Package 'ProbBreed'

April 5, 2024

Title Probability Theory for Selecting Candidates in Plant Breeding

Version 1.0.3.2

Description
Use probability theory under the Bayesian framework for calculating the risk of selecting candidates in a multi-environment context [Dias et al. (2022) <doi:10.1007/s00122-022-04041-y>]. Contained are functions used to fit a Bayesian multi-environment model (based on the available presets), extract posterior values and maximum posterior values, compute the variance components, check the model's convergence, and calculate the probabilities. For both across and within-environments scopes, the package computes the probability of superior performance and the pairwise probability of superior performance. Furthermore, the probability of superior stability and the pairwise probability of superior stability across environments is estimated. A joint probability of superior performance and stability is also provided.

URL https://github.com/saulo-chaves/ProbBreed,
https://saulo-chaves.github.io/ProbBreed_site/

BugReports https://github.com/saulo-chaves/ProbBreed/issues

License AGPL (>= 3)

Depends R (>= 3.5.0)

Imports ggplot2, rstan, rlang, lifecycle

Suggests knitr, plotly, rmarkdown

Encoding UTF-8

UseLTO true

NeedsCompilation yes

RoxygenNote 7.3.1

LazyData true

Author Saulo Chaves [aut, cre] (<https://orcid.org/0000-0002-0694-1798>),
Kaio Dias [aut, cph] (<https://orcid.org/0000-0002-9171-1021>),
Matheus Krause [aut] (<https://orcid.org/0000-0003-2411-9287>)

Maintainer Saulo Chaves <saulo.chaves@ufv.br>

Repository CRAN

Date/Publication 2024-04-05 18:23:00 UTC

R topics documented:

`bayes_met` *Bayesian model for multi-environment trials*

Description

This function runs a Bayesian model for analyzing data from Multi-environment trials using `rstan`, the R interface to Stan.

Usage

```
bayes_met(
  data,
  gen,
  loc,
  repl,
  trait,
  reg = NULL,
  year = NULL,
  res.het = FALSE,
  iter = 2000,
  cores = 2,
  chains = 4,
  pars = NA,
  warmup = floor(iter/2),
  thin = 1,
  seed = sample.int(.Machine$integer.max, 1),
  init = "random",
  verbose = FALSE,
  algorithm = c("NUTS", "HMC", "Fixed_param"),
  control = NULL,
  include = TRUE,
  show_messages = TRUE,
  ...
)
```

Arguments

data	A data frame containing the observations.
gen, loc	A string. The name of the column that corresponds to the evaluated genotype and location, respectively. If the environment is a combination of other factors (for instance, location-year), the name of the column that contains this information must be attributed to loc.
repl	A string, a vector, or NULL. If the trial is randomized in complete blocks, repl will be a string representing the name of the column that corresponds to the blocks. If the trial is randomized in incomplete blocks design, repl will be a string vector containing the name of the column that corresponds to the replicate and block effects on the first and second positions, respectively. If the data do not have replicates, repl will be NULL.
trait	A string. The name of the column that corresponds to the analysed variable.
reg	A string or NULL. If the data has information of regions, reg will be a string with the name of the column that corresponds to the region information. Otherwise, reg = NULL (default).
year	A string or NULL. If the data set has information of time-related environmental factors (years, seasons...), year will be a string with the name of the column that corresponds to the time information. Otherwise, year = NULL (default).
res.het	Logical, indicating if the model should consider heterogeneous residual variances. Default is FALSE. If TRUE, the model will estimate one residual variance per location.
iter	A positive integer specifying the number of iterations for each chain (including warmup). The default is 2000.
cores	Number of cores to use when executing the chains in parallel, which defaults to 1 but we recommend setting the mc.cores option to be as many processors as the hardware and RAM allow (up to the number of chains).
chains	A positive integer specifying the number of Markov chains. The default is 4.
pars	A vector of character strings specifying parameters of interest. The default is NA indicating all parameters in the model. If include = TRUE, only samples for parameters named in pars are stored in the fitted results. Conversely, if include = FALSE, samples for all parameters *except* those named in pars are stored in the fitted results.
warmup	A positive integer specifying the number of warmup (aka burnin) iterations per chain. If step-size adaptation is on (which it is by default), this also controls the number of iterations for which adaptation is run (and hence these warmup samples should not be used for inference). The number of warmup iterations should be smaller than iter and the default is iter/2.
thin	A positive integer specifying the period for saving samples. The default is 1, which is usually the recommended value.
seed	The seed for random number generation. The default is generated from 1 to the maximum integer supported by R on the machine. Even if multiple chains are used, only one seed is needed, with other chains having seeds derived from that of the first chain to avoid dependent samples. When a seed is specified by a

number, as.integer will be applied to it. If as.integer produces NA, the seed is generated randomly. The seed can also be specified as a character string of digits, such as "12345", which is converted to integer.

init — Initial values specification. See the detailed documentation for the init argument in stan.

verbose — TRUE or FALSE: flag indicating whether to print intermediate output from Stan on the console, which might be helpful for model debugging.

algorithm — One of sampling algorithms that are implemented in Stan. Current options are "NUTS" (No-U-Turn sampler, Hoffman and Gelman 2011, Betancourt 2017), "HMC" (static HMC), or "Fixed_param". The default and preferred algorithm is "NUTS".

control — A named list of parameters to control the sampler's behavior. See the details in the documentation for the control argument in stan.

include — Logical scalar defaulting to TRUE indicating whether to include or exclude the parameters given by the pars argument. If FALSE, only entire multidimensional parameters can be excluded, rather than particular elements of them.

show_messages — Either a logical scalar (defaulting to TRUE) indicating whether to print the summary of Informational Messages to the screen after a chain is finished or a character string naming a path where the summary is stored. Setting to FALSE is not recommended unless you are very sure that the model is correct up to numerical error.

... — Additional arguments can be chain_id, init_r, test_grad, append_samples, refresh, enable_random_init. See the documentation in stan.

Details

More details about the usage of bayes_met and other function of the ProbBreed package can be found at https://saulo-chaves.github.io/ProbBreed_site/. Information on solutions to solve convergence or mixing issue can be found at https://mc-stan.org/misc/warnings.html.

Value

An object of S4 class stanfit representing the fitted results. Slot mode for this object indicates if the sampling is done or not.

Methods

sampling signature(object = "stanmodel") Call a sampler (NUTS, HMC, or Fixed_param depending on parameters) to draw samples from the model defined by S4 class stanmodel given the data, initial values, etc.

See Also

rstan::sampling(), rstan::stan(), rstan::stanfit()

Examples

```
mod = bayes_met(data = maize,
                gen = "Hybrid",
                loc = "Location",
                repl = c("Rep", "Block"),
                year = NULL,
                reg = 'Region',
                res.het = FALSE,
                trait = 'GY',
                iter = 6000, cores = 4, chains = 4)
```

`extr_outs` *Extracting outputs from* `bayes_met()` *objects*

Description

This function extracts outputs of the Bayesian model fitted using `bayes_met()`, and provides some diagnostics about the model

Usage

```
extr_outs(
  data,
  trait,
  model,
  probs = c(0.025, 0.975),
  check.stan.diag = TRUE,
  verbose = FALSE,
  ...
)
```

Arguments

`data`	A data frame containing the observations
`trait`	A character representing the name of the column that corresponds to the analysed trait
`model`	An object containing the Bayesian model fitted using `rstan`
`probs`	A vector with two elements representing the probabilities (in decimal scale) that will be considered for computing the quantiles.
`check.stan.diag`	A logical value indicating whether the function should extract some diagnostic using native `rstan` functions.
`verbose`	A logical value. If `TRUE`, the function will indicate the completed steps. Defaults to `FALSE`
`...`	Passed to `rstan::stan_diag()`

Details

More details about the usage of `extr_outs`, as well as the other function of the `ProbBreed` package can be found at https://saulo-chaves.github.io/ProbBreed_site/.

Value

The function returns a list with:

- `post` : a list with the posterior of the effects, and the data generated by the model
- `map` : a list with the maximum posterior values of each effect
- `ppcheck` : a matrix containing the p-values of maximum, minimum, median, mean and standard deviation; effective number of parameters, WAIC2 value, Rhat and effective sample size.
- `plots` : a list with three types of ggplots: histograms, trace plots and density plots. These will be available for all effects declared at the `effects` argument.
- `stan_plots`: If `check.stan.diag = TRUE`, a list with plots generated by `rstan::stan_diag()`

See Also

`rstan::stan_diag()`, `ggplot2::ggplot()`, `rstan::check_hmc_diagnostics()`

Examples

```
mod = bayes_met(data = maize,
                gen = "Hybrid",
                loc = "Location",
                repl = c("Rep", "Block"),
                year = NULL,
                reg = 'Region',
                res.het = FALSE,
                trait = 'GY',
                iter = 6000, cores = 4, chains = 4)

outs = extr_outs(data = maize, trait = "GY", model = mod,
                 probs = c(0.05, 0.95),
                 check.stan.diag = TRUE,
                 verbose = TRUE)
```

---

maize *Maize real data set*

Description

This dataset belongs to value of cultivation and use maize trials of Embrapa Maize and Sorghum, and was used by Dias et al. (2022). It contains the grain yield of 32 single-cross hybrids and four commercial checks (36 genotypes in total) evaluated in 16 locations across five regions or mega-environments. These trials were laid out in incomplete blocks design, using a block size of 6 and two replications per trial.

Usage

```
maize
```

Format

`maize`:

A data frame with 823 rows and 6 columns:

Location 16 locations

Region 5 regions

Rep 2 replicates

Block 6 blocks

Hybrid 36 genotypes

GY Grain yield (phenotypes)

Source

Dias, K. O. G, Santos J. P. R., Krause, M. D., Piepho H. -P., Guimarães, L. J. M., Pastina, M. M., and Garcia, A. A. F. (2022). Leveraging probability concepts for cultivar recommendation in multi-environment trials. *Theoretical and Applied Genetics*, 133(2):443-455. doi:10.1007/s0012202204041y

---

prob_sup *Probabilities of superior performance and stability*

Description

This function estimates the probabilities of superior performance and stability across environments (`marginal` output). It also computes the probabilities of superior performance within environments (`conditional` output).

Usage

```
prob_sup(
  data,
  trait,
  gen,
  loc,
  reg = NULL,
  year = NULL,
  mod.output,
  int,
  increase = TRUE,
  save.df = FALSE,
  interactive = FALSE,
  verbose = FALSE
)
```

Arguments

`data`	A data frame containing the phenotypic data
`trait, gen, loc`	A string. The name of the columns that correspond to the trait, genotype and location information, respectively. If the environment is a combination of other factors (for instance, location-year), the name of the column that contains this information must be attributed to `loc`.
`reg`	A string or NULL. If the dataset has information about regions, `reg` will be a string with the name of the column that corresponds to the region information. Otherwise, `reg = NULL` (default).
`year`	A string or NULL. If the data set has information about time-related environmental factors (years, seasons...), `year` will be a string with the name of the column that corresponds to the time information. Otherwise, `year = NULL` (default).
`mod.output`	An object from the `extr_outs()` function
`int`	A number representing the selection intensity (between 0 and 1)
`increase`	Logical. Indicates the direction of the selection. `TRUE` (default) for increasing the trait value, `FALSE` otherwise.
`save.df`	Logical. Should the data frames be saved in the work directory? `TRUE` for saving, `FALSE` (default) otherwise.
`interactive`	Logical. Should ggplots be converted into interactive plots? If `TRUE`, the function loads the `plotly` package and uses the `plotly::ggplotly()` command.
`verbose`	A logical value. If `TRUE`, the function will indicate the completed steps. Defaults to `FALSE`.

Details

Probabilities provide the risk of recommending a selection candidate for a target population of environments or for a specific environment. The function `prob_sup()` computes the probabilities of superior performance and the probabilities of superior stability:

- Probability of superior performance

Let Ω represent the subset of selected genotypes based on their performance across environments. A given genotype j will belong to Ω if its genotypic marginal value ($\hat{g}_j$) is high or low enough compared to its peers. `prob_sup()` leverages the Monte Carlo discretized sampling from the posterior distribution to emulate the occurrence of S trials. Then, the probability of the j^{th} genotype belonging to Ω is the ratio of success ($\hat{g}_j \in \Omega$) events and the total number of sampled events, as follows:

$$Pr(\hat{g}_j \in \Omega|y) = \frac{1}{S}\sum_{s=1}^{S} I(\hat{g}_j^{(s)} \in \Omega|y)$$

where S is the total number of samples ($s = 1, 2, ..., S$), and $I(g_j^{(s)} \in \Omega|y)$ is an indicator variable that can assume two values: (1) if $\hat{g}_j^{(s)} \in \Omega$ in the s^{th} sample, and (0) otherwise. S is conditioned to the number of iterations and chains previously set at `bayes_met()`.

Similarly, the conditional probability of superior performance can be applied to individual environments. Let Ω_k represent the subset of superior genotypes in the k^{th} environment, so that the probability of the $j^{th} \in \Omega_k$ can calculated as follows:

$$Pr(\hat{g}_{jk} \in \Omega_k|y) = \frac{1}{S}\sum_{s=1}^{S} I(\hat{g}_{jk}^{(s)} \in \Omega_k|y)$$

where $I(\hat{g}_{jk}^{(s)} \in \Omega_k|y)$ is an indicator variable mapping success (1) if $\hat{g}_{jk}^{(s)}$ exists in Ω_k, and failure (0) otherwise, and $\hat{g}_{jk}^{(s)} = \hat{g}_{j}^{(s)} + \widehat{ge}_{jk}^{(s)}$. Note that when computing conditional probabilities (i.e., conditional to the k^{th} environment or mega-environment), we are accounting for the interaction of the j^{th} genotype with the k^{th} environment.

The pairwise probabilities of superior performance can also be calculated across or within environments. This metric assesses the probability of the j^{th} genotype being superior to another experimental genotype or a commercial check. The calculations are as follows, across and within environments, respectively:

$$Pr(\hat{g}_{j} > \hat{g}_{j'}|y) = \frac{1}{S}\sum_{s=1}^{S} I(\hat{g}_{j}^{(s)} > \hat{g}_{j'}^{(s)}|y)$$

or

$$Pr(\hat{g}_{jk} > \hat{g}_{j'k}|y) = \frac{1}{S}\sum_{s=1}^{S} I(\hat{g}_{jk}^{(s)} > \hat{g}_{j'k}^{(s)}|y)$$

These equations are set for when the selection direction is positive. If `increase = FALSE`, $>$ is simply switched by $<$.

- Probability of superior stability

Probabilities of superior performance highlight experimental genotypes with high agronomic stability. For ecological stability (invariance), the probability of superior stability is the more adequate. Making a direct analogy with the method of Shukla (1972), a stable genotype is the one that has a low variance of the GEI (genotype-by-environment interaction) effects $[var(\widehat{ge})]$. Using the same probability principles previously described, the probability of superior stability is given as follows:

$$Pr[var(\widehat{ge}_{jk}) \in \Omega|y] = \frac{1}{S}\sum_{s=1}^{S} I[var(\widehat{ge}_{jk}^{(s)}) \in \Omega|y]$$

where $I[var(\widehat{ge}_{jk}^{(s)}) \in \Omega|y]$ indicates if $var(\widehat{ge}_{jk}^{(s)})$ exists in Ω (1) or not (0). Pairwise probabilities of superior stability are also possible in this context:

$$Pr[var(\widehat{ge}_{jk}) < var(\widehat{ge}_{j'k})|y] = \frac{1}{S}\sum_{s=1}^{S} I[var(\widehat{ge}_{jk})^{(s)} < var(\widehat{ge}_{j'k})^{(s)}|y]$$

Note that j will be superior to j' if it has a **lower** variance of the genotype-by-environment interaction effect. This is true regardless if `increase` is set to `TRUE` or `FALSE`.

The joint probability independent events is the product of the individual probabilities. The estimated genotypic main effects and the variances of GEI effects are independent by design, thus the joint probability of superior performance and stability as follows:

$$Pr[\hat{g}_j \in \Omega \cap var(\widehat{ge}_{jk}) \in \Omega] = Pr(\hat{g}_j \in \Omega) \times Pr[var(\widehat{ge}_{jk}) \in \Omega]$$

The estimation of these probabilities are strictly related to some key questions that constantly arises in plant breeding:

- **What is the risk of recommending a selection candidate for a target population of environments?**
- **What is the probability of a given selection candidate having good performance if recommended to a target population of environments? And for a specific environment?**
- **What is the probability of a given selection candidate having better performance than a cultivar check in the target population of environments? And in specific environments?**
- **How probable is it that a given selection candidate performs similarly across environments?**
- **What are the chances that a given selection candidate is more stable than a cultivar check in the target population of environments?**
- **What is the probability that a given selection candidate having a superior and invariable performance across environments?**

More details about the usage of `prob_sup`, as well as the other function of the `ProbBreed` package can be found at https://saulo-chaves.github.io/ProbBreed_site/.

Value

The function returns two lists, one with the `marginal` probabilities, and another with the `conditional` probabilities.

The `marginal` list has:

- `df` : A list of data frames containing the calculated probabilities:
  - `perfo`: the probabilities of superior performance.
  - `pair_perfo`: the pairwise probabilities of superior performance.
  - `stabi`: the probabilities of superior stability. Can be `stabi_gl`, `stabi_gm` (when `reg` is not `NULL`) or `stabi_gt` (when `year` is not `NULL`).
  - `pair_stabi`: the pairwise probabilities of superior stability. Can be `pair_stabi_gl`, `pair_stabi_gm` (when `reg` is not `NULL`) or `pair_stabi_gt` (when `year` is not `NULL`).
  - `joint_prob`: the joint probabilities of superior performance and stability.
- `plot` : A list of ggplots illustrating the outputs:
  - `g_hpd`: a caterpillar plot representing the marginal genotypic value of each genotype, and their respective highest posterior density interval (95% represented by the thick line, and 97.5% represented by the thin line).
  - `perfo`: a bar plot illustrating the probabilities of superior performance
  - `pair_perfo`: a heatmap representing the pairwise probability of superior performance (the probability of genotypes at the *x*-axis being superior to those on the *y*-axis).

  - `stabi`: a bar plot with the probabilities of superior stability. Different plots are generated for `stabi_gl`, `stabi_gm` and `stabi_gt` if `reg` or/and `year` are not `NULL`.
  - `pair_stabi`: a heatmap with the pairwise probabilities of superior stability. Different plots are generated for `stabi_gl`, `stabi_gm` and `stabi_gt` if `reg` or/and `year` are not `NULL`. This plot represents the probability of genotypes at the *x*-axis being superior to those on *y*-axis.
  - `joint_prob`: a plot with the probabilities of superior performance, probabilities of superior stability and the joint probabilities of superior performance and stability.

The `conditional` list has:

- `df` : A list with:
  - `prob`: data frames containing the probabilities of superior performance within environments. Can be `prob_loc`, `prob_reg` (if `reg` is not `NULL`), and `prob_year` (if `year` is not `NULL`).
  - `pwprob`: lists with the pairwise probabilities of superior performance within environments. Can be `pwprob_loc`, `pwprob_reg` (if `reg` is not `NULL`), and `pwprob_year` (if `year` is not `NULL`).
- `plot` : A list with:
  - `prob`: heatmaps with the probabilities of superior performance within environments. Can be `prob_loc`, `prob_reg` (if `reg` is not `NULL`), and `prob_year` (if `year` is not `NULL`).
  - `pwprob`: a list of heatmaps representing the pairwise probability of superior performance within environments. Can be `pwprob_loc`, `pwprob_reg` (if `reg` is not `NULL`), and `pwprob_year` (if `year` is not `NULL`). The interpretation is the same as in the `pair_perfo` in the `marginal` list: the probability of genotypes at the *x*-axis being superior to those on *y*-axis.

References

Dias, K. O. G, Santos J. P. R., Krause, M. D., Piepho H. -P., Guimarães, L. J. M., Pastina, M. M., and Garcia, A. A. F. (2022). Leveraging probability concepts for cultivar recommendation in multi-environment trials. *Theoretical and Applied Genetics*, 133(2):443-455. doi:10.1007/s00122-02204041y

Shukla, G. K. (1972) Some statistical aspects of partioning genotype environmental componentes of variability. *Heredity*, 29:237-245. doi:10.1038/hdy.1972.87

Examples

```
mod = bayes_met(data = maize,
                gen = "Hybrid",
                loc = "Location",
                repl = c("Rep", "Block"),
                year = NULL,
                reg = 'Region',
                res.het = FALSE,
                trait = 'GY',
                iter = 6000, cores = 4, chains = 4)

outs = extr_outs(data = maize, trait = "GY", model = mod,
```

```
                  probs = c(0.05, 0.95),
                  check.stan.diag = TRUE,
                  verbose = TRUE)

results = prob_sup(data = maize,
                   trait = "GY",
                   gen = "Hybrid",
                   loc = "Location",
                   reg = 'Region',
                   year = NULL,
                   mod.output = outs,
                   int = .2,
                   increase = TRUE,
                   save.df = FALSE,
                   interactive = FALSE,
                   verbose = FALSE)
```

`soy` *Soybean real data set*

Description

This dataset belongs to the USDA Northern Region Uniform Soybean Tests, and it is a subset of the data used by Krause et al. (2023). It contains the empirical best linear unbiased estimates of genotypic means of the seed yield from 39 experimental genotypes evaluated in 14 locations across three regions or mega-environments. The original data, available at the package `SoyURT`, has 4,257 experimental genotypes evaluated at 63 locations and 31 years resulting in 591 location-year combinations (environments) with 39,006 yield values.

Usage

```
soy
```

Format

`soy`**:**

A data frame with 823 rows and 6 columns:

Loc 14 locations

Reg Regions containing the evaluated environments: 1, 2 and 3

Gen 39 experimental genotypes

Y 435 EBLUEs (phenotypes)

Source

Krause MD, Dias KOG, Singh AK, Beavis WD. 2023. Using soybean historical field trial data to study genotype by environment variation and identify mega-environments with the integration of genetic and non-genetic factors. bioRxiv : the preprint server for biology. doi: https://doi.org/10.1101/2022.04.11.487885

Index