Package 'DiscreteDLM'

July 21, 2025

Title Bayesian Distributed Lag Model Fitting for Binary and Count Response Data

Version 1.0.0

Maintainer Daniel Dempsey <daniel.dempsey0@gmail.com>

Description Tools for fitting Bayesian Distributed Lag Models (DLMs) to longitudinal response data that is a count or binary. Count data is fit using negative binomial regression and binary is fit using quantile regression. The contribution of the lags are fit via b-splines. In addition, infers the predictor inclusion uncertainty. Multimomial models are not supported. Based on Dempsey and Wyse (2025) <doi:10.48550/arXiv.2403.03646>.

License GPL-3

Encoding UTF-8

Imports statmod, BayesLogit, dlnm, splines, dplyr, ggplot2, ggridges, reshape2

RoxygenNote 7.3.2

URL https://github.com/DanDempsey/DiscreteDLM

BugReports https://github.com/DanDempsey/DiscreteDLM/issues

NeedsCompilation yes

Author Daniel Dempsey [aut, cre] (ORCID: <https://orcid.org/0000-0003-0899-5412>),
Jason Wyse [ctb] (ORCID: <https://orcid.org/0000-0003-1391-7371>),
Christian E. Galarza [ctb] (Author of the ald package, which this package derives some code from (see the COPYRIGHT file in the inst folder).),
Victor H. Lachos [ctb] (Author of the ald package, which this package derives some code from (see the COPYRIGHT file in the inst folder).)

Repository CRAN

Date/Publication 2025-02-06 19:40:10 UTC

Contents

`ALD` *The Asymmetric Laplacian Distribution*

Description

Density, distribution function, quantile function and random generation for the Asymmetric Laplacian Distribution (ALD). Also contains random number generation for a truncated ALD. Our code here is heavily derived from the "ald" package by Galarza and Lachos (2021), adapted to vectorize the mu parameter.

Usage

```
dALD(x, mu = 0, sigma = 1, p = 0.5)
pALD(q, mu = 0, sigma = 1, p = 0.5)
qALD(prob, mu = 0, sigma = 1, p = 0.5)
rALD(n, mu = 0, sigma = 1, p = 0.5)
rTALD(n, upper_tail, mu = 0, sigma = 1, p = 0.5)
```

Arguments

`mu`	vector of location parameters.
`sigma`	vector of scale parameters.
`p`	ALD skew parameter.
`prob`	vector of probabilities.
`n`	number of observations.
`x, q`	vector of quantiles.
`upper_tail`	Logical denoting what portion of the distribution is retained. If TRUE, only positive values are sampled, and negative if FALSE.

Details

These functions are based on the three parameter ALD:

$$f(x|\mu,\sigma,p) = \frac{p(1-p)}{\sigma} exp\left(-\rho_p(\frac{x-\mu}{\sigma})\right)$$

where

$$\rho_p(z) = z(p - I_{z<0})$$

These functions differ than the ones provided in the *ald* package by vectorising the mu parameter. In addition we have implemented a function to sample from the truncated ALD (rTALD), which is needed for Bayesian quantile regression. Note that truncation is fixed at zero; the user only decides whether the negative or positive axis is discarded.

Value

Each function returns a vector. dALD, pALD and qALD returns the PDF, CDF and inverse CDF of the ALD respectively. rALD returns a random sample from the ALD distribution, and rTALD returns a random sample from the ALD truncated at 0.

Author(s)

Daniel Dempsey (daniel.dempsey0@gmail.com). Original code from the *ald* package were written by Christian E. Galarza and Victor H. Lachos.

Examples

```
set.seed( 100 )

### Vectorised input
random_ALD <- rALD( 1000, mu = runif(1000, -100, 100) )
plot( random_ALD )

### Truncated version
trunc_random_ALD <- rTALD( 1000, TRUE, mu = 2, sigma = 3, p = 0.75 )
plot( dALD(sort(trunc_random_ALD), mu = 2, sigma = 3, p = 0.75) )
```

`dataframe_DLM` *Creating a DLM-Ready Dataframe*

Description

Transforms a dataframe to prepare it for distributed lag modelling.

Usage

```
dataframe_DLM(X, lag, dynamic_vars = NULL, arglag = list(fun = "bs"), ...)
```

Arguments

`X`	A dataframe, or something that can be coerced into one.
`lag`	Lag length of the dynamic variable. See `crossbasis` for more details.
`dynamic_vars`	The column name or indices that correspond to the longitudinal variables. If left missing, the dataset is not altered in any way and a warning is supplied.
`arglag`	A list that is passed into `onebasis` for generating a basis matrix for the lag space. See `crossbasis`.
`...`	Further arguments to be passed into the `crossbasis` function.

Details

The purpose of this function is to streamline the preperation of the data for distributed lag modelling in the case where every dynamic variable is handled the same way.

If no dynamic variables are given, the input dataframe is returned unaltered with a warning. Otherwise, the function returns a *dataframe_DLM* object, which is essentially treated like a standard dataframe but contains extra information to simplify the process of distributed lag modelling when using the *NB_MCMC* or *QB_MCMC* functions.

Value

The input is returned unchanged if no dynamic variables are given. Otherwise a *dataframe_DLM* object is returned. See details.

Author(s)

Daniel Dempsey (daniel.dempsey0@gmail.com)

Examples

```
X <- dplyr::select( dlnm::chicagoNMMAPS, c('cvd', 'dow', 'temp', 'dptp', 'o3') )
X <- na.omit( X )
arglag <- list( fun = 'bs', df = 4 )
DLM_dat <- dataframe_DLM( X, lag = 40, dynamic_vars = c('temp', 'dptp', 'o3'), arglag = arglag )
```

`DiscreteDLM-overview` *Fitting Discrete Distributed Lag Models with Variable Selection via MCMC*

Description

DiscreteDLM contains functionality for fitting and visualising Bayesian Distributed Lag Models (DLMs) when the response variable is either count or binary. More specifically, this package contains an implementation of Bayesian quantile binary regression (for binary response data) and negative-binomial regression (for count response data). Additionally, these functions implement variable uncertainty inference for each predictor. Multinomial regression is not implemented.

While these functions can be applied to any standard regression problem, the package was set up with DLMs in mind; the functions therein allow for an easy-to-use pipeline to go from defining the DLM, fitting the DLM, and visualising the results.

Details

Package:	DiscreteDLM
Type:	Package
Version:	0.9.8
Date:	2024-11-06
License:	GPL (>=3)

Author(s)

Daniel Dempsey <<`daniel.dempsey0@gmail.com`>>, with contributions and supervision by Jason Wyse <<`wyseja@tcd.ie`>>

References

Daniel Dempsey and Jason Wyse. 2025. "Bayesian Variable Selection in Distributed Lag Models: A Focus on Binary Quantile and Count Data Regressions." https://doi.org/10.48550/arXiv.2403.03646.

See Also

`dataframe_DLM`,`NB_MCMC`,`QB_MCMC`

`NB_MCMC` *Negative Binomial Regression via MCMC*

Description

Fits a negative binomial regression model using Gibbs sampling and performs Bayesian variable selection.

Usage

```
NB_MCMC(
  formula,
  data = NULL,
  nsamp = 1000,
  nburn = 1000,
  thin = 1,
  standardize = TRUE,
  prior_beta_mu = 0,
  prior_beta_sigma = 100,
  prior_gamma_p = 0.5,
  prior_xi_shape = 2,
  prior_xi_scale = 1/50,
  init_beta = 0,
  init_gamma = FALSE,
  init_xi = 1,
  verbose = TRUE
)
```

Arguments

`formula`	Formula object to set the symbolic description of the model to be fitted.
`data`	Optional dataframe. Ideally this should be a `dataframe_DLM` object if doing distributed lag modelling.
`nsamp`	The desired sample size from the posterior. Set to 5000 by default.
`nburn`	The number of iterations of the MCMC to be discarded as burn-in. Set to 5000 by default.
`thin`	Thinning factor of the MCMC chain after burn-in. Set to 1 by default.
`standardize`	Logical indicating if the data should be standardised prior to fitting the model to zero mean and unit variance. If there is no intercept, then only scaling is applied (with a warning). The posterior sample of beta is transformed back to the original scale upon completion of the algorithm.
`prior_beta_mu`	Mean of the Gaussian prior for the regression coefficients, beta. Either a vector of length equal to the number of predictors or a single numeric to represent a constant vector.
`prior_beta_sigma`	Covariance matrix of the Gaussian prior for the regression coefficients, beta. Either a square matrix of dimension equal to the number of predictors or a single numeric to represent an isotropic covariance matrix.
`prior_gamma_p`	Probability parameter of the Bernoulli prior for the predictor inclusion parameter, gamma. Either a vector of length equal to the number of predictors or a single numeric, to represent that all predictors have the same prior probability of inclusion.
`prior_xi_shape`	Shape parameter of the Gamma distribution prior for the negative binomial stopping parameter, xi.
`prior_xi_scale`	Scale parameter of the Gamma distribution prior for the negative binomial stopping parameter, xi.
`init_beta`	Initial MCMC values for the beta parameters. Either a vector of length equal to the number of predictors or a single numeric representing the same starting value for each beta component.
`init_gamma`	Initial MCMC values for the gamma parameters. Either a vector of length equal to the number of predictors or a single numeric representing the same starting value for each gamma component.
`init_xi`	Initial MCMC value for xi.
`verbose`	Logical indicating if a progress report should be printed to the console during the run.

Details

This function fits a negative binomial regression model via MCMC. Latent variable representation allows for Gibbs sampling of the parameters; see Pillow and Scott (2012) and Zhou et. al. (2012). The algorithm also includes predictor inclusion uncertainty inference (inferred via a Metroplis step) adapted from Holmes and Held (2006).

The parameters of interest for this model are the regression slopes (beta), the binary predictor inclusion indicator (gamma) and the negative binomial stopping parameter (xi). For further details, see Dempsey and Wyse (2024).

Note that when setting initial values and priors for gamma, the intercept must be included and therefore its initial value and prior are forced to be equal to 1. If an intercept is not used, then the first variable is forced to be included instead.

Value

An "MCMC_DLM" object; a list containing the MCMC-derived posterior sample, as well as the data that were used.

Author(s)

Daniel Dempsey (daniel.dempsey0@gmail.com)

References

Chris C. Holmes and Leonhard Held. 2006. "Bayesian auxiliary variable models for binary and multinomial regression." Bayesian Analysis 1(1): 145-168.

Daniel Dempsey and Jason Wyse. 2025. "Bayesian Variable Selection in Distributed Lag Models: A Focus on Binary Quantile and Count Data Regressions." https://doi.org/10.48550/arXiv.2403.03646

Jonathan Pillow and James Scott. 2012. "Fully Bayesian inference for neural models with negative-binomial spiking." Advances in neural information processing systems 25.

Mingyuan Zhou, Lingbo Li, David Dunson and Lawrence Carin. 2012. "Lognormal and gamma mixed negative binomial regression." Proceedings of the... International Conference on Machine Learning. International Conference on Machine Learning. Vol. 2012. NIH Public Access.

See Also

QB_MCMC, dataframe_DLM, plot.MCMC_DLM

Examples

```
### Set up data
X <- dplyr::select( dlnm::chicagoNMMAPS, c('cvd', 'dow', 'temp', 'dptp', 'o3') )
X <- na.omit( X )
arglag <- list( fun = 'bs', df = 4 )
DLM_dat <- dataframe_DLM( X, lag = 40, dynamic_vars = c('temp', 'dptp', 'o3'), arglag = arglag )

### Fit model
# NOTE: Only using 100 total iterations for illustration purposes!
myfit <- NB_MCMC( cvd ~ ., data = DLM_dat, nsamp = 50, nburn = 50 )
summary( myfit )
```

`plot.MCMC_DLM` *Visualises Results from MCMC_DLM Fits*

Description

Provides some ggplot visualisations of the posterior from MCMC_DLM objects.

Usage

```
## S3 method for class 'MCMC_DLM'
plot(x, type = "beta", include_intercept = FALSE, print_output = TRUE, ...)
```

Arguments

`x`	An MCMC_DLM object.
`type`	One of "beta", "gamma", "xi", or "lags". Partial matching is supported. See Details below.
`include_intercept`	Logical indicating if the intercept should be included in the graphs.
`print_output`	Logical indicating whether or not the ggplot objects should be printed to the screen.
`...`	Extra arguments; currently unused.

Details

The type of plot generated is determined by the *type* argument. The options are:

- *beta*: A ridgeplot of the regression slopes,
- *gamma*: A bar chart of the mean probability of inclusion,
- *xi*: A kernel density estimate plot of the negative binomial stopping parameter (of course, this only works for negative binomial regression fits),
- *lags*: A collection of plots (supplied in a list), each containing a line chart of the lag-response for each dynamic variable.

Value

A ggplot object, except when type = 'lags', in which case it will be a list of ggplot objects. Will display the ggplot visual if print_output = TRUE.

Author(s)

Daniel Dempsey (daniel.dempsey0@gmail.com)

`QB_MCMC` *Binary Quantile Regression via MCMC*

Description

Fits a quantile regression model to a binary response dataset using Gibbs sampling and performs Bayesian variable selection.

Usage

```
QB_MCMC(
  formula,
  data = NULL,
  quantile = 0.5,
  nsamp = 1000,
  nburn = 1000,
  thin = 1,
  standardize = TRUE,
  prior_beta_mu = 0,
  prior_beta_sigma = 100,
  prior_gamma_p = 0.5,
  init_beta = 0,
  init_gamma = FALSE,
  verbose = TRUE
)
```

Arguments

`formula`	Formula object to set the symbolic description of the model to be fitted.
`data`	Optional dataframe. Ideally this should be a `dataframe_DLM` object if doing distributed lag modelling.
`quantile`	The chosen quantile for regression.
`nsamp`	The desired sample size from the posterior. Set to 5000 by default.
`nburn`	The number of iterations of the MCMC to be discarded as burn-in. Set to 5000 by default.
`thin`	Thinning factor of the MCMC chain after burn-in. Set to 1 by default (no values discarded after burn-in).
`standardize`	Logical indicating if the data should be standardised prior to fitting the model to zero mean and unit variance. If there is no intercept, then only scaling is applied (with a warning). The posterior sample of beta is transformed back to the original scale upon completion of the algorithm.
`prior_beta_mu`	Mean of the Gaussian prior for the regression coefficients, beta. Either a vector of length equal to the number of predictors or a single numeric to represent a constant vector.

`prior_beta_sigma`	Covariance matrix of the Gaussian prior for the regression coefficients, beta. Either a square matrix of dimension equal to the number of predictors or a single numeric to represent an isotropic covariance matrix.
`prior_gamma_p`	Probability parameter of the Bernoulli prior for the predictor inclusion parameter, gamma. Either a vector of length equal to the number of predictors or a single numeric, to represent that all predictors have the same prior probability of inclusion.
`init_beta`	Initial MCMC values for the beta parameters. Either a vector of length equal to the number of predictors or a single numeric representing the same starting value for each beta component.
`init_gamma`	Initial MCMC values for the gamma parameters. Either a vector of length equal to the number of predictors or a single numeric representing the same starting value for each gamma component.
`verbose`	Logical indicating if a progress report should be printed to the console during the run.

Details

This function fits a quantile binary regression model via MCMC. Latent variable representation allows for Gibbs sampling of the parameters; see Benoit and Van den Poel (2017). The algorithm also includes predictor inclusion uncertainty inference (inferred via a Metroplis step) adapted from Holmes and Held (2006).

The parameters of interest for this model are the regression slopes (beta) and the binary predictor inclusion indicator (gamma). For further details, see Dempsey and Wyse (2024).

Note that when setting initial values and priors for gamma, the intercept must be included and therefore its initial value and prior are forced to be equal to 1. If an intercept is not used, then the first variable is forced to be included instead.

Value

An "MCMC_DLM" object; a list containing the MCMC-derived posterior sample, as well as the data that were used.

Author(s)

Daniel Dempsey (daniel.dempsey0@gmail.com)

References

Chris C. Holmes and Leonhard Held. 2006. "Bayesian auxiliary variable models for binary and multinomial regression." Bayesian Analysis 1(1): 145-168.

Daniel Dempsey and Jason Wyse. 2025. "Bayesian Variable Selection in Distributed Lag Models: A Focus on Binary Quantile and Count Data Regressions." https://doi.org/10.48550/arXiv.2403.03646

Dries F. Benoit and Dirk Van den Poel. 2017. "bayesQR: A Bayesian approach to quantile regression." Journal of Statistical Software 76: 1-32.

See Also

NB_MCMC, dataframe_DLM, plot.MCMC_DLM

Examples

```
### Set up data
set.seed( 100 )
binary_response <- dlnm::chicagoNMMAPS$cvd > 65
predictors <- dplyr::select( dlnm::chicagoNMMAPS, c('dow', 'temp', 'dptp', 'o3') )
X <- cbind( binary_response, predictors )
X <- na.omit( X )
arglag <- list( fun = 'bs', df = 4 )
DLM_dat <- dataframe_DLM( X, lag = 40, dynamic_vars =  c('temp', 'dptp', 'o3'), arglag = arglag )

### Fit model
# NOTE: Only using 100 total iterations for illustration purposes!
myfit <- QB_MCMC( binary_response ~ ., data = DLM_dat, nsamp = 50, nburn = 50 )
summary( myfit )
```

Index