Package: copCAR (via r-universe)

October 22, 2024

Version 2.0-4

Date 2021-01-08

Title Fitting the copCAR Regression Model for Discrete Areal Data

Type Package

Author Emily Goren <emily.goren@gmail.com> and John Hughes <drjphughesjr@gmail.com>

Maintainer John Hughes <drjphughesjr@gmail.com>

Depends mcmcse, numDeriv, Rcpp, spam

Suggests lattice, parallel, pbapply

LinkingTo Rcpp, RcppArmadillo

RcppModules copCARmod

Description Provides tools for fitting the copCAR (Hughes, 2015) <DOI:10.1080/10618600.2014.948178> regression model for discrete areal data. Three types of estimation are supported (continuous extension, composite marginal likelihood, and distributional transform), for three types of outcomes (Bernoulli, negative binomial, and Poisson).

Encoding UTF-8

RoxygenNote 6.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2021-01-08 16:30:02 UTC

Contents

`adjacency.matrix` *Return an adjacency matrix for a square lattice.*

Description

Return an adjacency matrix for a square lattice.

Usage

```
adjacency.matrix(m, n = NULL)
```

Arguments

`m`	the number of rows in the lattice.
`n`	the number of columns in the lattice. Defaults to `NULL`. If missing, the lattice is assumed to be `m` by `m`.

Details

This function builds the adjacency matrix for the `m` by `n` square lattice.

Value

A matrix A of 0s and 1s, where A_{ij} is equal to 1 iff vertices i and j are adjacent.

`copCAR` *Fit copCAR model to discrete areal data.*

Description

Fit the copCAR model to Bernoulli, negative binomial, or Poisson observations.

Usage

```
copCAR(formula, family, data, offset, A, method = c("CML", "DT", "CE"),
  confint = c("none", "bootstrap", "asymptotic"), model = TRUE, x = FALSE,
  y = TRUE, verbose = FALSE, control = list())
```

Arguments

`formula`	an object of class "`formula`" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of the model specification are given under "Details".
`family`	the marginal distribution of the observations at the areal units and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See `family` for details of family functions.) Supported families are `binomial`, `negbinomial`, and `poisson`. When the negative binomial family is used, an initial value for θ must be passed to the `negbinomial` family function.
`data`	an optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in data, the variables are taken from `environment(formula)`, typically the environment from which `copCAR` is called.
`offset`	this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. This should be `NULL` or a numeric vector of length equal to the number of observations. One or more `offset` terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See `model.offset`.
`A`	the symmetric binary adjacency matrix for the underlying graph.
`method`	the method for inference. `copCAR` supports the continous extension ("`CE`"), distributional transform ("`DT`"), and composite marginal likelihood ("`CML`").
`confint`	the method for computing confidence intervals. This defaults to "`none`". The other options are "`bootstrap`" (for parametric bootstrap intervals using the quantile method) and "`asymptotic`" (for intervals computed using an estimate of the asymptotic covariance matrix).
`model`	a logical value indicating whether the model frame should be included as a component of the returned value.
`x`	a logical value indicating whether the model matrix used in the fitting process should be returned as a component of the returned value.
`y`	a logical value indicating whether the response vector used in the fitting process should be returned as a component of the returned value.
`verbose`	a logical value indicating whether to print various messages to the screen, including progress updates. Defaults to `FALSE`.
`control`	a list of parameters for controlling the fitting process. `bootit` the size of the (parametric) bootstrap sample. This applies when `confint = "bootstrap"`, or when `confint = "asymptotic"` and `method = "CML"` or `method = "DT"`. Defaults to `500`. `m` the number of vectors of standard uniforms used to approximate the expected likelihood when `method = "CE"`. Defaults to `1000`. `rho.max` the value $\rho^{\max}$, which is the maximum value of ρ used to approximate the CAR variances when `method = "CE"` or `method = "DT"`. If missing, assumed to be `0.999`.

`epsilon` the tolerance $\epsilon > 0$ used to approximate the CAR variances when `method = "CE"` or `method = "DT"`. If missing, assumed to be `0.01`. itemmaxitthe maximum number of iterations to be used by `optim` when optimizing the objective function. Defaults to `1000`.

`parallel` a logical value indicating whether to parallelize the bootstrap. This defaults to `TRUE` if the **parallel** package can be loaded.

`type` the cluster type, one of "`SOCK`" (default), "`PVM`", "`MPI`", or "`NWS`".

`nodes` the number of slave nodes to create.

Details

This function performs frequentist inference for the copCAR model proposed by Hughes (2015). copCAR is a copula-based areal regression model that employs the proper conditional autoregression (CAR) introduced by Besag, York, and Mollié (1991). Specifically, copCAR uses the CAR copula, a Caussian copula based on the proper CAR.

The spatial dependence parameter $\rho \in [0, 1)$, regression coefficients $\beta = (\beta_1, \dots, \beta_p)' \in R^p$, and, for negative binomial margins, dispersion parameter $\theta > 0$ can be estimated using the continous extension (CE) (Madsen, 2009), distributional transform (DT) (Kazianka and Pilz, 2010), or composite marginal likelihood (CML) (Varin, 2008) approaches.

The CE approach transforms the discrete observations to continous outcomes by convolving them with independent standard uniforms (Denuit and Lambert, 2005). The true likelihood for the discrete outcomes is the expected likelihood for the transformed outcomes. An estimate (sample mean) of the expected likelihood is optimized to estimate the copCAR parameters. The number of standard uniform vectors, m, can be chosen by the user. The default value is 1,000. The CE approach is exact up to Monte Carlo standard error but is computationally intensive (the computational burden grows rapidly with increasing m). The CE approach tends to perform poorly when applied to Bernoulli outcomes, and so that option is not permitted.

The distributional transform stochastically "smoothes" the jumps of a discrete distribution function (Ferguson, 1967). The DT-based approximation (Kazianka and Pilz, 2010) for copCAR performs well for Poisson and negative binomial marginals but, like the CE approach, tends to perform poorly for Bernoulli outcomes.

The CML approach optimizes a composite marginal likelihood formed as the product of pairwise likelihoods of adjacent observations. This approach performs well for Bernoulli, negative binomial, and Poisson outcomes.

In the CE and DT approaches, the CAR variances are approximated. The quality of the approximation is determined by the values of control parameters $\epsilon > 0$ and $\rho^{\max} \in [0, 1)$. The default values are 0.01 and 0.999, respectively.

When `confint = "bootstrap"`, a parametric bootstrap is carried out, and confidence intervals are computed using the quantile method. Monte Carlo standard errors (Flegal et al., 2008) of the quantile estimators are also provided.

When `confint = "asymptotic"`, confidence intervals are computed using an estimate of the asymptotic covariance matrix of the estimator. For the CE method, the inverse of the observed Fisher information matrix is used. For the CML and DT methods, the objective function is misspecified, and so the asymptotic covariance matrix is the inverse of the Godambe information matrix (Godambe, 1960), which has a sandwich form. The "bread" is the inverse of the Fisher information matrix, and the "meat" is the covariance matrix of the score function. The former is estimated using the inverse of the observed Fisher information matrix. The latter is estimated using a parametric bootstrap.

Value

copCAR returns an object of class "copCAR", which is a list containing the following components:

- boot.sample: (if confint = "bootstrap") the bootstrap sample.
- call: the matched call.
- coefficients: a named vector of parameter estimates.
- confint: the value of confint supplied in the function call.
- control: a list containing the names and values of the control parameters.
- convergence: the integer code returned by optim.
- cov.hat: (if confint = "asymptotic") the estimate of the asymptotic covariance matrix of the parameter estimator.
- data: the data argument.
- family: the family object used.
- fitted.values: the fitted mean values, obtained by transforming the linear predictors by the inverse of the link function.
- formula: the formula supplied.
- linear.predictors: the linear fit on link scale.
- message: A character string giving any additional information returned by the optimizer, or NULL.
- method: the method (CE, CML, or DT) used for inference.
- model: if requested (the default), the model frame.
- npar: the number of model parameters.
- offset: the offset vector used.
- residuals: the response residuals, i.e., the outcomes minus the fitted values.
- terms: the terms object used.
- value: the value of the objective function at its minimum.
- x: if requested, the model matrix.
- xlevels: (where relevant) a record of the levels of the factors used in fitting.
- y: if requested (the default), the response vector used.

References

Besag, J., York, J., and Mollié, A. (1991) Bayesian image restoration, with two applications in spatial statistics. *Annals of the Institute of Statistical Mathematics*, **43**(1), 1–20.

Denuit, M. and Lambert, P. (2005) Constraints on concordance measures in bivariate discrete data. *Journal of Multivariate Analysis*, **93**, 40–57.

Ferguson, T. (1967) *Mathematical statistics: a decision theoretic approach*, New York: Academic Press.

Flegal, J., Haran, M., and Jones, G. (2008) Markov Chain Monte Carlo: can we trust the third significant figure? *Statistical Science*, 23(2), 250–260.

Godambe, V. (1960) An optimum property of regular maximum likelihood estimation. *The Annals of Mathmatical Statistics*, **31**(4), 1208–1211.

Hughes, J. (2015) copCAR: A flexible regression model for areal data. *Journal of Computational and Graphical Statistics*, **24**(3), 733–755.

Kazianka, H. and Pilz, J. (2010) Copula-based geostatistical modeling of continuous and discrete data including covariates. *Stochastic Environmental Research and Risk Assessment*, **24**(5), 661–673.

Madsen, L. (2009) Maximum likelihood estimation of regression parameters with spatially dependent discrete data. *Journal of Agricultural, Biological, and Environmental Statistics*, **14**(4), 375–391.

Varin, C. (2008) On composite marginal likelihoods. *Advances in Statistical Analysis*, **92**(1), 1–28.

Examples

```
## Not run:
# Simulate data and fit copCAR to them.

# Use the 20 x 20 square lattice as the underlying graph.

m = 20
A = adjacency.matrix(m)

# Create a design matrix by assigning coordinates to each vertex
# such that the coordinates are restricted to the unit square.

x = rep(0:(m - 1) / (m - 1), times = m)
y = rep(0:(m - 1) / (m - 1), each = m)
X = cbind(x, y)

# Set the dependence parameter, regression coefficients, and dispersion parameter.

rho = 0.995      # strong dependence
beta = c(1, 1)   # the mean surface increases in the direction of (1, 1)
theta = 2        # dispersion parameter

# Simulate negative binomial data from the model.

z = rcopCAR(rho, beta, X, A, family = negbinomial(theta))

# Fit the copCAR model using the continous extension, and compute 95% (default)
# asymptotic confidence intervals. Give theta the initial value of 1. Use m equal to 100.

fit.ce = copCAR(z ~ X - 1, A = A, family = negbinomial(1), method = "CE", confint = "asymptotic",
                control = list(m = 100))
summary(fit.ce)

# Fit the copCAR model using the DT approximation, and compute 90% confidence
# intervals. Bootstrap the intervals, based on a bootstrap sample of size 100.
# Do the bootstrap in parallel, using ten nodes.

fit.dt = copCAR(z ~ X - 1, A = A, family = negbinomial(1), method = "DT", confint = "bootstrap",
```

```
                control = list(bootit = 100, nodes = 10))
summary(fit.dt, alpha = 0.9)

# Fit the copCAR model using the composite marginal likelihood approach.
# Do not compute confidence intervals.

fit.cml = copCAR(z ~ X - 1, A = A, family = negbinomial(1), method = "CML", confint = "none")
summary(fit.cml)

## End(Not run)
```

`negbinomial` *Family function for negative binomial GLMs.*

Description

Provides the information required to apply copCAR with negative binomial marginal distributions.

Usage

```
negbinomial(theta = stop("'theta' must be specified."), link = "log")
```

Arguments

`theta`	the dispersion parameter (must be positive).
`link`	the link function, as a character string, name, or one-element character vector, specifying one of `log`, `sqrt`, or `identity`, or an object of class "`link-glm`"

Value

An object of class "`family`", a list of functions and expressions needed to fit a negative binomial GLM.

`rcopCAR` *Simulate areal data.*

Description

`rcopCAR` simulates areal data from the copCAR model.

Usage

```
rcopCAR(rho, beta, X, A, family)
```

Arguments

rho	the spatial dependence parameter ρ such that $\rho \in [0, 1)$.
beta	the vector of regression coefficients $\beta = (\beta_1, \ldots, \beta_p)'$.
X	the n by p design matrix X.
A	the symmetric binary adjacency matrix for the underlying graph.
family	the marginal distribution of the observations and link function to be used in the model. This can be a character string naming a family function, a family function, or the result of a call to a family function. (See `family` for details of family functions.) Supported familes are `binomial`, `poisson`, and `negbinomial`.

Details

This function simulates data from the copCAR model with the given spatial dependence parameter ρ, regression coefficients β, design matrix X, and adjacency structure A. For negative binomial marginal distributions, a value for the dispersion parameter $\theta > 0$ is also required; this value must be passed to the `negbinomial` family function. For more details on the copCAR model, see `copCAR`.

Value

A vector of length n distributed according to the specified copCAR model.

Examples

```
# Use the 20 x 20 square lattice as the underlying graph.

m = 20
A = adjacency.matrix(m)

# Create a design matrix by assigning coordinates to each vertex
# such that the coordinates are restricted to the unit square.

x = rep(0:(m - 1) / (m - 1), times = m)
y = rep(0:(m - 1) / (m - 1), each = m)
X = cbind(x, y)

# Set the dependence parameter and regression coefficients.

rho = 0.995      # strong dependence
beta = c(1, 1)   # the mean surface increases in the direction of (1, 1)

# Simulate Poisson data from the corresponding copCAR model.

z = rcopCAR(rho, beta, X, A, family = poisson(link = "log"))

# Simulate Bernoulli outcomes.

z = rcopCAR(rho, beta, X, A, family = binomial(link = "logit"))

# Set the dispersion parameter.
```

```
theta = 10

# Simulate negative binomial outcomes.

z = rcopCAR(rho, beta, X, A, family = negbinomial(theta))
```

`residuals.copCAR` *Extract model residuals.*

Description

Extract model residuals.

Usage

```
## S3 method for class 'copCAR'
residuals(object, type = c("deviance", "pearson",
  "response"), ...)
```

Arguments

`object`	an object of class `copCAR`, typically the result of a call to `copCAR`.
`type`	the type of residuals that should be returned. The alternatives are "`deviance`" (default), "`pearson`", and "`response`".
`...`	additional arguments.

Value

A vector of residuals.

See Also

`copCAR`, `residuals.glm`

`summary.copCAR` *Print a summary of a copCAR model fit.*

Description

Print a summary of a copCAR model fit.

Usage

```
## S3 method for class 'copCAR'
summary(object, alpha = 0.05, digits = 4, ...)
```

Arguments

`object`	an object of class `copCAR`, the result of a call to `copCAR`.
`alpha`	the significance level for the confidence intervals. The default is 0.05.
`digits`	the number of significant digits to display. The default is 4.
`...`	additional arguments.

Details

This function displays (1) the call to `copCAR`, (2) the values of the control parameters, (3) a table of estimates, and (when applicable) (4) confidence intervals and (5) Monte Carlo standard errors.

Each row of the table of estimates shows a parameter estimate and (when applicable) the confidence interval for the parameter. If `copCAR` was called with `confint = "bootstrap"`, Monte Carlo standard errors are provided.

References

Flegal, J., Haran, M., and Jones, G. (2008) Markov Chain Monte Carlo: can we trust the third significant figure? *Statistical Science*, **23**(2), 250–260.

See Also

`copCAR`

`vcov.copCAR` *Return the estimated covariance matrix for a* `copCAR` *model object.*

Description

Return the estimated covariance matrix for a `copCAR` model object.

Usage

```
## S3 method for class 'copCAR'
vcov(object, ...)
```

Arguments

`object`	a fitted `copCAR` model object.
`...`	additional arguments.

Details

Unless copCAR was called with confint = "none", this function returns an estimate of the covariance matrix of the CE/CML/DT estimator of the parameters. If confint = "bootstrap", cov is applied to the bootstrap sample to compute the estimate. If confint = "asymptotic", an estimate of the asymptotic covariance matrix is returned; this is an estimate of the inverse Fisher information matrix if method = "CE", or an estimate of the inverse of the Godambe information matrix if method = "CML" or method = "DT". Note that the entries involving the spatial dependence parameter are for $\gamma = \Phi^{-1}(\rho)$ rather than for ρ (Hughes, 2015).

Value

An estimate of the covariance matrix of the CE/CML/DT estimator of the parameters.

References

Hughes, J. (2015) copCAR: A flexible regression model for areal data. *Journal of Computational and Graphical Statistics*, **24**(3), 733–755.

Index