Package ‘spatialAtomizeR’

February 14, 2026

Type Package

Title Spatial Analysis with Misaligned Data Using Atom-Based Regression Models

Version 0.2.6

Date 2026-02-13

Description Implements atom-based regression models (ABRM) for analyzing spatially misaligned data. Provides functions for simulating misaligned spatial data, preparing NIMBLE model inputs, running MCMC diagnostics, and providing results. All main functions return
S3 objects with print(), summary(), and plot() methods for intuitive result exploration.
Methods originally described in Mugglin et al. (2000) <doi:10.1080/01621459.2000.10474279>, further investigated in Trevisani & Gelfand (2013), and applied in Nethery et al. (2023) <doi:10.1101/2023.01.10.23284410>.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.0.0), nimble

Imports sp, sf, spdep, MASS, raster, dplyr, tidyr, ggplot2, reshape2, coda, BiasedUrn, stats, utils, grDevices, methods

Suggests testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

URL `https://github.com/bellayqian/spatialAtomizeR`

BugReports `https://github.com/bellayqian/spatialAtomizeR/issues`

RoxygenNote 7.3.3

NeedsCompilation no

Author Yunzhe Qian [aut, cre],
Rachel Nethery [aut],
Nancy Krieger [ctb] (Contributed to the project conceptualization and manuscript),
Nykesha Johnson [ctb] (Contributed to the project conceptualization and manuscript)

Maintainer Yunzhe Qian <qyzanemos@gmail.com>

Repository CRAN

Date/Publication 2026-02-14 18:00:02 UTC

Contents

`gen_correlated_spat` *Generate Correlated Spatial Effects*

Description

Generate Correlated Spatial Effects

Usage

```
gen_correlated_spat(
  W,
  n_vars,
  rho = 0.6,
  var_spat = 1,
  correlation = 0.5,
  verify = FALSE
)
```

Arguments

`W`	Spatial adjacency matrix
`n_vars`	Number of variables
`rho`	Spatial correlation parameter (default = 0.6)
`var_spat`	Spatial variance (default = 1)
`correlation`	Correlation between variables (default = 0.5)
`verify`	Logical for verification (default = FALSE)

Value

Matrix of spatial effects

---

get_abrm_model — *Get ABRM Model Code for NIMBLE*

Description

Returns the NIMBLE code for the Atom-Based Regression Model with mixed-type variables. Automatically registers custom distributions if not already registered.

Usage

```
get_abrm_model()
```

Value

A nimbleCode object containing the model specification

---

plot.abrm — *Plot method for abrm objects*

Description

Plot method for abrm objects

Usage

```
## S3 method for class 'abrm'
plot(x, ...)
```

Arguments

`x`	An object of class "abrm"
`...`	Additional arguments (ignored)

Value

Invisibly returns the input object `x`. The function is called for its side effect of displaying MCMC diagnostic plots (trace plots and density plots) if they are available in the abrm object.

print.abrm *Print method for abrm objects*

Description

Print method for abrm objects

Usage

```
## S3 method for class 'abrm'
print(x, ...)
```

Arguments

x	An abrm object
...	Additional arguments (unused)

Value

Invisibly returns the input object x. The function is called for its side effect of printing a summary of the ABRM model results including convergence status, number of parameters estimated, and key fit statistics.

print.misaligned_data *Print method for misaligned_data objects*

Description

Print method for misaligned_data objects

Usage

```
## S3 method for class 'misaligned_data'
print(x, ...)
```

Arguments

x	A misaligned_data object
...	Additional arguments (unused)

Value

Invisibly returns the input object x. The function is called for its side effect of printing a summary of the simulated misaligned spatial data including grid dimensions and number of atoms.

print.vcov.abrm — *Print method for vcov.abrm objects*

Description

Print method for vcov.abrm objects

Usage

```
## S3 method for class 'vcov.abrm'
print(x, ...)
```

Arguments

`x`	A vcov.abrm object
`...`	Additional arguments (currently unused)

Value

Invisibly returns the input object

run_abrm — *Run ABRM Analysis*

Description

Runs the Atom-Based Regression Model on simulated data

Usage

```
run_abrm(
  gridx,
  gridy,
  atoms,
  model_code,
  true_params = NULL,
  norm_idx_x = NULL,
  pois_idx_x = NULL,
  binom_idx_x = NULL,
  norm_idx_y = NULL,
  pois_idx_y = NULL,
  binom_idx_y = NULL,
  dist_y = 2,
  niter = 50000,
  nburnin = 30000,
```

```
  nchains = 2,
  thin = 10,
  seed = NULL,
  sim_metadata = NULL,
  save_plots = TRUE,
  output_dir = NULL
)
```

Arguments

gridx	The X-grid sf dataframe, containing a numeric area ID variable named 'ID' and covariates named 'covariate_x_1','covariate_x_2',...
gridy	The Y-grid sf dataframe, containing a numeric area ID variable named 'ID', covariates named 'covariate_y_1','covariate_y_2',...., and an outcome named 'y'.
atoms	The atom sf dataframe, which should contain numeric variables named 'ID_x' and 'ID_y' holding the X-grid and Y-grid cell IDs for each atom, as well as an atom-level population count named 'population'.
model_code	NIMBLE model code from get_abrm_model()
true_params	The true outcome model regression coefficient parameters, if known (e.g., from simulate_misaligned_data())
norm_idx_x	Vector of numeric indices of X-grid covariates (ordered as 'covariate_x_1','covariate_x_2',...) that should be treated as normally-distributed
pois_idx_x	Vector of numeric indices of X-grid covariates (ordered as 'covariate_x_1','covariate_x_2',...) that should be treated as Poisson-distributed
binom_idx_x	Vector of numeric indices of X-grid covariates (ordered as 'covariate_x_1','covariate_x_2',...) that should be treated as binomial-distributed
norm_idx_y	Vector of numeric indices of Y-grid covariates (ordered as 'covariate_y_1','covariate_y_2',...) that should be treated as normally-distributed
pois_idx_y	Vector of numeric indices of Y-grid covariates (ordered as 'covariate_y_1','covariate_y_2',...) that should be treated as Poisson-distributed
binom_idx_y	Vector of numeric indices of Y-grid covariates (ordered as 'covariate_y_1','covariate_y_2',...) that should be treated as binomial-distributed
dist_y	Distribution type for outcome (1=normal, 2=poisson, 3=binomial)
niter	Number of MCMC iterations (default: 50000)
nburnin	Number of burn-in iterations (default: 30000)
nchains	Number of MCMC chains (default: 2)
thin	Thinning interval (default: 10)
seed	Integer seed for reproducibility. Each chain uses seed+(chain_number-1) (default: NULL)
sim_metadata	Optional simulation metadata list
save_plots	Logical, whether to save diagnostic plots (default: TRUE)
output_dir	Directory for saving outputs (default: NULL)

Value

List containing MCMC results and parameter estimates

---

run_nimble_model — *Run NIMBLE Model with Diagnostics*

Description

Run NIMBLE Model with Diagnostics

Usage

```
run_nimble_model(
  constants,
  data,
  inits,
  sim_metadata = NULL,
  model_code,
  niter = 50000,
  nburnin = 30000,
  nchains = 2,
  thin = 10,
  seed = NULL,
  save_plots = TRUE,
  output_dir = NULL
)
```

Arguments

`constants`	List of model constants
`data`	List of data
`inits`	List of initial values
`sim_metadata`	List with simulation metadata (optional)
`model_code`	NIMBLE code object
`niter`	Number of MCMC iterations (default: 50000)
`nburnin`	Number of burn-in iterations (default: 30000)
`nchains`	Number of MCMC chains (default: 2)
`thin`	Thinning interval (default: 10)
`seed`	Integer seed for reproducibility. Each chain uses seed+(chain_number-1) (default: NULL)
`save_plots`	Logical, whether to save diagnostic plots (default: TRUE)
`output_dir`	Directory for saving plots (default: NULL)

Value

List containing MCMC samples, summary, and convergence diagnostics simulate_misaligned_data *Simulate Misaligned Spatial Data*

Description

Simulate Misaligned Spatial Data

Usage

```
simulate_misaligned_data(
  seed = 2,
  dist_covariates_x = c("normal", "poisson", "binomial"),
  dist_covariates_y = c("normal", "poisson", "binomial"),
  dist_y = "poisson",
  x_intercepts = rep(0, 3),
  y_intercepts = rep(0, 3),
  rho_x = 0.6,
  rho_y = 0.6,
  x_correlation = 0.5,
  y_correlation = 0.5,
  beta0_y = NULL,
  beta_x = NULL,
  beta_y = NULL,
  diff_pops = TRUE,
  xy_cov_cor = FALSE
)
```

Arguments

`seed` Random seed (default = 2)

`dist_covariates_x` Vector specifying distribution type for each synthetic X-grid covariate ('poisson', 'binomial', or 'normal')

`dist_covariates_y` Vector specifying distribution type for each synthetic Y-grid covariate ('poisson', 'binomial', or 'normal')

`dist_y` Distribution type for synthetic outcome variable (one of 'poisson', 'binomial', or 'normal')

`x_intercepts` Intercepts for X covariates

`y_intercepts` Intercepts for Y covariates

`rho_x` Spatial correlation parameter for X-grid covariates (0 to 1 with higher values yielding more spatial correlation, default = 0.6)

`rho_y` Spatial correlation parameter for Y-grid covariates and outcome (0 to 1 with higher values yielding more spatial correlation, default = 0.6)

`x_correlation`	Between-variable correlation for all pairs of X-grid covariates (default = 0.5)
`y_correlation`	Between-variable correlation for all pairs of Y-grid covariates (default = 0.5)
`beta0_y`	Intercept for outcome model
`beta_x`	Outcome model coefficients for X-grid covariates
`beta_y`	Outcome model coefficients for Y-grid covariates
`diff_pops`	Logical, indicating whether the atoms should be generated with different population sizes (diff_pops = TRUE) or a common population size (diff_pops = FALSE)
`xy_cov_cor`	Logical, indicating whether the atom-level spatial random effects for X-grid and Y-grid covariates should be correlated (xy_cov_cor = TRUE) or not. When set to TRUE, the x_correlation and rho_x parameters are used to generate all covariates (separate correlation parameters are not allowed for X-grid and Y-grid covariates).

Value

List containing gridy, gridx, atoms, and true_params

`summary.abrm` *Summary method for abrm objects*

Description

Summary method for abrm objects

Usage

```
## S3 method for class 'abrm'
summary(object, ...)
```

Arguments

`object`	An abrm object
`...`	Additional arguments (unused)

Value

Invisibly returns the input object `object`. The function is called for its side effect of printing the ABRM model summary including detailed parameter estimates.

summary.misaligned_data

Summary method for misaligned_data objects

Description

Summary method for misaligned_data objects

Usage

```
## S3 method for class 'misaligned_data'
summary(object, ...)
```

Arguments

object	A misaligned_data object
...	Additional arguments (unused)

Value

Invisibly returns the input object `object`. The function is called for its side effect of printing the misaligned data summary including grid information and true parameter values (beta_x and beta_y).

vcov.abrm

Variance-covariance method for abrm objects Extracts variance-covariance matrices for regression coefficients from MCMC posterior samples. Returns separate matrices for X-grid and Y-grid coefficients.

Description

Variance-covariance method for abrm objects Extracts variance-covariance matrices for regression coefficients from MCMC posterior samples. Returns separate matrices for X-grid and Y-grid coefficients.

Usage

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

Arguments

object	An abrm object from run_abrm()
...	Additional arguments (unused)

Details

The variance-covariance matrices are computed from the posterior samples of the MCMC chains. If multiple chains were run, samples are combined across chains before computing covariances.

Value

A list with class "vcov.abrm" containing:

`vcov_beta_x`	Variance-covariance matrix for X-grid coefficients
`vcov_beta_y`	Variance-covariance matrix for Y-grid coefficients
`vcov_beta_0`	Variance of the intercept (scalar)
`vcov_all`	Combined variance-covariance matrix for all parameters

Examples

```
## Not run:
# Fit model
results <- run_abrm(...)

# Get variance-covariance matrices
vcov_mats <- vcov(results)

# Access specific matrices
vcov_mats$vcov_beta_x  # Covariance for X-grid coefficients
vcov_mats$vcov_beta_y  # Covariance for Y-grid coefficients

# Compute standard errors from diagonal
sqrt(diag(vcov_mats$vcov_beta_x))

# Compute correlation matrix
cov2cor(vcov_mats$vcov_beta_y)

## End(Not run)
```

Index