Package: MultRegCMP (via r-universe)

October 31, 2026

Type Package

Title Bayesian Multivariate Conway-Maxwell-Poisson Regression Model for Correlated Count Data

Version 0.1.0

Description Fits a Bayesian Regression Model for multivariate count data. This model assumes that the data is distributed according to the Conway-Maxwell-Poisson distribution, and for each response variable it is associate different covariates. This model allows to account for correlations between the counts by using latent effects based on the Chib and Winkelmann (2001) <http://www.jstor.org/stable/1392277> proposal.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports purrr, mvnfast, stats, progress, bayesplot, ggplot2, cowplot

Depends R (>= 2.10)

RoxygenNote 7.3.1

Config/Needs/website rmarkdown

Config/pak/sysreqs libicu-dev

Repository https://mauroflorez.r-universe.dev

Date/Publication 2024-06-19 19:29:51 UTC

RemoteUrl https://github.com/mauroflorez/cmp

RemoteRef HEAD

RemoteSha a30c24d6334c558e0fb2a977f8752a79f751761f

Contents

com_sampler 2
DIC_cmp 3
epl_20_21 3

fitting_plots 4
llk_cmp 4
log_cmp 5
mcmc_cmp 6
prod_list 8

Index **9**

`com_sampler` *Rejection Sampler - COM-Poisson*

Description

Sampler for the Conway-Maxwell-Poisson as described in Algorithm 2 - Benson & Friel (2021)

Usage

```
com_sampler(mu, nu, n = 1, ndraws = FALSE)
```

Arguments

`mu`	Location parameter
`nu`	Shape parameter
`n`	Number of draws (default = 1)
`ndraws`	Optional: Return the number of draws required to generate the n samples.

Value

A list or numeric in case ndraws = FALSE:

`sample`	Values sampled from the distribution
`drawsa`	Number of draws required in the rejection sampler
`log_Bf`	Log of the boundary of the rejection sampler

Examples

```
com_sampler(2, 0.2, n = 10, ndraws = TRUE)
com_sampler(1, 2)
```

DIC_cmp *DIC of the regression model*

Description

This function used an approach similar to the presented by Benson & Friel (2021) to calculate the BIC. We select S a sample size of the posterior samples to speed up computation

Usage

```
DIC_cmp(fit, S = 100)
```

Arguments

fit	An object from the mcmc_cmp_mh
S	Number of iterations used to calculate the DIC

Value

Vector of approximated DIC

epl_20_21 *Scores English Premier League Season 2020-2021*

Description

A data set with the scores of the games played during season 2020-2021 in the English Premier League (EPL)

Usage

```
epl_20_21
```

Format

A data frame with 380 rows and 4 variables:

HG Goals scored by home team.

AG Goals scored by away team.

HomeTeam Home team.

AwayTeam Away team.

Source

<https://www.football-data.co.uk>

`fitting_plots` *Rootograms plots - Multivariate CMP*

Description

Rootograms plots - Multivariate CMP

Usage

```
fitting_plots(fit, type = "rootogram", S = 100)
```

Arguments

`fit`	An element from ‘mcmc_cmp‘
`type`	Wheter to do a bar plot or a rootogram
`S`	Optional. Indicates the number of posterior samples used (Default 100)

Value

No return value, called for plotting only

Examples

```
n = 50; J = 2
X = list(matrix(rnorm(3*n), ncol = 3), matrix(rnorm(3*n), ncol = 3))
beta <- list(c(1,0.1, 1), c(0, 0.5, -0.5))
mu <- exp(prod_list(X, beta))
y = matrix(rpois(n = length(mu), lambda = mu), nrow = n)
fit <- mcmc_cmp(y, X, S = 1000, nburn = 1000, scale_cov_b = 0.8,
scale_cov_beta = 0.04, scale_cov_gamma = 0.06)
fitting_plots(fit)
```

`llk_cmp` *Log likelihood of the Conway-Maxwell-Poisson Distribution*

Description

This function calculates the log likelihood of the distribution as described by Benson and Friel (2021)

Usage

```
llk_cmp(y, mu, nu, r = 1000)
```

Arguments

y	Count value
mu	Location parameter
nu	Shape parameter
r	Number of acceptances

Value

Estimation of the log likelihood of the distribution

Examples

```
llk_cmp(10, 5, 2)
```

---

log_cmp *Log density of the normalized component of the Conway-Maxwell-Poisson*

Description

Log density of the normalized component of the Conway-Maxwell-Poisson

Usage

```
log_cmp(y, mu, nu)
```

Arguments

y	Value
mu	Location parameter
nu	Shape parameter

Value

Numeric corresponding to the log of the unnormalized component of the distribution

mcmc_cmp *MCMC Algorithm for Conway-Maxwell-Poisson Regression Model for Multivariate Correlated Count Data*

Description

MCMC Algorithm to estimate the parameters in the regression model for multivariate correlated count data

Usage

```
mcmc_cmp(
  y,
  X,
  S = 10000,
  nburn = 5000,
  initial_beta,
  initial_gamma,
  initial_b,
  prior_mean_beta,
  prior_var_beta,
  prior_mean_gamma,
  prior_var_gamma,
  v_0,
  R_0,
  intercept = FALSE,
  scale_b,
  scale_beta,
  scale_gamma,
  scale_cov_b,
  scale_cov_beta,
  scale_cov_gamma,
  inc_burn = FALSE,
  re_chain = TRUE,
  way = 2,
  random_seed,
  ...
)
```

Arguments

y	Matrix of observations
X	Covariates list, each element is the design matrix for each column of y
S	Number of MCMC samples to be drawn
nburn	Number of MCMC samples to burn-in
initial_beta	List with initial value of $beta$ for each response

initial_gamma	List with initial value of $gamma$ for each response
initial_b	Initital value of b.
prior_mean_beta	Prior mean for $beta$. (Default zero vector)
prior_var_beta	Prior covariance matrix for $beta$ (Default I)
prior_mean_gamma	Prior mean for $beta$. (Default zero vector)
prior_var_gamma	Prior covariance matrix for $gamma$ (Default I)
v_0	Prior degrees of freedom of random effects
R_0	Prior covariance matrix of random effects
intercept	Logical value indicating whether include the intercept
scale_b	Covariance matrix for RW proposals of the random effects (Default I)
scale_beta	List with initial values for the scale matrices of $beta$ (Default I)
scale_gamma	List with initial values for the scale matrices of $gamma$ (Default I)
scale_cov_b	Scale parameter for the RW of random effects. (Default $2.4/sqrt(2)$)
scale_cov_beta	Scale parameter for the covariance of the proposals.
scale_cov_gamma	Scale parameter for the covariance of the proposals.
inc_burn	logical: include burned samples in the return
re_chain	logical: If the posterior samples for the r.e are include. False return just the mean
way	How to calculate the MCMC updates, based on Chib (2001)
random_seed	Random seed
...	Additional parameters of the MCMC algorithm

Value

A list:

posterior_b	List with posterior values of the random effects
estimation_beta	Estimation of beta parameters
posterior_beta	List with posterior values of beta
estimation_gamma	Estimation of gamma parameters
posterior_gamma	List with posterior values of gamma
posterior_D	Values of covariance matrix D
fitted_mu	Posterior of location parameters for each response
fitted_nu	Posterior of shape parameters for ecah response
accept_rate_b	Acceptance rate of Random Effects

`accept_rate_beta`
: Acceptance rate of beta

`accept_rate_gamma`
: Acceptance rate of gamma

`scale_beta`
: Estimated Scale matrix for beta parameters

`scale_gamma`
: Estimated Scale matrix for gamma parameters

`X`
: List of covariates used

`y`
: Matrix of observed counts

Examples

```
n = 50; J = 2
X = list(matrix(rnorm(3*n), ncol = 3), matrix(rnorm(3*n), ncol = 3))
beta <- list(c(1,0.1, 1), c(0, 0.5, -0.5))
mu <- exp(prod_list(X, beta))
y = matrix(rpois(n = length(mu), lambda = mu), nrow = n)
fit <- mcmc_cmp(y, X, S = 10000, nburn = 1000, scale_cov_b = 0.8,
scale_cov_beta = 0.04, scale_cov_gamma = 0.06)
```

---

prod_list *Product of lists between matrices*

Description

Product of lists between matrices

Usage

```
prod_list(X, beta)
```

Arguments

`X`
: Data

`beta`
: Parameters

Value

A list with the products element-wise

Index

∗ **datasets**
epl_20_21, 3

com_sampler, 2

DIC_cmp, 3

epl_20_21, 3

fitting_plots, 4

llk_cmp, 4
log_cmp, 5

mcmc_cmp, 6

prod_list, 8