Package 'saeHB.Spatial.Beta'

July 1, 2026

Type Package

Title Small Area Estimation Hierarchical Bayes for Spatial Beta Model

Version 0.1.0

Description Provides several functions and datasets for area-level Small Area Estimation using the Hierarchical Bayesian (HB) method. Model-based estimators are designed for variables of interest that follow a Beta distribution. The package supports spatial structures under the Simultaneous Autoregressive (SAR) and Leroux Conditional Autoregressive (CAR) models, accommodating survey design effect (DEFF) adjustments. The 'rjags' package is employed to obtain parameter estimates via Gibbs Sampling. For references, see Rao and Molina (2015) <doi:10.1002/9781118735855>, Kubacki and Jedrzejczak (2016) <doi:10.59170/stattrans-2016-022>, Leroux et al. (2000) <doi:10.1007/978-1-4612-1284-3_4>, and Chung and Datta (2020) <https://www.census.gov/content/dam/Census/library/working-papers/2020/adrm/RRS2020-07.pdf>.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports rjags, coda, stats, grDevices, graphics, sf, spdep

SystemRequirements JAGS (http://mcmc-jags.sourceforge.net)

Suggests knitr, rmarkdown, testthat (>= 3.0.0), ggplot2

VignetteBuilder knitr

URL https://github.com/BobyIwan/saeHB.Spatial.Beta

BugReports https://github.com/BobyIwan/saeHB.Spatial.Beta/issues

Config/roxygen2/version 8.0.0

Config/testthat/edition 3

NeedsCompilation no

Author Boby Iwan [aut, cre],
Cucu Sumarni [aut]

Maintainer Boby Iwan <bobyiwanboby2122@gmail.com>

Repository CRAN

Date/Publication 2026-07-01 09:30:07 UTC

Contents

`adjacency_mat` *Binary Adjacency Matrix*

Description

A binary adjacency matrix (`B`) generated from a 6x6 regular grid using Queen contiguity. This matrix is mathematically suitable for the Leroux Conditional Autoregressive (CAR) model.

Usage

```
data(adjacency_mat)
```

Format

A `36` x `36` numeric matrix. The elements take a value of `1` if two areas share a common border (are neighbors), and `0` otherwise. All diagonal elements are `0`.

`betadeff_lerouxcar` *Small Area Estimation using Hierarchical Bayesian Method under Spatial Beta-Leroux CAR Model with Design Effect*

Description

This function gives small area estimator under Spatial Leroux CAR Model with Design Effect (DEFF) adjustment. It is implemented to a variable of interest (y) that is assumed to follow a Beta Distribution. The range of data is $0 < y < 1$.

Usage

```
betadeff_lerouxcar(
  formula,
  deff,
  n_i,
  proxmat,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 2,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.v = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

Arguments

formula	Formula that describes the fitted model.
deff	String specifying the name of the design effect variable in the data frame.
n_i	String specifying the name of the sample size variable in the data frame.
proxmat	$N \times N$ spatial binary adjacency matrix with values 0 or 1 representing the neighborhood structure between areas. The diagonal elements must be 0.
data	The data frame.
iter.update	Number of updates performed during Gibbs sampling. Default is 3.
iter.mcmc	Total number of MCMC iterations per chain. Default is 2000.
thin	Thinning rate for MCMC sampling. Must be a positive integer. Default is 1.
burn.in	Number of burn-in iterations discarded from each MCMC chain. Default is 1000.
chains	Number of parallel MCMC chains. Default is 2.
n.adapt	Number of iterations used for the adaptation phase in JAGS. Default is 1000.
coef	Optional vector containing the mean of the prior distribution of the regression model coefficients.
var.coef	Optional vector containing the variances of the prior distribution of the regression model coefficients.
tau.v	Initial value or shape for the random effect precision. Default is 1.
seed	An integer seed for the random number generator to ensure reproducibility. Default is 123.

quiet	Logical; if TRUE, suppresses the JAGS terminal output. Default is FALSE.
plot	Logical; if TRUE, generates MCMC diagnostic trace, autocorrelation, and density plots. Default is TRUE.
keep.fit	Logical; if TRUE, keeps the raw MCMC coda samples object in the output list. Default is FALSE.

Value

This function returns a list with the following objects:

est A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the small area means estimated using the Hierarchical Bayesian method.

randeff A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effects (v).

refvar A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effect variances $(a.var)$.

coefficient A dataframe containing the posterior mean estimates, posterior standard deviations, 95% credible intervals, Rhat convergence diagnostics, and Effective Sample Sizes (ESS) for the regression coefficients (β) and the spatial autoregressive parameter (ρ).

Examples

```
# Load dataset and proximity matrix
data(databeta)
data(adjacency_mat)


# Fit the Spatial Beta-Leroux CAR model with Design Effect
result <- betadeff_lerouxcar(
  formula = y ~ x1 + x2,
  deff = "deff",
  n_i = "n_i",
  proxmat = adjacency_mat,
  data = databeta
)

# View the estimation results
# 1. Small Area Estimates
result$est
# 2. Estimated area-specific random effects
result$randeff
# 3. Estimated variance of the random effects
result$refvar
# 4. Estimated regression coefficients and spatial parameter
result$coefficient
```

`betadeff_nonspatial` *Small Area Estimation using Hierarchical Bayesian Method under Non-Spatial Beta Model with Design Effect*

Description

This function gives small area estimator under Non-Spatial Model with Design Effect (DEFF) adjustment. It is implemented to a variable of interest (y) that is assumed to follow a Beta Distribution. The range of data is $0 < y < 1$.

Usage

```
betadeff_nonspatial(
  formula,
  deff,
  n_i,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 2,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.v = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

Arguments

`formula`	Formula that describes the fitted model.
`deff`	String specifying the name of the design effect variable in the data frame.
`n_i`	String specifying the name of the sample size variable in the data frame.
`data`	The data frame.
`iter.update`	Number of updates performed during Gibbs sampling. Default is `3`.
`iter.mcmc`	Total number of MCMC iterations per chain. Default is `2000`.
`thin`	Thinning rate for MCMC sampling. Must be a positive integer. Default is `1`.
`burn.in`	Number of burn-in iterations discarded from each MCMC chain. Default is `1000`.
`chains`	Number of parallel MCMC chains. Default is `2`.

n.adapt	Number of iterations used for the adaptation phase in JAGS. Default is 1000.
coef	Optional vector containing the mean of the prior distribution of the regression model coefficients.
var.coef	Optional vector containing the variances of the prior distribution of the regression model coefficients.
tau.v	Initial value or shape for the random effect precision. Default is 1.
seed	An integer seed for the random number generator to ensure reproducibility. Default is 123.
quiet	Logical; if TRUE, suppresses the JAGS terminal output. Default is FALSE.
plot	Logical; if TRUE, generates MCMC diagnostic trace, autocorrelation, and density plots. Default is TRUE.
keep.fit	Logical; if TRUE, keeps the raw MCMC coda samples object in the output list. Default is FALSE.

Value

This function returns a list with the following objects:

est A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the small area means estimated using the Hierarchical Bayesian method.

randeff A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effects (v).

refvar A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the global random effect variance (σ_v^2).

coefficient A dataframe containing the posterior mean estimates, posterior standard deviations, 95% credible intervals, Rhat convergence diagnostics, and effective sample sizes (ESS) for the regression coefficients (β).

Examples

```
# Load dataset
data(databeta)


# Fit the Non-Spatial Beta model with Design Effect
result <- betadeff_nonspatial(
  formula = y ~ x1 + x2,
  deff = "deff",
  n_i = "n_i",
  data = databeta
)

# View the estimation results
# 1. Small Area Estimates
result$est
# 2. Estimated area-specific random effects
result$randeff
# 3. Estimated global variance of the random effects
```

```
result$refvar
# 4. Estimated regression coefficients
result$coefficient
```

`betadeff_sar` *Small Area Estimation using Hierarchical Bayesian Method under Spatial Beta SAR Model with Design Effect*

Description

This function gives small area estimator under Spatial SAR Model with Design Effect (DEFF) adjustment. It is implemented to a variable of interest (y) that is assumed to follow a Beta Distribution. The range of data is $0 < y < 1$.

Usage

```
betadeff_sar(
  formula,
  deff,
  n_i,
  proxmat,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 2,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.u = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

Arguments

`formula`	Formula that describes the fitted model.
`deff`	String specifying the name of the design effect variable in the data frame.
`n_i`	String specifying the name of the sample size variable in the data frame.
`proxmat`	$N \times N$ row-standardized proximity matrix with values in the interval `[0,1]` representing the spatial proximity between areas. The rows sum to `1`.

data	The data frame.
iter.update	Number of updates performed during Gibbs sampling. Default is 3.
iter.mcmc	Total number of MCMC iterations per chain. Default is 2000.
thin	Thinning rate for MCMC sampling. Must be a positive integer. Default is 1.
burn.in	Number of burn-in iterations discarded from each MCMC chain. Default is 1000.
chains	Number of parallel MCMC chains. Default is 2.
n.adapt	Number of iterations used for the adaptation phase in JAGS. Default is 1000.
coef	Optional vector containing the mean of the prior distribution of the regression model coefficients.
var.coef	Optional vector containing the variances of the prior distribution of the regression model coefficients.
tau.u	Initial value or shape for the random effect precision. Default is 1.
seed	An integer seed for the random number generator to ensure reproducibility. Default is 123.
quiet	Logical; if TRUE, suppresses the JAGS terminal output. Default is FALSE.
plot	Logical; if TRUE, generates MCMC diagnostic trace, autocorrelation, and density plots. Default is TRUE.
keep.fit	Logical; if TRUE, keeps the raw MCMC coda samples object in the output list. Default is FALSE.

Value

This function returns a list with the following objects:

est A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the small area means estimated using the Hierarchical Bayesian method.

randeff A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effects (v).

refvar A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effect variances $(a.var)$.

coefficient A dataframe containing the posterior mean estimates, posterior standard deviations, 95% credible intervals, Rhat convergence diagnostics, and Effective Sample Sizes (ESS) for the regression coefficients (β) and the spatial autoregressive parameter (ρ).

Examples

```
# Load dataset and proximity matrix
data(databeta)
data(weight_mat)


# Fit the Spatial Beta-SAR model with Design Effect
result <- betadeff_sar(
  formula = y ~ x1 + x2,
```

```
  deff = "deff",
  n_i = "n_i",
  proxmat = weight_mat,
  data = databeta
)

# View the estimation results
# 1. Small Area Estimates
result$est
# 2. Estimated area-specific random effects
result$randeff
# 3. Estimated variance of the random effects
result$refvar
# 4. Estimated regression coefficients and spatial parameter
result$coefficient
```

---

beta_lerouxcar *Small Area Estimation using Hierarchical Bayesian Method under Spatial Beta-Leroux CAR Model*

Description

This function gives small area estimator under Spatial Leroux CAR Model. It is implemented to a variable of interest (y) that is assumed to follow a Beta Distribution. The range of data is $0 < y < 1$.

Usage

```
beta_lerouxcar(
  formula,
  proxmat,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 2,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.v = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

Arguments

formula	Formula that describes the fitted model.
proxmat	$N \times N$ spatial binary adjacency matrix with values 0 or 1 representing the neighborhood structure between areas. The diagonal elements must be 0.
data	The data frame.
iter.update	Number of updates performed during Gibbs sampling. Default is 3.
iter.mcmc	Total number of MCMC iterations per chain. Default is 2000.
thin	Thinning rate for MCMC sampling. Must be a positive integer. Default is 1.
burn.in	Number of burn-in iterations discarded from each MCMC chain. Default is 1000.
chains	Number of parallel MCMC chains. Default is 2.
n.adapt	Number of iterations used for the adaptation phase in JAGS. Default is 1000.
coef	Optional vector containing the mean of the prior distribution of the regression model coefficients.
var.coef	Optional vector containing the variances of the prior distribution of the regression model coefficients.
tau.v	Initial value or shape for the random effect precision. Default is 1.
seed	An integer seed for the random number generator to ensure reproducibility. Default is 123.
quiet	Logical; if TRUE, suppresses the JAGS terminal output. Default is FALSE.
plot	Logical; if TRUE, generates MCMC diagnostic trace, autocorrelation, and density plots. Default is TRUE.
keep.fit	Logical; if TRUE, keeps the raw MCMC coda samples object in the output list. Default is FALSE.

Value

This function returns a list with the following objects:

est A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the small area means estimated using the Hierarchical Bayesian method.

randeff A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effects (v).

refvar A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effect variances $(a.var)$.

coefficient A dataframe containing the posterior mean estimates, posterior standard deviations, 95% credible intervals, Rhat convergence diagnostics, and Effective Sample Sizes (ESS) for the regression coefficients (β), the spatial autoregressive parameter (ρ), and the global precision parameter (ϕ).

Examples

```
# Load dataset and proximity matrix
data(databeta)
data(adjacency_mat)


# Fit the Spatial Beta-Leroux CAR model
result <- beta_lerouxcar(
  formula = y ~ x1 + x2,
  proxmat = adjacency_mat,
  data = databeta
)

# View the estimation results
# 1. Small Area Estimates
result$est
# 2. Estimated area-specific random effects
result$randeff
# 3. Estimated variance of the random effects
result$refvar
# 4. Estimated regression coefficients, spatial, and precision parameters
result$coefficient
```

---

beta_nonspatial *Small Area Estimation using Hierarchical Bayesian Method under Non-Spatial Beta Model*

Description

This function gives small area estimator under Non-Spatial Model. It is implemented to a variable of interest (y) that is assumed to follow a Beta Distribution. The range of data is $0 < y < 1$.

Usage

```
beta_nonspatial(
  formula,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 2,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.v = 1,
```

```
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

Arguments

formula	Formula that describes the fitted model.
data	The data frame.
iter.update	Number of updates performed during Gibbs sampling. Default is 3.
iter.mcmc	Total number of MCMC iterations per chain. Default is 2000.
thin	Thinning rate for MCMC sampling. Must be a positive integer. Default is 1.
burn.in	Number of burn-in iterations discarded from each MCMC chain. Default is 1000.
chains	Number of parallel MCMC chains. Default is 2.
n.adapt	Number of iterations used for the adaptation phase in JAGS. Default is 1000.
coef	Optional vector containing the mean of the prior distribution of the regression model coefficients.
var.coef	Optional vector containing the variances of the prior distribution of the regression model coefficients.
tau.v	Initial value or shape for the random effect precision. Default is 1.
seed	An integer seed for the random number generator to ensure reproducibility. Default is 123.
quiet	Logical; if TRUE, suppresses the JAGS terminal output. Default is FALSE.
plot	Logical; if TRUE, generates MCMC diagnostic trace, autocorrelation, and density plots. Default is TRUE.
keep.fit	Logical; if TRUE, keeps the raw MCMC coda samples object in the output list. Default is FALSE.

Value

This function returns a list with the following objects:

est A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the small area means estimated using the Hierarchical Bayesian method.

randeff A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effects (v).

refvar A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the global random effect variance (σ_v^2).

coefficient A dataframe containing the posterior mean estimates, posterior standard deviations, 95% credible intervals, Rhat convergence diagnostics, and effective sample sizes (ESS) for the regression coefficients (β) and the global precision parameter (ϕ).

Examples

```
# Load dataset
data(databeta)


# Fit the Non-Spatial Beta model
result <- beta_nonspatial(
  formula = y ~ x1 + x2,
  data = databeta
)

# View the estimation results
# 1. Small Area Estimates
result$est
# 2. Estimated area-specific random effects
result$randeff
# 3. Estimated global variance of the random effects
result$refvar
# 4. Estimated regression coefficients and precision parameter
result$coefficient
```

---

beta_sar *Small Area Estimation using Hierarchical Bayesian Method under Spatial Beta SAR Model*

Description

This function gives small area estimator under Spatial SAR Model. It is implemented to a variable of interest (y) that is assumed to follow a Beta Distribution. The range of data is $0 < y < 1$.

Usage

```
beta_sar(
  formula,
  proxmat,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 2,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.u = 1,
  seed = 123,
```

```
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

Arguments

formula	Formula that describes the fitted model.
proxmat	$N \times N$ row-standardized proximity matrix with values in the interval [0,1] representing the spatial proximity between areas. The rows sum to 1.
data	The data frame.
iter.update	Number of updates performed during Gibbs sampling. Default is 3.
iter.mcmc	Total number of MCMC iterations per chain. Default is 2000.
thin	Thinning rate for MCMC sampling. Must be a positive integer. Default is 1.
burn.in	Number of burn-in iterations discarded from each MCMC chain. Default is 1000.
chains	Number of parallel MCMC chains. Default is 2.
n.adapt	Number of iterations used for the adaptation phase in JAGS. Default is 1000.
coef	Optional vector containing the mean of the prior distribution of the regression model coefficients.
var.coef	Optional vector containing the variances of the prior distribution of the regression model coefficients.
tau.u	Initial value or shape for the random effect precision. Default is 1.
seed	An integer seed for the random number generator to ensure reproducibility. Default is 123.
quiet	Logical; if TRUE, suppresses the JAGS terminal output. Default is FALSE.
plot	Logical; if TRUE, generates MCMC diagnostic trace, autocorrelation, and density plots. Default is TRUE.
keep.fit	Logical; if TRUE, keeps the raw MCMC coda samples object in the output list. Default is FALSE.

Value

This function returns a list with the following objects:

est A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the small area means estimated using the Hierarchical Bayesian method.

randeff A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effects (v).

refvar A dataframe containing the posterior mean estimates, posterior standard deviations, and 95% credible intervals of the area-specific random effect variances $(a.var)$.

coefficient A dataframe containing the posterior mean estimates, posterior standard deviations, 95% credible intervals, Rhat convergence diagnostics, and Effective Sample Sizes (ESS) for the regression coefficients (β), the spatial autoregressive parameter (ρ), and the global precision parameter (ϕ).

Examples

```
# Load dataset and proximity matrix
data(databeta)
data(weight_mat)


# Fit the Spatial Beta-SAR model
result <- beta_sar(
  formula = y ~ x1 + x2,
  proxmat = weight_mat,
  data = databeta
)

# View the estimation results
# 1. Small Area Estimates
result$est
# 2. Estimated area-specific random effects
result$randeff
# 3. Estimated variance of the random effects
result$refvar
# 4. Estimated regression coefficients, spatial, and precision parameters
result$coefficient
```

---

build_w *Build Spatial Weights Matrix*

Description

This function constructs spatial weights matrices (W) for spatial modeling. It supports various methods including Contiguity, Distance-based, and Kernel-based weights, and provides a robust fallback mechanism to automatically connect isolated areas (islands).

Usage

```
build_w(
  data,
  coords = NULL,
  method = c("contiguity", "distance", "kernel"),
  contiguity = c("queen", "rook", "bishop"),
  distance = c("knn", "inverse_distance", "exponential"),
  k = 2,
  dmax = NULL,
  power = 1,
  alpha = 1,
  epsilon = 1e-12,
  kernel = c("uniform", "gaussian", "triangular", "epanechnikov", "quartic"),
```

```
  bandwidth = NULL,
  lonlat = TRUE,
  style = "W",
  zero.policy = TRUE,
  fallback = c("knn", "distance", "none"),
  fallback_k = 2,
  fallback_dmax = NULL,
  output = c("all", "matrix", "listw", "nb")
)
```

Arguments

data	An sf object (polygons/points) or a standard data frame. If a standard data frame is provided, coords must be specified.
coords	An $N \times 2$ matrix of coordinates. Required if data is not an sf object.
method	A string indicating the spatial weight construction method. Options are "contiguity", "distance", or "kernel".
contiguity	A string indicating the contiguity type. Options are "queen", "rook", or "bishop".
distance	A string indicating the distance-based type. Options are "knn", "inverse_distance", or "exponential".
k	An integer specifying the number of nearest neighbors for KNN methods. Default is 2.
dmax	A numeric specifying the maximum distance threshold for distance-based neighbors. The unit depends on lonlat (kilometers if TRUE, native coordinate units/meters if FALSE).
power	A numeric specifying the decay power for inverse distance weights. Default is 1.
alpha	A numeric specifying the decay parameter for exponential distance weights. Default is 1.
epsilon	A small numeric value to prevent division by zero in inverse distance calculation. Default is 1e-12.
kernel	A string indicating the type of spatial kernel. Options are "uniform", "gaussian", "triangular", "epanechnikov", or "quartic".
bandwidth	A numeric specifying the bandwidth (h) for kernel weights. Required if method = "kernel". The unit depends on lonlat (kilometers if TRUE, native coordinate units/meters if FALSE).
lonlat	Logical; if TRUE, coordinates are treated as Longitude/Latitude, spherical distances are calculated, and limits are in **kilometers (km)**. If FALSE, coordinates are assumed to be planar (e.g., UTM), Euclidean distances are calculated, and limits are in the native unit of the coordinates (usually **meters**). Default is TRUE.
style	A character string specifying the spatial weights coding scheme ("W" for row-standardized or "B" for binary). Default is "W".
zero.policy	Logical; if TRUE, areas with no neighbors are allowed to have zero-weight rows. Default is TRUE.

fallback	A string indicating the fallback method for isolated areas (without neighbors) when using contiguity. Options are "knn", "distance", or "none". Default is "knn".
fallback_k	An integer specifying the number of neighbors for the fallback method. Default is 2.
fallback_dmax	A numeric specifying the maximum distance for the fallback method.
output	A string specifying the format of the output. Options are "all" (returns a comprehensive list), "matrix", "listw", or "nb". Default is "all".

Details

The function supports the following spatial weight construction methods:

- **Contiguity:** Queen, Rook, and Bishop.
- **Distance-based:** K-Nearest Neighbors (KNN), Inverse Distance, and Exponential.
- **Kernel-based:** Uniform, Gaussian, Triangular, Epanechnikov, and Quartic.

For distance and kernel methods, if lonlat = TRUE, spherical (great-circle) distances are calculated. For the kernel method specifically, distances are internally converted to kilometers.

Value

Depending on the output argument, this function returns:

- "matrix": An $N \times N$ spatial weights matrix.
- "listw": A listw object compatible with spdep functions.
- "nb": An nb (neighborhood) object.
- "all": A list containing W (matrix), listw, nb, info (method details), and diag (diagnostic metrics for isolates and fallback).

Examples

```
# Generate random Longitude and Latitude coordinates for 10 areas
set.seed(123)
lon <- runif(10, min = 100, max = 140)
lat <- runif(10, min = -10, max = 10)
coords <- cbind(lon, lat)

# 1. Build KNN distance-based weights (k = 2) using spherical distance
W_knn <- build_w(
  data = NULL,
  coords = coords,
  method = "distance",
  distance = "knn",
  k = 2,
  lonlat = TRUE,
  output = "matrix"
)
```

```
# View the first few rows of the matrix
head(W_knn)

# 2. Build Gaussian Kernel weights using 500 km bandwidth
W_kernel <- build_w(
  data = NULL,
  coords = coords,
  method = "kernel",
  kernel = "gaussian",
  bandwidth = 500,
  lonlat = TRUE,
  output = "matrix"
)

# View the first few rows of the matrix
head(W_kernel)
```

`databeta` *Synthetic Data for Small Area Estimation using Spatial Beta Model*

Description

A synthetic dataset generated for testing and tutorial purposes of the `saeHB.Spatial.Beta` package. The data is generated under a Spatial Simultaneous Autoregressive (SAR) process with a Beta distribution, accommodating survey design effects (DEFF).

This data is generated by these following steps:

1. Generate auxiliary variables $x1 \sim N(0,1)$ and $x2 \sim N(0,1)$.
2. Generate sample sizes $n_i \sim U(10,50)$ and survey design effects $deff_i \sim U(1,2.5)$. Calculate the precision parameter for each area: $\phi_i = (n_i/deff_i) - 1$.
3. Generate spatial random effects under the SAR model. First, generate independent normal errors $u \sim N(0,1)$. Then, calculate the spatial random effect $v = (I - \rho W)^{-1}u$, where I is an identity matrix, W is the row-standardized proximity matrix (`weight_mat`), and the spatial autoregressive parameter ρ is set to 0.70.
4. Calculate the true mean proportions $\mu = \text{logit}^{-1}(X\beta + v)$, where the regression coefficients are set as $\beta_0 = \beta_1 = \beta_2 = 1$.
5. Generate the response variable $y \sim \text{Beta}(\mu\phi, (1-\mu)\phi)$. Values are strictly bounded between 0 and 1.
6. Area ID `domain`, response variable `y`, auxiliary variables `x1`, `x2`, sample size `n_i`, and design effect `deff` are combined into a data frame called `databeta`.

Usage

```
data(databeta)
```

Format

A data frame with 36 rows and 6 columns:

domain Area ID/name

y Direct estimates of the proportion/variable of interest (0 < y < 1)

x1 Auxiliary variable 1 (Normal distribution)

x2 Auxiliary variable 2 (Normal distribution)

n_i Sample size for each area

deff Survey design effect for each area

---

databeta_na *Synthetic Data with Missing Values for Small Area Estimation*

Description

A synthetic dataset identical to `databeta`, but contains 5 missing values (NA) in the variable of interest (`y`) to demonstrate the prediction capability of the models for non-sampled areas.

Usage

```
data(databeta_na)
```

Format

A data frame with 36 rows and 6 columns:

domain Area ID/name

y Direct estimates of the proportion/variable of interest (0 < y < 1). Contains `NA` values.

x1 Auxiliary variable 1

x2 Auxiliary variable 2

n_i Sample size for each area

deff Survey design effect for each area

`moran_test` *Moran's I Test for Spatial Autocorrelation*

Description

This function provides a convenient wrapper to perform Moran's I test for spatial autocorrelation on a numeric vector. It seamlessly handles missing values (NA) by subsetting both the numeric vector and the spatial weights list simultaneously.

Usage

```
moran_test(
  x,
  listw,
  alternative = c("greater", "less", "two.sided"),
  mc = FALSE,
  nsim = 999,
  zero.policy = TRUE,
  na.rm = TRUE
)
```

Arguments

`x`	A numeric vector of the variable of interest (e.g., residuals, random effects, or raw data).
`listw`	A `listw` object containing spatial weights created by `build_w` or `spdep`.
`alternative`	A character string specifying the alternative hypothesis. Must be one of `"greater"` (default), `"less"`, or `"two.sided"`.
`mc`	Logical; if `TRUE`, performs Moran's I test using Monte Carlo permutations. Default is `FALSE` (analytical approach).
`nsim`	An integer specifying the number of permutations if `mc = TRUE`. Default is `999`.
`zero.policy`	Logical; if `TRUE`, allows areas with no neighbors (isolates) to be included in the calculation. Default is `TRUE`.
`na.rm`	Logical; if `TRUE`, missing values in `x` are removed, and the corresponding rows/columns in the spatial weights are automatically subsetted. Default is `TRUE`.

Details

This function supports two approaches to testing the significance of Moran's I:

1. Analytical Approach (Randomization - Default)
When `mc = FALSE`, the function uses the analytical approach (specifically, the assumption of randomization). It computes the theoretical expectation and variance of Moran's I under the null hypothesis of no spatial autocorrelation. This method assumes that the observed values could have occurred in any spatial location with equal probability.

When to use: Use this approach when your dataset is relatively large and follows standard statistical assumptions. It is computationally fast and provides reliable asymptotic p-values for large N.

2. Monte Carlo Permutation Approach (`mc = TRUE`)
When `mc = TRUE`, the function calculates the p-value empirically. It randomly permutes (shuffles) the observed values `x` across the spatial units `nsim` times. For each permutation, it calculates a pseudo-Moran's I. The final p-value is the proportion of simulated Moran's I values that are as extreme as or more extreme than the observed Moran's I.
When to use: Use this approach when your dataset has a relatively small number of areas or when you want to avoid relying on asymptotic theory. Because it computes the p-value empirically without assuming a specific theoretical distribution for the Moran's I statistic, the Monte Carlo approach is highly robust and is widely recommended for evaluating MCMC outputs.

Value

A list with class `htest` containing the following components:

- `statistic`: The value of the standard deviate of Moran's I.
- `p.value`: The p-value of the test.
- `estimate`: The value of the observed Moran's I, its expectation, and variance.
- `method`: A character string indicating the type of test performed.
- `data.name`: A character string giving the name(s) of the data.

Examples

```
# Load datasets
data(databeta)
data(weight_mat)

# Convert the spatial weights matrix to a 'listw' object
W_listw <- spdep::mat2listw(weight_mat, style = "W", zero.policy = TRUE)

# Perform Moran's I test (Analytical approach)
moran_test(x = databeta$y, listw = W_listw)

# Perform Moran's I test (Monte Carlo permutation approach)
moran_test(x = databeta$y, listw = W_listw, mc = TRUE, nsim = 99)

# Handling Missing Values automatically (na.rm = TRUE is default)
y_with_na <- databeta$y
y_with_na[c(2, 5)] <- NA
moran_test(x = y_with_na, listw = W_listw, na.rm = TRUE)
```

`weight_mat` *Row-Standardized Spatial Weight Matrix*

Description

A row-standardized proximity matrix (`W`) generated from a 6x6 regular grid using Queen contiguity. This matrix is mathematically suitable for the Spatial Simultaneous Autoregressive (SAR) model and Moran's I test.

Usage

```
data(weight_mat)
```

Format

A `36` x `36` numeric matrix. The values are numbers in the interval `[0,1]` representing the proximity of the row and column areas. The sum of the values in each row is exactly 1.

Index