BESSiE

Version 1.0

A program for Best Linear Unbiased Prediction and Bayesian analysis of linear mixed models

Vinzent Boerner[1] and Bruce Tier[1]

[1]Animal Genetics and Breeding Unit (AGBU)
University of New England, Armidale, 2351, NSW, Australia
vboerner@une.edu.au

Contents

List of Examples

1 General remarks

BESSiE is started from the command line via "BESSiE ParameterFileName". All run time options are given via the parameter file.

The current version of BESSiE is optimized for Intel architecture. Thus using BESSiE on AMD architecture will result in increased run time. Executables are available for Linux/Unix 64 bit environment only.

1.1 Disclaimer

BESSiE is under ongoing development and many of its features have been tested only a few times on a limited number of models and data sets. Thus, the users uses BESSiE completely on his/her own risk.

1.2 Conditions of use

BESSiE can be used by the scientific community free of charge, but users must credit BESSiE in any publications. Commercial users must obtain the explicit approval of the authors before using BESSiE.

1.3 Feedback and support

BESSiE comes without any guaranteed support and the user is strongly advised to study this manual thoroughly.

However, the user may provide feedback to the authors about the program functionality, possible aborts (segmentation faults), usability of output and comprehensibleness of the manual.

2 Program modes

BESSiE allows for three different modes, BLUP, GIBBS and CREATE.

Mode BLUP performs a best linear unbiased analysis of the specified linear mixed models given the observed data and supplied variances of random factors. The results are best linear unbiased estimations of levels of fixed factor and best linear unbiased prediction of levels of random factors.

Mode GIBBS performs a Gibbs sampling analysis of the specified linear mixed model given the observed data using supplied variances of random factors as starting values or a prior knowledge. The results for factor levels and for co-variances are draws from their conditional posterior distributions as given in Sorensen & Gianola (2002). In addition, for factors of which levels are modelled by genetic markers the results are draws from posterior distribution as specified in Meuwissen *et al.* (2001) ("BayesA" and "BayesB"), in Habier *et al.* (2011) ("BayesCπ) and in Erbe *et al.* (2012) ("BayesR").

Mode CREATE performs all necessary steps to perform the analysis (e.g. reading data, produce pre-analysis output etc.), but will not proceed further.

3 Models

BESSiE accommodates models fitting fixed and random factors in a uni-variate or multi-variate way.

3.1 Fixed factors.

Fixed factors may enter the model as integer dummy variables (e.g. contemporary group) or as real co-variables (e.g. age, weight). By default, BESSiE will model a fixed factor as dummy variable. This can be altered by setting the respective feature in the parameter file.

3.1.1 Integer dummy variables

By default BESSiE will cancel the last level of each dummy variable factor within trait except for that factor which has the fewest levels. However, this procedure does not guarantee that the resulting coefficient matrix of the mixed model equation is of full rank. While an insufficient rank will not cause BESSiE to terminate, BESSiE will also make no attempt to check for that. The user may set factor levels to zero to account for dependencies, but if BESSiE detects factor levels equal to zero where the corresponding observation is not missing, it will not perform the above cancellation procedure. Instead, BESSiE will assume that the user has sufficiently taken care for linear dependencies for that trait.

3.1.2 Real co-variables

Regression on a co-variables will be modelled as $\sum_{j=1}^{N}\sum_{l=1}^{M} b_{jl}x_{ij}^{l}$, where $b_{jl}x_{ij}^{l}$ are polynomials of order $l = 1, ..., M$ of the ith observation of the fixed co-variable $x_j, j = 1, ..., N$. The default is a first order polynomial, but a higher order can be given in the parameter file.

3.1.3 Fixed genetic groups

These factors are treated as a special case of real co-variables where factor levels are regression coefficients on genetic group proportions of those animals' genomes on which phenotypic observation where taken. Thus, the number of factor levels is equal to the number of genetic groups.

3.2 Random Factors.

BESSiE can model several different random factors. However, BESSiE assumes random factors with levels $i = 1, ..., N$ to follow $N(0, C\sigma^2)$, where C is a matrix of dimension $N \times N$ and σ^2 is a scalar variance. If the model suggest a co-variance between factors, BESSiE will assumes the n correlated factors, say $a_1, ..., a_n$ to follow $N[(0, ..., 0)', \Sigma \otimes C]$ where in this example Σ is the $n \times n$ co-variance matrix between $a_1, ..., a_n$.

BESSiE allows the user to specify the structure of the matrix C, which in turn allows BESSiE to exploit properties of C. However, some types of C may require additional input. In the following, possible random factors and their additional input requirements are described. Note that Σ is always the co-variance matrix between factors, and therefore will be a scalar if a factor has no covariance with any other factor in the model.

3.2.1 NRM factors

These factors $\sim N(0, \Sigma \otimes A)$ where A is a numerator relationship matrix derived from a provided pedigree, and is of dimension "number of individuals in the pedigree"×"number of individuals in the pedigree".

3.2.2 IDE factors

These factors $\sim N(0, \Sigma \otimes I)$ where I is an identity matrix of dimension "number of factor levels"×"number of factor levels". In addition, I may be replaced by an arbitrary matrix K which is provided in a file specified in the parameter file. Then, K must be symmetric, positive definite and of dimension equal to the number of levels of the related factors.

3.2.3 Random genetic group factors (NRMGG factors)

These factors are treated as a special case of IDE factors where factor levels are regression coefficients on genetic group proportions of those animals' genomes on which phenotypic

observation were taken. Thus, the number of factor levels is equal to the number of genetic groups.

3.2.4 GRM factors

These factors $\sim N(0, \Sigma \otimes G)$ where G is a genomic relationship matrix. G may be provided in a separate file or calculated from genotypes provided from a separate file. If G is calculated from genotypes it will be of dimension "number of genotypes"×"number of genotypes". If provided from file, its dimension must be equal to the number of factor levels.

3.2.5 SNP factors

These factors $\sim N(0, \Sigma \otimes I)$ where I is an identity matrix. However, they will only be modelled in conjunction with a matrix of marker genotypes (usually single nucleotide polymorphisms, SNP) supplied in a separate file. Moreover, solutions for levels of these factors will be obtained from specified algorithms which depend on the program mode. Contrarily to the factors above, the setting of Σ depends on the program mode.

3.2.5.1 SNP BLUP

SNP best linear unbiased prediction is only available in the BLUP mode, and Σ has the same structure as with any other aforementioned type of factor, but the co-variance ascribed to any single SNP will be Σ/n_{SNP} where n_{SNP} is the number of SNPs in the provided genotype file.

3.2.5.2 Bayesian alphabet

Currently available methods from the Bayesian alphabet are "BayesA" and "BayesB" as given by Meuwissen *et al.* (2001), "BayesCπ" as given by Habier *et al.* (2011) and "BayesR" as given by Erbe *et al.* (2012), and can only be used in the mode GIBBS. For all these methods, Σ will be a diagonal matrix with the number of rows/columns equal to $n_{SNP} \times n_{SNP}$ where n_{SNP} is the number of SNPs. The diagonal elements of Σ will be $\sigma^2_{SNP_i}$, which is the variance of SNP i. Thus, in a multi-variate setting no co-variance between SNP effects will be allowed for, and SNP variances will be generated according to the specified method where the method may differ between traits. An option in the parameter file allows these algorithms to run on the original marker matrix, or on a marker matrix which is normalised (centred and scaled), where centering and scaling is done for each SNP locus separately.

3.2.5.2.1 BayesA Method "BayesA" is characterised by an inverse chi-square prior distribution of SNP variances specified by scale and shape parameters of 0.002 and 4.012, respectively, as given by Meuwissen *et al.* (2001). However these parameters can be overridden via the parameter file.

3.2.5.2.2 BayesB The same as for "BayesA" applies to "BayesB", but the algorithm has two additional parameters, the number of Metropolis-Hastings cycles and the probability that an SNP has **NO** effect (π). The default setting for scale and shape are 0.0426 and 4.234 as given in Meuwissen *et al.* (2001), 0.95 for π and 100 for the number of Metropolis-Hastings cycles. As with "BayesA" all these parameters can be overridden via the parameter file.

3.2.5.2.3 BayesCπ Method "BayesCπ" assumes an equal variance for all SNPs in the model which is drawn from an inverse chi-square distribution with a fixed scale parameter and a shape parameter which is recalculated in every round of the sampling procedure according to the probability that an SNP was fitted (1-π) and the total variance explained by SNPs (Habier *et al.* 2011). New values for π, and its counterpart $1 - \pi$, are drawn from an beta distribution conditional on the number of SNPs included in the model in the previous sampling round. While a prior value of the total variance explained by the SNPs must be provided via the parameter file, the starting value for π is set by default to 0.95, as well as the default prior for α and β of the Beta distribution to (1,1) and the default shape parameter of the inverse chi-square distribution to 4.2. However, all default parameters can be overridden via the parameter file.

3.2.5.2.4 BayesR Method "BayesR" models SNP effects as coming from a mixture of normal distributions (Erbe *et al.* 2012). As for "BayesCπ" the total variance explained by SNPs must be provided via the parameter file. In addition, a file specifying the number of distributions and their variance in terms of proportion of the total variance must be provided. The number of distribution is up to the user and will be determined from the number of lines in that file. This file will also specify the initial probabilities that an SNP comes from one of the distributions, and a prior vector of pseudo-counts of SNPs assigned to the distributions.

3.2.6 Co-variances of random factors

A co-variance matrix of random factors must be supplied in a separate file. This matrix must contain a row/column for any factor of type NRM, GRM, IDE and SNP factors of type BLUP, but **NOT** for SNP factors modelled by the Bayesian alphabet. In addition a

row/column for each trait must be added to the co-variance matrix of random factors to allow for residual co-variances. Thus, even when no random factors are modelled a file with a co-variance matrix must be supplied which then contains the residual co-variances matrix only.

The matrix in that file must be quadratic but not necessarily symmetric because BESSiE will mirror the upper off-diagonal elements into the lower off-diagonal elements. No particular order of random factors or residuals must be considered as long as the resulting full matrix is invertible and positive definite. Values in the matrix must be space separated.

In mode BLUP, values in the co-variance matrix are used for solving the MME. In mode GIBBS values in the co-variance matrix are starting values for the sampling process, but can also be used as prior information. However, in mode GIBBS a blank line must be added below the co-variance matrix, and below this blank line BESSiE expects a vector of length equal to the number of columns of the co-variance matrix. This vector contains the prior degree of freedom (also called degree of belief), and its values must be ≥ 0. The highest value in the sub-vector related to a certain sub-matrix is used as prior degree of freedom for the sub-matrix. This sounds difficult but is easily understood:

BESSiE will try to detect independent sub-matrices in the co-variance matrix. If for example in a uni-variate analysis only a single random factor is modelled, there must be exactly two independent sub-matrices, one for the random factor and one for the residuals, where both sub-matrices will be of dimension 1×1. BESSiE will then use the column positions of the sub-matrices in the full co-variance matrix to separate the vectors of prior degrees of freedoms into sub-vectors related to the sub-matrices. If a sub-vector contains different values BESSiE will used the highest because only a single prior degree of freedom can be assigned to a sub-matrix. Thus, if a sub-vector contains only zeros, the variances in the related sub-matrix won't be used as prior knowledge.

3.3 Categorical observations

BESSiE allows for categorical phenotypic observations in uni-variate and multivariate analysis, thus, allowing for any combination of categorically and continuously measured traits. However, only when using mode GIBBS, estimated location and dispersion parameters are those conditional on the underlying scale (Albert & Chib 1993), whereas mode BLUP will provide location parameters conditional on the observed scale.

The number of categories is not limited, but must be at least two. For traits with two categories the residual variance will be fixed to one and the single threshold will be set to zero.

3.4 Weighted observations

BESSiE allows for analysis of weighted phenotypic observations, where weights must be supplied in the data file and their column position must be given in the parameter file. If weights are supplied, BESSiE will alter the assumed distribution of residuals from $N(0, I\sigma_e^2)$ to $N(0, D\sigma_e^2)$, where I is an identity matrix and D is a diagonal matrix of which diagonal elements are the weights provided.

In multivariate analysis, different weights may be supplied for every single trait. If weights a missing for a particular trait D will be replaced by I. However, BESSiE will alter the residual covariance between traits to $D_{1,2}\sigma_{e_1,e_2}$, where $D_{1,2}$ is the Cholesky factor of $D_1 D_2$, where D_1 and D_2 are diagonal matrices of weights of trait 1 and 2, respectively.

Weights cannot be applied to phenotypic observations of categorical scale.

3.5 Estimation of SNP effects in uni- and multivariate Gibbs sampling analysis

In mode GIBBS, BESSiE will estimates the effects of SNP factor levels (SNP effects) by methods "BayesA", "BayesB", "BayesCπ" or "BayesR". This involves a two-stage procedure: At stage 1 means of conditional posterior distributions of all factors except SNP factors are calculated, and replaced by draws from this distributions, where the response variables are the phenotypic observations pre-corrected for SNP effects. At stage 2, means of conditional posterior distributions of all SNP effects are calculated and replaced by draws from these distributions, where the response variables are phenotypic observations pre-corrected for all other factors. As an example consider a bi-variate model:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} + \begin{pmatrix} Z_1 M & 0 \\ 0 & Z_2 M \end{pmatrix} \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

where y, b, g and e are vectors of phenotypic observations, fixed effects and SNP effects, X, and Z are incidence matrices relating the effects to their respective observations, M is a matrix of marker genotypes of dimension "Number of genotypes"×"number of markers" and the subscripts are for trait 1 and 2. The equations to estimate $(b_1, b_2)'$ are

$$\left[\begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix}' R^{-1} \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \right] \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} =$$

$$\begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix}' R^{-1} \left[\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} - \begin{pmatrix} Z_1 M & 0 \\ 0 & Z_2 M \end{pmatrix} \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} \right]$$

where R is the residual co-variance matrix, whereas the equations to estimate $(g_1, g_2)'$ are

$$\left[\begin{pmatrix} Z_1M & 0 \\ 0 & Z_2M \end{pmatrix}' R^{-1} \begin{pmatrix} Z_1M & 0 \\ 0 & Z_2M \end{pmatrix} + \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}\right] \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} =$$
$$\begin{pmatrix} Z_1M & 0 \\ 0 & Z_2M \end{pmatrix}' R^{-1} \left[\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} - \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}\right]$$

where σ_1 and σ_2 are diagonal matrices of dimension "number of SNPs"$\times$ "number of SNPs" of which elements contain the variances of SNPs generated according to the Bayesian method specified for trait 1 and trait 2 in the parameter file.

Attention should be given to bi-variate models including an NRM factor (say u) where only the model for one trait (say trait 1) allows for the estimation of SNP effects. While all factor level effects will be estimated as outlined above, SNP effects will be estimated from:

$$\left[\ M'Z_1' \ (R^{-1})_{1,1} \ Z_1M \ + \ \sigma_1 \ \right] \ g_1 \ =$$
$$M'Z_1' \ \left[\ (R^{-1})_{1,1} \quad (R^{-1})_{1,2} \ \right] \left[\begin{pmatrix} y_1 \\ y_1 \end{pmatrix} - \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}\right]$$

While σ_{u_1,Z_1Mg_1} may be zero, σ_{u_2,Z_1Mg_1} is non-zero, but BESSiE will regard it as being zero.

4 Input files

4.1 Pedigree file

Records in the pedigree file must be of kind integer and placed in three space separated columns: individual, parent 1, parent 2. Individual ids must be numbered consecutively from one to the number of records, thus the highest individual id must be equal to the number of records in the file. Parents must precede progeny, and unknown parents must be coded with zero.

If a genetic group factor is modelled, the respective matrix of regressors will be derived from the pedigree. Thus, the pedigree must be expanded by phantom parents which have to appear at the beginning of the pedigree, and a fourth column assigning phantom parents to genetic groups (see Example 4.1)

4.2 Data file

The data file is structured in rows and space delimited columns. Each column contains either a vector of phenotypic observations or a the coding of a model factor or the observation of a co-variable. In multi-variate analysis, each phenotypic observation vector has its own column as each model factor/co-variable has. For example, the file providing the data for the model

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}, \tag{4.1}$$

, where the model for each trait has two fixed factors, may contain six columns: y_1, y_2, X_{1_1}, X_{1_2}, X_{2_1}, X_{2_2}. If fixed factors are equal for both traits, the number of columns can be reduced to four by using the columns for the fixed factors for both traits.

If the model is expanded by an animal factor, e.g.

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} + \begin{pmatrix} Z_1 & 0 \\ 0 & Z_2 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}, \tag{4.2}$$

Example 4.1: Including genetic groups in the pedigree

In the pedigree below, all males with unknown parents shall be assigned to genetic group one, whereas all females with unknown parents shall be assigned to genetic group two. Because normal pedigree rules apply (e.g. a sire ids cannot occur in the dam column), four phantom parents must be introduced, two as sire and dam of the females (3 and 4) and two as the sire and dam of the males (1 and 2). Then, these phantom parents are assigned to genetic groups via the fourth column in the pedigree file.

Original: without genetic groups

1	0	0
2	0	0
3	0	0
4	0	0
5	1	2
6	3	4

Expanded: with genetic groups

1	0	0	1
2	0	0	1
3	0	0	2
4	0	0	2
5	1	2	0
6	1	2	0
7	3	4	0
8	3	4	0
9	5	6	0
10	7	8	0

The number of phantom parents must be provided to BESSiE via the parameter file. Then, BESSiE will set up the matrix of regressors and regenerate the original pedigree without genetic groups.

the data file should contain these eight columns: y_1, y_2, X_{1_1}, X_{1_2}, X_{2_1}, X_{2_2}, Z_1, Z_2. Since values in columns Z_1 and Z_2 are always equal (e.g. either all individuals have records on both traits or records are coded as missing), the number of columns can be reduced to seven by dropping either column Z_1 or Z_2. The same holds if the models of both traits fit equal co-variables, e.g. age.

The number of rows and columns in the data file need not be given to BESSiE. Instead, BESSiE will determine the number of rows and columns itself. For the determination of the number of columns the program evaluates the first valid (not commented) line in the data file. However, the data file may contain more columns than those related to the analysis.

4.2.1 Factor coding.

The coding of factors depends on its' type, Note that BESSiE is **NOT** a re-numbering program. Thus, all factors must be coded numerically in a reasonable manner (see below).

4.2.1.1 Fixed dummy factor.

Levels of fixed dummy factors must be coded sequentially from 1 to N, where N is the number of levels of a fixed factor. The user may delete levels by setting them to zero to account for linear dependencies between factors. However, the coding of the remaining factor levels must still yield a sequence. Factor levels associated with missing observations must be zero.

4.2.1.2 Fixed co-variables.

For a fixed co-variable x, the program fits the model $b_1x+b_2x^2+.....b_nx^n$. Thus, co-variable observations equal to zero will be regarded as valid unless the corresponding phenotypic observation is missing.

4.2.1.3 Random or Fixed genetic groups.

Regression on genetic group genome proportion requires the initial coding of these factors being related to the pedigree coding. Thus the same requirements apply as for the coding of NRM factors (see 4.2.1.4).

4.2.1.4 NRM factors

Levels of these factors (NRM factors) must be consistent with the individuals' ids in the pedigree file. Thus, factor levels must be $\leq$ than the greatest id and $\geq$ zero. Note that when modelling genetic groups, the numbering must be consistent with the pedigree without the genetic groups because BESSiE regenerates and uses that pedigree.

4.2.1.5 IDE factors

For this type of factor, e.g. common environmental effects or maternal environmental effects, levels of each factor must be coded sequentially from 1 to N, where N is the number of factor levels. Coding of levels associated with missing observations may be set to zero or to the observed level. However, when modelling co-variance between IDE factors, the concatenation of all vectors of observed factor levels with non-zero co-variance, sorted and duplicates removed, must yield a sequence from 1 to N (see Example 4.2).

While the "pure" IDE setting in the parameter file assumes IDE factors to have a co-variance structure of $\Sigma \otimes I$, where I is in identity matrix, I may be replaced by an arbitrary squared, symmetric and positive definite matrix K provided in a file. Note that K must of dimension equal to the number of factor levels where the zero level does not count (see 3.2.2).

4.2.1.6 GRM factors

The smallest possible coding for levels of these factors is 1, the greatest equal to the dimension of the genomic relationship matrix (GRM). If the GRM is calculated from the SNP genotype file, the latter dimension is equal to the number of lines in the genotype file. If the GRM is provided from a file, the factor levels must correspond to the dimension of the GRM. Thus, the user may provide more genotypes than animals with phenotypic observations or a GRM of larger dimension, but not less genotypes/a smaller GRM than individuals with phenotypes. Note that the program identifies the genotypes/lines in the GRM related to an animal by the level coding. Thus, an individual which has the level coding "1" in the respective column of the data file will be assumed to have the genotype which occurs in line 1 of the genotype file.

4.2.1.7 SNP factors.

For SNP factors modelled according to the Bayesian alphabet or as BLUP_SNP, the same coding rules apply as those for a factor with GRM co-variance structure.

4.2.2 Factor nesting.

The program does not accommodate for factor nesting by any special source code. However, nesting of fixed factors can be achieved via appropriate coding of the nested factor. For co-variables nesting can be achieved via multiple columns of the particular co-variable and then setting in each column the co-variable observations to zero except those observed within the respective level of the nesting factor.

4.2.3 Data file headers and comments.

It is possible to comment whole lines in the data file by putting an exclamation mark at the beginning of the line. This allows to use headers in the data file and to exclude observations.

4.2.4 Missing records.

In a bi-variate analysis, individuals may have a record for trait 1, but not for trait 2. The missing records in trait 2 must be coded with "-1234567890", and the levels of all fixed factors and observations of all fixed co-variables related to that missing observation must be set to zero. However, random factors associated with a missing record can have a valid coding according to the rules lined out above (e.g. the animal id), or can be set to zero (see Example 4.2).

4.3 Genotype file

The genotype file has one genotype per line where SNP genotypes must be coded with "0", "1", or "2", and genotypes of different SNP are NOT space separated. BESSiE assumes that the ids of genotyped individuals are running from 1 to the number of lines in the genotype file. As with the data file, it will determine the number of genotypes and the number of SNPs itself, where for the latter it will evaluate the first line of the file. Both numbers are then written to the log file. If a factor that makes use of genotypes is included in the model, the consistency between the factor levels in the related column of the data file and the number of lines in the genotype file will be checked. Thus, a level of the related factor greater than the number of genotypes will lead the program to terminate. However, BESSiE accommodates for supplying more genotypes than individuals with observations.

Example 4.2: Data file for a bi-variate model with missing records

For example consider the bi-variate model

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} + \begin{pmatrix} Z_{1d} & 0 & Z_{1m} \\ 0 & Z_{2d} & 0 \end{pmatrix} \begin{pmatrix} u_{1d} \\ u_{2d} \\ u_{1m} \end{pmatrix} + \begin{pmatrix} Z_{1c} & 0 \\ 0 & Z_{2c} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix},$$

where subscripts d and m distinguish between direct genetic and maternal genetic NRM factors and c is the maternal common environmental IDE factor. Further assume that there is no overlap between individuals recorded for y_1 and y_2, and that all dams of individuals recorded for y_2 are unknown. Then, the y_1 records of individuals recorded for y_2 will be modelled as missing, and the dam ids in the respective column of the data file must be set to zero because these dams are unknown. However, even if dams are known it is not necessary to use the real ids instead of zero. The same applies to the IDE factor coding.
A possible data file for such a structure would look like:

!y_1	y_2	u_d	u_m	c_1
$y_{1,1}$	-1234567890	20	13	1
$y_{1,2}$	-1234567890	21	13	1
$y_{1,3}$	-1234567890	23	13	1
$y_{1,4}$	-1234567890	24	15	2
$y_{1,5}$	-1234567890	25	15	2
-1234567890	$y_{2,1}$	8	0	0
-1234567890	$y_{2,2}$	9	0	0
-1234567890	$y_{2,2}$	10	0	0

Note that the first line is the header which is exempted from reading due to the exclamation mark at the first position.

4.4 Genomic relationship matrix file

When modelling a GRM factor, the GRM may provided in a file. BESSiE will assume the GRM being stored in that file as full squared matrix, and its dimensions will be determined while reading. However, BESSiE will mirror the upper off-diagonal elements into the lower off-diagonal elements to ensure symmetry.

The numbering of individuals related to each row/column of the GRM is assumed to be sequential from 1 to the number of rows/columns. The dimension of the GRM and the levels of those factors having a GRM co-variance structure must be consistent.

4.5 External matrix K file

When modelling an IDE factor $N(0, \Sigma \otimes K)$ (see 3.2.2), matrix K may provided in a file. BESSiE will assume K being stored in that file as full squared matrix, and its dimensions will be determined while reading. However, BESSiE will mirror the upper off-diagonal elements into the lower off-diagonal elements to ensure symmetry. The dimension of K and the levels of those factors having the above co-variance structure must be consistent.

4.6 Co-variance file

For uni- or multi-variate BLUP or Gibbs sampling analysis a file with a squared co-variance matrix must be supplied containing non-zero elements at least in the diagonal and in the upper triangular. All values must be space separated. This even applies when only fixed factors are modelled. The dimension of the matrix depends on the number of random effects across the models of all traits in the analysis. However, only NRM, GRM and IDE factors require a record in the co-variance matrix, but SNP factors only in mode BLUP. In addition to the number of random effects, the dimension of the matrix is increased by the number of traits in the analysis because every trait requires a residual variance. Since the row/column position of an NRM/GRM/IDE/SNP and residual effect in the co-variance matrix is given in the parameter file, rows/columns can be shuffled (see Example 4.3).

However, BESSiE will check whether the matrix provided is invertible and positive definite.

For Gibbs sampling analysis, the file must also contain a line with prior degrees of freedom. This line must be placed below the co-variance matrix, separated from it by a blank line, and it must have the same number of records, separated by space, as the co-variance matrix has columns. Since for each sub-covariance matrix only one prior degree of freedom can

Example 4.3: The co-variance matrix file

Assume a bi-variate analysis with the model:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} + \begin{pmatrix} Z_1 & 0 \\ 0 & Z_2 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix},$$

and with several individuals having records for both traits. A file providing the co-variance matrix would look like A, B or C:

A

σ^2_{u1}	$\sigma_{u1,u2}$	0	0
$\sigma_{u1,u2}$	σ^2_{u2}	0	0
0	0	σ^2_{e1}	$\sigma_{e1,e2}$
0	0	$\sigma_{e1,e2}$	σ^2_{e2}

B

σ^2_{u1}	0	$\sigma_{u1,u2}$	0
0	σ^2_{e1}	0	$\sigma_{e1,e2}$
$\sigma_{u1,u2}$	0	σ^2_{u2}	0
0	$\sigma_{e1,e2}$	0	σ^2_{e2}

C

σ^2_{u1}	$\sigma_{u1,u2}$	0	0
0	σ^2_{u2}	0	0
0	0	σ^2_{e1}	$\sigma_{e1,e2}$
0	0	0	σ^2_{e2}

be assigned, BESSiE will use the largest prior value in the related sub-vector (see Example 4.4).

The values in the co-variance matrix govern the way BESSiE solves the mixed model equations and samples new co-variances. Setting off-diagonal values to zero may lead to co-variances between effects neither be modelled nor sampled (see Example 4.5).

4.7 Distribution file for BayesR

Modelling an effect according to the BayesR algorithm described in Erbe *et al.* (2012) requires the specification of the variance of a number of normal distributions. Moreover, a prior vector of pseudo counts of SNPs in each distribution and an initial vector of probabilities that a the effect of a certain SNP comes from a certain distribution are necessary. These parameters must be supplied to BESSiE via a distribution file. The number of distributions is defined by the number of lines in that file.

However, the file must have exactly three space separated columns. The first column contains a proportionality factor which is used to get the variance of this distribution by multiplying the factor with the variance maximum explainable by the modelled factor (supplied by the parameter file). Values in this column must not exceed 1. The second column contains pseudo counts of the number of SNPs in each distribution, which is used as prior knowledge when generating a draw from the Dirichlet distribution. Values in this column must not be negative. The third column contains the starting values for the probability that a SNPs comes from a certain distribution, where the sum of the elements in this column

Example 4.4: The prior degrees of freedom

In the co-variance matrix below two independent sub-matrices can be identified. One sub-matrix occupies the first and third row/column of the matrix, e.g. the co-variance matrix of animal effects, the other the second and fourth, e.g. the co-variance matrix of residual effects. Below the co-variance matrix, separated from it by a blank line, is the vector of prior degrees of freedom. BESSiE will assign the first and third value in that vector to the animal effects matrix, the second and fourth vector value to the residual effects matrix. Thus, the two sub-vectors are [100 0] and [0 50]. Since every sub-matrix can have only one prior degree of freedom, BESSiE will make use of the largest value in each sub-vector, sampling the animal matrix with a prior degree of freedom of 100, the residual matrix with 50.

$$\begin{array}{cccc} \sigma^2_{u1} & 0 & \sigma_{u1,u2} & 0 \\ 0 & \sigma^2_{e1} & 0 & \sigma_{e1,e2} \\ \sigma_{u1,u2} & 0 & \sigma^2_{u2} & 0 \\ 0 & \sigma_{e1,e2} & 0 & \sigma^2_{e2} \end{array}$$

$$\begin{array}{cccc} 100 & 0 & 0 & 50 \end{array}$$

Example 4.5: Interaction between the co-variance matrix and the sampling procedure in mode GIBBS

Assume a tri-variate analysis with the model

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} X_1 & 0 & 0 \\ 0 & X_2 & 0 \\ 0 & 0 & X_3 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} + \begin{pmatrix} Z_1 & 0 & 0 \\ 0 & Z_2 & 0 \\ 0 & 0 & Z_3 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix},$$

and with several individuals having records for all three traits. A file providing the co-variance matrix (omitting the vector of prior degrees of freedom) would look like:

$$\begin{array}{cccccc} \sigma^2_{u1} & \sigma_{u1,u2} & \sigma_{u1,u3} & 0 & 0 & 0 \\ 0 & \sigma^2_{u2} & \sigma_{u2,u3} & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{u3} & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{e1} & \sigma_{e1,e2} & \sigma_{e1,e3} \\ 0 & 0 & 0 & 0 & \sigma^2_{e2} & \sigma_{e2,e3} \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{e3} \end{array}$$

Setting $\sigma_{e1,e3}$ to zero will **NOT** result in this covariance not being regarded/sampled because y_1 is not independent of y_3 due to $\sigma_{e1,e2}$ and $\sigma_{e2,e3}$ being unequal to zero. However, if $\sigma_{e1,e2}$ is set to zero too, $\sigma_{e1,e2}$ and $\sigma_{e1,e3}$ will neither be regarded nor sampled. Moreover, setting $\sigma_{u1,u2}$ and $\sigma_{u1,u3}$ to zero as well will result in two independent analysis running at the same time, a uni-variate regarding only the model of y_1, and a bi-variate regarding only the model of (y_2, y_3).

must be equal to 1 (see Example 4.6).

Example 4.6: Distribution file required for method "BayesR"

The example file below will result in four distributions with variances $0\times\sigma^2$, $0.0001\times\sigma^2$, $0.001\times\sigma^2$ and $0.01\times\sigma^2$, where σ^2 is the variance provided by the parameter file. The pseudo count in the second column will result in a uniform prior. The values in third column provide initial probabilities that SNPs come from certain distributions. Thus, in the first round of the sampler, the probability that a SNP has a zero variance and, therefore, a zero effect is 0.9. A stronger prior for pseudo counts can be imposed by changing the values in the second column, for example from (1,1,1,1) to (40000,500,100,10). The sum over the value in the second column may not be related to the number of fitted SNPs.

```
0        1   0.9
0.0001   1   0.05
0.001    1   0.04
0.01     1   0.01
```

5 Parameter file

BESSiE is started from the command line via "BESSiE ParameterFileName". All run time options are given via the parameter file.

5.1 General description

The parameter file contains only four structural elements: **objects, object spaces, object features and comments**. It is essential to understand the following part to fully exploit the flexibility of the program in terms of modes, models, input and output.

5.1.1 Comments

Comments start with an exclamation mark which can be placed anywhere in a text line. However, all text between the exclamation mark and the end of the line will be regarded as comment.

5.1.2 Objects I

An object is initialised with an initialiser and is finalised with a finaliser. This concept is best understood by Example 5.1.

Example 5.1: Initialiser and Finaliser of an object

```
1 BEGIN PARAMETERFILE !initaliser for the object "PARAMETERFILE"
2   ......
3   ......
4 END PARAMETERFILE !finaliser for the object "PARAMETERFILE"
```

Initialiser have the structure BEGIN OBJECTNAME, finaliser END OBJECTNAME. The initialiser must be placed always in a row which is located above the finaliser. The initialiser tells BESSiE where to start when searching for certain object features, the finaliser where

to end. Thus, in the following "reading an object" by BESSiE means the reading and evaluation of the lines between the initialiser and finaliser. When the parameter file governs the BESSiE to read a certain object, it will search for its initialiser and finaliser. If one cannot be found, the BESSiE will terminate with an error message. While BEGIN and END must always be written in capital letters, the spelling of the OBJECTNAME depends whether it can be set by the user or is hard-coded in the program.

Since PARAMETERFILE itself is an object, initialised with BEGIN PARAMETERFILE and finalised with END PARAMETERFILE,it can be placed into an arbitrary text file allowing the user to include text (e.g. job descriptions) above the initaliser or below the finaliser without commenting it (see Example 5.2).

Example 5.2: Object property of the parameter file itself

```
1 Analysis mice weights from 14/01/2014
2 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
3  ....
4  ....
5 END PARAMETERFILE !finaliser for the object PARAMETERFILE
6 The analysis was actually crap. Next trial.
```

Thus, although the first and the last line is not commented, it will not be regarded as a part of the object PARAMETERFILE.

5.1.3 Object space I

The object space is the space which starts below the initaliser and ends above the finaliser. The object space hosts all the information related to the object. In addition, the object space also hosts the initaliser/finaliser of nested objects and their object spaces. In Example 5.2 the space of object PARAMETERFILE expands from line 3 to line 4.

5.1.4 Object Features I

Object Features provide necessary information about an object. They are placed in the space of the related object and have the structure KEYWORD: VALUE or KEYWORD: variable. KEYWORD must always be written in capital letters, followed by a colon which is used by BESSiE to separate KEYWORD from VALUE or variable, respectively (see Example 5.3).

In some cases, VALUE and variable may contain a concatenation of words or number which can have different separators (see Example 5.4).

Example 5.3: Object features

```
1 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
2   JOBNAME: Test !feature of object PARAMETERFILE
3   INPUT: PEDIGREE !feature of object PARAMETERFILE
4 END PARAMETERFILE !finaliser for the object PARAMETERFILE
```

Here, JOBNAME: TEST (line 2) and INPUT: PEDIGREE (line 3) are object features which belong to the object PARAMETERFILE.

Example 5.4: Object features

```
1 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
2   JOBNAME: Test !feature of object PARAMETERFILE
3   INPUT: PEDIGREE,TRAITS !feature of object PARAMETERFILE
4 END PARAMETERFILE !finaliser for the object PARAMETERFILE
```

Here, INPUT: PEDIGREE,TRAITS (line 3) in an object feature which belong to the object PARAMETERFILE of which VALUE is a concatenation of PEDIGREE and TRAITS, separated by a comma.

5.1.4.1 Variable object features.

In Example 5.3 JOBNAME: TEST is a variable object feature which has the structure KEYWORD VARIABLE. The type of VARIABLE is hard-coded in the program (e.g. character string, real number or integer), but its content is set by the user. If the VARIABLE is of type character, the spelling is up to the user. However, it then might be reused as an OBJECTNAME. Then the spelling of the OBJECTNAME must match that of VARIABLE (see Example 5.8).

5.1.4.2 Value object features.

In Example 5.3 INPUT: PEDIGREE is a value object feature which has the structure KEYWORD: VALUE. The possible values of VALUE and their type (character, logical etc.) are hard-coded in the program, and if VALUE is of type character all letters must be capitalised and the spelling must be correct. Thus, in the latter example, spelling like "pedigree", "Pedigree", "pedigre" etc. will lead BESSiE to terminate.

5.1.5 Objects II: nesting.

All objects in the parameter file are nested within a host object except the main host object PARAMETERFILE. Moreover, some objects may be nested within already nested host objects. An example:

Example 5.5: Nested objects

```
1 Analysis mice heights 14/09/2014
2 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
3   JOBNAME: Test !variable feature of object PARAMETERFILE
4   INPUT: PEDIGREE,TRAITS !value feature of object PARAMETERFILE
5   BEGIN PEDIGREE !initaliser for the nested object PEDIGREE
6     ......
7   END PEDIGREE !finaliser for the nested object PEDIGREE
8   BEGIN TRAITS !initaliser for the nested object TRAITS
9     NAMES: height !variable feature for object TRAITS
10    BEGIN height !initaliser for the nested object height
11      ......
12    END height !finaliser for the nested object height
13  END TRAITS !finaliser for the nested object TRAITS
14 END PARAMETERFILE !finaliser for the object PARAMETERFILE
15 This analysis failed.
```

Here the objects PEDIGREE (line 5 - line 7) and TRAITS (line 8 - line 13) are nested within the object PARAMETERFILE, whereas the trait "height" is nested as object HEIGHT (line 10 - line 12) within the object TRAITS. Note that the spelling of PEDIGREE and TRAITS is hard-coded, while the spelling of HEIGHT is user defined via the variable feature NAMES (line 9) of object TRAITS.

5.1.6 Object space II.

As shown in Example 5.5, the space of a certain object contains features of this object as well as spaces and features of other objects. However, when searching for features of a certain object, the program will search only in the related object space, not in the space of nested objects. An example:

5.1.7 Objects III: compulsory and conditional objects

Objects are either compulsory or conditional. Compulsory objects must have an initialiser/finaliser in the parameter file. The only totally compulsory object is PARAMETERFILE. The reading of conditional objects is triggered by features of the host object, where names of these conditional objects are derived from VARIABLE or VALUE of these features. Thus,

Example 5.6: Wrong placement of object features

```
1 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
2   INPUT: PEDIGREE,TRAITS !value feature of object PARAMETERFILE
3   BEGIN PEDIGREE !initaliser for the nested object PEDIGREE
4     JOBNAME: Test !variable feature of object PARAMETERFILE
5     ......
6   END PEDIGREE !finaliser for the nested object PEDIGREE
7 END PARAMETERFILE !finaliser for the object PARAMETERFILE
```

The object feature JOBNAME: TEST (line 4), which belongs to the object PARAMETERFILE, is wrongly placed in space of object PEDIGREE (line 3 - line 6). The program will search for features related to the object PARAMETERFILE in its object space, but will skip the PEDIGREE space, which occupies the lines 4 and 5. Thus, the space of object PARAMETERFILE left for searching for features of this object consists only of a single line, which is line 2. Since the program will not find the feature JOBNAME: TEST it will terminate with an error message.

for BESSiE this objects exists conditional on a triggering feature of the host object (see Example 5.7)

Example 5.7: Reading of conditional objects

```
1 BEGIN PARAMETERFILE !initaliser for the compulsory object PARAMETERFILE
2   JOBNAME: Test !variable feature of object PARAMETERFILE
3   INPUT: PEDIGREE !value feature of object PARAMETERFILE
4   BEGIN PEDIGREE !initaliser for the nested conditional object PEDIGREE
5     ......
6   END PEDIGREE !finaliser for the nested conditional object PEDIGREE
7 END PARAMETERFILE !finaliser for the compulsory object PARAMETERFILE
```

The search and reading of the nested object with name PEDIGREE is triggered by the value object feature INPUT: PEDIGREE (line 3). Thus, the value of feature INPUT, which is PEDIGREE, triggers the search for the initialiser BEGIN PEDIGREE and the finaliser END PEDIGREE, and the subsequent reading and evaluation of object features in the space of object PEDIGREE.

If the search for the initialiser/finaliser is triggered by a variable object feature, VARIABLE is re-used for the object name in the same way as VALUE of feature INPUT in Example 5.7. Remember that the spelling of variables of type character is up to the user. Thus, the spelling must be kept when naming the object (see Example 5.8).

The reading of a conditional nested object can also be triggered by the reading of the host object without evaluating any feature of the host object. This is the case for objects

Example 5.8: Reading of conditional objects triggered by variable object features

```
1 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
2   JOBNAME: Test !variable feature of object PARAMETERFILE
3   INPUT: PEDIGREE,TRAITS !value feature of object PARAMETERFILE
4   BEGIN PEDIGREE !initaliser for the nested object PEDIGREE
5     ......
6   END PEDIGREE !finaliser for the nested object PEDIGREE
7   BEGIN TRAITS !initaliser for the nested object TRAITS
8     NAMES: height !variable feature for object TRAITS
9     BEGIN height !initaliser for the nested object height
10      ......
11    END height !finaliser for the nested object height
12  END TRAITS !finaliser for the nested object TRAITS
13 END PARAMETERFILE !finaliser for the object PARAMETERFILE
```

As in Example 5.7, the reading of PEDIGREE is triggered by the value object feature INPUT: PEDIGREE,TRAITS (line 3). Moreover, INPUT triggers also the reading of the object TRAITS. The variable object feature NAMES in the TRAITS space (line 8) triggers the reading of the object height. Thus, the program searches for BEGIN height/END height in the space of TRAITS (which expands between line 8 and line 11), and any deviation from the spelling of height defined in NAMES: height will lead to an error message and the termination of the program (e.g. BEGIN Height etc.).

which become immediately necessary if the host object is read. The name of these objects is hard-coded. Currently, only one object belongs to this class: COVAR (see Example 5.9).

5.1.8 Object Features II: compulsory and optional object features.

In addition to the above outlined classification of object features into variable and value object features, object features can be compulsory or optional. Compulsory object features must be given in the space of the related object, and their absence will cause a program termination with a related error message. Optional object features can be given but have a hard-coded default value. Thus, if an optional object feature is missing, the program will continue to run but will use the default variable/value of the missing object feature (see Example 5.10).

5.1.9 Objects III: optional objects.

So far we have only seen objects which have at least one compulsory feature. Thus, if these objects are missing, BESSiE will terminate with a related error message. However, if all object features of an object are optional, the object itself will become optional. If the creation of such an object is triggered by a feature of the host object, BESSiE will search

Example 5.9: Reading of conditional objects triggered by the creation of a host object

```
1 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
2   JOBNAME: Test !variable feature of object PARAMETERFILE
3   INPUT: TRAITS !value feature of object PARAMETERFILE
4   BEGIN TRAITS !initaliser for the nested object TRAITS
5     NAMES: height !variable feature for object TRAITS
6     BEGIN height !initaliser for the nested object height
7       .......
8     END height !finaliser for the nested object height
9     BEGIN COVAR !initaliser for the nested object COVAR
10      FILE: Test.txt !variable feature of object COVAR
11    END COVAR !finaliser for the nested object COVAR
12  END TRAITS !finaliser for the nested object TRAITS
13 END PARAMETERFILE !finaliser for the object PARAMETERFILE
```

The reading of object COVAR (line 9 - line 11) is triggered by the reading of object TRAITS (line 4 - line 12) without the evaluation any feature of object TRAITS, because without a co-variance matrix for the trait "height" neither a BLUP analysis nor a Gibbs sampling analysis would be possible.

for the initialiser/finaliser of this object. If they are missing BESSiE will continue to run with the default values (see Example 5.11).

5.1.10 Object Features III: conditional object features.

A feature of an object can be conditional on other features of the same object. Thus, these features may become necessary due to reading other features of the same object. However, if these conditional features have default values, they are optional, otherwise compulsory (see Example 5.12).

Example 5.10: Optional and compulsory object features

```
1 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
2   MODE: BLUP !compulsory value feature of object PARAMETERFILE
3   JOBNAME: Test !compulsory variable feature of object PARAMETERFILE
4   INPUT: PEDIGREE,TRAITS !compulsory value feature of object PARAMETERFILE
5   BEGIN PEDIGREE !initaliser for the nested object PEDIGREE
6     ......
7   END PEDIGREE !finaliser for the nested object PEDIGREE
8   BEGIN TRAITS !initaliser for the nested object TRAITS
9     NAMES: height !compulsory variable feature for object TRAITS
10    BEGIN height !initaliser for the nested object height
11      MODEL: mean+weight+animal !compulsory variable feature of object height
12      BEGIN mean !initaliser for the nested object mean
13        .......
14      END mean !finaliser for the nested object mean
15      BEGIN weight !initaliser for the nested object weight
16        TYPE: FIX !compulsory value feature of object weight
17        COV: T !optional variable feature of object weight
18        EXP: 3 !optional variable feature of object weight
19        ......
20      END weight !finaliser for the nested object weight
21      BEGIN animal !initaliser for the nested object animal
22        ......
23      END animal !finaliser for the nested object animal
24    END height !finaliser for the nested object height
25  END TRAITS !finaliser for the nested object TRAITS
26 END PARAMETERFILE !finaliser for the object PARAMETERFILE
```

In the above example INPUT (line 4) is a compulsory feature of PARAMETERFILE, NAMES (line 9) is an compulsory feature of TRAITS , MODEL (line 11) is a compulsory feature of HEIGHT (because a trait without a model wouldn't make sense), and TYPE (line 16) is a compulsory feature of WEIGHT. However, for a fixed factor (because TYPE: FIX) BESSiE assumes by default regression on dummy variables, which is equal to setting the feature COV: F (line 17). Thus, COV has a default value, is therefore optional, and may be omitted from the parameter file. But if the fixed factor should be modelled as linear co-variable, COV: T must be set. If COV: T is set, BESSiE assumes by default linear regression (EXP: 1, line 18). Thus, EXP has a default value of 1 and may be omitted. If BESSiE should model a higher order polynomial, EXP must occur in the parameter file and set to the desired value.

Example 5.11: Optional objects

```
1 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
2   MODE: BLUP !compulsory value feature of object PARAMETERFILE
3   JOBNAME: Test !compulsory variable feature of object PARAMETERFILE
4   INPUT: TRAITS !compulsory value feature of object PARAMETERFILE
5   BEGIN TRAITS !initaliser for the nested object TRAITS
6     ......
7   END TRAITS !finaliser for the nested object TRAITS
8   BEGIN BLUP !initaliser for the object BLUP
9     MMESOLVER: PCG !optional variable feature of object BLUP
10  END BLUP !finaliser for the object BLUP
11 END PARAMETERFILE !finaliser for the object PARAMETERFILE
```

In the above example the reading of object BLUP (line 8 - line 10) is triggered by the compulsory value feature MODE (line 2) of the host object PARAMETERFILE. However, all features of object BLUP are optional. Thus, omitting the object space (line 9) as well as the initialiser and finaliser (line 8 and line 10) of object BLUP would still yield a valid parameter file, and BESSiE will use the default values for all features of object BLUP.

Example 5.12: Optional and compulsory object features

```
1 BEGIN PARAMETERFILE !initaliser for the object PARAMETERFILE
2   MODE: BLUP !compulsory value feature of object PARAMETERFILE
3   JOBNAME: Test !compulsory variable feature of object PARAMETERFILE
4   INPUT: PEDIGREE,TRAITS !compulsory value feature of object PARAMETERFILE
5   BEGIN PEDIGREE !initaliser for the nested object PEDIGREE
6     ......
7   END PEDIGREE !finaliser for the nested object PEDIGREE
8   BEGIN TRAITS !initaliser for the nested object TRAITS
9     NAMES: height !compulsory variable feature for object TRAITS
10    BEGIN height !initaliser for the nested object height
11      MODEL: mean+weight+animal !compulsory variable feature of object height
12      ......
13      BEGIN mean !initaliser for the nested object mean
14        .......
15      END mean !finaliser for the nested object mean
16      BEGIN weight !initaliser for the nested object weight
17        TYPE: FIX !compulsory value feature of object weight
18        COV: T !conditional optional variable feature of object weight
19        EXP: 3 !conditional optional variable feature of object weight
20        ......
21      END weight !finaliser for the nested object weight
22      BEGIN animal !initaliser for the nested object animal
23        TYPE: RAN !compulsory value feature of object animal
24        STRUCTURE: NRM !conditional compulsory variable feature of object animal
25        COVARPOS: 1 !conditional compulsory variable feature of object animal
26        ......
27      END animal !finaliser for the nested object animal
28    END height !finaliser for the nested object height
29  END TRAITS !finaliser for the nested object TRAITS
30 END PARAMETERFILE !finaliser for the object PARAMETERFILE
```

As already lined out in Example 5.11, feature COV (line 18) and feature EXP (line 19) are optional features of object WEIGHT. However, in addition they are also conditional features such that BESSiE will only recognise them if the value of feature TYPE has been set to FIX (line 17). Since COV and EXP both have default values, the can be regarded as conditional optional features. For object ANIMAL (line 22 - 27), the object space contains three features. While feature TYPE (line 23) is compulsory, STRUCTURE (line 24) is a conditional feature which will be read by BESSiE only if the value of TYPE was RAN. Since STRUCTURE has no default value, it is a conditional compulsory feature. The same applies to COVARPOS (line 25), which is conditional on the value of STRUCTURE, and has no default value.

5.2 Description of parameter file objects

For the following chapter the reader should be familiar with the terminology introduced in the last chapter:

- Initialiser
- Finaliser
- Object
- Host object
- Nested object
- Object space
- Object feature
- Compulsory object
- Conditional object
- Optional object
- Value object feature
- Variable object feature
- Compulsory feature
- Conditional feature
- Optional feature

In the following, **U.d.** will abbreviate "user defined". The sub-headings will have the structure "Object OBJECTNAME". OBJECTNAME is hard-coded, thus, spelling must be abide by. However, for the three exemptions

- Object "U.D. EXTERNAL MATRIX"
- Object "U.D. SINGLE TRAIT"
- Object "U.D. SINGLE FACTOR"

"U.D. ....." means that OBJECTNAME will be derived from a variable feature of the host object. If you are unfamiliar with this please go back to 5.1.7.

5.2.1 Object PARAMETERFILE

Object PARAMETERFILE is the only compulsory object in the parameter file. When reading the parameter file, BESSiE will only evaluate those file lines placed between the initialiser and the finaliser. Thus, above and below, respectively, arbitrary text can be written without the need of commenting it out. Several object may be placed in the same file where BESSiE will regard only the first. However, by commenting the initialiser/finaliser of the first object, objects below this can be supplied to the program.

5.2.1.1 Features

PARAMETERFILE has the following features:

- JOBNAME: optional variable feature, value is a u.d. job description. If provided, all output file names will have a prefix equal to the variable except the logfile.
- SEED: optional variable feature, value is numeric and provides a seed for the random number generator. Default: 12345
- INPUT: compulsory value feature, value is any combination of PEDIGREE, GENOTYPES, TRAITS and GRM, where a comma must be used as separator.
- MODE: compulsory value feature, value is either GIBBS, BLUP or CREATE

A note on feature INPUT

Triggers the reading of other objects may be necessary for the analysis, where the object names are derived from the feature value using a comma as a separator. Thus, running a Gibbs sampling analysis without a random NRM factor may not require PEDIGREE to be among the comma-separated value list. However, missing PEDIGREE although the model requires it will lead BESSiE to terminate. The same applies to GENOTYPES and GRM, but BESSiE will generally terminate if TRAITS is missing.

5.2.2 Object GIBBSSAMPLER

Object GIBBSSAMPLER is conditional on the value of feature MODE of object PARAMETERFILE, is nested within the space of the latter and provides information about Gibbs sampler features. Since all features of this objects are optional and have default values, object GIBBSSAMPLER is also optional.

5.2.2.1 Features

- MODE: optional value feature. Sets the sampling mode. Currently value is BLOCK only for blocked sampling as given by Sorensen & Gianola (2002). Default: BLOCK.
- CYCLES: optional variable feature. Variable is a u.d. integer value >=1 for the number of cycles to perform by the sampler. Default: 10000.
- BURNIN: optional variable feature. Variable is a u.d. integer value >=0 for the number of cycles to discard as burn-in. Default: 1000.
- PRINTINT: optional variable feature. Variable is a u.d. integer value >=1 for the sequence of printing results/calculating the mean of factor level effects. Note that this number must be consistent with CYCLES and BURNIN. Default: 100.
- MMESOLVER: optional value feature. Conditional on MODE: BLOCK. Set the solver for solving the mixed model equation when using the blocked sampler. Currently, value is PCG only. Default: PCG.

5.2.3 Object BLUP

Object BLUP is conditional on the value of feature MODE of object PARAMETERFILE, is nested within the space of the latter and provides information about BLUP features. Since all features of this object are optional and have default values, object BLUP is also optional.

5.2.3.1 Features

- MMESOLVER: optional value feature. Sets the solver for solving the mixed model equation. Currently, value is PCG only. Default: PCG.

5.2.4 Object PCG

Object PCG is conditional on the value of feature MMESOLVER of objects GIBBSSAMPLER or BLUP, can be nested in both the latter, and provides information about PCG features. Since all features of this object are optional and have default values, object PCG is also optional.

5.2.4.1 Features

- NROUNDS: optional variable feature. Sets the maximum number of iterations performed by the solver in each solver call. Variable is a u.d. integer value >=1. Default:

50000.

- CONV: optional variable feature. Sets the convergence criterion for the solver. Variable is a u.d. numeric value >=0. Default: 0.000000001.
- PRINT: optional value feature. Enables output of the solver time. Value is TIME. Default: none.
- THREADS: optional varibale feature. Allows to set the number of threads used by the PCG at runtime. Variable is a u.d. integer >= 1. Default: 1.

5.2.5 Object PEDIGREE

Object PEDIGREE is conditional on the value of INPUT of object PARAMETERFILE, is nested within the space of object PARAMETERFILE and provides information related to the pedigree.

5.2.5.1 Features

- FILE: compulsory variable feature. Variable is a u.d. file name of which file provides the pedigree
- PHAN: optional variable feature. Variable is a u.d. numeric value >=1 which provides the number of lines on top of the pedigree related to phantom parents.

5.2.6 Object GENOTYPES

Object GENOTYPES is conditional on the value of INPUT of object PARAMETERFILE, is nested within the space of object PARAMETERFILE and provides information related to the genotypes.

5.2.6.1 Features

- FILE: compulsory variable feature. Variable is a u.d. file name of which file provides the genotypes.

5.2.7 Object GRM

Object GRM is conditional on the value of INPUT of object PARAMETERFILE, is nested within the object PARAMETERFILE and provides information related to the genomic relationship matrix. Since all unconditional features are optional, the object is optional itself.

5.2.7.1 Features

- FROMGENO: optional value feature. Value is T or F, which tells BESSiE whether the GRM should be constructed from genotypes or should be read from file. Default: T.
- FILE: compulsory variable feature, conditional on FROMGENO: F. Variable is a u.d. file name of which file provides the GRM.
- METHOD: optional value feature. Value is either VR or YG where the first constructs the GRM according to VanRaden (2008), and the last according to Yang *et al.* (2010). Default: VR.
- ALFR: optional value feature for setting the method to calculate the allele frequencies underlying the GRM. Value is either GOF (observed allele frequency), GMF (average minor allele frequency) or G05 (0.5). Default is GOF.
- MAKEPD: optional value feature for making the GRM positive definite. Value is either ADD or BEND, where ADD adds 0.0001 to the diagonal elements. BEND performs an eigenvalue decomposition of the GRM, sets all eigenvalues < 0.0000001 to this value and retrieves the "bent" GRM.
- CALCEIGENVAL: optional value feature. Value is either T or F. Default: F. Useful for obtaining output of eigenvalues and eigenvectors in mode CREATE (see PRINT).
- ADDDIAG: optional variable feature, conditional on MAKEPD: ADD. Variable is a u.d. numeric value which will be added to the diagonal elements of the GRM to achieve positive definiteness. Default: 0.0001.
- BENDTRES: optional variable feature, conditional on MAKEPD: BEND. Variable is a u.d. numeric value which will be used as a threshold. Eigenvalues of the GRM below this value will be set to this value. Default: 0.0000001.
- PRINT: optional value feature. Value is any comma-separated combination of GM, GMI, GEVA, and GEVE. Default: None.
  - GM: prints GRM to file "GRM.out"
  - GMI: prints GRM to file "GRMInverse.out"
  - GEVA: prints GRM eigenvalues to "GRMEigenValues.out". Will only take effect if CALCEIGENVAL: T.
  - GEVE: prints GRM eigenvectors to "GRMEigenVectors.out". Will only take effect if CALCEIGENVAL: T.

5.2.8 Object TRAITS

Object TRAITS is conditional on the value of feature INPUT of object PARAMETERFILE, is nested within object PARAMETERFILE and provides information about the modelled traits.

5.2.8.1 Features

- NAMES: compulsory variable feature. Variable is a concatenation of u.d. trait names, separated by comma.
- DATAFILE: compulsory variable feature. Variable is a u.d. file name of which file provides the data to analyse.
- CAT: optional variable feature. Variable is a concatenation of u.d. trait names, separated by comma. The spelling of these names must be the same as in NAMES. If so, the observations of these traits will be treated as of categorical scale.

A note on feature NAMES

Feature NAMES triggers the reading of objects of which names are derived from the trait names provided in the variable where the comma is used as a trait name separator. Thus the names of these object are u.d..

5.2.9 Object COVAR

Object COVAR is nested within object TRAITS, and is conditional on the existence of this object. Thus is will be read automatically if object TRAITS is read.

5.2.9.1 Features

- FILE: compulsory variable feature. Variable is a u.d. name of the file which provides the co-variance matrix between random factors.
- PRINT: optional value feature. Values are a comma-separated concatenation of VA and SUB, where the first enables printing the sampled variances as given by the initial co-variance matrix, and the last enables printing of the initial sub-covariance matrices derived from the full co-variance matrix. Default: none.

5.2.10 Object "U.D. EXTERNAL MATRIX"

In case the matrix K (see 3.2.2) should be read from outside, this matrix must be named (e.g. "test") and an object with this name (object TEST), nested within COVAR, must be created to provide necessary information. If several different matrices should be used, an object for each must be created.

5.2.10.1 Features

- FILE: compulsory variable feature. Variable is a u.d. name of the file which provides the matrix.

5.2.11 Object "U.D. SINGLE TRAIT"

The name of this object is derived from the trait names provided by the feature NAMES of object TRAITS, where the spelling of both, the object name and the trait name in the variable of NAMES, must match. Thus, the object is conditional on feature NAME of object TRAITS and is nested within the latter.

5.2.11.1 Features

- MODEL: compulsory variable feature. Variable are u.d. names of factors to be included in the model concatenated by a "+".
- OBSERVPOS: compulsory variable feature. Variable is a u.d. integer value >=1 which contains the column position of the observations of this trait in the data file.
- RESVARPOS: compulsory variable feature. Variable is a u.d. integer value >=1 which contains the column/row position of the residual variance of this trait in the co-variance matrix.
- WEIGHTPOS: optional variable feature. Variable is a u.d. integer value >=1 which contains the column position of the weights related to the observations of this trait in the data file.
- PRINT: optional value feature. Enables various output. Default: none. Value is one or a comma-separated concatenation of the following:
  - RA or RB: write trait residuals in ascii or binary, respectively.

Feature MODEL Feature MODEL triggers the reading of objects of which names are derived from the factor names provided in the variable where the "+" is used as a factor

name separator. Thus the names of the objects describing the factors are u.d..

5.2.12 Object 'U.D. SINGLE FACTOR"

The name of this object is derived from the factor names provided by the feature MODEL of object 'U.D. TRAIT NAME", where the spelling of both, the object name and the factor name in the variable of MODEL, must match. Thus, the object is conditional on feature MODEL of object 'U.D. TRAIT NAME", and is nested within the latter.

5.2.12.1 Features

- TYPE: compulsory value feature. Value is either FIX or RAN. Provides information whether a factor should be modelled as fixed or as random.
- FILEPOS: compulsory variable feature. Variable is a u.d. integer providing the column position of this factor in the data file.
- COV: optional value feature conditional on TYPE: FIX. Value is T or F. Provides information whether a fixed factor is an integer dummy variable or a linear co-variable. Default: F.
- GG: optional value feature conditional on TYPE: FIX and COV: T. Value is T or F. Provides information whether a fixed linear co-variable should be derived from genetic groups. Default: F.
- EXP: optional variable feature conditional on COV: T and GG: F. Variable is a u.d. integer value. Provides information about the order of the polynomial used to model a fixed factor which is linear co-variable. Default: 1.
- STRUCTURE: compulsory value feature conditional on TYPE: RAN. Provides information about the co-variance structure for this random effect. Possible values are NRM, NRMGG, IDE, GRM or SNP.
- METHOD: compulsory value feature conditional on STRUCTURE: SNP. Provides information which method to use for the estimation of SNP effects. Possible values are BLUP, BAYESA, BAYESB, BAYESCPI or BAYESR.
- COVARPOS: compulsory variable feature conditional the value of STRUCTURE is NRM, NRMGG, GRM, IDE or SNP, where the latter applies only if METHOD: BLUP. Variable is a u.d. integer which provides information about the row/column position of this factor in the co-variance matrix.
- SCALE: optional variable feature conditional on METHOD: BAYESA and METHOD:

BAYESB. Variable is a real number >0 setting the scale parameter of the scaled inverse chi-square distribution. Default for METHOD: BAYESA: 0.002, and for METHOD: BAYESB: 0.0429.

- SHAPE: optional variable feature conditional on METHOD: BAYESA, METHOD: BAYESB and METHOD: BAYESCPI. Variable is a real number >0 setting the shape parameter of the scaled inverse chi-square distribution. Default for METHOD: BAYESA: 4.012, for METHOD: BAYESB: 4.234 and for METHOD: BAYESCPI: 4.2.
- NORMSNP: optional variable feature conditional on METHOD: BAYESA, METHOD: BAYESB, METHOD: BAYESCPI or METHOD: BAYESR. Value is T or F. Provides information whether the marker matrix used for this factor should be normalised (centered and scaled). Default: F.
- PI: optional variable feature conditional on METHOD: BAYESB and METHOD: BAYESCPI. Variable is a real number between 0 and 1 which provides the general (BAYESB) or the initial (BAYESCPI) probability that a SNP has a zero effect. Default for METHOD: BAYESB: 0.95, for METHOD: BAYESCPI: 0.95.
- NMETRO: optional variable feature conditional on METHOD: BAYESB. Variable is an integer >0 which sets the number of Metropolis-Hasting cycles. Default: 100.
- PALPHA: optional variable feature conditional on METHOD: BAYESCPI. Variable is an integer >0 which provides prior knowledge about the alpha parameter of the beta distribution. Default: 1.
- PBETA: optional variable feature conditional on METHOD: BAYESCPI. Variable is an integer >0 which provides prior knowledge about the beta parameter of the beta distribution. Default: 1.
- TOTALSNPVAR: compulsory variable feature conditional on METHOD: BAYESR or METHOD: BAYESCPI. Variable is a u.d. real number providing the variance maximum explainable by SNPs.
- DISTFILE: compulsory variable feature conditional on METHOD: BAYESR. Variable is the u.d. name of the file which provides information about the distributions to model.
- PRINT: optional value feature. Enables various output. Default: none. Value is one or a comma-separated concatenation of the following:
  - EA or EB: write factor level effects in ascii or binary, respectively.
  - SA: write factor level effect statistics (mean and standard deviation) in ascii.
  - VSNPA or VSNPB: write variances of SNP effects sampled by a method from the Bayesian alphabet in ascii or binary, respectively.

- GBVA or GBVB: write genomic values for animals with genotypes for SNP effects modelled by a method from the Bayesian alphabet in ascii or binary, respectively.
- SVSNPA or SVSNPB: write variance explained by SNP effects modelled by a method from the Bayesian alphabet in ascii or binary, respectively. This variance is calculated as $\sum_i^N 2p_i(1-p_i)a_i^2$, where N is the number of SNPs, p_i is the minor allele frequency and a_i is the effect of SNP i.
- PIA or PIB: conditional on METHOD: BAYESCPI or METHOD: BAYESR. For BayesCπ this will trigger output of π, the re-calculated locus variance, the re-calculated scale parameter and the number of SNPs with a non-zero effect. For BayesR, this will trigger output of the vector of distribution probabilities (BayesR) and the related SNP counts for each distribution in ascii or binary, respectively.
- DC: conditional on METHOD: BAYESR. write distribution counts for every SNP including burn-in.

5.3 Parameter file examples

5.3.1 Uni-variate analysis

5.3.1.1 Factors: fix, fixed co-variables, ran. NRM direct, ran. NRM maternal and ran. IDE

Consider the model

$$y = X_d b_d + X_c b_c + Z_d u_d + Z_m u_m + Cc + e$$

with y being a vector of phenotypic observations, capital letters are matrices linking the effects to their respective observations, and the following factors:

b_d: fixed effect of dummy variable (f1),

b_c: fixed effect of a co-variable modelled as third order polynomial (f2),

u_d: random direct animal genetic effect,

u_m: random direct maternal genetic effect,

c: random common environmental effect,

e: residual effect.

A Gibbs sampling analysis of the data with this model would require the following parameter file:

Example 5.13: Parameter file for uni-variate fix, fixed co-variables, ran. NRM direct, ran. NRM maternal and ran. IDE

```
1 BEGIN PARAMETERFILE
2   JOBNAME: Test !provide a job name
3   MODE: GIBBS !provide mode
4   INPUT: PEDIGREE,TRAITS !provide names of objects to be read
5   BEGIN TRAITS
6     DATAFILE: Data.txt !provide the name of the data file
7     NAMES: weight !provide trait name
8     BEGIN weight
9       OBSERVPOS: 6 !provide the observation position in the data file
10      RESVARPOS: 4 !provide the row/column position in the co-variance matrix file
11      MODEL: f1+f2+animal+maternal+common !provide the model
12      BEGIN f1
13        TYPE: FIX !provide the type
14        FILEPOS: 1 !provide the position of this factor in the data file
15      END f1
16      BEGIN f2
```

```
17        TYPE: FIX !provide the type
18        FILEPOS: 2 !provide the position of this factor in the data file
19        COV: T !provide the switch to co-variables
20        EXP: 3 !override the order of the polynomial
21      END f2
22      BEGIN animal
23        TYPE: RAN !provide the type
24        FILEPOS: 3 !provide the position of this factor in the data file
25        STRUCTURE: NRM !provide the structure of the random factor
26        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
27      END animal
28      BEGIN maternal
29        TYPE: RAN !provide the type
30        FILEPOS: 4 !provide the position of this factor in the data file
31        STRUCTURE: NRM !provide the structure of the random factor
32        COVARPOS: 2 !provide the row/column position in the co-variance matrix file
33      END maternal
34      BEGIN common
35        TYPE: RAN !provide the type
36        FILEPOS: 5 !provide the position of this factor in the data file
37        STRUCTURE: IDE !provide the structure of the random factor
38        COVARPOS: 3 !provide the row/column position in the co-variance matrix file
39      END common
40    END weight
41    BEGIN COVAR
42      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
43      PRINT: VA !set switch for printing sampled variances
44    END COVAR
45  END TRAITS
46  BEGIN GIBBSSAMPLER
47    BURNIN: 1000 !override default setting for burn-in
48    PRINTINT: 100 !override default setting for interval for printing/mean calculation
49    CYCLES: 10000 !override default number of Gibbs cycles
50  END GIBBSSAMPLER
51  BEGIN PEDIGREE
52    FILE: Pedigree.txt !provide the name of the file containing the pedigree
53  END PEDIGREE
54 END PARAMETERFILE
```

892 Setting the feature MODE (line 3) to BLUP switches to BLUP analysis. Dropping fac-
893 tors from the model can be achieved by deleting these factor from VARIABLE of feature
894 MODEL (line 11). However, if a random factor is dropped which requires a record in the
895 co-variance matrix, the matrix and the respective positions must be adjusted.

5.3.1.2 Factors: fix, ran. NRM direct and ran. NRMGG (random genetic groups)

Consider the model

$$y = X_d b_d + Z_d u_d + Z_{gg} Q u_{gg} + e$$

with y being a vector of phenotypic observations, capital letters are matrices linking the effects to their respective observations, and the following factors:

b_d: fixed effect of dummy variable (f1),

u_d: random direct animal genetic effect,

u_{gg}: random genetic group effect where Q is the respective matrix of regressors,

e: residual effect.

A Gibbs sampling analysis of the data with this model would require the following parameter file:

Example 5.14: Parameter file for uni-variate fix, ran. NRM direct and ran. NRMGG (random genetic groups)

```
BEGIN PARAMETERFILE
  JOBNAME: Test !provide a job name
  MODE: GIBBS !provide mode
  INPUT: PEDIGREE,TRAITS !provide names of objects to be read
  BEGIN TRAITS
    DATAFILE: Data.txt !provide the name of the data file
    NAMES: weight !provide trait name
    BEGIN weight
      OBSERVPOS: 6 !provide the observation position in the data file
      RESVARPOS: 4 !provide the row/column position in the co-variance matrix file
      MODEL: f1+animal+gg !provide the model
      BEGIN f1
        TYPE: FIX !provide the type
        FILEPOS: 1 !provide the position of this factor in the data file
      END f1
      BEGIN animal
        TYPE: RAN !provide the type
        FILEPOS: 3 !provide the position of this factor in the data file
        STRUCTURE: NRM !provide the structure of the random factor
        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
      END animal
      BEGIN gg
        TYPE: RAN !provide the type
        FILEPOS: 3 !provide the position of this factor in the data file
```

```
25        STRUCTURE: NRMGG !provide the structure of the random factor
26        COVARPOS: 2 !provide the row/column position in the co-variance matrix file
27      END gg
28    END weight
29    BEGIN COVAR
30      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
31      PRINT: VA !set switch for printing sampled variances
32    END COVAR
33  END TRAITS
34  BEGIN GIBBSSAMPLER
35    BURNIN: 1000 !override default setting for burn-in
36    PRINTINT: 100 !override default setting for interval for printing/mean calculation
37    CYCLES: 10000 !override default number of Gibbs cycles
38  END GIBBSSAMPLER
39  BEGIN PEDIGREE
40    FILE: Pedigree.txt !provide the name of the file containing the pedigree
41  END PEDIGREE
42 END PARAMETERFILE
```

Note line 30 of the parameter file. The file position of factor "gg" is equal to the file position of factor "animal". Thus, in this setting the genetic group effect will be estimated on the animal level. However, the file position of "gg" may be different to that in "animal", but numbers must be integer and related to the individual ids in the pedigree without genetic groups.

5.3.1.3 Factors: fix, ran. NRM direct and fixed genetic groups

Consider the model

$$y = X_d b_d + Z_d u_d + X_{gg} Q b_{gg} + e$$

with y being a vector of phenotypic observations, capital letters are matrices linking the effects to their respective observations, and the following factors:

b_d: fixed effect of dummy variable (f1),

u_d: random direct animal genetic effect,

b_{gg}: fixed genetic group effect where Q is the respective matrix of regressors,

e: residual effect.

A Gibbs sampling analysis of the data with this model would require the following parameter file:

Example 5.15: Parameter file for uni-variate fix, ran. NRM direct and fixed genetic groups

```
1 BEGIN PARAMETERFILE
2   JOBNAME: Test !provide a job name
3   MODE: GIBBS !provide mode
4   INPUT: PEDIGREE,TRAITS !provide names of objects to be read
5   BEGIN TRAITS
6     DATAFILE: Data.txt !provide the name of the data file
7     NAMES: weight !provide trait name
8     BEGIN weight
9       OBSERVPOS: 6 !provide the observation position in the data file
10      RESVARPOS: 4 !provide the row/column position in the co-variance matrix file
11      MODEL: f1+animal+gg !provide the model
12      BEGIN f1
13        TYPE: FIX !provide the type
14        FILEPOS: 1 !provide the position of this factor in the data file
15      END f1
16      BEGIN animal
17        TYPE: RAN !provide the type
18        FILEPOS: 3 !provide the position of this factor in the data file
19        STRUCTURE: NRM !provide the structure of the random factor
20        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
21      END animal
22      BEGIN gg
23        TYPE: FIX !provide the type
24        FILEPOS: 3 !provide the position of this factor in the data file
25        COV: T !provide the switch to co-variables
26        GG: T !provide the switch to pedigree derived genetic groups
27      END gg
28    END weight
29    BEGIN COVAR
30      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
31      PRINT: VA !set switch for printing sampled variances
32    END COVAR
33  END TRAITS
34  BEGIN GIBBSSAMPLER
35    BURNIN: 1000 !override default setting for burn-in
36    PRINTINT: 100 !override default setting for interval for printing/mean calculation
37    CYCLES: 10000 !override default number of Gibbs cycles
38  END GIBBSSAMPLER
39  BEGIN PEDIGREE
40    FILE: Pedigree.txt !provide the name of the file containing the pedigree
41  END PEDIGREE
42 END PARAMETERFILE
```

As in the previous example, note line 30 of the parameter file. The file position of factor "gg" is equal to the file position of factor "animal". Thus, in this setting the genetic group effect will be estimated on the animal level.

5.3.1.4 Factors: fix and ran. NRM direct, Weighted phenotypes

Consider the model

$$y = X_d b_d + Z_d u_d + e$$

with y being a vector of phenotypic observations, capital letters are matrices linking the effects to their respective observations, and the following factors:

b_d: fixed effect of dummy variable (f1),

u_d: random direct animal genetic effect,

e: residual effect,

and residuals distributed $N(0, D\sigma_e^2)$, where D is a matrix of weights provided via the data file. A Gibbs sampling analysis of the data with this model would require the following parameter file:

Example 5.16: Parameter file for uni-variate fix and ran. NRM direct, Weighted phenotypes

```
BEGIN PARAMETERFILE
  JOBNAME: Test !provide a job name
  MODE: GIBBS !provide mode
  INPUT: PEDIGREE,TRAITS !provide names of objects to be read
  BEGIN TRAITS
    DATAFILE: Data.txt !provide the name of the data file
    NAMES: weight !provide trait name
    BEGIN weight
      OBSERVPOS: 6 !provide the observation position in the data file
      RESVARPOS: 4 !provide the row/column position in the co-variance matrix file
      WEIGHTPOS: 7 !provide the weight position in the data file
      MODEL: f1+f2+animal+maternal+common !provide the model
      BEGIN f1
        TYPE: FIX !provide the type
        FILEPOS: 1 !provide the position of this factor in the data file
      END f1
      BEGIN animal
        TYPE: RAN !provide the type
        FILEPOS: 3 !provide the position of this factor in the data file
        STRUCTURE: NRM !provide the structure of the random factor
        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
      END animal
    END weight
    BEGIN COVAR
      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
      PRINT: VA !set switch for printing sampled variances
```

```
27    END COVAR
28  END TRAITS
29  BEGIN GIBBSSAMPLER
30    BURNIN: 1000 !override default setting for burn-in
31    PRINTINT: 100 !override default setting for interval for printing/mean calculation
32    CYCLES: 10000 !override default number of Gibbs cycles
33  END GIBBSSAMPLER
34  BEGIN PEDIGREE
35    FILE: Pedigree.txt !provide the name of the file containing the pedigree
36  END PEDIGREE
37 END PARAMETERFILE
```

Note the optional variable feature WEIGHTPOS in line 11, which tells BESSiE to alter the distribution of residuals as given above.

5.3.1.5 Factors: fix, ran. NRM direct and ran. IDE with external matrix K

Consider the model

$$y = X_d b_d + Z_d u_d + Z_m u_m + Cc + e$$

with y being a vector of phenotypic observations, capital letters are matrices linking the effects to their respective observations, and with the following factors:

b_d: fixed effect of dummy variable (f1),

u_d: random direct animal genetic effect,

c: random effect $\sim N(0, K\sigma^2)$, where K in an arbitrary squared, symmetric and positive definite matrix provided by the user,

e: residual effect.

A Gibbs sampling analysis of the data with this model would require the following parameter file:

Example 5.17: Parameter file for uni-variate fix, ran. NRM direct and ran. IDE with external matrix K

```
1 BEGIN PARAMETERFILE
2   JOBNAME: Test !provide a jobname
3   MODE: GIBBS !provide mode
4   INPUT: PEDIGREE,TRAITS !provide names of objects to be read
5   BEGIN TRAITS
6     DATAFILE: Data.txt !provide the name of the data file
7     NAMES: weight !provide trait name
8     BEGIN weight
```

```
9       OBSERVPOS: 4 !provide the observation position in the data file
10      RESVARPOS: 3 !provide the row/column position in the co-variance matrix file
11      MODEL: f1+animal+common !provide the model
12      BEGIN f1
13        TYPE: FIX !provide the type
14        FILEPOS: 1 !provide the position of this factor in the data file
15      END f1
16      BEGIN animal
17        TYPE: RAN !provide the type
18        FILEPOS: 2 !provide the position of this factor in the data file
19        STRUCTURE: NRM !provide the structure of the random factor
20        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
21      END animal
22      BEGIN common
23        TYPE: RAN !provide the type
24        FILEPOS: 3 !provide the position of this factor in the data file
25        STRUCTURE: IDE !provide the structure of the random factor
26        COVARPOS: 2 !provide the row/column position in the co-variance matrix file
27        EXT: k !provide a name for the matrix K
28      END common
29    END weight
30    BEGIN COVAR
31      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
32      PRINT: VA !set switch for printing sampled variances
33      BEGIN k
34        File: K_Matrix.txt !provide the name of the file containing matrix K
35      END k
36    END COVAR
37  END TRAITS
38  BEGIN GIBBSSAMPLER
39    BURNIN: 1000 !override default setting for burn-in
40    PRINTINT: 100 !override default setting for interval for printing/mean calculation
41    CYCLES: 10000 !override default number of Gibbs cycles
42  END GIBBSSAMPLER
43  BEGIN PEDIGREE
44    FILE: Pedigree.txt !provide the name of the file containing the pedigree
45  END PEDIGREE
46 END PARAMETERFILE
```

948 In the example above, the space of object COMMON (line 22 - 28), nested in space of object
949 WEIGHT (line 8 - 29), has been extended by the feature EXT (line 27) of which variable is
950 k. This tells BESSiE to search for the space of an object named k (note that the spelling
951 must be equal) in the space of variable COVAR (line 30 - 36). This space is initialised with
952 BEGIN K (line 33) and finalised with END K (line 35), and it is compulsory because it has
953 a compulsory feature FILE of which variable provides the name of the file containing the
954 matrix K. This structure allows to impose co-variances between factors of type $\Sigma \otimes K$ (see

3.2.2). 955

5.3.1.6 Factors: fix and ran. GRM 956

Consider the model

$$y = Xb + Z_{GRM}u_{GRM} + e$$

with y being a vector of phenotypic observations, all capital letters are matrices linking factor levels to their respective observations, and the following factors: 957 958

b: fixed effect of dummy variable (f1), 959

u_{GRM}: effect of genomic breeding value $\sim N(0, GRM\sigma^2)$, 960

e: residual effect. 961

A Gibbs sampling analysis of the data with this model would require the following parameter file: 962 963

Example 5.18: Parameter file for uni-variate fix and ran. GRM 964

```
BEGIN PARAMETERFILE
  JOBNAME: Test !provide a jobname
  MODE: GIBBS !provide mode
  INPUT: TRAITS,GENOTYPES,GRM !provide names of objects to be read separated by comma
  BEGIN TRAITS
    DATAFILE: Data.txt !provide the name of the data file
    NAMES: weight !provide trait name
    BEGIN weight
      OBSERVPOS: 3 !provide the observation position in the data file
      RESVARPOS: 2 !provide the row/column position in the co-variance matrix file
      MODEL: f1+gbv !provide the model
      BEGIN f1
        TYPE: FIX !provide the type
        FILEPOS: 1 !provide the position of this factor in the data file
      END f1
      BEGIN gbv
        TYPE: RAN !provide the type
        FILEPOS: 2 !provide the position of this factor in the data file
        STRUCTURE: GRM !provide the structure of the random factor
        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
        PRINT: EA !set switches for printing various output
      END gbv
    END weight
    BEGIN COVAR
      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
      PRINT: VA !set switch for printing sampled variances
```

```
27    END COVAR
28  END TRAITS
29  BEGIN GIBBSSAMPLER
30    BURNIN: 1000 !override default setting for burn-in
31    PRINTINT: 100 !override default setting for interval for printing/mean calculation
32    CYCLES: 10000 !override default number of Gibbs cycles
33  END GIBBSSAMPLER
34  BEGIN PEDIGREE
35    FILE: Pedigree.txt !provide the name of the file containing the pedigree
36  END PEDIGREE
37  BEGIN GENOTYPES
38    FILE: Genotypes.txt !provide the name of the file containing the genotypes
39  END GENOTYPES
40 END PARAMETERFILE
```

Note the additional values of feature INPUT of object PARAMETERFILE (line 4). GRM and GENOTYPES have been added to the comma separated list, thus triggering the reading of object GENOTYPES and GRM. While the object GENOTYPES is defined at the bottom of the parameter file (line 37 - 39), the object GRM is fully determined by default values, thus neither an initialiser/finaliser nor any object space or feature is necessary for object GRM. In that case, GRM will constructed from genotypes, and a missing of the value GENOTYPES of feature INPUT will cause BESSiE to terminate with a related error message. Note that PEDIGREE has been omitted from INPUT. Thus, BESSiE will not attempt to read the space of object PEDIGREE, even when included in the parameter file.

5.3.1.7 Factors: fix, ran. NRM direct and BayesR SNP

Consider the model

$$y = Xb + Z_d u + Z_m M g + e$$

with y being a vector of phenotypic observations, M is a matrix of SNP marker genotypes, all other capital letters are matrices linking the effects to their respective observations, and with the following factors:

- b: fixed effect of dummy variable (f1),
- u_d: random direct animal genetic effect, (animal),
- g: effect of SNP marker (snp),
- e: residual effect.

A Gibbs sampling analysis of the data with this model applying the "BayesR" algorithm would require the following parameter file:

Example 5.19: Parameter file for uni-variate fix, ran. NRM direct and BayesR SNP

```
BEGIN PARAMETERFILE
  JOBNAME: Test !provide a jobname
  MODE: GIBBS !provide mode
  INPUT: PEDIGREE,TRAITS,GENOTYPES !provide names of objects to be read separated by comma
  BEGIN TRAITS
    DATAFILE: Data.txt !provide the name of the data file
    NAMES: weight !provide trait name
    BEGIN weight
      OBSERVPOS: 4 !provide the observation position in the data file
      RESVARPOS: 2 !provide the row/column position in the co-variance matrix file
      MODEL: f1+animal+snp !provide the model
      BEGIN f1
        TYPE: FIX !provide the type
        FILEPOS: 1 !provide the position of this factor in the data file
      END f1
      BEGIN animal
        TYPE: RAN !provide the type
        FILEPOS: 2 !provide the position of this factor in the data file
        STRUCTURE: NRM !provide the structure of the random factor
        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
      END animal
      BEGIN snp
        TYPE: RAN !provide the type
        FILEPOS: 3 !provide the position of this factor in the data file
        STRUCTURE: SNP !provide the structure of the random factor
        METHOD: BAYESR !provide a method which allows for structure SNP
        TOTALSNPVAR: 300 !provide the variance totally explainable by SNP
        DISTFILE: SNPVar.txt !provide the name of the file containing the SNP variance proportions
        PRINT: EA,VSNPA,SVSNPA,GBVA,PIA,DC !set switches for printing various output
      END snp
    END weight
    BEGIN COVAR
      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
      PRINT: VA !set switch for printing sampled variances
    END COVAR
  END TRAITS
  BEGIN GIBBSSAMPLER
    BURNIN: 1000 !override default setting for burn-in
    PRINTINT: 100 !override default setting for interval for printing/mean calculation
    CYCLES: 10000 !override default number of Gibbs cycles
  END GIBBSSAMPLER
  BEGIN PEDIGREE
    FILE: Pedigree.txt !provide the name of the file containing the pedigree
  END PEDIGREE
  BEGIN GENOTYPES
    FILE: Genotypes.txt !provide the name of the file containing the genotypes
```

```
47  END GENOTYPES
48 END PARAMETERFILE
```

Note the additional value of feature INPUT of object PARAMETERFILE (line 4). GENOTYPES has been added to the comma separated list, thus triggering the reading of object GENOTYPES at the bottom of the parameter file (line 45 - 47). Since "BAYESA" and "BAYESB" will not require any additional input, and the only compulsory feature when using "BAYESCPI" is TOTALSNPVAR, changing to these methods can easily be achieved by setting the feature METHOD (line 26) to "BAYESA", "BAYESB" or "BAYESCPI". Note that when changing METHOD, compulsory features unique to "BAYESR" (e.g. DISTFILE) neither need to be omitted nor commented because BESSiE will simply not regard them while reading the space of object SNP.

5.3.1.8 Factors: fix, ran. NRM direct and BayesR SNP, categorical observations

Consider the model

$$y = Xb + Z_d u + Z_m M g + e$$

with y being a vector of categorical phenotypic observations, M is a matrix of SNP marker genotypes, all other capital letters are matrices linking the effects to their respective observations, and with the following factors:

b: fixed effect of dummy variable (f1),

u_d: random direct animal genetic effect, (animal),

g: effect of SNP marker (snp),

e: residual effect.

A Gibbs sampling analysis of the data with this model applying the "BayesR" algorithm would require the following parameter file:

Example 5.20: Parameter file for uni-variate fix, ran. NRM direct and BayesR SNP with categorical observations

```
1 BEGIN PARAMETERFILE
2   JOBNAME: Test !provide a jobname
3   MODE: GIBBS !provide mode
4   INPUT: PEDIGREE,TRAITS,GENOTYPES !provide names of objects to be read separated by comma
5   BEGIN TRAITS
6     DATAFILE: Data.txt !provide the name of the data file
7     NAMES: weight !provide trait name
```

```
8     CAT: weight !switch to categorical analysis for this trait
9     BEGIN weight
10      OBSERVPOS: 4 !provide the observation position in the data file
11      RESVARPOS: 2 !provide the row/column position in the co-variance matrix file
12      MODEL: f1+animal+snp !provide the model
13      BEGIN f1
14        TYPE: FIX !provide the type
15        FILEPOS: 1 !provide the position of this factor in the data file
16      END f1
17      BEGIN animal
18        TYPE: RAN !provide the type
19        FILEPOS: 2 !provide the position of this factor in the data file
20        STRUCTURE: NRM !provide the structure of the random factor
21        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
22      END animal
23      BEGIN snp
24        TYPE: RAN !provide the type
25        FILEPOS: 3 !provide the position of this factor in the data file
26        STRUCTURE: SNP !provide the structure of the random factor
27        METHOD: BAYESR !provide a method which allows for structure SNP
28        TOTALSNPVAR: 300 !provide the variance totally explainable by SNP
29        DISTFILE: SNPVar.txt !provide the name of the file containing the SNP variance proportions
30        PRINT: EA,VSNPA,SVSNPA,GBVA,PIA,DC !set switches for printing various output
31      END snp
32    END weight
33    BEGIN COVAR
34      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
35      PRINT: VA !set switch for printing sampled variances
36    END COVAR
37  END TRAITS
38  BEGIN GIBBSSAMPLER
39    BURNIN: 1000 !override default setting for burn-in
40    PRINTINT: 100 !override default setting for interval for printing/mean calculation
41    CYCLES: 10000 !override default number of Gibbs cycles
42  END GIBBSSAMPLER
43  BEGIN PEDIGREE
44    FILE: Pedigree.txt !provide the name of the file containing the pedigree
45  END PEDIGREE
46  BEGIN GENOTYPES
47    FILE: Genotypes.txt !provide the name of the file containing the genotypes
48  END GENOTYPES
49 END PARAMETERFILE
```

Note the additional value of feature INPUT of object PARAMETERFILE (line 4). GENOTYPES has been added to the comma separated list, thus triggering the reading of object GENOTYPES at the bottom of the parameter file (line 45 - 47). Since “BAYESA”

and "BAYESB" will not require any additional input, and the only compulsory feature when using "BAYESCPI" is TOTALSNPVAR, changing to these methods can easily be achieved by setting the feature METHOD (line 26) to "BAYESA", "BAYESB" or "BAYESCPI". Note that when changing METHOD, compulsory features unique to "BAYESR" (e.g. DISTFILE) neither need to be omitted nor commented because BESSiE will simply not regard them while reading the space of object SNP.

5.3.2 Bi-variate models

5.3.2.1 Factors: fix and ran. NRM direct

Consider the model

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} + \begin{pmatrix} Z_{u1} & 0 \\ 0 & Z_{u2} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

with subscripts 1 and 2 for traits 1 and 2, y being a vector of phenotypic observations, all capital letters are matrices linking the effects to their respective observations, and the following factors:

b: fixed effect of dummy variable (f1),

u_d: random direct animal genetic effect,

e: residual effect.

A Gibbs sampling analysis of the data with this model would require the following parameter file:

Example 5.21: Parameter file for bi-variate fix and ran. NRM direct

```
1 BEGIN PARAMETERFILE
2   JOBNAME: Test !provide a jobname
3   MODE: GIBBS !provide mode
4   INPUT: PEDIGREE,TRAITS !provide names of objects to be read separated by comma
5   BEGIN TRAITS
6     DATAFILE: Data.txt !provide the name of the data file
7     NAMES: weight,height !provide trait names separated by comma
8     ! start defining trait weight
9     BEGIN weight
10      OBSERVPOS: 3 !provide the observation position in the data file
11      RESVARPOS: 3 !provide the row/column position in the co-variance matrix file
12      MODEL: f1+animal !provide the model
13      BEGIN f1
14        TYPE: FIX !provide the type
15        FILEPOS: 1 !provide the position of this factor in the data file
```

```
16      END f1
17      BEGIN animal
18        TYPE: RAN !provide the type
19        FILEPOS: 2 !provide the position of this factor in the data file
20        STRUCTURE: NRM !provide the structure of the random factor
21        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
22      END animal
23    END weight
24    ! start defining trait height
25    BEGIN height
26      OBSERVPOS: 5 !provide the observation position in the data file
27      RESVARPOS: 4 !provide the row/column position in the co-variance matrix file
28      MODEL: f1+animal !provide the model
29      BEGIN f1
30        TYPE: FIX !provide the type
31        FILEPOS: 4 !provide the position of this factor in the data file
32      END f1
33      BEGIN animal
34        TYPE: RAN !provide the type
35        FILEPOS: 2 !provide the position of this factor in the data file
36        STRUCTURE: NRM !provide the structure of the random factor
37        COVARPOS: 2 !provide the row/column position in the co-variance matrix file
38      END animal
39    END height
40    BEGIN COVAR
41      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
42      PRINT: VA !set switch for printing sampled variances
43    END COVAR
44  END TRAITS
45  BEGIN GIBBSSAMPLER
46    BURNIN: 1000 !override default setting for burn-in
47    PRINTINT: 100 !override default setting for interval for printing/mean calculation
48    CYCLES: 10000 !override default number of Gibbs cycles
49  END GIBBSSAMPLER
50  BEGIN PEDIGREE
51    FILE: Pedigree.txt !provide the name of the file containing the pedigree
52  END PEDIGREE
53 END PARAMETERFILE
```

Note the differences to the parameter file for uni-variate analysis:

- a second trait, "height" has been added to the variable of feature NAMES of object TRAITS (line 7), separated from the first trait by a comma.
- an space for object HEIGHT has been added to the parameter file (line 25 - 39)
- positions in the data file and in the co-variance file have been adjusted.

Note that the variables for features FILEPOS in line 19 and 35 are equal because animal ids are the same for both traits.

5.3.2.2 Factors: fix, ran. NRM direct and BayesR SNP

Consider the model

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} + \begin{pmatrix} Z_{u1} & 0 \\ 0 & Z_{u2} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} + \begin{pmatrix} Z_{m1}M & 0 \\ 0 & Z_{m2}M \end{pmatrix} \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

with subscripts 1 and 2 for traits 1 and 2, y being a vector of phenotypic observations, M is a matrix of SNP marker genotypes, all other capital letters are matrices linking the effects to their respective observations, and the following factors:

b: fixed effect of dummy variable (f1),

u_d: random direct animal genetic effect,

g: effect of SNP marker,

e: residual effect.

A Gibbs sampling analysis of the data with this model applying the "BayesR" for the first trait and a "BayesB" algorithm would require the following parameter file:

Example 5.22: Parameter file for bi-variate fix, ran. NRM direct and BayesR SNP

```
1 BEGIN PARAMETERFILE
2   JOBNAME: Test !provide a jobname
3   MODE: GIBBS !provide mode
4   INPUT: PEDIGREE,TRAITS,GENOTYPES !provide names of objects to be read separated by comma
5   BEGIN TRAITS
6     DATAFILE: Data.txt !provide the name of the data file
7     NAMES: weight,height !provide trait names separated by comma
8     ! start defining trait weight
9     BEGIN weight
10  a    OBSERVPOS: 4 !provide the observation position in the data file
11      RESVARPOS: 3 !provide the row/column position in the co-variance matrix file
12      MODEL: f1+animal+snp !provide the model
13      BEGIN f1
14        TYPE: FIX !provide the type
15        FILEPOS: 1 !provide the position of this factor in the data file
16      END f1
17      BEGIN animal
18        TYPE: RAN !provide the type
19        FILEPOS: 2 !provide the position of this factor in the data file
20        STRUCTURE: NRM !provide the structure of the random factor
```

```
21        COVARPOS: 1 !provide the row/column position in the co-variance matrix file
22      END animal
23      BEGIN snp
24        TYPE: RAN !provide the type
25        FILEPOS: 3 !provide the position of this factor in the data file
26        STRUCTURE: SNP !provide the structure of the random factor
27        METHOD: BAYESR !provide a method which is allowed for structure SNP
28        TOTALSNPVAR: 300 !provide the variance totally explainable by SNP
29        DISTFILE: SNPVar.txt !provide the name of the file containing the SNP variance proportions
30        PRINT: EA,VSNPA,SVSNPA,GBVA,PIA,DC !set switches for printing various output separated by comma
31      END snp
32    END weight
33    ! start defining trait height
34    BEGIN height
35      OBSERVPOS: 6 !provide the observation position in the data file
36      RESVARPOS: 4 !provide the row/column position in the co-variance matrix file
37      MODEL: f1+animal+snp !provide the model
38      BEGIN f1
39        TYPE: FIX !provide the type
40        FILEPOS: 5 !provide the position of this factor in the data file
41      END f1
42      BEGIN animal
43        TYPE: RAN !provide the type
44        FILEPOS: 2 !provide the position of this factor in the data file
45        STRUCTURE: NRM !provide the structure of the random factor
46        COVARPOS: 2 !provide the row/column position in the co-variance matrix file
47      END animal
48      BEGIN snp
49        TYPE: RAN !provide the type
50        FILEPOS: 3 !provide the position of this factor in the data file
51        STRUCTURE: SNP !provide the structure of the random factor
52        METHOD: BAYESB !provide a method which is allowed for structure SNP
53        PRINT: EA,VSNPA,SVSNPA,GBVA,PIA,DC !set switches for printing various output separated by comma
54      END snp
55    END height
56    BEGIN COVAR
57      FILE: Covar.txt !provide the name of the file containing the co-variance matrix
58      PRINT: VA !set switch for printing sampled variances
59    END COVAR
60  END TRAITS
61  BEGIN GIBBSSAMPLER
62    BURNIN: 1000 !override default setting for burn-in
63    PRINTINT: 100 !override default setting for interval for printing/mean calculation
64    CYCLES: 10000 !override default number of Gibbs cycles
65  END GIBBSSAMPLER
66  BEGIN PEDIGREE
67    FILE: Pedigree.txt !provide the name of the file containing the pedigree
```

```
68   END PEDIGREE
69   BEGIN GENOTYPES
70     FILE: Genotypes.txt !provide the name of the file containing the genotypes
71   END GENOTYPES
72 END PARAMETERFILE
```

Note that the major difference to the parameter file set up for uni-variate analysis is the feature NAMES (line 7) in object TRAITS. The name of the second trait has been added to the variable with the new variable "weight,height". This is already sufficient to tell BESSiE that it should perform a bi-variate analysis. The space of object HEIGHT (line 34 - 55), which is added for defining the second trait, is simply a copy of the space of object WEIGHT (line 9 - 32), with features adjusted accordingly (OBSERVPOS, RESVARPOS, COVARPOS, FILEPOS, METHOD)

6 Program output

BESSiE will write output to the command line and to files. The command line output is limited to a note about the success/failure of the program. All other output is written to various files, where by default only a log file is created. Any other output files will be generated only if a respective command is set in the parameter file. All files will be written into the directory where BESSiE was started. Output files are either report files or result files.

6.1 Report files

6.1.1 Logfile.out

The main report file is named "Logfile.out". It contains the current status of the program and all error messages, and its generation cannot be suppressed. The file can be parsed for error messages by searching for the keyword "ERROR".

6.1.2 PCGTime.out

This file contains information about the CPU time, real time and number of iterations, the preconditioned gradient solver (PCG) has used to solve the mixed model equation.
The files contains six columns:

1 number of the PCG calls (when doing Gibbs sampling, the number of PCG calls is equal to the number of rounds of the sampler)

2 number of iterations the PCG has used to solve the MME

3 CPU time in seconds used to solve the MME

4 ratio of column two and column three, thus stating how many iterations have been preformed per CPU second

5 as column three, but contains the real time

6 as column four but contains the number of round per real second

Note that when running a BLUP analysis, this file will contain a single line only.

6.2 Result files

Result files are produced only if BESSiE has found the respective optional feature in the parameter file.

6.2.1 Factor level results.

BESSiE can generate files for every single factor in the model containing the results for every level of the respective factor. These files are named "[TraitName][FactorName]Eff.out", where the trait name and the factor name are deduced from the parameter file.
In BLUP mode, each line in these files contains the result of a single factor level, starting with level 1 in line 1 and ending with level N in line N.
In GIBBS mode all levels are printed in one line of the file, and are separated by space. Thus, the number of columns depends on the number of levels of the respective factor, the number of lines on feature PRINTINT of object GIBBSSAMPLER.

6.2.2 Factor level result statistics

In GIBBS mode, BESSiE can generate files for every factor in the model containing the mean and standard deviation of every level of the respective factor. These files are named "[TraitName][FactorName]EffMean.out", where the trait name and the factor name are deduced from the parameter file.
These files have as many lines as the respective factor has levels, and two columns. The first column contains the means, the second the standard deviations of the levels. The recruiting of samples to calculate these values depend on the setting of the feature PRINTINT of object GIBBSSAMPLER. Thus, only those samples are used for calculation which would also be printed into the factor level results file. Note that the generation of these statistics does not depend on the printing of factor level results to the respective files.

6.2.3 Co-variance matrices.

In mode "GIBBS" co-variances of random factors are sampled in every iteration. If the parameter file contains the respective optional directive, the samples are written to a file named "[TraitName1][TraitName2]...[TraitNameN]SampleVariances.out". The file has as many column as diagonal elements and upper off-diagonal elements in unique sub-covariance matrices. The first two lines in that file contain the row and column numbers of these values in the original co-variance matrix, and all subsequent lines the samples variances which replaced the starting values in a particular cycle of the Gibbs sampling.

6.2.4 Variances of SNP effects

Variances of SNP effects generated by a method from the Bayesian Alphabet are written to "[TraitName][EffectName]Var.out". Since the methods are limited to mode GIBBS, the file structure is the same as that of the effect output files in mode GIBBS.

6.2.5 Genomic values of SNP effects

The same as to the above variances applies to the genomic values except that the file name will be "[TraitName][EffectName]GBV.out", and the number of columns will be equal to the number of individuals with genomic values, which is determined by the number of genotypes.

6.2.6 Summarised variance of SNP effects

Values calculated from $\sum_i^N 2p_i(1-p_i)a_i^2$, where N is the number of modelled SNPs, p_i is the minor allele frequency and a_i the effect of SNP i generated by a method from the Bayesian alphabet, will be written to "[TraitName][EffectName]SumSnpVar.out". This file will have one column, and the number of lines depends on the feature PRINTINT of object GIBBSSAMPLER.

6.2.7 BayesCπ specific output

Conditional on METHOD: BAYESCPI, an output file "[TraitName][EffectName]Pi.out" can be generated which contains four columns

- π
- re-calculated locus variance
- re-calculated scale parameter
- number of SNPs with a non-zero effect in the last sampling round.

The number of lines in that file depends on the feature PRINTINT of object GIBBSSAMPLER.

6.2.8 BayesR specific output

Conditional on METHOD: BAYESR, an output file "[TraitName][EffectName]Pi.out" can be generated of which number of columns is $N \times 2$, where N is the number of distributions. The first N columns contain the values of the probability vector drawn for the Dirichlet distribution for the last sampling round, and the second N columns contain the number of SNPs assigned to the different distributions in the last sampling round. The number of lines in this file depends on the feature PRINTINT of object GIBBSSAMPLER.

In addition a file "[TraitName][EffectName]Count.out" can be generated which contains N columns and as many lines as SNPs fitted. Each line contains a vector which reflects how often an SNPs was assigned to one of the N distributions. Note that in the current version these counts will reflect all rounds of the sampler, also the burn-in.

6.2.9 Residuals

Conditional on feature PRINT: RA or PRINT: RB of object "U.D. SINGLE TRAIT", BESSiE will write the residuals for each trait to a file named "[TraitName]Res.out". The format of this file will be the same as in 6.2.1, but the number of lines (MODE: BLUP)/columns (MODE: GIBBS) depends on the number of observations of the respective trait.

7 Recommendations, bugs and workarounds

7.1 Recommendations

7.1.1 Categorical observations

When analysing binary categorical observations, the user should impose a prior degree of freedom > 0 to the starting variances. If the prior degrees of freedom are zero, the prior distribution is improper which, depending on the amount and structure of the data, may result in poor/non convergence of the MCMC chain.

7.2 Bugs

7.2.1 Print bugs

In 6.2.1 it is described how BESSiE writes factor level effects into an output file when operating in mode GIBBS. Essentially, all factor level effects generated in a particular cycle of the Gibbs chain are supposed to appear in a single line of the respective output file. However, due to a compiler bug, a line break is introduced when writing more than 54,000 factor level effects. This bug while also affect output of marker variances because the lines in the related files will have the same length as those in the marker effect files.
An example: The result file of a factor modelling 500,000 genetic markers in a Gibbs chain with 50,000 cycles of which 10,000 are discarded as burn-in and using/printing every 200th sample should have 200 lines ((50000-10000)/200). Instead, it will have 200+x lines. However, the number of lines per printed cycle can be deduced by dividing the total number of lines by the expected number of lines. The expected number of lines may be calculated or derived from output files where such line break is unlikely to appear (e.g. variance file, files of factors with less than 50,000 levels). In the above example this would be (200+x)/200. The result must be an integer in any case, and is the number of file lines per print. Such a file may be read via the R function "scan" with appropriate settings for the "multi.line" argument.

8 Frequently asked questions

- Does BESSiE exploits parallelization?
  - No in general. Yes for the solver. Solving linear mixed models may involve multiplication of a dense matrices with a vector, e.g. when doing GBLUP. In this case, parallelisation may help, but it depends on the matrix dimension whether it will speed up or slow down the process. Thus, BESSiE allows the user set the number of threads for the solver via the parameter file to infer the optimum settings. But the uses should check via the solver output whether that really leads to a decrease in real time.

Bibliography

Albert, J. H. , Chib, S. (1993) Bayesian analysis of binary and polychotomous response data. *J. Am. Stat. Assoc.*, **88**, 669 – 679.

Erbe, M., Hayes, B. J., Matukumalli, L. K., Goswami, S., Bowman, P. J., Reich, C. M., Mason, B. A. , Goddard, M. E. (2012) Improving accuracy of genomic predictions within and between dairy cattle breeds with imputed high-density single nucleotide polymorphism panels. *J Dairy Sci*, **95**(7), 4114–4129.

Habier, D., Fernando, R. L., Kizilkaya, K. , Garrick, D. J. (2011) Extension of the bayesian alphabet for genomic selection. *BMC Bioinformatics*, **12**, 186.

Meuwissen, T. H., Hayes, B. J. , Goddard, M. E. (2001) Prediction of total genetic value using genome-wide dense marker maps. *Genetics*, **157**(4), 1819–1829.

Sorensen, D. , Gianola, D. (2002) *Likelihood, Bayesian, and MCMC Methods in Quantitative Genetics*. Springer, New York.

VanRaden, P. M. (2008) Efficient methods to compute genomic predictions. *J. Dairy Sci.*, **91**(11), 4414–4423.

Yang, J., Benyamin, B., McEvoy, B. P., Gordon, S., Henders, A. K., Nyholt, D. R., Madden, P. A., Heath, A. C., Martin, N. G., Montgomery, G. W., Goddard, M. E. , Visscher, P. M. (2010) Common snps explain a large proportion of the heritability for human height. *Nat Genet*, **42**(7), 565–569.