

Package ‘sfclust’

September 24, 2026

Title Bayesian Spatial Functional Clustering

Version 1.1.1

Description Bayesian clustering of spatial regions with similar functional shapes using spanning trees and latent Gaussian models. The method enforces spatial contiguity within clusters and supports a wide range of latent Gaussian models, including non-Gaussian likelihoods, via the R-INLA framework. The algorithm is based on Zhong, R., Chacón-Montalván, E. A., and Moraga, P. (2026) <[doi:10.1002/sim.70597](https://doi.org/10.1002/sim.70597)>, extending the approach of Zhang, B., Sang, H., Luo, Z. T., and Huang, H. (2023) <[doi:10.1214/22-AOAS1643](https://doi.org/10.1214/22-AOAS1643)>. The package includes tools for model fitting, convergence diagnostics, visualization, and summarization of clustering results.

License MIT + file LICENSE

URL <https://erickchacon.github.io/sfclust/>,
<https://github.com/ErickChacon/sfclust>

BugReports <https://github.com/ErickChacon/sfclust/issues>

Encoding UTF-8

Imports igraph, sf, SparseM, stars, dplyr, methods, Matrix, ggplot2,
patchwork

Suggests ggraph, here, INLA, knitr, rmarkdown, testthat (>= 3.0.0)

Additional_repositories <https://inla.r-inla-download.org/R/stable>

Depends R (>= 4.1.0)

Config/testthat/edition 3

VignetteBuilder knitr

LazyData true

Config/Needs/website rmarkdown, fda, maps, pbs

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Erick A. Chacón-Montalván [aut, cre] (ORCID:
<<https://orcid.org/0000-0001-8068-1034>>),
Ruiman Zhong [aut] (ORCID: <<https://orcid.org/0000-0002-4681-9199>>),
Paula Moraga [aut] (ORCID: <<https://orcid.org/0000-0001-5266-0201>>)

Maintainer Erick A. Chacón-Montalván <erick.chaconmontalvan@wur.nl>
Repository CRAN
Date/Publication 2026-09-24 10:30:02 UTC

Contents

chapa	2
data_all	3
filter_df	4
fitted.sfclust	4
genclust	5
log_mlik_all	7
plot.sfclust	8
plot_clusters_fitted	9
plot_clusters_map	10
plot_clusters_series	10
plot_log_mlik	11
print.sfclust	12
sfclust	12
stbinom	17
stgaus	17
summary.sfclust	18
update.sfclust	19

Index	21
-------	----

chapa	<i>Spatio-temporal NDWI2 raster data from El Chaparrillo</i>
-------	--

Description

A stars raster object containing the Normalized Difference Water Index 2 (NDWI2) derived from Sentinel-2 satellite imagery over El Chaparrillo, an agricultural research center in Spain, across 86 time points.

Usage

data(chapa)

Format

A stars object with:

ndwi2 NDWI2 values (numeric) derived from Sentinel-2 bands. Non-NA values define the study area mask.

dimensions Three dimensions: x (26 columns), y (27 rows), time (86 dates from 2025-05-15 to 2025-12-01). CRS: WGS 84 / UTM zone 30N.

Examples

```
library(sfclust)

data(chapa)
chapa
plot(chapa["ndwi2"])
```

data_all	<i>Prepare data in long format</i>
----------	------------------------------------

Description

Convert spatio-temporal data to long format with a flat spatial index (`ids`) and ordered observation indices (`id_<dimname>` for each non-spatial dimension). This is a pure converter: all rows are returned including cells that are all-NA. No filtering is applied. Use `filter_df()` downstream to restrict to valid cells.

Usage

```
data_all(x, spnames = "geometry")
```

Arguments

- `x` A stars object containing the spatial and functional dimensions.
- `spnames` Character vector with the names of the spatial dimensions of `x`. Use a single name (e.g. "geometry") for vector geometry data, or two names (e.g. `c("x", "y")`) for raster data. Functional dimensions are derived automatically as all remaining dimensions.

Value

A long-format data frame with columns `id` (flat array position in the original stars object, column-major over all dimensions), `ids` (flat spatial index, column-major over `spnames`, `spnames[1]` varies fastest), `id_<dimname>` for each functional dimension, and all variables from `x`. All rows are returned; `ids` is not remapped to `1:n_valid`.

Examples

```
library(sfclust)
library(stars)

dims <- st_dimensions(
  geometry = st_sfc(lapply(1:5, function(i) st_point(c(i, i)))),
  time = seq(as.Date("2024-01-01"), by = "1 day", length.out = 3)
)
stdata <- st_as_stars(cases = array(1:15, dim = c(5, 3)), dimensions = dims)
```

`data_all(stdata)`

<code>filter_df</code>	<i>Filter a long-format data frame to valid spatial cells</i>
------------------------	---

Description

Keeps only rows whose spatial index (`ids`) appears in `valid_ids` and adds a `sid` column (1..`n_valid`) giving each valid cell's rank among the valid cells. `sid` matches the indexing of membership vectors returned by `genclust()`. If `valid_ids` is `NULL`, all rows are kept and `sid` is assigned by rank of `ids`.

Usage

`filter_df(df, valid_ids = NULL)`

Arguments

- `df` A long-format data frame as returned by `data_all()`.
- `valid_ids` Integer vector of valid flat spatial positions, as returned in `genclust(...)$valid_ids`. If `NULL`, all spatial cells are treated as valid.

Value

A filtered data frame with rows corresponding to valid spatial cells only, with an additional `sid` column (integer, 1..`n_valid`).

<code>fitted.sfclust</code>	<i>Fitted Values for sfclust Objects</i>
-----------------------------	--

Description

This function calculates the fitted values for a specific clustering sample in an `sfclust` object, based on the estimated models for each cluster. The fitted values are computed using the membership assignments and model parameters associated with the selected clustering sample.

Usage

```
## S3 method for class 'sfclust'  
fitted(object, sample = object$clust$ids, sort = FALSE, aggregate = FALSE, ...)
```

Arguments

object	An object of class 'sfclust', containing clustering results and models.
sample	An integer specifying the clustering sample number for which the fitted values should be computed. The default is the id of the current clustering. The value must be between 1 and the total number of clustering (membership) samples.
sort	Logical value indicating if clusters should be relabel based on number of elements.
aggregate	Logical value indicating if fitted values are desired at cluster level. Only supported for sfclust_stars results.
...	Additional arguments, currently not used.

Value

A data frame with fitted values and cluster assignments, keyed by id. For sfclust_stars objects, a stars object is returned instead.

Examples

```
if (requireNamespace("INLA", quietly = TRUE)) {
  library(sfclust)

  data(stgaus)
  result <- sfclust(stgaus, formula = y ~ f(id_time, model = "rw1"), niter = 10,
    nmessage = 1)

  # Estimated values ordering clusters by size
  df_est <- fitted(result, sort = TRUE)

  # Estimated values aggregated by cluster
  df_est <- fitted(result, aggregate = TRUE)

  # Estimated values using a particular clustering sample
  df_est <- fitted(result, sample = 3)
}
```

genclust

Generate initial cluster assignments

Description

Creates an undirected graph from spatial data or a weighted adjacency matrix, computes its minimum spanning tree (MST), and partitions it into nclust clusters. Accepts weighted matrix or Matrix objects, and stars objects with either vector geometry (contiguity via `sf::st_touches()`) or raster dimensions (4-connected grid adjacency for non-NA pixels).

Usage

```

genclust(x, ...)

## Default S3 method:
genclust(x, ...)

## S3 method for class 'matrix'
genclust(x, nclust = 10, weights = NULL, ...)

## S3 method for class 'Matrix'
genclust(x, nclust = 10, weights = NULL, ...)

## S3 method for class 'stars'
genclust(x, nclust = 10, spnames = NULL, response = NULL, weights = NULL, ...)

```

Arguments

<code>x</code>	Spatial data or adjacency matrix. Accepted classes: <code>matrix</code> , <code>Matrix</code> , or <code>stars</code> .
<code>...</code>	Not used; required for S3 method consistency.
<code>nclust</code>	Integer. Number of initial clusters (default 10).
<code>weights</code>	Optional numeric vector of edge weights with n^2 elements, where n is the total number of spatial units (before filtering). If <code>NULL</code> , random weights are assigned.
<code>spnames</code>	Character vector with the names of the spatial dimensions. If <code>NULL</code> , auto-detected as dimensions with a non-NA regular delta in <code>stars::st_dimensions()</code> . Currently supports 1D (vector geometry) and 2D raster grids; 3D spatial grids (e.g. x, y, z) are not yet supported.
<code>response</code>	Character. Name of the attribute in <code>x</code> to use for determining valid spatial cells (cells where all observations are NA are excluded). If <code>NULL</code> (default), all spatial cells are treated as valid.

Value

A list with:

- `graph`: undirected igraph object representing spatial contiguity.
- `mst`: minimum spanning tree of graph.
- `membership`: integer vector of cluster assignments (length = number of valid spatial units).
- `valid_ids`: integer vector of flat spatial positions included in the graph (only present for `stars` input; `NULL` for `matrix/Matrix` input).

Examples

```

library(sfclust)
library(sf)
library(stars)

# stars object with vector geometry

```

```

geom <- st_make_grid(cellsize = c(1, 1), offset = c(0, 0), n = c(3, 2))
x <- st_as_stars(st_sf(z = 1:6, geometry = geom))
clust <- genclust(x, nclust = 3)
plot(st_sf(geom, cluster = factor(clust$membership)))

# stars raster input
x <- st_as_stars(cluster = matrix(1:35, 5))
clust <- genclust(x, nclust = 4)
x$cluster <- clust$membership
plot(x, col = rainbow(4))

# matrix input
x <- matrix(c(0,1,0,1, 1,0,1,0, 0,1,0,1, 1,0,1,0), nrow = 4)
clust <- genclust(x, nclust = 2)
clust$membership

```

log_mlik_all

*Fit models and compute the log marginal likelihood for all clusters***Description**

Fit the specified INLA model to each cluster and compute the log marginal likelihood for each cluster specified in the membership vector.

Usage

```

log_mlik_all(
  membership,
  data,
  correction = TRUE,
  detailed = FALSE,
  inla_args = NULL
)

```

Arguments

membership	Integer, character or factor vector indicating the cluster membership for each spatial unit.
data	A long-format data frame as returned by filter_df() , with columns id (flat array position), ids (flat spatial position), sid (valid-cell rank, 1 to n_valid, matching membership indices), observation index columns (id_<dimname>), and all response/covariate variables.
correction	Logical value indicating whether a correction for dispersion.
detailed	Logical value indicating whether to return the INLA model instead of the log marginal likelihood. The argument correction is not applied in this case.
inla_args	A named list or pairlist of arguments passed to inla() (e.g. formula, family, E).

Value

A numeric vector containing the log marginal likelihood for each cluster or the the fitted INLA model for each cluster when detailed = TRUE.

plot.sfclust	<i>Plot function for sfclust objects</i>
--------------	--

Description

Visualizes fitted cluster functions (plot 1) and a log marginal likelihood traceplot (plot 2). For sfclust_stars objects, also includes a spatial map of cluster assignments (plot 1 in that method, shifting the others to 2 and 3).

Usage

```
## S3 method for class 'sfclust'
plot(
  x,
  sample = x$clust$id,
  which = 1:2,
  clusters = NULL,
  sort = FALSE,
  legend = FALSE,
  fnames = NULL,
  ...
)
```

Arguments

x	An sfclust object.
sample	Integer specifying the clustering sample to display. Defaults to the last sample.
which	Integer vector indicating which plots to show. For sfclust: 1 = cluster functions, 2 = log marginal likelihood. For sfclust_stars: 1 = map, 2 = cluster functions, 3 = log marginal likelihood.
clusters	Optional vector of cluster IDs to include. If NULL, all clusters are shown.
sort	Logical; if TRUE, clusters are relabeled by decreasing size. Default is FALSE.
legend	Logical; if TRUE, a legend is included. Default is FALSE.
fnames	Character. Column name for the x-axis of cluster function plots. If NULL, taken from the result's stored args.
...	Additional arguments passed to underlying plot functions.

Value

A composed patchwork object.

plot_clusters_fitted *Plot functional shapes of cluster linear predictors*

Description

Plots the estimated mean functional shape (linear predictor or inverse-link scale) for each cluster in a given MCMC sample of an `sfclust` object.

Usage

```
plot_clusters_fitted(
  x,
  sample = x$clust$id,
  clusters = NULL,
  sort = FALSE,
  legend = FALSE,
  inv_link = TRUE,
  fnames = NULL,
  ...
)
```

Arguments

<code>x</code>	An <code>sfclust</code> object.
<code>sample</code>	Integer specifying the clustering sample to display. Defaults to the last sample.
<code>clusters</code>	Optional vector of cluster IDs to include. If <code>NULL</code> , all clusters are shown.
<code>sort</code>	Logical; if <code>TRUE</code> , clusters are relabeled by decreasing size. Default is <code>FALSE</code> .
<code>legend</code>	Logical; if <code>TRUE</code> , a color legend is included. Default is <code>FALSE</code> .
<code>inv_link</code>	Logical; if <code>TRUE</code> (default), values are shown on the inverse-link (mean) scale.
<code>fnames</code>	Character. Name of the column to use as the x-axis (functional dimension). If <code>NULL</code> , taken from the result's stored args (set automatically by <code>sfclust()</code>).
<code>...</code>	Additional arguments passed to <code>geom_line()</code> .

Value

A `ggplot2` object.

plot_clusters_map	<i>Plot a spatial map of cluster assignments</i>
-------------------	--

Description

Produces a ggplot2 map of spatial regions colored by their cluster assignment for a given MCMC sample of an sfclust_stars object.

Usage

```
plot_clusters_map(
  x,
  sample = x$clust$id,
  clusters = NULL,
  sort = FALSE,
  legend = FALSE,
  geom_before = NULL,
  ...
)
```

Arguments

x	An sfclust_stars object.
sample	Integer specifying the clustering sample to display. Defaults to the last sample.
clusters	Optional vector of cluster IDs to include. If NULL, all clusters are shown.
sort	Logical; if TRUE, clusters are relabeled by decreasing size. Default is FALSE.
legend	Logical; if TRUE, a fill legend is included. Default is FALSE.
geom_before	An optional ggplot2 geom layer to add before the cluster fill layer. Default is NULL.
...	Additional arguments passed to geom_sf().

Value

A ggplot2 object.

plot_clusters_series	<i>Plot observed time series by cluster</i>
----------------------	---

Description

Plots individual region time series faceted by cluster, overlaid with the cluster mean, for a given variable in an sfclust object.

Usage

```
plot_clusters_series(x, var, clusters = NULL, sort = FALSE, fnames = NULL, ...)
```

Arguments

<code>x</code>	An <code>sfclust</code> object.
<code>var</code>	An unquoted variable name to plot on the y-axis.
<code>clusters</code>	Optional vector of cluster IDs to include. If <code>NULL</code> , all clusters are shown.
<code>sort</code>	Logical; if <code>TRUE</code> , clusters are relabeled by decreasing size. Default is <code>FALSE</code> .
<code>fnames</code>	Character. Name of the column to use as the x-axis (functional dimension). If <code>NULL</code> , taken from the result's stored args (set automatically by <code>sfclust()</code>).
<code>...</code>	Additional arguments passed to <code>geom_line()</code> for individual region series.

Value

A `ggplot2` object with one facet per cluster.

plot_log_mlik

Plot log marginal likelihood convergence trace

Description

Plots the log marginal likelihood across MCMC samples for an `sfclust` object, highlighting the selected sample.

Usage

```
plot_log_mlik(x, sample = x$clust$id, ...)
```

Arguments

<code>x</code>	An <code>sfclust</code> object.
<code>sample</code>	Integer specifying the clustering sample to highlight. Defaults to the last sample.
<code>...</code>	Additional arguments passed to <code>geom_line()</code> .

Value

A `ggplot2` object.

<code>print.sfclust</code>	<i>Print method for sfclust objects</i>
----------------------------	---

Description

Prints details of an sfclust object, including the (i) within-cluster formula; (ii) hyperparameters used for the MCMC sample such as the number of clusters penalty (q) and the movement probabilities (move_prob); (iii) the number of movement type counts during the MCMC sampling; and (iv) the log marginal likelihood of the model of the last clustering sample.

Usage

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

Arguments

<code>x</code>	An object of class 'sfclust'.
<code>...</code>	Additional arguments passed to <code>print.default</code> .

Value

Invisibly returns the input sfclust object `x`. The function also prints a summary of:

- the within-cluster model formula,
- clustering hyperparameters,
- movement counts from the MCMC sampler,
- and the log marginal likelihood of the selected sample.

<code>sfclust</code>	<i>Bayesian spatial functional clustering</i>
----------------------	---

Description

`sfclust()` is the main user-facing function for Bayesian spatial functional clustering via reversible-jump MCMC. It dispatches on the class of the first argument:

Usage

```
sfclust(x, ...)

## Default S3 method:
sfclust(x, ...)

## S3 method for class 'data.frame'
sfclust(
  x,
  adjacency,
  graphdata = NULL,
  fnames = NULL,
  nclust = 10,
  move_prob = c(0.425, 0.425, 0.1, 0.05),
  logpen = log(1 - 0.5),
  correction = TRUE,
  niter = 100,
  burnin = 0,
  thin = 1,
  nmessage = 10,
  path_save = NULL,
  nsave = nmessage,
  ...
)

## S3 method for class 'stars'
sfclust(
  x,
  nclust = 10,
  graphdata = NULL,
  spnames = NULL,
  move_prob = c(0.425, 0.425, 0.1, 0.05),
  logpen = log(1 - 0.5),
  correction = TRUE,
  niter = 100,
  burnin = 0,
  thin = 1,
  nmessage = 10,
  path_save = NULL,
  nsave = nmessage,
  ...
)
```

Arguments

x	A <code>data.frame</code> (core interface) or a <code>stars</code> object (stars interface). Dispatch is based on this argument's class. For <code>sfclust.data.frame</code> : a long-format data frame with at least columns <code>id</code> (unique row index) and <code>ids</code> (integer spatial unit
---	---

	index, 1 to ns), plus any response and covariate columns referenced in formula. For <code>sfclust.stars</code> : a stars object containing response variables, covariates, and other necessary data.
<code>...</code>	Additional arguments such as <code>formula</code> , <code>family</code> , and others passed to <code>inla()</code> .
<code>adjacency</code>	A square weighted adjacency matrix ($ns \times ns$) encoding spatial contiguity and edge weights. Can be a dense matrix or a sparse Matrix. Typically obtained via <code>igraph::as_adjacency_matrix()</code> on the graph returned by <code>genclust()</code> .
<code>graphdata</code>	A list with components <code>graph</code> , <code>mst</code> , and <code>membership</code> as returned by <code>genclust()</code> . If NULL, it is built automatically.
<code>fnames</code>	Character. Name of the column in <code>x</code> that holds the functional index (e.g. <code>"id_time"</code>). Used by plot methods; not required by the algorithm itself. Default is NULL.
<code>nclust</code>	Integer. Initial number of clusters when <code>graphdata = NULL</code> . Ignored if <code>graphdata</code> is provided (default is 10).
<code>move_prob</code>	A numeric vector of probabilities for the MCMC move types: birth, death, change, and hyperparameter (default is <code>c(0.425, 0.425, 0.1, 0.05)</code>).
<code>logpen</code>	A negative numeric value representing the log-scale penalty for increasing the number of clusters by one. The number of clusters is assumed to follow a geometric prior with probability <code>q</code> , making this penalty equal to $\log(1 - q)$. For example, if <code>logpen = -50</code> , then a proposal that increases the number of clusters will only be favored if it improves the log marginal likelihood by more than 50.
<code>correction</code>	A logical indicating whether correction to compute the marginal likelihoods should be applied (default is TRUE). This depends on the type of effects included in the INLA model.
<code>niter</code>	An integer specifying the number of MCMC iterations after burn-in (default is 100).
<code>burnin</code>	An integer specifying the number of burn-in iterations to discard (default is 0).
<code>thin</code>	An integer specifying the thinning interval for recording the results (default is 1).
<code>nmessage</code>	An integer specifying how often progress messages should be printed (default is 10).
<code>path_save</code>	A character string specifying the file path to save the results (default is NULL).
<code>nsave</code>	An integer specifying the number of iterations between saved results (default is <code>nmessage</code>).
<code>sptnames</code>	Character vector with the names of the spatial dimensions of <code>x</code> . If NULL (default), auto-detected from the stars object dimensions.

Details

- `sfclust.data.frame()`: core interface — takes a pre-built long-format data frame and a weighted adjacency matrix. Use this when working with any data format after converting it yourself.
- `sfclust.stars()`: stars wrapper — takes a stars spatio-temporal object, converts it to long format, builds the spatial graph, and calls the core algorithm.

This implementation draws inspiration from the methods described in the paper: *"Bayesian Clustering of Spatial Functional Data with Application to a Human Mobility Study During COVID-19"* by Bohai Zhang, Huiyan Sang, Zhao Tang Luo, and Hui Huang, published in *The Annals of Applied Statistics*, 2023. For further details on the methodology, please refer to:

- The paper: [doi:10.1214/22AOAS1643](https://doi.org/10.1214/22AOAS1643)
- Supplementary material: [doi:10.1214/22AOAS1643SUPPB](https://doi.org/10.1214/22AOAS1643SUPPB)

The MCMC algorithm in this implementation is largely based on the supplementary material provided in the paper. However, we have generalized the computation of the marginal likelihood ratio by leveraging INLA (Integrated Nested Laplace Approximation). This generalization enables integration over all parameters and hyperparameters, allowing for inference within a broader family of distribution functions and model terms, thereby extending the scope and flexibility of the original approach. Further details of our approach can be found in our paper *"Bayesian spatial functional data clustering: applications in disease surveillance"* by Ruiman Zhong, Erick A. Chacón-Montalván, Paula Moraga:

- The paper: [doi:10.1002/sim.70597](https://doi.org/10.1002/sim.70597)

Value

An `sfclust` object (from `sfclust.data.frame`) or an `sfclust_stars` object inheriting from `sfclust` (from `sfclust.stars`). Both contain:

- `samples`: MCMC trace with membership, `log_mlike`, and `move_counts`.
- `clust`: selected clustering with `id`, membership, and fitted models.

`sfclust_stars` additionally carries `input_args` with `stars` (structural shell of the input), `spnames`, and `fnames`, used by spatial plot methods.

Author(s)

Ruiman Zhong <ruiman.zhong@kaust.edu.sa>, Erick A. Chacón-Montalván <erick.chaconmontalvan@wur.nl>, Paula Moraga <paula.moraga@kaust.edu.sa>

Examples

```
if (requireNamespace("INLA", quietly = TRUE)) {
  library(sfclust)

  # Stars interface: Gaussian model
  data(stgaus)
  result <- sfclust(stgaus, formula = y ~ f(id_time, model = "rw1"),
    niter = 10, nmessage = 1)

  print(result)
  summary(result, sort = TRUE)
  fitted(result, sort = TRUE)

  plot(result)
  plot_clusters_series(result, var = y)
```

```

result2 <- update(result, niter = 2, nmessage = 1)
plot(result2)

# Stars interface: Binomial model
data(stbinom)
result <- sfclust(stbinom, formula = cases ~ poly(id_time, 2) + f(id),
  family = "binomial", Ntrials = population, niter = 10, nmessage = 1)

print(result)
summary(result, sort = TRUE)
fitted(result, sort = TRUE)

plot(result)
plot_clusters_series(result, var = cases/population)

result2 <- update(result, niter = 2, nmessage = 1)
plot(result2)

# data.frame interface: Poisson model
ns <- 6L; nt <- 4L
set.seed(4)
df <- data.frame(
  id      = seq_len(ns * nt),
  ids     = rep(seq_len(ns), nt),
  id_time = rep(seq_len(nt), each = ns),
  expected = rep(10L, ns * nt)
)
df <- transform(df,
  y = rpois(ns * nt, expected * exp(0.5 * id_time * rep(c(-1, 1), each = ns / 2, nt)))
)
adj <- Matrix::sparseMatrix(
  i = c(1, 2, 4, 5, 1, 2, 3), j = c(2, 3, 5, 6, 4, 5, 6),
  x = 1L, dims = c(ns, ns), symmetric = TRUE
)

result <- sfclust(df, adjacency = adj, nclust = 3, fnames = "id_time",
  formula = y ~ 1 + id_time, family = "poisson", E = expected,
  niter = 3, burnin = 0, thin = 1, nmessage = 1)

print(result)
summary(result, sort = TRUE)
fitted(result, sort = TRUE)

plot(result)
plot_clusters_series(result, var = y)

result2 <- update(result, niter = 2, nmessage = 1)
plot(result2)
}

```

stbinom	<i>Spatio-temporal Binomial data</i>
---------	--------------------------------------

Description

A simulated stars object containing binomial response data with a functional clustering pattern defined by polynomial fixed effects. This dataset includes the variables cases and population observed across 100 simulated spatial regions over 91 time points.

Usage

```
data(stbinom)
```

Format

A stars object with:

cases Number of observed cases (integer)

population Population at risk (integer)

dimensions Two dimensions: geometry (spatial features) and time (daily observations)

Examples

```
library(sfclust)

data(stbinom)
stbinom
plot(stbinom["cases"])
```

stgaus	<i>Spatio-temporal Gaussian data</i>
--------	--------------------------------------

Description

A simulated stars object containing Gaussian response data with a functional clustering pattern using random walk processes. This dataset includes the response variable y observed across 100 simulated spatial regions over 91 time points.

Usage

```
data(stgaus)
```

Format

A stars object with:

y Response variable

Examples

```
library(sfclust)

data(stgaus)
stgaus
plot(stgaus["y"])
```

summary.sfclust	<i>Summary method for sfclust objects</i>
-----------------	---

Description

This function summarizes the cluster assignments from the desired clustering sample.

Usage

```
## S3 method for class 'sfclust'
summary(object, sample = object$clust$id, sort = FALSE, ...)
```

Arguments

object	An object of class 'sfclust'.
sample	An integer specifying the clustering sample number to be summarized (default is the last sample).
sort	Logical value indicating if clusters should be relabel based on number of elements.
...	Additional arguments passed to <code>print.default</code> .

Value

Invisibly returns a table with the number of regions in each cluster for the selected sample. The function also prints a summary that includes:

- the within-cluster model formula,
- the total number of MCMC clustering samples,
- the cluster membership counts for the specified sample (optionally sorted),
- and the log marginal likelihood of the selected clustering sample.

update.sfclust

*Update MCMC Clustering Procedure***Description**

This function continues the MCMC sampling of a `sfclust` object based on previous results or update the model fitting for a specified sample clustering if the argument `sample` is provided.

Usage

```
## S3 method for class 'sfclust'
update(
  object,
  niter = NULL,
  burnin = 0,
  thin = 1,
  nmessage = 10,
  sample = NULL,
  path_save = NULL,
  nsave = nmessage,
  ...
)
```

Arguments

<code>object</code>	A <code>sfclust</code> object.
<code>niter</code>	An integer specifying the number of additional MCMC iterations to perform. If <code>NULL</code> (default), no additional iterations are run and the within-cluster INLA models are refitted for the current clustering sample.
<code>burnin</code>	An integer specifying the number of burn-in iterations to discard.
<code>thin</code>	An integer specifying the thinning interval for recording results.
<code>nmessage</code>	An integer specifying the number of messages to display during the process.
<code>sample</code>	An integer specifying the clustering sample number to be executed. The default is the last sample (i.e., <code>nrow(x\$samples\$membership)</code>).
<code>path_save</code>	A character string specifying the file path to save the results. If <code>NULL</code> , results are not saved.
<code>nsave</code>	An integer specifying how often to save results. Defaults to <code>nmessage</code> .
<code>...</code>	Additional arguments (currently not used).

Details

This function takes the last state of the Markov chain from a previous `sfclust_fit` / `sfclust` execution and uses it as the starting point for additional MCMC iterations. If `niter = NULL` (the default), no additional iterations are run; instead, the within-cluster INLA models are fitted for the clustering given by `sample` (or the current `clust$id` if `sample` is `NULL`). This is useful when loading a checkpoint saved during a run, where the models were not yet stored: `result <- update(result)`.

Value

An updated `sfclust` object with (i) new clustering samples if `sample` is not specified, or (ii) updated within-cluster model results if `sample` is given.

Index

* datasets

- chapa, [2](#)
- stbinom, [17](#)
- stgaus, [17](#)

chapa, [2](#)

data_all, [3](#)
data_all(), [4](#)

filter_df, [4](#)
filter_df(), [7](#)
fitted.sfclust, [4](#)

genclust, [5](#)
genclust(), [4](#), [14](#)

igraph::as_adjacency_matrix(), [14](#)

log_mlik_all, [7](#)

plot.sfclust, [8](#)
plot_clusters_fitted, [9](#)
plot_clusters_map, [10](#)
plot_clusters_series, [10](#)
plot_log_mlik, [11](#)
print.sfclust, [12](#)

sf::st_touches(), [5](#)
sfclust, [12](#)
sfclust(), [9](#), [11](#)
stars::st_dimensions(), [6](#)
stbinom, [17](#)
stgaus, [17](#)
summary.sfclust, [18](#)

update.sfclust, [19](#)