Package 'ProfileGLMM'

December 18, 2025

Type Package

Title Bayesian Profile Regression using Generalised Linear Mixed Models

Version 1.0.2

Description Implements a Bayesian profile regression using a generalized linear mixed model as output model. The package allows for binary (probit mixed model) and continuous (linear mixed model) outcomes and both continuous and categorical clustering variables. The package utilizes 'RcppArmadillo' and 'RcppDist' for high-performance statistical computing in C++. For more details see Amestoy & al. (2025) <doi:10.48550/arXiv.2510.08304>.

License GPL-2

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.3.2

LinkingTo Rcpp, RcppArmadillo, RcppDist

Imports Rcpp, LaplacesDemon, MCMCpack, Matrix, Spectrum, mvtnorm

Depends R (>= 3.5)

URL https://github.com/MatteoAmestoy/ProfileGLMM-package

BugReports https://github.com/MatteoAmestoy/ProfileGLMM-package/issues

NeedsCompilation yes

Author Matteo Amestoy [aut, cre, cph],
Mark van de Wiel [ths],
Wessel van Wieringen [ths]

Maintainer Matteo Amestoy <m.amestoy@amsterdamumc.nl>

Repository CRAN

Date/Publication 2025-12-18 13:50:07 UTC

Contents

`encodeCat` *One-Hot Encodes Factor Variables (FIRST Level as Reference)*

Description

This function takes a dataframe, identifies all columns of class `factor`, and converts them into **dummy variables** using one-hot encoding via `stats::model.matrix`. For each factor, the function explicitly removes the first dummy variable generated, effectively making the **first level** of the factor the **reference level** (omitted category). Non-factor columns are retained as is.

Usage

```
encodeCat(dataframe)
```

Arguments

`dataframe`	A `data.frame` containing the data to be processed, which may include factor variables.

Value

A `data.frame` where:

- All original non-factor columns are present.
- All original factor columns are replaced by a set of binary (0/1) dummy variables. The first level of the factor is excluded from the generated dummies, making the last level the reference.

Examples

```
data("exposure_data")
exp_data = exposure_data$df
covList = {}
covList$FE = c('X')
XFE = encodeCat(exp_data[,covList$FE, drop = FALSE])
```

examp — *List of the different outputs of the main function for examples*

Description

A list of the different outputs of the main function for examples

Usage

```
examp
```

Format

A list with 4 components:

dataProfile Output of the profileGLMM_preprocess() function example

MCMC_Obj Output of the profileGLMM_Gibbs() function example

post_Obj Output of the profileGLMM_postprocess() function example

pred_Obj Output of the profileGLMM_predict() function example

Source

Generated synthetically by the package authors.

exposure_data — *Simulated Data and Parameters for a exposure profile linear mixed model*

Description

A list containing a simulated exposure dataset (`df`) and the ground-truth parameters (`theta0`) used to generate it.

The dataset `df` contains $N = 4500$ observations across $n_{Ind} = 1500$ individuals, with $n_R = 3$ repeated measures per individual.

Usage

```
exposure_data
```

Format

A list with 2 components:

df A data frame with 4,500 rows and 6 variables (the simulated data).

theta0 A list of 11 elements containing the true parameters used for simulation.

Details

The underlying model for the response $\boldsymbol{Y}$ is:

$$\boldsymbol{Y} = \boldsymbol{X}_{Fe}\boldsymbol{\beta} + \boldsymbol{X}_{Int}\boldsymbol{\alpha}_{Lat} + \boldsymbol{X}_{Re}\boldsymbol{\alpha}_{RE} + \boldsymbol{\epsilon}$$

`df` Data Variables

X Continuous predictor ($\sim N(0, 1)$).

t Time-like variable (structured around 0, 1, 2).

indiv **Individual ID** (1 to 1500), the grouping factor.

Exp1, Exp2 Exposure continuous predictors.

Y The **Simulated Response Variable** calculated as: $\boldsymbol{Y} = y_{Fe} + y_{Int} + y_{Re} + \epsilon$, where $\epsilon\ N(0, 1)$.

`theta0` Parameters

The list `theta0` holds the true values used to generate `Y`, including:

- `Lat`: **Categorical Factor** (9 levels), defining the clusters for interaction effects.
- `beta`: True fixed effects for the global intercept and $\boldsymbol{X}$ (i.e., $(3, 2)$).
- `alphaLat`: Vector of 18 coefficients defining the cluster-specific intercepts and slopes for $\boldsymbol{X}$ within the 9 `Lat` categories.
- `alphaRE`: Vector of 1500 random slopes for the time variable $\boldsymbol{t}$, drawn from $N(0, 1)$.
- `sigma`: Residual standard deviation (1).

Source

Generated synthetically by the package authors.

`piecewise_data` *Simulated Data and Parameters for a Piecewise Example*

Description

A list containing a second simulated dataset (`df`) and its ground-truth parameters (`theta0`). This dataset is generated from a **piecewise linear model**, where the continuous predictor `x` is segmented into 6 bins, and different intercept and slope coefficients are applied to each segment.

The dataset `df` contains $N = 3000$ observations.

Usage

```
piecewise_data
```

Format

A list with 2 components:

df A data frame with 3,000 rows and 2 variables (the simulated data).

theta0 A list of 5 elements containing the true parameters used for simulation.

Details

The underlying model for the response $\boldsymbol{Y}$ is:

$$\boldsymbol{Y} = \boldsymbol{X}_{Fe}\boldsymbol{\beta} + \boldsymbol{X}_{Lat}\boldsymbol{\alpha}_{Lat} + \boldsymbol{\epsilon}$$

where $\boldsymbol{X}_{Fe}$ is the global intercept, and $\boldsymbol{X}_{Lat}\boldsymbol{\alpha}_{Lat}$ models the piecewise relationship of `x` across the 6 categories defined in `theta0$Lat`. The error term $\boldsymbol{\epsilon}$ $N(0, 1)$.

`df` Data Variables

x A continuous predictor, uniformly distributed between -3 and 3.

Y The **Simulated Response Variable** defined by the piecewise linear model.

`theta0` Parameters

The list `theta0` holds the true values used for simulation, including:

- `beta`: True global intercept (i.e., (0.5)).
- `Lat`: The categorical factor (1 to 6) derived from segmenting `x`.
- `alphaLat`: Vector of $2 * 6 = 12$ coefficients defining the specific intercept and slope for `x` within each of the 6 segments.

Source

Generated synthetically by the package authors.

`prior_init` *Initialize the prior hyperparameters for the Profile GLMM*

Description

This function establishes the prior distributions for all parameters in the Profile GLMM. It sets up vague, non-informative priors (often using small precision/large variance or conjugate forms like Wishart/Dirichlet) for the fixed effects ($beta_{FE}$), residual variance (σ^2), random effects covariance (Σ_{RE}), latent effects covariance (Σ_{Lat}), cluster parameters (means and covariances), and the Dirichlet Process parameters (α).

Usage

```
prior_init(params)
```

Arguments

`params` A list containing dimensional parameters of the model (often the output of `process_Data_outcome`). Important fields used for prior setup include:

`qFE`: Number of fixed effects coefficients.

`qRE`: Dimension of the random effects vector.

`qLat`: Dimension of the latent effects vector.

`qUCont`: Number of continuous profile variables.

`qUCat`: Number of categorical profile variables.

Value

A list (`prior`) containing the hyperparameter values structured by the parameter block they govern:

`FE`: Priors for fixed effects and residual variance (e.g., `lambda`, `a`, `b` for conjugate Normal-Gamma).

`RE`: Inverse-Wishart priors for random effects covariance (Σ_{RE}) (e.g., `Phi`, `eta`).

`assign`: Priors for the cluster assignment parameters, nested under `Cont` (Normal-Inverse-Wishart for continuous) and `Cat` (Dirichlet for categorical).

`Lat`: Inverse-Wishart prior for the latent effects covariance (Σ_{Lat}) (e.g., `Phi`, `eta`).

`DP`: Parameters for the Dirichlet Process prior (e.g., `scale`, `shape`).

Examples

```
# Load dataProfile, the result of profileGLMM_preProcess()
data("examp")
dataProfile = examp$dataProfile
prior_config <- prior_init(dataProfile$params)
```

`profileGLMM_Gibbs` *R Wrapper for Profile GLMM Gibbs Sampler (C++ backend)*

Description

This is the main function for fitting the Profile Generalized Linear Mixed Model (Profile GLMM) using a blocked Gibbs sampling algorithm. It acts as an R wrapper, passing pre-processed data, initial values, and prior hyperparameters contained in the `model` object directly to the C++ implementation `GSLoopCPP`. The function simulates the posterior distribution of all model parameters, including fixed effects, random effects variance, profile cluster parameters, latent effects, and cluster assignments.

Usage

```
profileGLMM_Gibbs(model, nIt, nBurnIn)
```

Arguments

model	A list object containing all data, initial parameter values, model dimensions, prior hyperparameters, and model configuration (e.g., regression type). This object is typically the output of a data processing function like process_Data_outcome. Key components include: d: Data matrices (Y, XFE, XRE, XLat, UCont, UCat). params: Model dimension parameters (e.g., nC, qRE, qUCont). theta: Initial values for parameters (β_{FE}, σ^2, Σ_{RE}, cluster means, cluster covariance, cluster prob. vectors, Σ_{Lat}, γ_{Lat}). prior: Hyperparameters for all prior distributions (e.g., Normal, Inverse-Wishart, Dirichlet). regType: The type of regression being performed.
nIt	Integer, the total number of MCMC iterations *counting* the burn-in period. The sampler will run for nIt - nBurnIn iterations in total.
nBurnIn	Integer, the number of initial MCMC iterations that are discarded (not saved) to allow the chain to converge.

Value

A list containing the saved Gibbs-sampled MCMC chains for all model parameters (e.g., beta, Z, gamma, pvec, muClus, PhiClus, etc.) and the variable names from the original data. This output is ready for post-processing with profileGLMM_postProcess.

Examples

```
# Load dataProfile, the result of profileGLMM_pREProcess()
data("examp")
dataProfile = examp$dataProfile
MCMC_Obj = profileGLMM_Gibbs(model = dataProfile,
   nIt = 100,
   nBurnIn = 10
 )
```

---

profileGLMM_postProcess

Post-process the MCMC chain from profileGLMM_Gibbs

Description

This function performs essential post-processing of the MCMC output generated by the profileGLMM_Gibbs function. It calculates the posterior means and credible intervals for the fixed effects (population parameters) and, optionally, computes a representative cluster partition using methods like Least Squares (LS) or Ng's spectral clustering (NG) on the co-occurrence matrix. It also provides estimated cluster characteristics (centroids, probability vectors, and outcome effects) for the representative partition.

Usage

```
profileGLMM_postProcess(
  MCMC_Obj,
  modeClus = "NG",
  comp_cooc = TRUE,
  alpha = 0.05
)
```

Arguments

MCMC_Obj	Profile GLMM MCMC output of the profileGLMM_Gibbs function. This object must contain the raw MCMC samples for fixed effects (beta), cluster assignments (Z), cluster parameters (pvec, muClus, PhiClus), outcome effects (gamma), and names (names).
modeClus	A character string specifying the clustering method to determine the representative partition. Options are 'NG' (Ng's spectral clustering, default) or 'LS' (Least Squares clustering).
comp_cooc	A logical value. If TRUE (default), the co-occurrence matrix is computed and clustering is performed to find a representative partition. If FALSE, only the population parameters are processed.
alpha	A numeric value between 0 and 1, specifying the significance level for calculating the posterior credible intervals (CIs) of the fixed effects. Defaults to 0.05 (yielding 95% CIs).

Value

A list with three elements:

coocMat: The co-occurrence matrix of the MCMC cluster assignments (MCMC_Obj$Z).

clust: A list containing the results of the representative clustering (if comp_cooc = TRUE), including the optimal partition (Zstar), number of clusters (Kstar), representative cluster parameters (cen, pvec, gamma), and full posterior samples for the cluster characteristics.

pop: A list containing the posterior mean and (1-alpha) credible intervals for the fixed effects (betaFE).

Examples

```
# Load MCMC_Obj, the result of profileGLMM_Gibbs()
data("examp")
MCMC_Obj = examp$MCMC_Obj
post_Obj = profileGLMM_postProcess(MCMC_Obj, modeClus='LS')
print(post_Obj$pop$betaFE)
```

profileGLMM_predict *Prediction of cluster memberships and outcomes*

Description

This function uses the results of the post-processed Profile GLMM MCMC chain to predict cluster memberships and outcomes for new or existing data. It first calculates the fixed effect (FE) contribution and then, if a representative clustering is available in post_Obj, computes the predicted cluster membership and the corresponding latent effect (Lat) contribution to the outcome.

Usage

```
profileGLMM_predict(post_Obj, XFE, XLat, UCont, UCat)
```

Arguments

post_Obj	The post-processed output from the profileGLMM_postProcess function. Must contain pop for population constant parameters and optionally clust for cluster-specific parameters.
XFE	A numeric matrix of fixed effects covariates for the prediction data.
XLat	A numeric matrix of latent effect covariates. This matrix is used for the interaction term with the predicted cluster membership.
UCont	A numeric matrix or vector of continuous profile variables (used for predicting cluster membership). Set to NULL if no continuous variables were used in the model.
UCat	A numeric matrix or vector of categorical profile variables (used for predicting cluster membership). Set to NULL if no categorical variables were used in the model.

Value

A list with the following elements:

FE: A numeric vector of the predicted fixed effects contribution to the outcome.

Y: A numeric vector of the total predicted outcome (FE + Lat).

classPred: A factor vector of the predicted cluster membership for each observation. NULL if no representative clustering was provided in post_Obj.

Int: A numeric vector of the predicted latent effect contribution to the outcome. NULL if no representative clustering was provided.

Examples

```
# Load post_Obj, the result of profileGLMM_postProcess()
data("examp")
post_Obj = examp$post_Obj
# Load dataProfile, the result of profileGLMM_preProcess()
dataProfile = examp$dataProfile
pred_Obj = profileGLMM_predict(post_Obj,
                               dataProfile$d$XFE,
                               dataProfile$d$XLat,
                               dataProfile$d$UCont,
                               dataProfile$d$UCat)
```

`profileGLMM_preprocess` *Preprocess the data from a list describing the profile LMM model*

Description

Preprocess the data from a list describing the profile LMM model

Usage

```
profileGLMM_preprocess(
  regType,
  covList,
  dataframe,
  nC,
  intercept = list(FE = TRUE, RE = TRUE, Lat = TRUE)
)
```

Arguments

`regType`	A string, current possibilities: linear or probit
`covList`	A list with fields: • FE fixed effect covariates names/index in dataframe • RE random effect covariates names/index in dataframe • Lat latent effect covariates names/index in dataframe • Assign assignement variables list with fields: – Cont Continuous variables names/index in dataframe – Cat Categorical variables names/index in dataframe • REunit statistical unit of the RE colomn name/index • Y outcome (Continuous)
`dataframe`	A dataframe containing outcome anf covariates
`nC`	int: maximal number of cluster for the DP truncation
`intercept`	(optionnal): A list with fields

- RE bool indicating if FE have an intercept
- FE bool indicating if RE have an intercept
- Lat bool indicating if Latent have an intercept

Value

A list with

- d dictionary with [XFE,XRE,XLat,UCont,UCat,ZRE] design matrices
- [[params]] list of the parameters of the data
  - n int nb of obs
  - qFE lint, number of covariates of FE
  - nRE int, number of stat units of RE
  - qRE int, number of covariates of RE
  - qLat int, number of covariates interacting with the latent clusters
  - qUCont int, number of continuous clustering covariates
  - qUCat int, number of categorical clustering covariates
  - nC int, maximal number of clusters
- prior a list with all the specification of the default prior used
- theta a list with a default set of parameters to start the chain, drawn from the prior
- regType an int. Currently 0 for linear, 1 for probit

Examples

```
data("exposure_data")
exp_data = exposure_data$df
theta0 = exposure_data$theta0
covList = {}
covList$FE = c('X')
covList$RE = c('t')
covList$REunit = c('indiv')

covList$Lat = c('X')

covList$Assign$Cont = c('Exp1','Exp2')
covList$Assign$Cat = NULL

covList$Y = c('Y')
dataProfile = profileGLMM_preprocess(regType = 'linear',
                                     covList = covList,
                                     dataframe = exp_data,
                                     nC = 30,
                                    intercept = list(FE = TRUE, RE = FALSE, Lat = TRUE))
```

theta_init *Initialize the variables for the Gibbs sampler chain*

Description

This function generates initial values (theta) for all parameters in the Profile GLMM Gibbs sampler by drawing from the specified prior distributions. These initial values are crucial for starting the MCMC chain in profileGLMM_Gibbs. The initialization includes parameters for fixed effects, random effects variance, latent effects, and the profile cluster parameters (centroids, covariances, and categorical probability vectors).

Usage

```
theta_init(prior, params)
```

Arguments

prior	A list containing the prior configuration to draw initialization from. This list should match the structure produced by the prior_init function, including hyperparameters for FE, RE, Latent, and cluster assignment priors.
params	A list containing the problem's dimensional parameters and indices (e.g., number of observations, number of covariates). This list should match the structure of the output from process_Data_outcome.

Value

A list (theta) containing the sampled initialization values for the Gibbs sampler. Key elements include:

sig2: Initial residual variance.

betaFE: Initial fixed effects coefficients.

SigRE: Initial random effects covariance matrix.

SigLat: Initial latent effects covariance matrix.

gammaLat: Initial latent effects coefficients, organized by cluster.

ClusCont: List containing initial continuous cluster parameters (mu and Sigma).

ClusCat: List containing initial categorical cluster parameters (pvecClus).

Examples

```
# Load dataProfile, the result of profileGLMM_preProcess()
data("examp")
dataProfile = examp$dataProfile
theta = theta_init(dataProfile$prior,dataProfile$params)
```

Index