Package ‘FBMS’

December 21, 2023

Type Package
Title Flexible Bayesian Model Selection and Model Averaging
Version 1.0
Date 2023-12-20
Encoding UTF-8

Description Implements MJMCMC (mode jumping MCMC) described in Hubin and Storvik (2018) <doi:10.1016/j.csda.2018.05.020> and GMJMCMC (genetically modified MJMCMC) described in Hubin et al. (2021) <doi:10.1613/jair.1.13047> algorithms as well as the subsampling counterpart described in Lachmann et al. (2022) <doi:10.1016/j.ijar.2022.08.018> for flexible Bayesian model selection and model averaging.

License GPL-2
Depends R (>= 3.5.0), fastglm, GenSA, parallel, stats, graphics
Imports Rcpp
LinkingTo Rcpp
Suggests testthat, knitr, rmarkdown, markdown
RoxygenNote 7.2.3
VignetteBuilder knitr

Author Jon Lachmann [cre, aut], Aliaksandr Hubin [aut]
Maintainer Jon Lachmann <jon@lachmann.nu>

repository CRAN
Date/Publication 2023-12-21 16:30:11 UTC

R topics documented:

FBMS-package ......................................................... 3
breastcancer ......................................................... 4
compute_effects ....................................................... 5
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>cos_deg</td>
<td>6</td>
</tr>
<tr>
<td>diag_plot</td>
<td>6</td>
</tr>
<tr>
<td>erf</td>
<td>7</td>
</tr>
<tr>
<td>exoplanet</td>
<td>8</td>
</tr>
<tr>
<td>exp_db1</td>
<td>9</td>
</tr>
<tr>
<td>fbms</td>
<td>9</td>
</tr>
<tr>
<td>gauss</td>
<td>11</td>
</tr>
<tr>
<td>gaussian.loglik</td>
<td>11</td>
</tr>
<tr>
<td>gaussian.loglik.alpha</td>
<td>12</td>
</tr>
<tr>
<td>gelu</td>
<td>13</td>
</tr>
<tr>
<td>gen.params.gmjmcmc</td>
<td>13</td>
</tr>
<tr>
<td>gen.params.mjmcmc</td>
<td>14</td>
</tr>
<tr>
<td>gen.probs.gmjmcmc</td>
<td>14</td>
</tr>
<tr>
<td>gen.probs.mjmcmc</td>
<td>15</td>
</tr>
<tr>
<td>gmjmcmc</td>
<td>15</td>
</tr>
<tr>
<td>gmjmcmc.parallel</td>
<td>17</td>
</tr>
<tr>
<td>hs</td>
<td>18</td>
</tr>
<tr>
<td>linear.g.prior.loglik</td>
<td>19</td>
</tr>
<tr>
<td>logistic.loglik</td>
<td>19</td>
</tr>
<tr>
<td>logistic.loglik.alpha</td>
<td>20</td>
</tr>
<tr>
<td>marginal.probs</td>
<td>21</td>
</tr>
<tr>
<td>merge_results</td>
<td>21</td>
</tr>
<tr>
<td>mjjmcmc</td>
<td>23</td>
</tr>
<tr>
<td>mjjmcmc.parallel</td>
<td>24</td>
</tr>
<tr>
<td>model.string</td>
<td>25</td>
</tr>
<tr>
<td>ngeLU</td>
<td>25</td>
</tr>
<tr>
<td>nhs</td>
<td>26</td>
</tr>
<tr>
<td>not</td>
<td>26</td>
</tr>
<tr>
<td>nrelu</td>
<td>27</td>
</tr>
<tr>
<td>p0</td>
<td>27</td>
</tr>
<tr>
<td>p05</td>
<td>28</td>
</tr>
<tr>
<td>p0p0</td>
<td>28</td>
</tr>
<tr>
<td>p0p05</td>
<td>29</td>
</tr>
<tr>
<td>p0p1</td>
<td>29</td>
</tr>
<tr>
<td>p0p2</td>
<td>30</td>
</tr>
<tr>
<td>p0p3</td>
<td>30</td>
</tr>
<tr>
<td>p0pm05</td>
<td>31</td>
</tr>
<tr>
<td>p0pm1</td>
<td>31</td>
</tr>
<tr>
<td>p0pm2</td>
<td>32</td>
</tr>
<tr>
<td>p2</td>
<td>32</td>
</tr>
<tr>
<td>p3</td>
<td>33</td>
</tr>
<tr>
<td>plot.gmjmcmc</td>
<td>33</td>
</tr>
<tr>
<td>plot.gmjmcmc_merged</td>
<td>34</td>
</tr>
<tr>
<td>plot.mjmcmc</td>
<td>35</td>
</tr>
<tr>
<td>plot.mjmcmc_parallel</td>
<td>36</td>
</tr>
<tr>
<td>pm05</td>
<td>36</td>
</tr>
<tr>
<td>pm1</td>
<td>37</td>
</tr>
<tr>
<td>pm2</td>
<td>37</td>
</tr>
</tbody>
</table>
FBMS-package

Flexible Bayesian Model Selection and Model Averaging

Description

Implements MJMCMC (mode jumping MCMC) described in Hubin and Storvik (2018) <doi:10.1016/j.csda.2018.05.020> and GMJMCMC (genetically modified MJMCMC) described in Hubin et al. (2021) <doi:10.1613/jair.1.13047> algorithms as well as the subsampling counterpart described in Lachmann et al. (2022) <doi:10.1016/j.ijar.2022.08.018> for flexible Bayesian model selection and model averaging.

Author(s)

Maintainer: Jon Lachmann <jon@lachmann.nu>

Authors:

• Jon Lachmann <jon@lachmann.nu>
• Aliaksandr Hubin <aliaksah@math.uio.no>

Other contributors:

• Florian Frommlet <florian.frommlet@meduniwien.ac.at> [contributor]
• Geir Storvik <geirs@math.uio.no> [contributor]
References


breastcancer

Breast Cancer Wisconsin (Diagnostic) Data Set

Description

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

Usage

data(breastcancer)

Format

A data frame with 569 rows and 32 variables

Details

Separating plane described above was obtained using Multisurface Method-Tree (MSM-T) (K. P. Bennett, "Decision Tree Construction Via Linear Programming." Proceedings of the 4th Midwest Artificial Intelligence and Cognitive Science Society, pp. 97-101, 1992), a classification method which uses linear programming to construct a decision tree. Relevant features were selected using an exhaustive search in the space of 1-4 features and 1-3 separating planes.

The actual linear program used to obtain the separating plane in the 3-dimensional space is that described in: (K. P. Bennett and O. L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", Optimization Methods and Software 1, 1992, 23-34).

The variables are as follows:

- ID number
- Diagnosis (1 = malignant, 0 = benign)
- Ten real-valued features are computed for each cell nucleus
compute_effects

Source


Creators:

1. Dr. William H. Wolberg, General Surgery Dept. University of Wisconsin, Clinical Sciences Center Madison, WI 53792 wolberg 'at' eagle.surgery.wisc.edu
2. W. Nick Street, Computer Sciences Dept. University of Wisconsin, 1210 West Dayton St., Madison, WI 53706 street 'at' cs.wisc.edu 608-262-6619
3. Olvi L. Mangasarian, Computer Sciences Dept. University of Wisconsin, 1210 West Dayton St., Madison, WI 53706 olvi 'at' cs.wisc.edu

Donor: Nick Street

References


compute_effects

Compute effects for specified in labels covariates using a fitted model.

Description

This function computes model averaged effects for specified covariates using a fitted model object. The effects are expected change in the BMA linear predictor having an increase of the corresponding covariate by one unit, while other covariates are fixed to 0. Users can provide custom labels and specify quantiles for the computation of effects.

Usage

```r
compute_effects(object, labels, quantiles = c(0.025, 0.5, 0.975))
```

Arguments

- **object**: A fitted model object, typically the result of a regression or predictive modeling.
- **labels**: A vector of labels for which effects are to be computed.
- **quantiles**: A numeric vector specifying the quantiles to be calculated. Default is c(0.025, 0.5, 0.975).

Value

A matrix of treatment effects for the specified labels, with rows corresponding to labels and columns to quantiles.
See Also
    predict

Examples

```r
data <- data.frame(matrix(rnorm(600), 100))
result <- mjmcmc.parallel(runs = 2, cores = 1, data, gaussian.loglik)
compute_effects(result, labels = names(data)[-1])
```

---

<table>
<thead>
<tr>
<th>cos_deg</th>
<th>Cosine function for degrees</th>
</tr>
</thead>
</table>

Description

Cosine function for degrees

Usage

```r
cos_deg(x)
```

Arguments

- `x` The vector of values in degrees

Value

The cosine of `x`

Examples

```r
cos_deg(0)
```

---

<table>
<thead>
<tr>
<th>diagn_plot</th>
<th>Plot convergence of best/median/mean/other summary log posteriors in time</th>
</tr>
</thead>
</table>

Description

Plot convergence of best/median/mean/other summary log posteriors in time

Usage

```r
diagn_plot(res, FUN = median, conf = 0.95, burnin = 0, window = 10000)
```
Arguments

res Object corresponding gmjmcmc output
FUN The summary statistics to check convergence
conf which confidence intervals to plot
burnin how many first populations to skip
window sliding window for computing the standard deviation

Value
A list of summary statistics for checking convergence with given confidence intervals

Examples
result <- gmjmcmc(matrix(rnorm(600), 100), P = 2, gaussian.loglik, NULL, c("p0", "exp_dbl"))
diagnstats <- diagn_plot(result)

erf erf function

Description
erf function

Usage
erf(x)

Arguments
x The vector of values

Value
2 * pnorm(x * sqrt(2)) - 1

Examples
erf(2)
Excerpt from the Open Exoplanet Catalogue data set

Description

Data fields include planet and host star attributes.

Usage

data(exoplanet)

Format

A data frame with 223 rows and 11 variables

Details

The variables are as follows:

• TypeFlag: Flag indicating the type of data
• PlanetaryMassJpt: Mass of the planetary object in Jupiter masses
• RadiusJpt: Radius of the planetary object in Jupiter radii
• PeriodDays: Orbital period of the planetary object in days
• SemiMajorAxisAU: Semi-major axis of the planetary object’s orbit in astronomical units
• Eccentricity: Eccentricity of the planetary object’s orbit
• HostStarMassSlrMass: Mass of the host star in solar masses
• HostStarRadiusSlrRad: Radius of the host star in solar radii
• HostStarMetallicity: Metallicity of the host star
• HostStarTempK: Effective temperature of the host star in Kelvin
• PlanetaryDensJpt: Density of the planetary object up to a constant

Source

Dataset downloaded from the Open Exoplanet Catalogue Repository. https://github.com/OpenExoplanetCatalogue/oec_tables/

Creators:

1. Prof. Hanno Rein, Department for Physical and Environmental Sciences. University of Toronto at Scarborough Toronto, Ontario M1C 1A4 hanno.rein 'at' utoronto.ca
**exp_dbl**

*Double exponential function*

**Description**

Double exponential function

**Usage**

```r
exp_dbl(x)
```

**Arguments**

- `x` The vector of values

**Value**

\[ e^{-|x|} \]

**Examples**

```r
exp_dbl(2)
```

---

**fbms**

Fit a BGNLM model using Genetically Modified Mode Jumping Markov Chain Monte Carlo (MCMC) sampling. Or Fit a BGLM model using Modified Mode Jumping Markov Chain Monte Carlo (MCMC) sampling.

**Description**

This function fits a model using the relevant MCMC sampling. The user can specify the formula, family, data, transforms, and other parameters to customize the model.

**Usage**

```r
fbms(
  formula = NULL,
  family = "gaussian",
  data = NULL,
  transforms = NULL,
  loglik.pi = gaussian.loglik,
  loglik.alpha = gaussian.loglik.alpha,
  P = 10,
  runs = 10,
)```

---
cores = 1,
verbose = FALSE,
...
}

Arguments

- **formula**: A formula object specifying the model structure. Default is NULL.
- **family**: The distribution family of the response variable. Currently supports "gaussian" and "binomial". Default is "gaussian".
- **data**: A data frame containing the variables in the model. If NULL, the variables are taken from the environment of the formula. Default is NULL.
- **transforms**: A list of transformations for BGNLM model. Default is NULL.
- **loglik.pi**: The log-likelihood function for estimating the marginal likelihood and posterior modes (only used if family = "custom")
- **loglik.alpha**: The log-likelihood function for the alpha parameter in the model. Default is gaussian.loglik.alpha.
- **P**: The number of GMJMCMC generations. Default is 10.
- **runs**: The number of parallel chains in case of parallel processing. Default is 2.
- **cores**: The number of CPU cores to use for parallel processing. Default is 2.
- **verbose**: If TRUE, print detailed progress information during the fitting process. Default is FALSE.
- **...**: Additional parameters to be passed to the underlying MCMC fitting functions.

Value

An object containing the results of the fitted model and MCMC sampling.

See Also

- [mjmcmc](#), [gmjmcmc](#), [gmjmcmc.parallel](#)

Examples

```r
# Fit a Gaussian multivariate time series model
fbms_result <- fbms(
  X1 ~ .,
  family = "gaussian",
  data = data.frame(matrix(rnorm(600), 100)),
  P = 10,
  runs = 1,
  cores = 1
)
summary(fbms_result)
plot(fbms_result)
```
gauss

Gaussian function

Description
Gaussian function

Usage
gauss(x)

Arguments
x The vector of values

Value
e^{-x^2}

Examples
gauss(2)

gaussian.loglik

Log likelihood function for gaussian regression with a prior p(m)=r*sum(total_width).

Description
Log likelihood function for gaussian regression with a prior p(m)=r*sum(total_width).

Usage
gaussian.loglik(y, x, model, complex, params)

Arguments
y A vector containing the dependent variable
x The matrix containing the precalculated features
model The model to estimate as a logical vector
complex A list of complexity measures for the features
params A list of parameters for the log likelihood, supplied by the user
value

A list with the log marginal likelihood combined with the log prior (crit) and the posterior mode of
the coefficients (coefs).

Examples

gaussian.loglik(rnorm(100), matrix(rnorm(100)), TRUE, list(oc = 1), NULL)

description

Log likelihood function for gaussian regression for alpha calculation
This function is just the bare likelihood function Note that it only gives
a proportional value and is equivalent to least squares

usage

gaussian.loglik.alpha(a, data, mu_func)

arguments

a A vector of the alphas to be used
data The data to be used for calculation
mu_func The function linking the mean to the covariates, as a string with the alphas as a[i].

value

A numeric with the log likelihood.

examples

gaussian.loglik.alpha(1, matrix(rnorm(100), 50), "a * data[, 2]"")
### gelu

**GELU function**

**Description**

GELU function

**Usage**

```r
gelu(x)
```

**Arguments**

- `x` The vector of values

**Value**

```r
x*pnorm(x)
```

**Examples**

```r
gelu(2)
```

---

### gen.params.gmjmcmc

Generate a parameter list for GMJMCMC (Genetically Modified MJMCMC)

**Description**

Generate a parameter list for GMJMCMC (Genetically Modified MJMCMC)

**Usage**

```r
gen.params.gmjmcmc(data)
```

**Arguments**

- `data` The dataset that will be used in the algorithm

**Value**

A list of parameters to use when running the mjmc function.

**Examples**

```r
gen.params.gmjmcmc(matrix(rnorm(600), 100))
```
**Description**
Generate a parameter list for MJMCMC (Mode Jumping MCMC)

**Usage**
```
gen.params.mjmcmc(data)
```

**Arguments**
- **data**
  The dataset that will be used in the algorithm

**Value**
A list of parameters to use when running the mjmcmc function. 
Note that the $loglik$ item is an empty list, which is passed to the log likelihood function of the model, intended to store parameters that the estimator function should use.

**Examples**
```
gen.params.mjmcmc(matrix(rnorm(600), 100))
```

---

**Description**
Generate a probability list for GMJMCMC (Genetically Modified MJMCMC)

**Usage**
```
gen.probs.gmjmcmc(transforms)
```

**Arguments**
- **transforms**
  A list of the transformations used (to get the count).

**Value**
A list of probabilities to be used as input for the gmjmcmc function.
Examples

```r
gen.probs.gmjmcmc(c("p0", "exp_dbl"))
```

---

**gen.probs.mjmcmc**

*Generate a probability list for MJMCMC (Mode Jumping MCMC)*

**Description**

Generate a probability list for MJMCMC (Mode Jumping MCMC)

**Usage**

```r
gen.probs.mjmcmc()
```

**Value**

A list of probabilities to be used as input for the mjmcmc function.

**Examples**

```r
gen.probs.mjmcmc()
```

---

**gmjmcmc**

*Main algorithm for GMJMCMC (Genetically Modified MJMCMC)*

**Description**

Main algorithm for GMJMCMC (Genetically Modified MJMCMC)

**Usage**

```r
gmjmcmc(
data,
    loglik.pi = gaussian.loglik,
    loglik.alpha = gaussian.loglik.alpha,
    transforms,
    P = 10,
    N.init = 100,
    N.final = 100,
    probs = NULL,
    params = NULL,
    sub = FALSE,
    verbose = TRUE)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>data</code></td>
<td>A matrix containing the data to use in the algorithm, first column should be the dependent variable, second should be the intercept and the rest of the columns should be the independent variables.</td>
</tr>
<tr>
<td><code>loglik.pi</code></td>
<td>The (log) density to explore</td>
</tr>
<tr>
<td><code>loglik.alpha</code></td>
<td>The likelihood function to use for alpha calculation</td>
</tr>
<tr>
<td><code>transforms</code></td>
<td>A Character vector including the names of the non-linear functions to be used by the modification and the projection operator.</td>
</tr>
<tr>
<td><code>P</code></td>
<td>The number of generations for GMJMCMC (Genetically Modified MJMCMC). The default value is $P = 10$. A larger value like $P = 50$ might be more realistic for more complicated examples where one expects a lot of non-linear structures.</td>
</tr>
<tr>
<td><code>N.init</code></td>
<td>The number of iterations per population (total iterations = (T-1)*N.init+N.final)</td>
</tr>
<tr>
<td><code>N.final</code></td>
<td>The number of iterations for the final population (total iterations = (T-1)*N.init+N.final)</td>
</tr>
<tr>
<td><code>probs</code></td>
<td>A list of the various probability vectors to use</td>
</tr>
<tr>
<td><code>params</code></td>
<td>A list of the various parameters for all the parts of the algorithm</td>
</tr>
<tr>
<td><code>sub</code></td>
<td>An indicator that if the likelihood is inexact and should be improved each model visit (EXPERIMENTAL!)</td>
</tr>
<tr>
<td><code>verbose</code></td>
<td>A logical denoting if messages should be printed</td>
</tr>
</tbody>
</table>

Value

A list containing the following elements:

- `models` All models per population.
- `lo.models` All local optimization models per population.
- `populations` All features per population.
- `marg.probs` Marginal feature probabilities per population.
- `model.probs` Marginal feature probabilities per population.
- `model.probs.idx` Marginal feature probabilities per population.
- `best.margs` Best marginal model probability per population.
- `accept` Acceptance rate per population.
- `accept.tot` Overall acceptance rate.
- `best` Best marginal model probability throughout the run, represented as the maximum value in `unlist(best.margs)`.

Examples

```r
result <- gmjmcmc(matrix(rnorm(600), 100), P = 2, gaussian.loglik, NULL, c("p0", "exp_dbl"))
summary(result)
plot(result)
```
Run multiple gmjmcmc (Genetically Modified MJMCMC) runs in parallel returning a list of all results.

Usage

```r
gmjmcmc.parallel(
  runs,
  cores = getOption("mc.cores", 2L),
  merge.options = list(populations = "best", complex.measure = 2, tol = 1e-07),
  data,
  loglik.pi = gaussian.loglik,
  loglik.alpha = gaussian.loglik.alpha(),
  transforms,
  ...
)
```

Arguments

- `runs` The number of runs to run
- `cores` The number of cores to run on
- `merge.options` A list of options to pass to the `merge_results()` function run after the
- `data` A matrix containing the data to use in the algorithm, first column should be the
  dependent variable, second should be the intercept and the rest of the columns
  should be the independent variables.
- `loglik.pi` The (log) density to explore
- `loglik.alpha` The likelihood function to use for alpha calculation
- `transforms` A Character vector including the names of the non-linear functions to be used
  by the modification and the projection operator.
- `...` Further params passed to mjmc.

Value

Results from multiple gmjmcmc runs

Examples

```r
result <- gmjmcmc.parallel(
  runs = 1,
  cores = 1,
```
hs

heavy side function

Description

heavy side function

Usage

hs(x)

Arguments

x The vector of values

Value

as.integer(x>0)

Examples

hs(2)
linear.g.prior.loglik  \hspace{1em} Log likelihood function for linear regression using Zellner's g-prior

Description

Log likelihood function for linear regression using Zellner's g-prior

Usage

linear.g.prior.loglik(y, x, model, complex, params = list(g = 4))

Arguments

y  \hspace{1em} A vector containing the dependent variable
x  \hspace{1em} The matrix containing the precalculated features
model  \hspace{1em} The model to estimate as a logical vector
complex  \hspace{1em} A list of complexity measures for the features
params  \hspace{1em} A list of parameters for the log likelihood, supplied by the user

Value

A list with the log marginal likelihood combined with the log prior (crit) and the posterior mode of the coefficients (coefs).

Examples

linear.g.prior.loglik(rnorm(100), matrix(rnorm(100)), TRUE, list(oc=1))

logistic.loglik  \hspace{1em} Log likelihood function for logistic regression with a prior \hspace{1em} \hspace{1em} p(m)=\text{sum}(total\_width) This function is created as an example of how to create an estimator that is used to calculate the marginal likelihood of a model.

Description

Log likelihood function for logistic regression with a prior \hspace{1em} \hspace{1em} p(m)=\text{sum}(total\_width) This function is created as an example of how to create an estimator that is used to calculate the marginal likelihood of a model.

Usage

logistic.loglik(y, x, model, complex, params = list(r = 1))
**logistic.loglik.alpha**

*Log likelihood function for logistic regression for alpha calculation
This function is just the bare likelihood function*

**Description**

Log likelihood function for logistic regression for alpha calculation. This function is just the bare likelihood function.

**Usage**

```r
logistic.loglik.alpha(a, data, mu_func)
```

**Arguments**

- `a`  
  A vector of the alphas to be used
- `data`  
  The data to be used for calculation
- `mu_func`  
  The function linking the mean to the covariates, as a string with the alphas as `a[i].`

**Value**

A numeric with the log likelihood.
marginal.probs

Function for calculating marginal inclusion probabilities of features given a list of models

Description

Function for calculating marginal inclusion probabilities of features given a list of models

Usage

marginal.probs(models)

Arguments

models The list of models to use.

Value

A numeric vector of marginal model probabilities based on relative frequencies of model visits in MCMC.

Examples

result <- gmjmcmc(matrix(rnorm(600), 100), P = 2, gaussian.loglik, NULL, c("p0", "exp_dbl"))
marginal.probs(result$models[[1]])

merge_results

Merge a list of multiple results from many runs. This function will weight the features based on the best mlk in that population and merge the results together, simplifying by merging equivalent features (having high correlation).

Description

Merge a list of multiple results from many runs. This function will weight the features based on the best mlk in that population and merge the results together, simplifying by merging equivalent features (having high correlation).

Usage

merge_results(
  results,
  populations = NULL,
  complex.measure = NULL,
  tol = NULL,
  data = NULL
)
Arguments

results A list containing multiple results from GMJMCMC (Genetically Modified MJMCMC).

populations Which populations should be merged from the results, can be "all", "last" (default) or "best".

complex.measure The complex measure to use when finding the simplest equivalent feature, 1=total width, 2=operation count and 3=depth.

tol The tolerance to use for the correlation when finding equivalent features, default is 0.

data Data to use when comparing features, default is NULL meaning that mock data will be generated, if data is supplied it should be of the same form as is required by gmjmcmc, i.e. with both x, y and an intercept.

Value

An object of class "gmjmcmc_merged" containing the following elements:

features The features where equivalent features are represented in their simplest form.

marg.probs Importance of features.

counts Counts of how many versions that were present of each feature.

results Results as they were passed to the function.

pop.best The population in the results which contained the model with the highest log marginal posterior.

thread.best The thread in the results which contained the model with the highest log marginal posterior.

crit.best The highest log marginal posterior for any model in the results.

reported The highest log marginal likelihood for the reported populations as defined in the populations argument.

rep.pop The index of the population which contains reported.

best.log.posteriors A matrix where the first column contains the population indices and the second column contains the model with the highest log marginal posterior within that population.

rep.thread The index of the thread which contains reported.

result <- gmjmcmc.parallel( runs = 1, cores = 1, list(populations = "best", complex.measure = 2, tol = 0.0000001), matrix(rnorm(600), 100), P = 2, gaussian.loglik, loglik.alpha = gaussian.loglik.alpha, c("p0", "exp_db1") )

summary(result)

plot(result)

merge_results(result$results)
Main algorithm for MJMCMC (Genetically Modified MJMCMC)

Usage

```
mjmcmc(
  data, 
  loglik.pi, 
  N = 100, 
  probs = NULL, 
  params = NULL, 
  sub = FALSE, 
  verbose = TRUE 
)
```

Arguments

data A matrix containing the data to use in the algorithm, first column should be the dependent variable, second should be the intercept and the rest of the columns should be the independent variables.

loglik.pi The (log) density to explore

N The number of iterations to run for

probs A list of the various probability vectors to use

params A list of the various parameters for all the parts of the algorithm

sub An indicator that if the likelihood is inexact and should be improved each model visit (EXPERIMENTAL!)

verbose A logical denoting if messages should be printed

Value

A list containing the following elements:

- models All visited models.
- accept Average acceptance rate of the chain.
- lo.models All models visited during local optimization.
- best.crit The highest log marginal probability of the visited models.
- marg.probs Marginal probabilities of the features.
- model.probs Marginal probabilities of all of the visited models.
- model.probs.idx Indices of unique visited models.
- populations The covariates represented as a list of features.
Examples

result <- mjmcmc.parallel(matrix(rnorm(600), 100), gaussian.loglik)
summary(result)
plot(result)

---

mjmcmc.parallel  Run multiple mjmcmc runs in parallel, merging the results before returning.

Description

Run multiple mjmcmc runs in parallel, merging the results before returning.

Usage

mjmcmc.parallel(runs, cores = getOption("mc.cores", 2L), ...)

Arguments

- runs: The number of runs to run
- cores: The number of cores to run on
- ...: Further params passed to mjmcmc.

Value

Merged results from multiple mjmcmc runs

Examples

result <- mjmcmc.parallel(runs = 1, cores = 1, matrix(rnorm(600), 100), gaussian.loglik)
summary(result)
plot(result)
model.string

Function to generate a function string for a model consisting of features

Description
Function to generate a function string for a model consisting of features

Usage
model.string(model, features, link = "I", round = 2)

Arguments
model A logical vector indicating which features to include
features The population of features
link The link function to use, as a string
round Rounding error for the features in the printed format

Value
A character representation of a model

Examples
result <- gmjmcmc(matrix(rnorm(600), 100), P = 2, gaussian.loglik, NULL, c("p0", "exp_dbl"))
summary(result)
plot(result)
model.string(c(TRUE, FALSE, TRUE, FALSE, TRUE), result$populations[[1]])
model.string(result$models[[1]][[1]][[1]][[1]]$model, result$populations[[1]])

ngelu

Negative GELU function

Description
Negative GELU function

Usage
ngelu(x)

Arguments
x The vector of values
Value
-x*pnorm(-x)

Examples
ngelu(2)

---

nhs negative heavy side function

Description
negative heavy side function

Usage
nhs(x)

Arguments
x The vector of values

Value
as.integer(x<0)

Examples
nhs(2)

---

not not x

Description
not x

Usage
not(x)

Arguments
x The vector of binary values
nrelu

Value
1-x

Examples
not(TRUE)

Description
negative ReLu function

Usage
nrelu(x)

Arguments
x The vector of values

Value
max(-x,0)

Examples
nrelu(2)

p0

Description
p0 polynomial term

Usage
p0(x)

Arguments
x The vector of values
<table>
<thead>
<tr>
<th>Value</th>
<th>( \log(\text{abs}(x) + .\text{Machine}$\text{double}.\text{eps}) )</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th>( p0(2) )</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>( p0 ) polynomial term</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
<th>( p0(x) )</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Arguments</th>
<th>x \hspace{2em} The vector of values</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>( (\text{abs}(x) + .\text{Machine}$\text{double}.\text{eps})^{0.5} )</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th>( p0(2) )</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>( p0p0 ) polynomial term</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
<th>( p0p0(x) )</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Arguments</th>
<th>x \hspace{2em} The vector of values</th>
</tr>
</thead>
</table>
Value
\[ p0(x)^*p0(x) \]

Examples
\[ p0p0(2) \]

Description
\[ p0p0 \] polynomial term

Usage
\[ p0p0(x) \]

Arguments
\[ x \] The vector of values

Value
\[ p0(x)^*(abs(x)+.Machine$double.eps)^{(0.5)} \]

Examples
\[ p0p0(2) \]

Description
\[ p0p1 \] polynomial term

Usage
\[ p0p1(x) \]

Arguments
\[ x \] The vector of values
Value

\( p_0(x) \cdot x \)

Examples

\( p_0p1(2) \)

---

\( p_0p2 \) \hspace{1cm} \text{p0p2 polynomial term}

Description

\( p_0p2 \) polynomial term

Usage

\( p_0p2(x) \)

Arguments

\( x \) \hspace{1cm} The vector of values

Value

\( p_0(x) \cdot x^2 \)

Examples

\( p_0p2(2) \)

---

\( p_0p3 \) \hspace{1cm} \text{p0p3 polynomial term}

Description

\( p_0p3 \) polynomial term

Usage

\( p_0p3(x) \)

Arguments

\( x \) \hspace{1cm} The vector of values
**p0pm05**  

**Value**  
p0(x)*x^(3)

**Examples**  
p0p3(2)

---

**Description**  
p0pm05 polynomial term

**Usage**  
p0pm05(x)

**Arguments**  
x The vector of values

**Value**  
p0(x)sign(x)(abs(x)+.Machine$double.eps)^(-0.5)

**Examples**  
p0pm05(2)

---

**Description**  
p0pm1 polynomial terms

**Usage**  
p0pm1(x)

**Arguments**  
x The vector of values
Value
\[ p0(x)^*(x+.Machine\$double.eps)^(-1) \]

Examples
\[ p0pm1(2) \]

---

\[ p0pm2 \quad p0pm2 \text{ polynomial term} \]

Description

\[ p0pm2 \text{ polynomial term} \]

Usage
\[ p0pm2(x) \]

Arguments
\[ x \quad \text{The vector of values} \]

Value
\[ p0(x)\text{sign}(x)\text{abs}(x)+.Machine\$double.eps)^(-2) \]

Examples
\[ p0pm2(2) \]

---

\[ p2 \quad p2 \text{ polynomial term} \]

Description

\[ p2 \text{ polynomial term} \]

Usage
\[ p2(x) \]

Arguments
\[ x \quad \text{The vector of values} \]
p³

Value
x³(2)

Examples
p²(2)

---

p³ polynomial term

Description
p³ polynomial term

Usage
p³(x)

Arguments
x The vector of values

Value
x³(3)

Examples
p³(2)

---

plot.gmjmcmc

Function to plot the results, works both for results from gmjmcmc and merged results from merge.results

Description
Function to plot the results, works both for results from gmjmcmc and merged results from merge.results

Usage
## S3 method for class 'gmjmcmc'
plot(x, count = "all", pop = "last", ...)

Arguments

- **x**  The results to use
- **count**  The number of features to plot, defaults to all
- **pop**  The population to plot, defaults to last
- **...**  Not used.

Value

No return value, just creates a plot

Examples

```r
result <- gmjmcmc(matrix(rnorm(600), 100), P = 2, gaussian.loglik, NULL, c("p0", "exp_dbl"))
plot(result)
```

---

**Description**

Plot a gmjmcmc_merged run

**Usage**

```r
## S3 method for class 'gmjmcmc_merged'
plot(x, count = "all", ...)
```

**Arguments**

- **x**  The results to use
- **count**  The number of features to plot, defaults to all
- **...**  Not used.

**Value**

No return value, just creates a plot
Examples

```r
result <- gmjmcmc.parallel(
  runs = 1,
  cores = 1,
  list(populations = "best", complex.measure = 2, tol = 0.0000001),
  matrix(rnorm(600), 100),
  P = 2,
  gaussian.loglik,
  loglik.alpha = gaussian.loglik.alpha,
  c("p0", "exp_dbl")
)
plot(result)
```

---

### Description

Function to plot the results, works both for results from gmjmcmc and merged results from merge.results

### Usage

```r
## S3 method for class 'mjmcmc'
plot(x, count = "all", ...) 
```

### Arguments

- `x` The results to use
- `count` The number of features to plot, defaults to all
- `...` Not used.

### Value

No return value, just creates a plot

### Examples

```r
result <- mjmcmc(matrix(rnorm(600), 100), gaussian.loglik)
plot(result)
```
### plot.mjmcmc_parallel

**Plot a mjmcmc_parallel run**

**Description**

Plot a `mjmcmc_parallel` run

**Usage**

```r
## S3 method for class 'mjmcmc_parallel'
plot(x, count = "all", ...)
```

**Arguments**

- `x`: The results to use
- `count`: The number of features to plot, defaults to all
- `...`: Not used.

**Value**

No return value, just creates a plot

**Examples**

```r
result <- mjmcmc.parallel(runs = 1, cores = 1, matrix(rnorm(600), 100), gaussian.loglik)
plot(result)
```

### pm05

**pm05 polynomial term**

**Description**

`pm05` polynomial term

**Usage**

```r
pm05(x)
```

**Arguments**

- `x`: The vector of values

**Value**

```r
(abs(x)+.Machine$double.eps)^(-0.5)
```
**pm1**

**Examples**

pm05(2)

---

**pm1 polynomial term**

**Description**

pm1 polynomial term

**Usage**

pm1(x)

**Arguments**

x The vector of values

**Value**

sign(x)*(abs(x)+.Machine$double.eps)^(-1)

**Examples**

pm1(2)

---

**pm2 polynomial term**

**Description**

pm2 polynomial term

**Usage**

pm2(x)

**Arguments**

x The vector of values

**Value**

sign(x)*(abs(x)+.Machine$double.eps)^(-2)
Examples

`pm2(2)`

---

**predict.gmjmcmc**

*Predict using a gmjmcmc result object.*

**Description**

Predict using a gmjmcmc result object.

**Usage**

```r
## S3 method for class 'gmjmcmc'
predict(object, x, link = function(x) x, quantiles = c(0.025, 0.5, 0.975), ...)
```

**Arguments**

- `object`: The model to use.
- `x`: The new data to use for the prediction, a matrix where each row is an observation.
- `link`: The link function to use.
- `quantiles`: The quantiles to calculate credible intervals for the posterior modes (in model space).
- `...`: Not used.

**Value**

A list containing aggregated predictions and per model predictions.

- `aggr`: Aggregated predictions with mean and quantiles.
- `preds`: A list of lists containing individual predictions per model per population in object.

**Examples**

```r
result <- gmjmcmc(
  matrix(rnorm(600), 100),
  P = 2,
  gaussian.loglik,
  loglik.alpha = gaussian.loglik.alpha,
  c("p0", "exp dbl")
)
preds <- predict(result, matrix(rnorm(600), 100))
```
predict.gmjmcmc_merged

**Description**

Predict using a merged gmjmcmc result object.

**Usage**

```r
## S3 method for class 'gmjmcmc_merged'
predict(object, x, link = function(x) x, quantiles = c(0.025, 0.5, 0.975), ...)
```

**Arguments**

- **object**
  - The model to use.
- **x**
  - The new data to use for the prediction, a matrix where each row is an observation.
- **link**
  - The link function to use
- **quantiles**
  - The quantiles to calculate credible intervals for the posterior models (in model space).
- **...**
  - Not used.

**Value**

A list containing aggregated predictions and per model predictions.

- **aggr**
  - Aggregated predictions with mean and quantiles.
- **preds**
  - A list of lists containing individual predictions per model per population in object.

**Examples**

```r
result <- gmjmcmc.parallel(
  runs = 1,
  cores = 1,
  list(populations = "best", complex.measure = 2, tol = 0.0000001),
  matrix(rnorm(600), 100),
  P = 2,
  gaussian.loglik,
  loglik.alpha = gaussian.loglik.alpha,
  c("p0", "exp_dbl")
)
preds <- predict(result, matrix(rnorm(600), 100))
```
predict.gmjmcmc_parallel

*Predict using a gmjmcmc result object from a parallel run.*

### Description

Predict using a gmjmcmc result object from a parallel run.

### Usage

```r
## S3 method for class 'gmjmcmc_parallel'
predict(object, x, link = function(x) x, quantiles = c(0.025, 0.5, 0.975), ...)
```

### Arguments

- **object**: The model to use.
- **x**: The new data to use for the prediction, a matrix where each row is an observation.
- **link**: The link function to use.
- **quantiles**: The quantiles to calculate credible intervals for the posterior models (in model space).
- **...**: Additional arguments to pass to `merge_results`.

### Value

A list containing aggregated predictions and per model predictions.

- **aggr**: Aggregated predictions with mean and quantiles.
- **preds**: A list of lists containing individual predictions per model per population in object.

### Examples

```r
result <- gmjmcmc.parallel(
  runs = 1,
  cores = 1,
  list(populations = "best", complex.measure = 2, tol = 0.0000001),
  matrix(rnorm(600), 100),
  P = 2,
  gaussian.loglik,
  loglik.alpha = gaussian.loglik.alpha,
  c("p0", "exp_dbl")
)
preds <- predict(result$results, matrix(rnorm(600), 100))
```
predict.mjmcmc

Predict using a mjmcmc result object.

Description

Predict using a mjmcmc result object.

Usage

```r
## S3 method for class 'mjmcmc'
predict(object, x, link = function(x) x, quantiles = c(0.025, 0.5, 0.975), ...)
```

Arguments

- `object` The model to use.
- `x` The new data to use for the prediction, a matrix where each row is an observation.
- `link` The link function to use
- `quantiles` The quantiles to calculate credible intervals for the posterior modes (in model space).
- `...` Not used.

Value

A list containing aggregated predictions.

- `mean` Mean of aggregated predictions.
- `quantiles` Quantiles of aggregated predictions.

Examples

```r
result <- mjmcmc(matrix(rnorm(600), 100), gaussian.loglik)
preds <- predict(result, matrix(rnorm(500), 100))
```

predict.mjmcmc_parallel

Predict using a mjmcmc result object from a parallel run.

Description

Predict using a mjmcmc result object from a parallel run.
print.feature

Usage

## S3 method for class 'mjmmcm_parallel'
predict(object, x, link = function(x) x, quantiles = c(0.025, 0.5, 0.975), ...)

Arguments

- **object**: The model to use.
- **x**: The new data to use for the prediction, a matrix where each row is an observation.
- **link**: The link function to use.
- **quantiles**: The quantiles to calculate credible intervals for the posterior models (in model space).
- **...**: Not used.

Value

A list containing aggregated predictions.

- **mean**: Mean of aggregated predictions.
- **quantiles**: Quantiles of aggregated predictions.

Examples

```r
result <- mjmmcm_parallel(runs = 1, cores = 1, matrix(rnorm(600), 100), gaussian.loglik)
preds <- predict(result, matrix(rnorm(500), 100))
```

print.feature

Print method for "feature" class

Description

Print method for "feature" class

Usage

## S3 method for class 'feature'
print(x, dataset = FALSE, alphas = FALSE, labels = FALSE, round = FALSE, ...)

Arguments

- **x**: An object of class "feature"
- **dataset**: Set the regular covariates as columns in a dataset.
- **alphas**: Print a "?" instead of actual alphas to prepare the output for alpha estimation.
- **labels**: Should the covariates be named, or just referred to as their place in the data.frame.
- **round**: Should numbers be rounded when printing? Default is FALSE, otherwise it can be set to the number of decimal places.
- **...**: Not used.
**relu**

String representation of a feature

**Examples**

```r
gmjmcmc(matrix(rnorm(600), 100), P = 2, gaussian.loglik, NULL, c("p0", "exp_dbl"))
print(result$populations[[1]][[1]])
```

---

**relu**

**ReLU function**

---

**Description**

ReLU function

**Usage**

```r
relu(x)
```

**Arguments**

- `x` The vector of values

**Value**

```r
max(x, 0)
```

**Examples**

```r
relu(2)
```

---

**set.transforms**

Set the transformations option for GMJMCMC (Genetically Modified MJMCMC), this is also done when running the algorithm, but this function allows for it to be done manually.

---

**Description**

Set the transformations option for GMJMCMC (Genetically Modified MJMCMC), this is also done when running the algorithm, but this function allows for it to be done manually.

**Usage**

```r
set.transforms(transforms)
```
Arguments

arguemnts

Value

No return value, just sets the gmjmcmc-transformations option

Examples

set.transforms(c("p0", "p1"))

sigmoid

Sigmoid function

Description

Sigmoid function

Usage

sigmoid(x)

Arguments

x

The vector of values

Value

The sigmoid of x

Examples

sigmoid(2)
**sin_deg**

---

**Sine function for degrees**

**Description**
Sine function for degrees

**Usage**

\[ \text{sin}_\text{deg}(x) \]

**Arguments**

- \( x \)  The vector of values in degrees

**Value**

The sine of \( x \)

**Examples**

\[ \text{sin}_\text{deg}(0) \]

---

**sqroot**

---

**Square root function**

**Description**
Square root function

**Usage**

\[ \text{sqroot}(x) \]

**Arguments**

- \( x \)  The vector of values

**Value**

The square root of the absolute value of \( x \)

**Examples**

\[ \text{sqroot}(4) \]
string.population.models

Function to get a character representation of a list of models

Description
Function to get a character representation of a list of models

Usage
string.population.models(features, models, round = 2, link = "I")

Arguments
features A list of feature objects on which the models are build
models A list of model objects
round Rounding precision for parameters of the features
link The link function to use, as a string

Examples
result <- gmjmcmc(matrix(rnorm(600), 100), P = 2, gaussian.loglik, NULL, c("p0", "exp_dbl"))
string.population.models(result$populations[[1]])
Value

A matrix of character representations of a list of models.

Examples

```r
result <- gmjmcmc(matrix(rnorm(600), 100), P = 2, gaussian.loglik, NULL, c("p0", "exp_dbl"))
string.population.models(result$populations[[2]], result$models[[2]])
```

---

**summary.gmjmcmc**

*Function to print a quick summary of the results*

Description

Function to print a quick summary of the results

Usage

```r
## S3 method for class 'gmjmcmc'
summary(object, pop = "last", tol = 1e-04, labels = FALSE, effects = NULL, ...)
```

Arguments

- `object`: The results to use
- `pop`: The population to print for, defaults to last
- `tol`: The tolerance to use as a threshold when reporting the results.
- `labels`: Should the covariates be named, or just referred to as their place in the data.frame.
- `effects`: Quantiles for posterior modes of the effects across models to be reported, if either effects are NULL or if labels are NULL, no effects are reported.
- `...`: Not used.

Value

A data frame containing the following columns:

- `feats.strings`: Character representation of the features ordered by marginal probabilities.
- `marg.probs`: Marginal probabilities corresponding to the ordered feature strings.

Examples

```r
result <- gmjmcmc(matrix(rnorm(600), 100), P = 2, gaussian.loglik, NULL, c("p0", "exp_dbl"))
summary(result)
```
Function to print a quick summary of the results

Summary

Function to print a quick summary of the results

Usage

```r
## S3 method for class 'gmjmcmc_merged'
summary(object, tol = 1e-04, labels = FALSE, effects = NULL, ...)
```

Arguments

- `object`: The results to use
- `tol`: The tolerance to use as a threshold when reporting the results.
- `labels`: Should the covariates be named, or just referred to as their place in the data.frame.
- `effects`: Quantiles for posterior modes of the effects across models to be reported, if either effects are NULL or if labels are NULL, no effects are reported.
- `...`: Not used.

Value

A data frame containing the following columns:

- `feats.strings`: Character representation of the features ordered by marginal probabilities.
- `marg.probs`: Marginal probabilities corresponding to the ordered feature strings.

Examples

```r
result <- gmjmcmc.parallel(
  runs = 1,
  cores = 1,
  list(populations = "best", complex.measure = 2, tol = 0.0000001),
  matrix(rnorm(600), 100),
  P = 2,
  gaussian.loglik,
  loglik.alpha = gaussian.loglik alpha,
  c("p0", "exp_dbl")
)
summary(result)
```
Function to print a quick summary of the results

### Summary

#### Description
Function to print a quick summary of the results

#### Usage
```r
## S3 method for class 'mjmcmc'
summary(object, tol = 1e-04, labels = FALSE, effects = NULL, ...)
```

#### Arguments
- `object`: The results to use
- `tol`: The tolerance to use as a threshold when reporting the results.
- `labels`: Should the covariates be named, or just referred to as their place in the data.frame.
- `effects`: Quantiles for posterior modes of the effects across models to be reported, if either effects are NULL or if labels are NULL, no effects are reported.
- `...`: Not used.

#### Value
A data frame containing the following columns:
- `feats.strings`: Character representation of the covariates ordered by marginal probabilities.
- `marg.probs`: Marginal probabilities corresponding to the ordered feature strings.

#### Examples
```r
result <- mjmcmc(matrix(rnorm(600), 100), gaussian.loglik)
summary(result)
```

---

Function to print a quick summary of the results

### Summary

#### Description
Function to print a quick summary of the results

#### Usage
```r
## S3 method for class 'mjmcmc_parallel'
summary(object, tol = 1e-04, labels = FALSE, effects = NULL, ...)
```

---
Arguments

object  The results to use
tol    The tolerance to use as a threshold when reporting the results.
labels Should the covariates be named, or just referred to as their place in the data.frame.
effects Quantiles for posterior modes of the effects across models to be reported, if either effects are NULL or if labels are NULL, no effects are reported.
...    Not used.

Value

A data frame containing the following columns:

feats.strings  Character representation of the covariates ordered by marginal probabilities.
marg.probs     Marginal probabilities corresponding to the ordered feature strings.

Examples

result <- mjmcmc.parallel(runs = 1, cores = 1, matrix(rnorm(600), 100), gaussian.loglik)
summary(result)

to23

To the 2.3 power function

Description

To the 2.3 power function

Usage

to23(x)

Arguments

x       The vector of values

Value

x^2.3

Examples

to23(2)
to25  

To 2.5 power

Description
To 2.5 power

Usage
to25(x)

Arguments
x  
The vector of values

Value
x^2.5

Examples
to25(2)

---

to35  

To 3.5 power

Description
To 3.5 power

Usage
to35(x)

Arguments
x  
The vector of values

Value
x^3.5

Examples
to35(2)
**to72**  
*To the 7/2 power function*

**Description**  
To the 7/2 power function

**Usage**  
\texttt{to72(x)}

**Arguments**  
\texttt{x}  
The vector of values

**Value**  
\texttt{x^{7/2}}

**Examples**  
\texttt{to72(2)}

---

**troot**  
*Cube root function*

**Description**  
Cube root function

**Usage**  
\texttt{troot(x)}

**Arguments**  
\texttt{x}  
The vector of values

**Value**  
The cube root of \texttt{x}

**Examples**  
\texttt{troot(27)}
Index

* datasets
  breastcancer, 4
  exoplanet, 8

breastcancer, 4
compute_effects, 5
cos_deg, 6
diagn_plot, 6
erf, 7
exoplanet, 8
exp_db1, 9

FBMS (FBMS-package), 3
fbms, 9
FBMS-package, 3

gauss, 11
gaussian.loglik, 11
gaussian.loglik.alpha, 12
gelu, 13
gen.params.gmjmcmc, 13
gen.params.mjmcnc, 14
gen.probs.gmjmcmc, 14
gen.probs.mjmcnc, 15
gmjmcmc, 10, 15
gmjmcmc.parallel, 10, 17

hs, 18

linear.g.prior.loglik, 19
logistic.loglik, 19
logistic.loglik.alpha, 20

marginal.probs, 21
merge_results, 21
merge_results(), 17
mjmcnc, 10, 23
mjmcnc.parallel, 24

model.string, 25
ngelu, 25
nhs, 26
not, 26
nrelu, 27
p0, 27
p05, 28
p0p0, 28
p0p05, 29
p0p1, 29
p0p2, 30
p0p3, 30
p0pm05, 31
p0pm1, 31
p0pm2, 32
p2, 32
p3, 33
plot.gmjmcmc, 33
plot.gmjmcmc_merged, 34
plot.mjmcnc, 35
plot.mjmcnc_parallel, 36
pm05, 36
pm1, 37
pm2, 37
predict, 6
predict.gmjmcmc, 38
predict.gmjmcmc_merged, 39
predict.mjmcnc_parallel, 40
predict.mjmcnc, 41
predict.mjmcnc_parallel, 41
print.feature, 42

relu, 43

set.transforms, 43
sigmoid, 44
sin_deg, 45
sqroot, 45
string.population, 46
string.population.models, 46
summary.gjmcmc, 47
summary.gjmcmc_merged, 48
summary.mjmcmc, 49
summary.mjmcmc_parallel, 49

to23, 50
to25, 51
to35, 51
to72, 52
troot, 52