# Package 'saeHB.Spatial.Beta'

September 20, 2026

**Type** Package

**Title** Small Area Estimation Hierarchical Bayes for Spatial Beta Model

**Version** 0.2.0

**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 (proportions bounded between 0 and 1). The package supports both non-spatial and spatial models based on Simultaneous Autoregressive (SAR) and Leroux Conditional Autoregressive (CAR) structures for area-level random effects, with optional survey design effect (DEFF) adjustments for sampling variances. In addition, it provides utility functions for constructing spatial weights matrices and performing spatial autocorrelation diagnostics. The 'runjags' package is used to obtain posterior estimates via Markov Chain Monte Carlo (MCMC) with parallel computing capabilities. For references, see Rao and Molina (2015) <doi:10.1002/9781118735855>, Liu et al. (2014) <https://www150.statcan.gc.ca/n1/pub/12-001-x/2014001/article/14030-eng.pdf>, Kubacki and Jedrzejczak (2016) <doi:10.59170/stattrans-2016-022>, Leroux et al. (2000) <doi:10.1007/978-1-4612-1284-3_4>, Chung and Datta (2020) <https://www.census.gov/content/dam/Census/library/working-papers/2020/adrm/RRS2020-07.pdf>, Figueroa-Zúñiga et al. (2013) <doi:10.1016/j.csda.2012.12.002>, Denwood (2016) <doi:10.18637/jss.v071.i09>, Anselin (1988) <doi:10.1007/978-94-015-7799-1>, and Anselin and Morrison (2019) <https://spatialanalysis.github.io/lab_tutorials/Spatial_Weights_as_Distance_Functions.html>.

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 4.1.0)

**Imports** runjags, 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** 2026-09-20 18:30:25 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 Hierarchical Bayesian (HB) Spatial Beta-Leroux 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 estimates small area proportions using a Hierarchical Bayes (HB) method under a Spatial Leroux CAR Model with a Beta distribution, incorporating survey design effect (DEFF) adjustments. It is designed for a variable of interest ($y$) that represents proportions, strictly bounded between 0 and 1 ($0 < y < 1$).

## Usage

```
betadeff_lerouxcar(
  formula,
  deff,
  n_i,
  proxmat,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 4,
  n.sims = chains,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.v = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

## Arguments

| | |
|---|---|
| `formula` | An object of class `formula` that describes the fitted model. |
| `deff` | A character string specifying the name of the design effect (DEFF) variable in the data frame. |
| `n_i` | A character string specifying the name of the sample size variable in the data frame. |
| `proxmat` | An $N \times N$ spatial binary adjacency matrix with values `0` or `1` representing the neighborhood structure between areas. The diagonal elements must be `0` and the matrix must be symmetric. |
| `data` | The data frame containing the variables named in `formula`, `deff`, and `n_i`. |

| | |
|---|---|
| iter.update | Number of iterative model updates used to refine the prior distributions for the regression coefficients and random effect precision. 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 4. |
| n.sims | The maximum number of parallel JAGS simulations allowed to run concurrently. Default is equal to chains. |
| n.adapt | Number of iterations used for the adaptation phase in JAGS. Default is 1000. |
| coef | Optional vector specifying the prior means of the regression coefficients, including the intercept. |
| var.coef | Optional vector containing the variances of the prior distribution of the regression model coefficients. |
| tau.v | Initial value for the random effect precision $(\tau_v)$. 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 $(\beta)$ and the spatial autoregressive parameter $(\rho)$.

**fit** The raw MCMC coda object (included only if keep.fit = TRUE).

## 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 estimates small area proportions using a Hierarchical Bayes (HB) method under a Non-Spatial Model with a Beta distribution and Independent and Identically Distributed (IID) random effects, incorporating DEFF adjustments. It is designed for a variable of interest ($y$) that represents proportions, strictly bounded between 0 and 1 ($0 < y < 1$).

## Usage

```
betadeff_nonspatial(
  formula,
  deff,
  n_i,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 4,
  n.sims = chains,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
```

```
  tau.v = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

## Arguments

| | |
|---|---|
| formula | An object of class formula that describes the fitted model. |
| deff | A character string specifying the name of the design effect (DEFF) variable in the data frame. |
| n_i | A character string specifying the name of the sample size variable in the data frame. |
| data | The data frame containing the variables named in formula, deff, and n_i. |
| iter.update | Number of iterative model updates used to refine the prior distributions for the regression coefficients and random effect precision. 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 4. |
| n.sims | The maximum number of parallel JAGS simulations allowed to run concurrently. Default is equal to chains. |
| n.adapt | Number of iterations used for the adaptation phase in JAGS. Default is 1000. |
| coef | Optional vector specifying the prior means of the regression coefficients, including the intercept. |
| var.coef | Optional vector containing the variances of the prior distribution of the regression model coefficients. |
| tau.v | Initial value 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 data frame containing the posterior mean estimate, posterior standard deviation, and 95% credible interval of the global area-level random effect variance $\sigma_v^2$.

**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 $(\beta)$.

**fit** The raw MCMC coda object (included only if `keep.fit = TRUE`).

## 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 estimates small area proportions using a Hierarchical Bayes (HB) method under a Spatial Simultaneous Autoregressive (SAR) Model with a Beta distribution, incorporating survey design effect (DEFF) adjustments. It is designed for a variable of interest $(y)$ that represents proportions, strictly bounded between 0 and 1 $(0 < y < 1)$.

## Usage

```
betadeff_sar(
  formula,
  deff,
  n_i,
  proxmat,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 4,
  n.sims = chains,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.u = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

## Arguments

| | |
|---|---|
| formula | An object of class formula that describes the fitted model. |
| deff | A character string specifying the name of the design effect (DEFF) variable in the data frame. |
| n_i | A character string specifying the name of the sample size variable in the data frame. |
| proxmat | An $N \times N$ row-standardized spatial weights matrix (style = "W") representing the spatial proximity between areas. The rows must sum to 1 and diagonal elements must be 0. |
| data | The data frame containing the variables named in formula, deff, and n_i. |
| iter.update | Number of iterative model updates used to refine the prior distributions for the regression coefficients and random effect precision. 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 4. |
| n.sims | The maximum number of parallel JAGS simulations allowed to run concurrently. Default is equal to chains. |
| n.adapt | Number of iterations used for the adaptation phase in JAGS. Default is 1000. |

| | |
|---|---|
| coef | Optional vector specifying the prior means of the regression coefficients, including the intercept. |
| var.coef | Optional vector containing the variances of the prior distribution of the regression model coefficients. |
| tau.u | Initial value for the random effect precision $(\tau_u)$. 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 $(\beta)$ and the spatial autoregressive parameter $(\rho)$.

**fit** The raw MCMC coda object (included only if keep.fit = TRUE).

## 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 estimates small area proportions using a Hierarchical Bayes (HB) method under a Spatial Leroux CAR Model with a Beta distribution without DEFF adjustments, by estimating the unknown precision parameter. It is designed for a variable of interest ($y$) that represents proportions, strictly bounded between 0 and 1 ($0 < y < 1$).

## Usage

```
beta_lerouxcar(
  formula,
  proxmat,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 4,
  n.sims = chains,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.v = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

## Arguments

| | |
|---|---|
| `formula` | An object of class `formula` that describes the fitted model. |
| `proxmat` | An $N \times N$ spatial binary adjacency matrix with values `0` or `1` representing the neighborhood structure between areas. The diagonal elements must be `0` and the matrix must be symmetric. |

| | |
|---|---|
| data | The data frame containing the variables named in formula. |
| iter.update | Number of iterative model updates used to refine the prior distributions for the regression coefficients and random effect precision. 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 4. |
| n.sims | The maximum number of parallel JAGS simulations allowed to run concurrently. Default is equal to chains. |
| n.adapt | Number of iterations used for the adaptation phase in JAGS. Default is 1000. |
| coef | Optional vector specifying the prior means of the regression coefficients, including the intercept. |
| var.coef | Optional vector containing the variances of the prior distribution of the regression model coefficients. |
| tau.v | Initial value for the random effect precision $(\tau_v)$. 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 $(\beta)$, the spatial autoregressive parameter $(\rho)$, and the global precision parameter $(\phi)$.

**fit** The raw MCMC coda object (included only if keep.fit = TRUE).

## 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 estimates small area proportions using a Hierarchical Bayes (HB) method under a Non-Spatial Model with a Beta distribution and Independent and Identically Distributed (IID) random effects without DEFF adjustments, by estimating the unknown precision parameter. It is designed for a variable of interest ($y$) that represents proportions, strictly bounded between 0 and 1 ($0 < y < 1$).

## Usage

```
beta_nonspatial(
  formula,
  data,
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 4,
  n.sims = chains,
```

```
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.v = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

## Arguments

| | |
|---|---|
| formula | An object of class formula that describes the fitted model. |
| data | The data frame containing the variables named in formula. |
| iter.update | Number of iterative model updates used to refine the prior distributions for the regression coefficients and random effect precision. 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 4. |
| n.sims | The maximum number of parallel JAGS simulations allowed to run concurrently. Default is equal to chains. |
| n.adapt | Number of iterations used for the adaptation phase in JAGS. Default is 1000. |
| coef | Optional vector specifying the prior means of the regression coefficients, including the intercept. |
| var.coef | Optional vector containing the variances of the prior distribution of the regression model coefficients. |
| tau.v | Initial value 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 data frame containing the posterior mean estimate, posterior standard deviation, and 95% credible interval of the global area-level random effect variance $\sigma_v^2$.

**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 $(\beta)$ and the global precision parameter $(\phi)$.

**fit** The raw MCMC coda object (included only if `keep.fit = TRUE`).

## 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 estimates small area proportions using a Hierarchical Bayes (HB) method under a Spatial Simultaneous Autoregressive (SAR) Model with a Beta distribution without DEFF adjustments, by estimating the unknown precision parameter. It is designed for a variable of interest $(y)$ that represents proportions, strictly bounded between 0 and 1 $(0 < y < 1)$.

## Usage

```
beta_sar(
  formula,
  proxmat,
  data,
```

```
  iter.update = 3,
  iter.mcmc = 2000,
  thin = 1,
  burn.in = 1000,
  chains = 4,
  n.sims = chains,
  n.adapt = 1000,
  coef = NULL,
  var.coef = NULL,
  tau.u = 1,
  seed = 123,
  quiet = FALSE,
  plot = TRUE,
  keep.fit = FALSE
)
```

## Arguments

| | |
|---|---|
| formula | An object of class formula that describes the fitted model. |
| proxmat | An $N \times N$ row-standardized spatial weights matrix (style = "W") representing the spatial proximity between areas. The rows must sum to 1 and diagonal elements must be 0. |
| data | The data frame containing the variables named in formula. |
| iter.update | Number of iterative model updates used to refine the prior distributions for the regression coefficients and random effect precision. 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 4. |
| n.sims | The maximum number of parallel JAGS simulations allowed to run concurrently. Default is equal to chains. |
| n.adapt | Number of iterations used for the adaptation phase in JAGS. Default is 1000. |
| coef | Optional vector specifying the prior means of the regression coefficients, including the intercept. |
| var.coef | Optional vector containing the variances of the prior distribution of the regression model coefficients. |
| tau.u | Initial value for the random effect precision ($\tau_u$). 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 $(\beta)$, the spatial autoregressive parameter $(\rho)$, and the global precision parameter $(\phi)$.

**fit** The raw MCMC coda object (included only if `keep.fit = TRUE`).

## 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 (row-standardized $W$ or binary adjacency $B$) for Hierarchical Bayesian (HB) Beta Spatial modeling under Spatial Autoregressive (SAR) and Leroux Conditional Autoregressive (CAR) models. 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"),
  fallback = c("knn", "distance", "none"),
  fallback_k = 2,
  fallback_dmax = NULL,
  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 = c("W", "B"),
  zero.policy = TRUE,
  output = c("all", "matrix", "listw", "nb")
)
```

## Arguments

| | |
|---|---|
| `data` | An `sf` object containing spatial polygons/points, or a standard data frame. |
| `coords` | An $N \times 2$ matrix of spatial coordinates. Required only 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"`. Required if `method = "contiguity"`. |
| `fallback` | A string indicating the fallback method for isolated areas 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. Used if `fallback = "knn"`. Default is `2`. |
| `fallback_dmax` | A numeric specifying the maximum distance for the fallback method. Required if `fallback = "distance"`. |

| | |
|---|---|
| distance | A string indicating the distance-based type. Options are "knn", "inverse_distance", or "exponential". Required if method = "distance". |
| k | An integer specifying the number of nearest neighbors. Used if distance = "knn". Default is 2. |
| dmax | A numeric specifying the maximum distance threshold. Required if distance is "inverse_distance" or "exponential". |
| power | A numeric specifying the decay power. Used if distance = "inverse_distance". Default is 1. |
| alpha | A numeric specifying the decay parameter. Used if distance = "exponential". Default is 1. |
| epsilon | A small numeric value to prevent division by zero in inverse distance calculations. Default is 1e-12. |
| kernel | A string indicating the type of spatial kernel. Options are "uniform", "gaussian", "triangular", "epanechnikov", or "quartic". Required if method = "kernel". |
| bandwidth | A numeric specifying the bandwidth ($h$) for kernel weights. Required if method = "kernel". |
| lonlat | Logical; if TRUE, coordinates are treated as Longitude/Latitude (degrees), and distances are calculated in kilometers. If FALSE, distances are calculated using Euclidean geometry in the native units of the coordinates (typically meters for projected CRS). Default is TRUE. |
| style | A character string specifying the spatial weights coding scheme. Options are "W" (row-standardized, for HB Beta SAR models) or "B" (binary adjacency, for HB Beta Leroux CAR models). Default is "W". |
| zero.policy | Logical; if TRUE, areas with no neighbors are allowed to have zero-weight rows. Default is TRUE. |
| output | A character string specifying the desired format of the returned object. Options are "all", "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.

**Important note on `style` for HB Beta Spatial models:**

- `style = "W"`: Returns a row-standardized weights matrix, where non-isolated rows are standardized to sum to 1. Isolated areas may have zero-weight rows when `zero.policy = TRUE`. This is the required format for HB Beta Spatial Autoregressive (SAR) models.
- `style = "B"`: Returns a binary adjacency matrix (all positive weights are forcefully converted to 1). This is the required format for HB Beta Conditional Autoregressive (CAR) models, specifically the Leroux CAR model. When `style = "B"`, the resulting neighbour structure is forced to be symmetric (using `spdep::make.sym.nb()`) for methods that can otherwise produce asymmetric neighbour relations (Contiguity, particularly when a fallback connection is applied, KNN, and Inverse Distance/Exponential without `dmax`).

For continuous weighting methods (Inverse Distance, Exponential, Kernel), using `style = "B"` will ignore the calculated continuous weights and binarize the connections. Contiguity or KNN are therefore a more natural fit for Leroux CAR models, since their neighbourhood structure is already binary by design. Continuous methods remain usable with `style = "B"`, but their distance-based weighting information is discarded in the process.

## Value

Depending on the `output` argument, this function returns:

- `"matrix"`: An $N \times N$ spatial weights matrix (`W`). Used as the spatial weight input for SAR and Leroux CAR models.
- `"listw"`: A `listw` object. Used as the input for `moran_test()`.
- `"nb"`: An `nb` (neighborhood) object representing the list of neighbors for each area.
- `"all"`: A comprehensive list containing `W`, `listw`, `nb`, `info`, and `diag`.

## Examples

```
library(sf)

# 1. Contiguity Method (Requires sf polygons)
# Create a simple 3x3 polygon grid
bbox <- st_bbox(c(xmin = 0, ymin = 0, xmax = 3, ymax = 3))
grid <- st_make_grid(bbox, n = c(3, 3))
grid_sf <- st_sf(id = 1:9, geometry = grid)

# Build spatial weights with output = "all"
W_obj <- build_w(
  data = grid_sf,
  method = "contiguity",
  contiguity = "queen",
  style = "W",
  output = "all"
)

# Extract outputs from the list
W_mat   <- W_obj$W      # Spatial weights matrix (for SAR/CAR)
lw_obj  <- W_obj$listw  # listw object (for moran_test)
W_diag  <- W_obj$diag   # Diagnostic info (isolated areas, etc.)
head(W_mat)

# Setup Coordinates for Distance & Kernel
# 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)

# 2. Distance Method (Using coordinates)
# Build row-standardized KNN weights (style = "W") for SAR models
W_knn <- build_w(
```

```
  data = NULL,
  coords = coords,
  method = "distance",
  distance = "knn",
  k = 2,
  lonlat = TRUE,
  style = "W",
  output = "matrix"
)
head(W_knn)

# 3. Kernel Method (Using coordinates)
# Build binary adjacency Kernel weights (style = "B") for Leroux CAR models
W_kernel <- build_w(
  data = NULL,
  coords = coords,
  method = "kernel",
  kernel = "epanechnikov",
  bandwidth = 500,
  lonlat = TRUE,
  style = "B",
  output = "matrix"
)
head(W_kernel)

# 4. Fallback Mechanism (Handling Isolated Areas)
# Create polygons where one area is isolated (e.g., an island)
p1 <- st_polygon(list(rbind(c(0,0), c(1,0), c(1,1), c(0,1), c(0,0))))
p2 <- st_polygon(list(rbind(c(1,0), c(2,0), c(2,1), c(1,1), c(1,0))))
p3 <- st_polygon(list(rbind(c(10,10), c(11,10), c(11,11), c(10,11), c(10,10))))
islands_sf <- st_sf(id = 1:3, geometry = st_sfc(p1, p2, p3))

# Without fallback, the isolated area (area 3) remains isolated
W_none <- build_w(islands_sf, method = "contiguity", contiguity = "queen",
                  fallback = "none", style = "B", output = "all")
W_none$diag$n_isolates_after

# With KNN fallback, the isolated area is connected automatically,
# and symmetry is guaranteed for style = "B"
W_fallback <- build_w(islands_sf, method = "contiguity", contiguity = "queen",
                      fallback = "knn", fallback_k = 1, style = "B", output = "all")
W_fallback$diag$n_isolates_after
isSymmetric(W_fallback$W)
```

---

`databeta` *Synthetic Data for Small Area Estimation using Hierarchical Bayesian Spatial Beta Models*

---

## 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 $n_i$ from a discrete uniform distribution over integers between 10 and 50, and survey design effects $deff_i \sim U(1, 2.5)$ (rounded to two decimal places). 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 $\rho$ 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 using Hierarchical Bayesian Spatial Beta Models*

### 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 performs Moran's I test for detecting global spatial autocorrelation. It provides a convenient wrapper around `spdep::moran.test()` and `spdep::moran.mc()`, with automatic handling of missing values (NA) by seamlessly subsetting both the response vector and the spatial weights object 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, typically created by build_w(). |
| 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. If FALSE, performs the analytical test under the randomisation assumption. Default is FALSE. |
| 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 approach relies on an asymptotic approximation of the sampling distribution of Moran's I.
*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. Because it computes the p-value empirically without relying on asymptotic theory, the Monte Carlo permutation approach is particularly useful for small datasets or when the assumptions of the analytical test may not hold.

## Value

An object returned by moran.test when mc = FALSE, or by moran.mc when mc = TRUE. The returned components vary depending on the selected test.

## Examples

```
library(sf)

# 1. Prepare dummy data
bbox <- st_bbox(c(xmin = 0, ymin = 0, xmax = 3, ymax = 3))
grid <- st_make_grid(bbox, n = c(3, 3))
grid_sf <- st_sf(id = 1:9, geometry = grid)
set.seed(123)
grid_sf$y <- rnorm(9)
```

```
# 2. Build spatial weights using the package's native function
W_obj <- build_w(
  data = grid_sf,
  method = "contiguity",
  contiguity = "queen",
  output = "all"
)

# 3. Perform Moran's I test (Analytical approach)
moran_test(x = grid_sf$y, listw = W_obj$listw)

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

# 5. Handling Missing Values automatically
y_with_na <- grid_sf$y
y_with_na[c(2, 5)] <- NA
moran_test(x = y_with_na, listw = W_obj$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 Hierarchical Bayesian (HB) Spatial Beta-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