = list(V = 1, nu = 0.002), G4 = list(
  V = 1,
  nu = 0.002
)), R = list(V = 1, nu = 0.002))
model3.5 <- MCMCglmm(laydate ~ 1 + age,
  random = ~ animal + animal_pe +
```

```

    year + byear, ginv = list(animal = Ainv), data = gryphonRM, prior = prior3.5,
    verbose = FALSE
)
posterior.mode(model3.5$VCV)

```

```

##      animal  animal_pe      year      byear      units
## 4.831458287 7.851537615 7.925240216 0.001934676 7.751678678

```

This model will return additional variance components corresponding to year of measurement effects and birth year (of the female effects). The latter were not simulated as should be apparent from the parameter estimate (and by the support interval derivable from the posterior distribution and from DIC-based comparison of model3.5 and a model from which the birth year term had been eliminated, see tutorial 1). However, year effects were simulated as should be apparent from the modal estimate and from the support interval (try this yourself using HPDinterval()) and this could be formally confirmed by comparison of DIC. year effects could alternatively be included as fixed effects (try this, you should be able to handle the new prior specification at this point). Since we simulated large year of measurement effects this treatment will reduce V R and increase the estimates of heritability and repeatability which must now be interpreted as proportions of phenotypic variance after conditioning on both age and year of measurement effects.

## 4.5 brms

```

library(brms)
Amat <- nadiv::makeA(Ped)

model_simple1.1 <- brm(
  bwt ~ 1 + (1 | animal) + (1 | animal_pe),
  data = gryphonRM,
  family = gaussian(), cov_ranef = list(animal = Amat),
  chains = 2, cores = 2, iter = 1000
)

summary(model_simple1.1)
plot(model_simple1.1)

```

## 4.6 stan

to do

# Chapitre 5

## Quick comparison of codes

### 5.1 Univariate model with repeated measures

#### 5.1.1 Asreml-R

#### 5.1.2 gremlin

#### 5.1.3 MCMCglmm

#### 5.1.4 brms

### 5.2 bivariate model

#### 5.2.1 Asreml-R

#### 5.2.2 gremlin

#### 5.2.3 MCMCglmm

#### 5.2.4 brms



# Bibliography

- Hadfield, J. (2021). *MCMCglmm: MCMC Generalised Linear Mixed Models*. R package version 2.32.
- Hadfield, J. D. (2010). Mcmc methods for multi-response generalized linear mixed models: The MCMCglmm R package. *Journal of Statistical Software*, 33(2):1–22.
- Wilson, A. J., Réale, D., Clements, M. N., Morrissey, M. M., Postma, E., Walling, C. A., Kruuk, L. E. B., and Nussey, D. H. (2010). An ecologist’s guide to the animal model. *Journal of Animal Ecology*, 79(1):13–26.