Package ‘dirinla’

December 30, 2021

Type Package

Title Hierarchical Bayesian Dirichlet regression models using Integrated Nested Laplace Approximation

Version 1.0.0

Author Joaquín Martínez-Minaya «jomarminaya@gmail.com>>

Maintainer Joaquín Martínez-Minaya «jomarminaya@gmail.com>> and Finn Lindgren «Finn.Lindgren@ed.ac.uk>>

Description The R-package dirinla allows the user to fit models in the CoDa context. In particular, it allows fit Dirichlet regression models using the Integrated Nested Laplace Approximation (INLA) methodology.

License GPL-2

Imports utils,
Matrix,
dplyr,
samplingDataCRT,
purrr,
magrittr,
stringr,
tidyverse,
stats,
ggplot2,
Rfast,
Rfast2,
ggtern,
gridExtra,
sn,
knitr,
plyr,
methods,
INLA

Suggests DirichletReg

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.1.2

R topics documented:

`bdiag_m` *Fast version of Matrix :: .bdiag() – for the case of *many* (k x k) matrices: Copyright (C) 2016 Martin Maechler, ETH Zurich*

Description

Fast version of Matrix :: .bdiag() – for the case of *many* (k x k) matrices: Copyright (C) 2016 Martin Maechler, ETH Zurich

Usage

```
bdiag_m(lmat)
```

Arguments

`lmat`	list(<mat1>, <mat2>, ..., <mat_N>) where each mat_j is a k x k 'matrix'

Value

a sparse (N*k x N*k) matrix of class "`dgCMatrix`".

`data_stack_dirich` *Preparing the data*

Description

'data_stack_dirich' prepares the data using inla.stack from the package INLA.

Usage

```
data_stack_dirich(y, covariates, share = NULL, data, d, n)
```

Arguments

`y`	Response variable in a matrix format.
`covariates`	String with the name of covariates.
`share`	Covariates to share in all the cateogries. TODO
`data`	Data.frame which contains all the covariates.
`d`	Number of categories.
`n`	Number of locations.

Value

Matrix A such as eta = A

Author(s)

Joaquín Martínez-Minaya <<jomarminaya@gmail.com>>

Examples

```
n <- 100
d <- 4

V <- matrix(rnorm(4*n, 0, 1), ncol=4)
V <- as.data.frame(V)
names(V) <- c('v1', 'v2', 'v3', 'v4' )

covariates <- names(V)


formula <- y ~ 1 + v1 + v2 | 1 + v1 | 1 + v1

names_cat <- formula_list(formula)

data_stack_construct <- data_stack_dirich(y          = as.vector(rep(NA, n*d)),
                                          covariates = names_cat,
                                          share      = NULL,
                                          data       = V,
                                          d          = d,
                                          n          = n )
```

data_stack_dirich_formula *Preparing the data*

Description

'data_stack_dirich_formula' prepares the data using inla.stack from the package INLA.

Usage

```
data_stack_dirich_formula(y, covariates, share = NULL, data, d, n)
```

Arguments

y	Response variable in a matrix format.
covariates	String with the name of covariates.
share	Covariates to share in all the cateogries. Not implemented yet.
data	Data.frame which contains all the covariates.
d	Number of categories.
n	Number of locations.

Value

List with two objects - Object of class inla.stack - Object with class formula

Author(s)

Joaquín Martínez-Minaya <<joaquin.martinez-minaya@uv.es>>

Examples

```
n <- 100
d <- 4

V <- matrix(rnorm(4*n, 0, 1), ncol=4)
V <- as.data.frame(V)
names(V) <- c('v1', 'v2', 'v3', 'v4' )

covariates <- names(V)


formula <- y ~ 1 + v1 + v2 | 1 + v1 | 1 + v1

names_cat <- formula_list(formula)

data_stack_construct <- data_stack_dirich(y          = as.vector(rep(NA, n*d)),
                                          covariates = names_cat,
                                          share      = NULL,
                                          data       = V,
                                          d          = d,
                                          n          = n )
```

`digamma_red` *Computing the function digamma*

Description

'digamma_red' is the function digamma appropiate for really small values

Usage

```
digamma_red(x, ...)
```

Arguments

`x`	Argument to applied the function digamma.
`...`	Rest of arguments used in the case of digamma functions.

Value

Result of applying digamma function

Author(s)

Joaquín Martínez-Minaya <<`joaquin.martinez-minaya@uv.es`>>

`dirichlet_log_pos_x` *Dirichlet log posterior function*

Description

'dirichlet_log_pos_x' returns the -log posterior Dirichlet distribution asumming multivariate normal prior with precision matrix Qx for elements of the latent field.

Usage

```
dirichlet_log_pos_x(A = A, x, Qx = Qx, y)
```

Arguments

`A`	A matrix which links eta with the latent field, i.e., eta = A x.
`x`	Vector with the elements of the latent field, i.e., eta = A x.
`Qx`	Precision matrix for the priors of the latent field.
`y`	Vector with the response variable.

Value

A real value showing the -log posterior density is returned

Author(s)

Joaquín Martínez-Minaya <<`jomarminaya@gmail.com`>>

`dirinlareg` *Fitting a Dirichlet regression*

Description

'dirinlareg' Main function to do a Dirichlet Regression

Usage

```
dirinlareg(
  formula,
  y,
  data.cov,
  share = NULL,
  x0 = NULL,
  tol0 = 1e-05,
  tol1 = 0.1,
  k0 = 20,
  k1 = 5,
  a = 0.5,
  strategy = "ls-quasi-newton",
  prec = prec,
  verbose = FALSE,
  cores = 1,
  sim = 1000,
  prediction = FALSE,
  data.pred.cov = NULL,
  ...
)
```

Arguments

formula	object of class formula indicating the response variable and the covariates of the Dirichlet regression
y	matrix containing the response variable R^nxd, being n number of individuals and d the number of categories
data.cov	data.frame with the covarites, only the covariates!
share	parameters to be fitted jointly.
x0	initial optimization value
tol0	tolerance
tol1	tolerance for the gradient such that |grad| < tol1 * max(1, |f|)
k0	number of iterations
k1	number of iterations including the calling to inla
a	step length in the optimization algorithm
strategy	strategy to use to optimize
prec	precision for the prior of the fixed effects
verbose	if TRUE all the computing process is shown. Default is FALSE

cores	Number of cores for parallel computation. The package parallel is used.
sim	Simulations to call inla.posterior.sample and extract linear predictor, alphas and mus. The bigger it is, better is the approximation, but more computational time.
prediction	if TRUE we will predict with the new values of the covariates given in data.pred.cov.
data.pred.cov	data.frame with the values for the covariates where we want to predict.
...	arguments for the inla command

Value

model dirinlaregmodel object

Author(s)

Joaquín Martínez-Minaya <<joaquin.martinez-minaya@uv.es>>

Examples

```
#' ### In this example, we show how to fit a model using the dirinla package ###
### --- 1. Loading the libraries --- ####
library(INLA)
library(DirichletReg)


### --- 2. Simulating from a Dirichlet likelihood --- ####
set.seed(1000)
N <- 50 #number of data
V <- as.data.frame(matrix(runif((4) * N, 0, 1), ncol = 4)) #Covariates
names(V) <- paste0('v', 1:4)

formula <- y ~ 1 + v1 | 1 + v2 | 1 + v3 | 1 + v4
(names_cat <- formula_list(formula))

x <- c(-1.5, 1, -3, 1.5,
       2, -3 , -1, 5)

mus <- exp(x) / sum(exp(x))
C <- length(names_cat)
data_stack_construct <-
  data_stack_dirich(y = as.vector(rep(NA, N * C)),
                    covariates = names_cat,
                    data       = V,
                    d          = C,
                    n          = N)

A_construct <- data_stack_construct
A_construct[1:8, ]

eta <- A_construct %*% x
alpha <- exp(eta)
alpha <- matrix(alpha,
                ncol  = C,
                byrow = TRUE)
y_o <- rdirichlet(N, alpha)
colnames(y_o) <- paste0("y", 1:C)
head(y_o)
```

```
### --- 3. Fitting the model --- ####
y <- y_o
model.inla <- dirinlareg(
  formula  = y ~ 1 + v1 | 1 + v2 | 1 + v3 | 1 + v4,
  y        = y,
  data.cov = V,
  prec     = 0.0001,
  verbose  = FALSE)


summary(model.inla)
```

`dirinlaregmodel` *Defining a new class*

Description

'dirinlaregmodel' is a new object class

Usage

```
dirinlaregmodel(
  call = NULL,
  formula = NULL,
  summary_fixed = NULL,
  marginals_fixed = NULL,
  summary_random = NULL,
  marginals_random = NULL,
  summary_hyperpar = NULL,
  marginals_hyperpar = NULL,
  summary_linear_predictor = NULL,
  marginals_linear_predictor = NULL,
  summary_alphas = NULL,
  marginals_alphas = NULL,
  summary_precision = NULL,
  marginals_precision = NULL,
  summary_means = NULL,
  marginals_means = NULL,
  summary_predictive_alphas = NULL,
  marginals_predictive_alphas = NULL,
  summary_predictive_means = NULL,
  marginals_predictive_means = NULL,
  summary_predictive_precision = NULL,
  marginals_predictive_precision = NULL,
  dic = NULL,
  waic = NULL,
  cpo = NULL,
  nobs = NULL,
  ncat = NULL,
```

```
  y = NULL,
  data.cov = NULL
)
```

Arguments

- `call` The call of the function dirinlareg.
- `formula` Formula introduced by the user.
- `summary_fixed` List containing a summary of the marginal posterior distributions of the fixed effects.
- `marginals_fixed` List containing the marginal posterior distributions of the fixed effects.
- `summary_random` List containing a summary of the marginal posterior distributions of the random effects.
- `marginals_random` List containing the marginal posterior distributions of the random effects.
- `summary_hyperpar` List containing a summary of the marginal posterior distributions of the hyperparameters.
- `marginals_hyperpar` List containing the marginal posterior distributions of the hyperparameters.
- `summary_linear_predictor` List containing a summary of the marginal posterior distributions of the linear predictor.
- `marginals_linear_predictor` List containing the marginal posterior distributions of the linear predictor.
- `summary_alphas` List containing a summary of the marginal posterior distributions of the alphas.
- `marginals_alphas` List containing the marginal posterior distributions of the alphas.
- `summary_precision` List containing a summary of the marginal posterior distributions of the precision.
- `marginals_precision` List containing the marginal posterior distributions of the precision.
- `summary_means` List containing a summary of the marginal posterior distributions of the means.
- `marginals_means` List containing the marginal posterior distributions of the means.
- `summary_predictive_alphas` List containing a summary of the marginal posterior predictive distribution of the alphas.
- `marginals_predictive_alphas` List containing the marginal posterior predictive distribution of the alphas.
- `summary_predictive_means` List containing a summary fo the marginal posterior predictive distribution of the means.
- `marginals_predictive_means` List containing the marginal posterior predictive distribution of the means.
- `summary_predictive_precision` List containing a summary of the marginal posterior predictive distribution of the precision.

`marginals_predictive_precision`
List containing the marginal posterior predictive distribution of the precision.

`dic` List containing the inla output for dic.

`waic` List containing the inla output for waic.

`cpo` List containing the inla output for cpo.

`nobs` Number of observations.

`ncat` Number of categories.

`y` matrix containing the response variable R^nxd, being n number of individuals and d the number of categories

`data.cov` data.frame with the covarites, only the covariates!

Value

object of list and dirinlaregmodel class.

`extract_fixed` *Extracting marginals fixed of an inla object*

Description

'extract_fixed' is a function to extract summary and marginals distribution corresponding to the fixed effects

Usage

```
extract_fixed(inla_model, names_cat)
```

Arguments

`inla_model` Object of inla class.

`names_cat` List generated with extract_formula.

Value

summary_fixed Summary of fixed effects for each category.

marginals_fixed Marginals for each parameter estimated.

Author(s)

Joaquín Martínez-Minaya <<joaquin.martinez-minaya@uv.es>>

extract_linear_predictor

Extracting posterior distributions from the linear predictor

Description

'extract_linear_predictor' extracts the posterior distribution from the linear predictor

Usage

```
extract_linear_predictor(
  inla_model,
  n,
  d,
  Lk_eta,
  names_cat = names_cat,
  sim,
  verbose,
  cores
)
```

Arguments

inla_model	An object of class inla.
n	Number of observations.
d	Number of categories.
Lk_eta	Cholesky decomposition of the Hessian matrix.
names_cat	List generated with extract_formula.
sim	simulations for the function inla.posterior.sample
verbose	if TRUE all the computing process is shown. Default is FALSE
cores	number of cores to be used in the computations

Value

summary_linear_predictor List containing a summary of the marginal posterior distributions of the linear predictor.

marginals_linear_predictor List containing simulations of marginal posterior distributions of the linear predictor.

summary_alphas List containing a summary of the marginal posterior distributions of the alphas.

marginals_alphas List containing simulations of the marginal posterior distributions of the alphas.

summary_precision List containing a summary of the marginal posterior distributions of the precision.

marginals_precision List containing simulations of the marginal posterior distributions of the precision.

summary_means List containing a summary of the marginal posterior distributions of the means.

marginals_means List containing the simulations of the marginal posterior distributions of the means.

Author(s)

Joaquín Martínez-Minaya <<jomarminaya@gmail.com>>

---

formula_list *Formula in to list*

Description

'formula_list' reads the formula and generates a list with the name of the covariates used in each category

Usage

```
formula_list(form, y = NULL)
```

Arguments

form	Object of class formula.
y	Matrix containing the response variable R^nxd, being n number of individuals and d the number of categories.

Value

A list with the names of the variables used in each category.

Author(s)

Joaquín Martínez-Minaya <<jomarminaya@gmail.com>>

Examples

```
formula <- y ~ 1 + v1 + v2 | -1 + v1 | 0 + v2
formula_list(formula)
```

---

g0_vector_eta_1 *Computing gradient vector in eta*

Description

'g0_vector_eta' computes the gradient of -loglikelihood

Usage

```
g0_vector_eta_1(A = A, x, y)
```

Arguments

A	Matrix which links eta with the latent field, i.e., eta = A x.
x	Vector with the elements of the latent field, i.e., eta = A x.
y	Vector with the response variable.

Value

A numeric vector with the gradient in eta.

Author(s)

Joaquín Martínez-Minaya <<`joaquin.martinez-minaya@uv.es`>>

`H0_matrix_eta1` *Computing expected Hessian for a vector eta*

Description

‘H0_matrix_eta_1‘ computes the expected Hessian in eta of -loglikelihood

‘H0_matrix_eta_1‘ computes the expected Hessian in eta of -loglikelihood

Usage

```
H0_matrix_eta1(eta, d)
```



```
H0_matrix_eta1(eta, d)
```


Arguments

`eta`	eta vector to compute the expected Hessian.
`d`	Dimension

Value

Expected Hessian in eta.

Expected Hessian in eta.

Author(s)

Joaquín Martínez-Minaya <<`jomarminaya@gmail.com`>>

Joaquín Martínez-Minaya <<`jomarminaya@gmail.com`>>

H0_matrix_eta_x *Computing expected Hessian in eta*

Description

'H0_matrix_eta_x' computes the expected Hessian in eta of -loglikelihood

'H0_matrix_eta_x' computes the expected Hessian in eta of -loglikelihood

Usage

```
H0_matrix_eta_x(eta, d, cores)
```



```
H0_matrix_eta_x(eta, d, cores)
```


Arguments

eta	Linear predictor resulting of the product Ax.
d	Dimension.
cores	Number of cores for parallel computation. The package parallel is used.

Value

Expected Hessian in eta.

Expected Hessian in eta.

Author(s)

Joaquín Martínez-Minaya <<jomarminaya@gmail.com>>

Joaquín Martínez-Minaya <<jomarminaya@gmail.com>>

H_matrix_eta_diag *Computing additional diagonal part for the real Hessian H = H0 + diag*

Description

'H_matrix_eta_diag' computes the expected Hessian in eta of -loglikelihood

Usage

```
H_matrix_eta_diag(eta, d, y)
```

Arguments

eta	eta vector to compute the expected Hessian.
d	Dimension
y	Data corresponding to the i-individual

Value

Elements of the diagonal such as H = H0 + diag

Author(s)

Joaquín Martínez-Minaya <<`jomarminaya@gmail.com`>>

`log_beta_mult_eta` *Calculating the log beta function in eta*

Description

'beta_mult_eta' computes the log beta function in eta

Usage

```
log_beta_mult_eta(x)
```

Arguments

`x`	Vector of elements.

Value

Numeric value.

Author(s)

Joaquín Martínez-Minaya <<`joaquin.martinez-minaya@uv.es`>>

`look_for_mode_x` *Finding the mode of the full posterior distribution*

Description

'look_for_mode_x' computes optimization algorithms to find the mode of the posterior

Usage

```
look_for_mode_x(
  A = A,
  x0,
  tol0,
  tol1,
  k0,
  a = 0.5,
  y,
  d,
  n,
  strategy = "ls-quasi-newton",
```

```
  Qx,
  verbose,
  cores
)
```

Arguments

A	Matrix which links latent field with linear predictor.
x0	Initial optimization value.
tol0	Tolerance for |x_new - x_old| and |f_new - f_old|.
tol1	Tolerance for the gradient such that |grad| < tol1 * max(1, |f|)
k0	Number of iterations.
a	Step length in the algorithm.
y	Response variable. Number of columns correspond to the number of categories.
d	Number of categories.
n	Number of individuals.
strategy	Strategy to use to optimize.
Qx	Prior precision matrix for the fixed effects.
verbose	By default is FALSE. If TRUE, the computation process is shown in the scream.
cores	Number of cores for parallel computation. The package parallel is used.

Value

x_hat Matrix with the x of the iterations.

Hk Hessian in eta. This Hessian is a combination of the real Hessian (when it is positive definite) and the expected Hessian (when the real Hessian is not positive definite).

gk Gradient in eta.

Lk Cholesky decomposition matrix.

eta Linear predictor.

z New pseudo observation conditioned to eta.

Author(s)

Joaquín Martínez-Minaya <<joaquin.martinez-minaya@uv.es>>

---

newton_x *Newton-Raphson algorithm*

Description

'newton_x' computes optimization algorithms to find the mode of the posterior. Line search strategy with Armijo conditions is implemented

Usage

```
newton_x(A, x_hat, gk, Hk, a, Qx, strategy, y, d = d)
```

Arguments

`A`	Matrix which links eta with the latent field, i.e., eta = A x
`x_hat`	Vector with the elements of the latent field, i.e., eta_hat = A x_hat
`gk`	Gradient in eta.
`Hk`	Hessian in eta.
`a`	Step length.
`Qx`	Precision matrix for the prior of the latent field.
`strategy`	Strategy to use to optimize. Now, line search strategy with quasi-newton algorithm is the only one avaliable.
`y`	Vector with the response variable
`d`	Number of categories.

Value

g0 : Gradient in x_hat_new. A numeric vector with the gradient in x_hat_new.

x_hat_new: New value of x after apply one iteration.

Author(s)

Joaquín Martínez-Minaya <<`joaquin.martinez-minaya@uv.es`>>

`plot.dirinlaregmodel` *plot of dirinlaregmodel xs*

Description

'plot.dirinlaregmodel' Method which plots a dirinlaregmodel x

Usage

```
## S3 method for class 'dirinlaregmodel'
plot(x, ...)
```

Arguments

`x`	Object of class dirinlaregmodel.
`...`	Other arguments.

Value

Plotting the posterior of the fixed effects.

Author(s)

Joaquín Martínez-Minaya <<`jomarminaya@gmail.com`>>

predict.dirinlaregmodel
Finding the mode of the full posterior distribution

Description

'predict.dirinlaregmodel' computes the posterior predictive distribution for some given values of the covariates

Usage

```
## S3 method for class 'dirinlaregmodel'
predict(object, data.pred.cov, ...)
```

Arguments

`object`	dirinlaregmodel object.
`data.pred.cov`	Data.frame with the covariate values for the variables to predict.
`...`	Other arguments.

Value

model dirinlaregmodel object

Author(s)

Joaquín Martínez-Minaya <<jomarminaya@gmail.com>>

Examples

```
### In this example, we show how to fit a model using the dirinla package ###
### --- 1. Loading the libraries --- ####
library(INLA)
library(DirichletReg)


### --- 2. Simulating from a Dirichlet likelihood --- ####
set.seed(1000)
N <- 50 #number of data
V <- as.data.frame(matrix(runif((4) * N, 0, 1), ncol = 4)) #Covariates
names(V) <- paste0('v', 1:4)

formula <- y ~ 1 + v1 | 1 + v2 | 1 + v3 | 1 + v4
(names_cat <- formula_list(formula))

x <- c(-1.5, 1, -3, 1.5,
       2, -3 , -1, 5)

mus <- exp(x) / sum(exp(x))
C <- length(names_cat)
data_stack_construct <-
  data_stack_dirich(y = as.vector(rep(NA, N * C)),
                    covariates = names_cat,
```

```
                    data       = V,
                    d          = C,
                    n          = N)

A_construct <- data_stack_construct
A_construct[1:8, ]

eta <- A_construct %*% x
alpha <- exp(eta)
alpha <- matrix(alpha,
                ncol  = C,
                byrow = TRUE)
y_o <- rdirichlet(N, alpha)
colnames(y_o) <- paste0("y", 1:C)
head(y_o)


### --- 3. Fitting the model --- ####
y <- y_o
model.inla <- dirinlareg(
  formula  = y ~ 1 + v1 | 1 + v2 | 1 + v3 | 1 + v4,
  y        = y,
  data.cov = V,
  prec     = 0.0001,
  verbose  = FALSE)


summary(model.inla)
### --- 4. Predicting for v1 = 0.25, v2 = 0.5, v3 = 0.5, v4 = 0.1 --- ####
model.prediction <- predict(model.inla,
                data.pred.cov= data.frame(v1 = 0.25,
                                       v2 = 0.5,
                                       v3 = 0.5,
                                       v4 = 0.1))
model.prediction$summary_predictive_means
```

`summary.dirinlaregmodel` *Summary of dirinlaregmodel objects*

Description

'summary.dirinlaregmodel' is a function which gives a summary of a dirinlaregmodel object

Usage

```
## S3 method for class 'dirinlaregmodel'
summary(object, ...)
```

Arguments

`object`	Object of class dirinlaregmodel.
`...`	Other arguments.

Value

Print summary.

Author(s)

Joaquín Martínez-Minaya <<jomarminaya@gmail.com>>

---

summary_fast — *Summary using the packages Rfast and Rfast2 of a matrix by rows*

Description

'summary_fast' summarise a matrix by rows

Usage

```
summary_fast(A)
```

Arguments

A	matrix to be summarised

Value

A matrix whose columns are "mean", "sd", "0.025quant", "0.5quant", "0.975quant"

Author(s)

Joaquín Martínez-Minaya <<jomarminaya@gmail.com>>

Examples

```
A <- matrix(rnorm(10000), ncol = 1000)
summary_fast(A)
```

---

trigamma_red — *Computing the function trigamma*

Description

'trigamma_red' is the function trigamma appropiate for really small values

Usage

```
trigamma_red(x, ...)
```

Arguments

x	Argument to applied the function trigamma.
...	Rest of arguments used in the case of digamma functions.

Value

Result of applying trigamma function.

Author(s)

Joaquín Martínez-Minaya <<joaquin.martinez-minaya@uv.es>>

Index