

# Package ‘StageWise’

December 12, 2021

**Title** Two-stage analysis of multi-environment trials for genomic selection and GWAS

**Version** 0.13

**Author** Jeffrey B. Endelman

**Maintainer** Jeffrey Endelman <endelman@wisc.edu>

**Description** Two-stage analysis of multi-environment trials for genomic selection and GWAS

**Depends** R (>= 4.0)

**License** GPL-3

**LazyData** true

**RoxygenNote** 7.1.1

**Encoding** UTF-8

**Imports** Matrix, ggplot2, methods, ggrepel, rlang, ggpubr, SpATS, spam

**Suggests** knitr, rmarkdown, asreml

**Collate** 'Stage1.R'

'Stage2.R'

'blup.R'

'blup\_prep.R'

'class\_geno.R'

'class\_prep.R'

'class\_var.R'

'gwas\_threshold.R'

'manhattan\_plot.R'

'predict.geno.R'

'private\_functions.R'

'quantile.geno.R'

'read\_geno.R'

'summary.var.R'

## R topics documented:

blup . . . . .	2
blup_prep . . . . .	3
class_geno-class . . . . .	3
class_prep-class . . . . .	4
class_var-class . . . . .	4
gwas_threshold . . . . .	5
manhattan_plot . . . . .	5

predict . . . . .	6
quantile . . . . .	6
read_genotype . . . . .	7
Stage1 . . . . .	7
Stage2 . . . . .	9
summary . . . . .	10

<b>Index</b>	<b>11</b>
--------------	-----------

---

blup	<i>BLUP</i>
------	-------------

---

## Description

BLUP

## Usage

```
blup(data, geno = NULL, what, index.weights = NULL, gwas.ncore = 0L)
```

## Arguments

data	object of <code>class_prep</code> from <code>blup_prep</code>
geno	object of <code>class_genotype</code> from <code>read_genotype</code>
what	"id" or "marker"
index.weights	named vector of index weights for the locations or traits
gwas.ncore	Integer indicating number of cores to use for GWAS (default is 0 for no GWAS). Requires <code>what="markers"</code> .

## Details

Argument `what="id"` leads to prediction of breeding values (BV) and genotypic values (GV), including the average fixed effect of the environments and any fixed effect markers. For `what="marker"`, environment fixed effects are not included in the BLUP. Argument `index.weights` is a named vector (matching the names of the locations or traits), and the values are interpreted for standardized traits. An overall scaling factor is also applied so that the sum of the squared index coefficients equals 1.

## Value

Data frames of BLUPs

---

blup_prep	<i>Prepare data for BLUP</i>
-----------	------------------------------

---

### Description

Prepare data for BLUP

### Usage

```
blup_prep(data, vcov = NULL, geno = NULL, vars, mask = NULL)
```

### Arguments

data	data frame of BLUEs from Stage 1
vcov	list of variance-covariance matrices for the BLUEs
geno	object of <a href="#">class_geno</a> from <a href="#">read_geno</a>
vars	object of <a href="#">class_var</a> from <a href="#">Stage2</a>
mask	(optional) data frame with possible columns "id","env","trait"

### Value

Object of [class\\_prep](#)

---

class_geno-class	<i>S4 class for marker genotype data</i>
------------------	--

---

### Description

S4 class for marker genotype data

### Slots

map	Marker map positions
coeff	Coefficients of the marker effects (dim: indiv x marker)
scale	Scaling factor between markers and indiv
G	Additive (genomic) relationship matrix
eigen.G	list of eigenvalues and eigenvectors

---

class_prep-class	<i>S4 class to prepare for blup</i>
------------------	-------------------------------------

---

### Description

S4 class to prepare for blup

### Slots

y Stage 1 BLUEs  
 Z Incidence matrix for random effects  
 id genotype identifiers  
 var.u variance of random effects  
 Vinv inverted covariance matrix of the Stage 1 BLUEs  
 Pmat P matrix from Searle  
 fixed fixed effect estimates  
 random random effect estimates  
 add matrix of additive variances from [class\\_var](#)  
 loc.env data frame with loc, env  
 trait.env data frame with trait, env  
 index.scale sqrt of genetic variances for the locations/traits  
 fixed.marker names of fixed effect markers

---

class_var-class	<i>S4 class for variances</i>
-----------------	-------------------------------

---

### Description

S4 class for variances

### Slots

add additive  
 g.resid genetic residual  
 resid residual  
 meanG mean of diagonal of G  
 meanOmega mean of diagonal of Omega  
 fixed.marker.var variance of marker fixed effects  
 fixed.marker.cov contribution of marker fixed effects to additive covariance between locations

---

gwas_threshold	<i>Compute GWAS discovery threshold</i>
----------------	---

---

**Description**

Compute GWAS discovery threshold

**Usage**

```
gwas_threshold(geno, alpha = 0.05, exclude.chrom = NULL, n.core = 1)
```

**Arguments**

geno	object of <a href="#">class_geno</a>
alpha	genome-wide significance level
exclude.chrom	chromosomes to exclude
n.core	number of cores to use

**Details**

Uses a Bonferroni-type correction based on an effective number of markers that accounts for LD (Moskvina and Schmidt, 2008).

**Value**

$-\log_{10}(p)$  threshold

**References**

Moskvina V, Schmidt KM (2008) On multiple-testing correction in genome-wide association studies. Genetic Epidemiology 32:567-573. doi:10.1002/gepi.20331

---

manhattan_plot	<i>Create Manhattan plot</i>
----------------	------------------------------

---

**Description**

Create Manhattan plot

**Usage**

```
manhattan_plot(data, chrom = NULL, thresh = NULL, rotate.label = FALSE)
```

**Arguments**

data	data frame with columns for marker, chrom, position, and gwas.score
chrom	optional, to plot only one chromosome
thresh	optional, to include horizontal line at discovery threshold
rotate.label	TRUE/FALSE whether to rotate x-axis labels to be perpendicular

**Details**

Assumes position in bp

**Value**

ggplot2 object

---

predict	<i>Predict additive (breeding) values from marker effects</i>
---------	---

---

**Description**

Predict additive (breeding) values from marker effects

**Arguments**

object                    object of `class_geno`  
marker.effects   data frame with columns "marker","add.effect"

**Details**

Use the `blup` function with `what="marker"` to generate the data frame for `marker.effects`.

**Value**

data frame with columns "id", "BV"

---

quantile	<i>G matrix quantile</i>
----------	--------------------------

---

**Description**

G matrix quantile

**Arguments**

x                        object of `class_geno`  
prob                    probability

**Details**

Unlike the S3 method, `prob` must have `length = 1`

**Value**

data frame with the quantile of the G matrix coefficients for each id

---

read_genotype	<i>Read marker genotype data</i>
---------------	----------------------------------

---

### Description

Read marker genotype data

### Usage

```
read_genotype(filename, ploidy, map, eigen.tol = 1e-09)
```

### Arguments

filename	Name of CSV file
ploidy	2,4,6,etc. (even numbers)
map	TRUE/FALSE
eigen.tol	1e-9

### Details

When map=TRUE, first three columns of the file are marker, chrom, position. When map=FALSE, the first column is marker. Subsequent columns contain the allele dosage for individuals/clones, coded 0,1,2,...ploidy (fractional values are allowed). The input file for diploids can also be coded using -1,0,1 (fractional values allowed). Additive coefficients are computed by subtracting the population mean from each marker, and the additive (genomic) relationship matrix is computed as  $G = \text{tcrossprod}(\text{coeff})/\text{scale}$ . The scale parameter ensures the mean of the diagonal elements of G equals 1 under panmictic equilibrium. Missing genotype data is replaced with the population mean. For numerical conditioning, eigenvalues of G smaller than eigen.tol are replaced by eigen.tol. Monomorphic markers are removed.

### Value

Variable of class `class_genotype`.

---

Stage1	<i>Stage 1 analysis of multi-environment trials</i>
--------	---

---

### Description

Computes genotype BLUEs within each environment

**Usage**

```

Stage1(
  filename,
  traits,
  effects = NULL,
  solver = "asreml",
  spline = NULL,
  silent = TRUE,
  workspace = c("500mb", "500mb")
)

```

**Arguments**

filename	Name of CSV file
traits	trait names (see Details)
effects	data frame specifying other effects in the model (see Details)
solver	one of the following: "asreml", "spats"
spline	vector of variable names for 2D spline with SpATS
silent	TRUE/FALSE, whether to suppress REML output
workspace	memory limits for ASRreml-R

**Details**

The input file must have one column labeled "id" for the individuals and one labeled "env" for the environments. The data for each environment are analyzed independently with a linear mixed model. Although not used in Stage1, to include a genotype x location effect in [Stage2](#), a column labeled "loc" should be present in the input file.

Argument effects is used to specify other i.i.d. effects besides genotype and has three columns: name, fixed, factor. The "name" column is a string that must match a column in the input file. The fixed column is a logical variable to indicate whether the effect is fixed (TRUE) or random (FALSE). The factor column is a logical variable to indicate whether the effect is a factor (TRUE) or numeric (FALSE).

Argument solver specifies which software to use for REML. Current options are "asreml" and "spats". For "spats", the argument spline must be a vector of length two, with the names of the x and y variables (respectively) for the 2D spline.

The heritability and residuals in the output are based on a random effects model for id.

Missing response values are omitted for single-trait analysis but retained for multi-trait analysis (unless both traits are missing), to allow for prediction in Stage 2.

Argument workspace is a vector of length two containing the workspace and pworkspace limits for ASReML-R, with default values of 500mb. If you get an error about insufficient memory, try increasing the appropriate value (workspace for variance estimation and pworkspace for BLUE computation).

For multiple traits, only "asreml" is supported, and only the BLUE model is run, so the returned object does not contain H2 or resid.

**Value**

List containing



**blue** data frame of BLUEs for all environments  
**vcov** list of variance-covariance matrices for the BLUEs, one per env  
**H2** broad-sense H2 on a plot basis  
**resid** list containing diagnostic plots and data frame of residuals  
**aic** AIC for the BLUE model (only with asreml)

Stage2

*Stage 2 analysis of multi-environment trials*

## Description

Stage 2 analysis of multi-environment trials

## Usage

```
Stage2(
  data,
  vcov = NULL,
  geno = NULL,
  fix.eff.marker = NULL,
  silent = TRUE,
  workspace = "500mb"
)
```

## Arguments

<code>data</code>	data frame of BLUEs from Stage 1 (see Details)
<code>vcov</code>	named list of variance-covariance matrices for the BLUEs
<code>geno</code>	output from <a href="#">read_geno</a>
<code>fix.eff.marker</code>	markers in <code>geno</code> to include as additive fixed effect covariates
<code>silent</code>	TRUE/FALSE, whether to suppress ASReml-R output
<code>workspace</code>	Memory limit for ASReml-R variance estimation

## Details

Stage 2 of the two-stage approach described by Damesa et al. 2017, using ASReml-R for variance component estimation. The variable `data` has three mandatory column: `id`, `env`, `BLUE`. Optionally, `data` can have a column labeled `"loc"`, which changes the main effect for genotype into a separable genotype-within-location effect, using a FA2 covariance model for the locations. Optionally, `data` can have a column labeled `"trait"`, which uses an unstructured covariance model. The multi-location and multi-trait analyses cannot be combined. Missing data are allowed in the multi-trait but not the single-trait analysis. The argument `geno` is used to partition genetic values into additive and non-additive (`g.resid`) components. Any individuals in `data` that are not present in `geno` are discarded.

The argument `vcov` is used to partition the macro- and micro-environmental variation, which are called GxE and residual in the output. `vcov` is a named list of variance-covariance matrices for the BLUEs within each environment, with `id` for rownames (single trait) or `id:trait`. The order in `vcov` and `data` should match. Both `data` and `vcov` can be created using the function [Stage1](#).

Because ASReml-R can only use relationship matrices defined in the global environment, this function creates and then removes global variables when either `vcov` or `geno` is used. By default, the workspace memory for ASReml-R is set at 500mb. If you get an error about insufficient memory, try increasing it. ASReml-R version 4.1.0.148 or later is required.

**Value**

List containing

**aic** AIC

**vars** variances, as variable of class `class_var`

**fixed** Fixed effect estimates for env and markers

**random** Random effect predictions

**uniplot** uniplot of the genetic correlation between locations

**References**

Damesa et al. 2017. Agronomy Journal 109: 845-857. doi:10.2134/agronj2016.07.0395

---

summary	<i>Summarize variances and correlations</i>
---------	---

---

**Description**

Summarize variances and correlations

**Arguments**

object                      object of `class_var`

**Details**

When reporting the partitioning of variance, the variance component for the additive effect is multiplied by the mean diagonal of the G matrix.

**Value**

For univariate analysis, a matrix with the proportion of variance ( $R^2$ ). For multi-location or multi-trait analysis, a correlation matrix is also returned. If a G matrix was included in the analysis, the correlation is between additive values; otherwise, it is between genotypic values. If fixed effect markers are present, the additive correlation is above the diagonal, and below the diagonal is the proportion of additive covariance due to the fixed effect markers.

# Index

blup, [2](#), [6](#)  
blup\_prep, [2](#), [3](#)  
  
class\_geno, [2](#), [3](#), [5–7](#)  
class\_geno (class\_geno-class), [3](#)  
class\_geno-class, [3](#)  
class\_prep, [2](#), [3](#)  
class\_prep (class\_prep-class), [4](#)  
class\_prep-class, [4](#)  
class\_var, [3](#), [4](#), [10](#)  
class\_var (class\_var-class), [4](#)  
class\_var-class, [4](#)  
  
gwas\_threshold, [5](#)  
  
manhattan\_plot, [5](#)  
  
predict, [6](#)  
  
quantile, [6](#)  
  
read\_geno, [2](#), [3](#), [7](#), [9](#)  
  
Stage1, [7](#), [9](#)  
Stage2, [3](#), [8](#), [9](#)  
summary, [10](#)