Package ‘pema’

December 15, 2021

Title Penalized Meta-Analysis

Version 0.1.0

Description Conduct penalized meta-analysis, see Van Lissa & Van Erp (2021). <doi:10.31234/osf.io/6phs5>. In meta-analysis, there are often between-study differences. These can be coded as moderator variables, and controlled for using meta-regression. However, if the number of moderators is large relative to the number of studies, such an analysis may be overfit. Penalized meta-regression is useful in these cases, because it shrinks the regression slopes of irrelevant moderators towards zero.

License GPL (>= 3)

Encoding UTF-8

LazyData true

URL https://github.com/cjvanlissa/pema

RoxygenNote 7.1.2

Biarch true

Depends R (>= 3.4.0), rstan (>= 2.18.1)

Imports methods, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstantools (>= 2.1.1), sn

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

Suggests testthat (>= 3.0.0)

SystemRequirements GNU make

Config/testthat/edition 3

NeedsCompilation yes

Author Caspar J van Lissa [aut, cre] (<https://orcid.org/0000-0002-0808-5024>), Sara J van Erp [aut]

Maintainer Caspar J van Lissa <c.j.vanlissa@uu.nl>

Repository CRAN

Date/Publication 2021-12-15 09:20:02 UTC
## R topics documented:

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pema-package</td>
<td>Conduct penalized meta-regression.</td>
</tr>
<tr>
<td>as.stan</td>
<td>Convert an object to stanfit</td>
</tr>
</tbody>
</table>

### Description

Penalized meta-regression shrinks the regression slopes of irrelevant moderators towards zero (Van Lissa & Van Erp, 2021).

### References


### as.stan

Create a stanfit object from an object for which a method exists, so that all methods for stanfit objects can be used.

#### Usage

```r
as.stan(x, ...)  
```

#### Arguments

- `x`: An object for which a method exists.
- `...`: Arguments passed to or from other methods.

#### Value

An object of class stanfit, as documented in rstan::stan.
Examples

```r
stanfit <- "a"
class(stanfit) <- "stanfit"
brmaobject <- list(fit = stanfit)
class(brmaobject) <- "brma"
converted <- as.stan(brmaobject)
```

---

### bonapersona

Data from 'The behavioral phenotype of early life adversity'

---

### Description

This meta-analysis of rodent studies examined whether early life adversity (ELA) alters cognitive performance in several domains. The data include over 400 independent experiments, involving approximately 8600 animals.

### Usage

```r
bonapersona
```

### Format

A data.frame with 734 rows and 65 columns.

### References


---

### brma

Conduct Bayesian Regularized Meta-Analysis

---

### Description

This function conducts Bayesian regularized meta-regression (Van Lissa & Van Erp, 2021). It uses the `stan` function `rstan::sampling` to fit the model. A lasso or horseshoe prior is used to shrink the regression coefficients of irrelevant moderators towards zero. See Details.
Usage

brma(
  formula,
  data,
  vi = "vi",
  study = NULL,
  method = "hs",
  standardize = TRUE,
  prior = switch(method, lasso = c(df = 1, scale = 1), hs = c(df = 1, df_global = 1, df_slab = 4, scale_global = 1, scale_slab = 1, par_ratio = NULL)),
  mute_stan = TRUE,
  ...
)

Arguments

formula An object of class formula (or one that can be coerced to that class), see \texttt{lm}.
data Optional data.frame containing the variables in the model, see \texttt{lm}.
vi Character. Name of the column in the data that contains the variances of the effect sizes. This column will be removed from the data prior to analysis. Defaults to "vi".
study Character. Name of the column in the data that contains the study id. Use this when the data includes multiple effect sizes per study. This column can be a vector of integers, or a factor. This column will be removed from the data prior to analysis. See Details for more information about analyzing dependent data.
method Character, indicating the type of regularizing prior to use. Supports one of \texttt{c("lasso" ,"hs")}, see Details. Defaults to "lasso".
standardize Logical, indicating whether or not to standardize the predictors (defaults to \texttt{TRUE}, which is recommended so that shrinking affects all parameters similarly.
prior Numeric vector, specifying the prior to use. Note that the different methods require this vector to contain specific named elements.
mute_stan Logical, indicating whether mute all `Stan` output or not.
... Additional arguments passed on to \texttt{rstan::sampling()}. Use this, e.g., to override default arguments of that function.

Details

The Bayesian regularized meta-analysis algorithm (Van Lissa & Van Erp, 2021) penalizes meta-regression coefficients either via the lasso prior (Park & Casella, 2008) or the regularized horseshoe prior (Piironen & Vehtari, 2017).

lasso The Bayesian equivalent of the lasso penalty is obtained when placing independent Laplace (i.e., double exponential) priors on the regression coefficients centered around zero. The scale of the Laplace priors is determined by a global scale parameter \texttt{scale}, which defaults to 1 and an inverse-tuning parameter \frac{1}{\lambda} which is given a chi-square prior governed by a degrees of freedom parameter \texttt{df} (defaults to 1). If \texttt{standardize = TRUE}, shrinkage will affect all
coefficients equally and it is not necessary to adapt the scale parameter. Increasing the df parameter will allow larger values for the inverse-tuning parameter, leading to less shrinkage.

**hs** One issue with the lasso prior is that it has relatively light tails. As a result, not only does the lasso have the desirable behavior of pulling small coefficients to zero, it also results in too much shrinkage of large coefficients. An alternative prior that improves upon this shrinkage pattern is the horseshoe prior (Carvalho, Polson & Scott, 2010). The horseshoe prior has an infinitely large spike at zero, thereby pulling small coefficients toward zero but in addition has fat tails, which allow substantial coefficients to escape the shrinkage. The regularized horseshoe is an extension of the horseshoe prior that allows the inclusion of prior information regarding the number of relevant predictors and can be more numerically stable in certain cases (Piironen & Vehtari, 2017). The regularized horseshoe has a global shrinkage parameter that influences all coefficients similarly and local shrinkage parameters that enable flexible shrinkage patterns for each coefficient separately. The local shrinkage parameters are given a Student’s t prior with a default df parameter of 1. Larger values for df result in lighter tails and a prior that is no longer strictly a horseshoe prior. However, increasing df slightly might be necessary to avoid divergent transitions in Stan (see also https://mc-stan.org/misc/warnings.html).

Similarly, the degrees of freedom for the Student’s t prior on the global shrinkage parameter df_global can be increased from the default of 1 to, for example, 3 if divergent transitions occur although the resulting prior is then strictly no longer a horseshoe. The scale for the Student’s t prior on the global shrinkage parameter scale_global defaults to 1 and can be decreased to achieve more shrinkage. Moreover, if prior information regarding the number of relevant moderators is available, it is recommended to include this information via the par_ratio argument by setting it to the ratio of the expected number of non-zero coefficients to the expected number of zero coefficients. When par_ratio is specified, scale_global is ignored and instead based on the available prior information. Contrary to the horseshoe prior, the regularized horseshoe applies additional regularization on large coefficients which is governed by a Student’s t prior with a scale_slab defaulting to 2 and df_slab defaulting to 4. This additional regularization ensures at least some shrinkage of large coefficients to avoid any sampling problems.

**Value**

A list object of class brma, with the following structure:

```r
list(
  fit # An object of class stanfit, for compatibility with rstan
coefficients # A numeric matrix with parameter estimates; these are
  # interpreted as regression coefficients, except tau2 and tau,
  # which are interpreted as the residual variance and standard
  # deviation, respectively.
formula # The formula used to estimate the model
terms # The predictor terms in the formula
X # Numeric matrix of moderator variables
Y # Numeric vector with effect sizes
vi # Numeric vector with effect size variances
tau2 # Numeric, estimated tau2
R2 # Numeric, estimated heterogeneity explained by the moderators
k # Numeric, number of effect sizes
vi_column # Optional, name of the column in the original data
```
References


Examples

```r
data("curry")
df <- curry[c(1:5, 50:55), c("d", "vi", "sex", "age", "donorcode")]
suppressWarnings({res <- brma(d~., data = df, iter = 10)})
```

Description

A systematic review and meta-analysis of the effects of performing acts of kindness on the well-being of the actor.

Usage

curry

Format

A data.frame with 56 rows and 18 columns.

- **study_id**: factor, Unique identifier of the study
- **effect_id**: integer, Unique identifier of the effect size
- **d**: numeric, Standardized mean difference between the control group and intervention group
- **vi**: numeric, Variance of the effect size
- **n1i**: numeric, Number of participants in the intervention group
- **n1c**: numeric, Number of participants in the control group
- **sex**: numeric, Percentage of male participants

---

Data from 'Happy to Help?'
maxap

### Description

Find the parameter estimate with the highest posterior probability density given a vector of samples.

### Usage

```r
maxap(x, dens = NULL, ...)
```

### Arguments

- `x`  
  Numeric vector.

- `dens`  
  Optional object of class `density`. Defaults to `NULL`.

- `...`  
  Arguments passed to `density`.

### Value

Atomic numeric vector with the maximum a-posteriori estimate of vector `x`.

### Examples

```r
maxap(c(1, 2, 3, 4, 5))
```
Simulate a meta-analytic dataset

**Description**

This function simulates a meta-analytic dataset based on the random-effects model. The simulated effect size is Hedges’ G, an estimator of the Standardized Mean Difference (Hedges, 1981; Li, Dusseldorp, & Meulman, 2017). The functional form of the model can be specified, and moderators can be either normally distributed or Bernoulli-distributed. See Van Lissa, in preparation, for a detailed explanation of the simulation procedure.

**Usage**

```r
simulate_smd(
  k_train = 20,
  k_test = 100,
  mean_n = 40,
  es = 0.5,
  tau2 = 0.04,
  alpha = 0,
  moderators = 5,
  distribution = "normal",
  model = "es * x[, 1]"
)
```

**Arguments**

- `k_train`  Atomic integer. The number of studies in the training dataset. Defaults to 20.
- `k_test`  Atomic integer. The number of studies in the testing dataset. Defaults to 100.
- `mean_n`  Atomic integer. The mean sample size of each simulated study in the meta-analytic dataset. Defaults to 40. For each simulated study, the sample size n is randomly drawn from a normal distribution with mean mean_n, and sd mean_n/3.
- `es`  Atomic numeric vector. The effect size, also known as beta, used in the model statement. Defaults to .5.
- `tau2`  Atomic numeric vector. The residual heterogeneity. For a range of realistic values encountered in psychological research, see Van Erp, Verhagen, Grasman, & Wagenmakers, 2017. Defaults to 0.04.
- `alpha`  Vector of slant parameters, passed to `sn::rsn`.
- `moderators`  Atomic integer. The number of moderators to simulate for each study. Make sure that the number of moderators to be simulated is at least as large as the number of moderators referred to in the model parameter. Internally, the matrix of moderators is referred to as "x". Defaults to 5.
- `distribution`  Atomic character. The distribution of the moderators. Can be set to either "normal" or "bernoulli". Defaults to "normal".
**simulate_smd**

expression to specify the model from which to simulate the mean true effect size, mu. This formula may use the terms "es" (referring to the es parameter of the call to `simulate_smd`), and "x[, ]" (referring to the matrix of moderators, x). Thus, to specify that the mean effect size, mu, is a function of the effect size and the first moderator, one would pass the value `model = "es * x[, 1]"`. Defaults to "es * x[, 1]".

**Value**

List of length 4. The "training" element of this list is a data.frame with k_train rows. The columns are the variance of the effect size, vi; the effect size, yi, and the moderators, X. The "testing" element of this list is a data.frame with k_test rows. The columns are the effect size, yi, and the moderators, X. The "housekeeping" element of this list is a data.frame with k_train + k_test rows. The columns are n, the sample size n for each simulated study; mu_i, the mean true effect size for each simulated study; and theta_i, the true effect size for each simulated study.

**Examples**

```r
set.seed(8)
simulate_smd()
simulate_smd(k_train = 50, distribution = "bernoulli")
simulate_smd(distribution = "bernoulli", model = "es * x[, 1] * x[, 2]"
```
# Index

<table>
<thead>
<tr>
<th><em>datasets</em></th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>bonapersona</td>
<td>3</td>
</tr>
<tr>
<td>curry</td>
<td>6</td>
</tr>
<tr>
<td>as.stan</td>
<td>2</td>
</tr>
<tr>
<td>bonapersona</td>
<td>3</td>
</tr>
<tr>
<td>brma</td>
<td>3</td>
</tr>
<tr>
<td>curry</td>
<td>6</td>
</tr>
<tr>
<td>density</td>
<td>7</td>
</tr>
<tr>
<td>lm</td>
<td>4</td>
</tr>
<tr>
<td>maxap</td>
<td>7</td>
</tr>
<tr>
<td>pema (pema-package)</td>
<td>2</td>
</tr>
<tr>
<td>pema-package</td>
<td>2</td>
</tr>
<tr>
<td>rstan::sampling</td>
<td>3</td>
</tr>
<tr>
<td>rstan::sampling()</td>
<td>4</td>
</tr>
<tr>
<td>rstan::stan</td>
<td>2</td>
</tr>
<tr>
<td>simulate_smd</td>
<td>8</td>
</tr>
<tr>
<td>sn::rsn</td>
<td>8</td>
</tr>
</tbody>
</table>