Package 'spCF'

June 29, 2026

Type Package

Title Coarse-to-Fine Spatial Modeling

Version 0.1.2

Imports FNN, fields, nloptr, dbscan, ranger, withr, Rcpp

LinkingTo Rcpp

Suggests sp, sf, knitr, rmarkdown, CARBayesdata, lightgbm

Description Provides functions for coarse-to-fine spatial modeling (CFSM), enabling fast spatial prediction, regression, and uncertainty quantification. This method is suitable for moderate to large samples. For methodological details, see Murakami et al. (2026) <doi:10.1111/gean.70034> and related works on its generalized-linear <doi:10.48550/arXiv.2605.01157> and downscaling extensions.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.3.3

VignetteBuilder knitr

NeedsCompilation yes

Author Daisuke Murakami [aut, cre],
Alexis Comber [aut],
Takahiro Yoshida [aut],
Narumasa Tsutsumida [aut],
Chris Brunsdon [aut],
Tomoki Nakaya [aut]

Maintainer Daisuke Murakami <dmuraka@ism.ac.jp>

Repository CRAN

Date/Publication 2026-06-29 07:50:08 UTC

Contents

`cf_downscale` *Coarse-to-fine spatial downscaling (CF-DS)*

Description

Scalable downscaling via CF-DS for predicting disaggregate-level responses from aggregate-level response `Y`, while ensuring that predictions aggregate exactly to the observed aggregate-level values.

Usage

```
cf_downscale(
  Y,
  x = NULL,
  prop_weight = NULL,
  coords,
  agg_id,
  mod_hv,
  adj = TRUE,
  nonneg = TRUE
)
```

Arguments

`Y`	Vector of aggregate-level response variables (length `N`).
`x`	Matrix of disaggregate-level covariates (`n` x `K`), assumed to match the `x` used in `cf_downscale_hv`.
`prop_weight`	Vector of disaggregate-level proportional allocation weights (length `n`), assumed to match the `prop_weight` used in `cf_downscale_hv`. See `cf_downscale_hv` for the role and examples of choices.
`coords`	Matrix of disaggregate-level coordinates (`n` x `2`).
`agg_id`	Area ID for each disaggregate-level unit (length `n`).
`mod_hv`	Output object from `cf_downscale_hv`.
`adj`	Logical (default `TRUE`). When `TRUE`, a per-area multiplicative adjustment is applied to satisfy the aggregation constraint so that the downscaled predictions aggregate exactly to the observed 'Y'. When `FALSE`, the constraint is satisfied only approximately, which may be preferable when 'Y' contains noise.
`nonneg`	If `TRUE` (default), clip negative predictions to zero before the multiplicative adjustment.

Value

A list with the following elements:

beta Regression coefficients, their standard errors, and the lower and upper limits of the 95 percent confidence intervals.

sd_summary Standard deviation of the regression term (xb), spatial processes (spatial_scale1, spatial_scale2,...), and residuals.

e_summary Aggregate-level holdout validation accuracy, evaluated on the validation units: R-squared (validation_R2), root mean squared error (validation_RMSE), and mean absolute error (validation_MAE). All are `NA` when no validation areas are available (e.g. `train_rat = 1`).

pred Predictive mean (`pred`) and standard deviation (`pred_sd`) of the disaggregate-level response. The spatial-process contribution to `pred_sd` is rescaled by a holdout-calibrated factor (stored as `other$tau`) estimated on the validation areas.

bands Bandwidth values for each accepted scale during the holdout validation in `cf_downscale_hv`.

Z Predictive mean of each single-scale spatial process at the disaggregate-level (data.frame; one column per scale).

Z_sd Predictive standard deviation of the single-scale process at the disaggregate-level units (data.frame).

other Other internally used output objects.

Author(s)

Daisuke Murakami

References

Murakami, D., Chun, Y., Yoshida, T., & Seya, H. (2026). Scalable coarse-to-fine spatial downscaling. *ArXiv preprint*.

See Also

`cf_downscale_hv`, `cf_lm`

Examples

```
## Not run:
set.seed(123)
require(sf); require(CARBayesdata)
data(GGHB.IZ)
data(pollutionhealthdata)
d  <- pollutionhealthdata[pollutionhealthdata$year == 2010, ]
ar <- merge(GGHB.IZ, d, by = "IZ")

### Disaggregate-level data (271 units)
coords <- st_coordinates(suppressWarnings(st_centroid(ar)))
x      <- data.frame(pm10 = ar$pm10, jsa = ar$jsa, price = ar$price)
prop_weight <- as.numeric(ar$expected)
```

```
### Aggregate-level data (30 units).
agg_id <- as.integer(stats::kmeans(coords, centers = 30)$cluster)

### Two types of response variables are possible:
# Y_type = "sum"  : Y_I = sum(response variable for each aggregate unit)
# Y_type = "mean" : Y_I = mean(response variable for each aggregate unit)
Y_type <- "sum"   # change to "mean" for the density-type data
Y      <- as.numeric(stats::aggregate(ar$observed, by = list(agg_id),
                       FUN = if (Y_type == "sum") sum else mean)[, 2])

### Downscaling
mh <- cf_downscale_hv(Y = Y, Y_type = Y_type, x = x,
                      prop_weight = prop_weight,
                      coords = coords, agg_id = agg_id)
md <- cf_downscale(Y = Y, x = x, prop_weight = prop_weight,
                   coords = coords, agg_id = agg_id, mod_hv = mh)

### Mapping
ar$agg_id <- agg_id
agg_poly  <- stats::aggregate(ar["agg_id"], by = list(agg_id = agg_id),
                              FUN = function(z) z[1])
agg_poly$Y<- Y
ar$pred   <- md$pred$pred
plot(agg_poly["Y"], nbreaks = 20, main = "Aggregated data")
plot(ar["pred"], nbreaks = 20, main = "Downscaling result")

## End(Not run)
```

`cf_downscale_hv` *Holdout validation for the coarse-to-fine spatial downscaling (CF-DS)*

Description

Trains the CF-DS model and selects the number of spatial scales through sequential holdout validation.

Usage

```
cf_downscale_hv(
  Y,
  Y_type = "sum",
  x = NULL,
  prop_weight = NULL,
  coords,
  agg_id,
  train_rat = 0.75,
  id_train = NULL,
  alpha = 0.9,
```

```
  kernel = "exp",
  rel_tol = 1e-04,
  seed = 123
)
```

Arguments

Y	Vector of aggregate-level response values (length N).
Y_type	Aggregation type of Y: "sum" for extensive (count-like) data (e.g., population) or "mean" for intensive (density-like) data (e.g., population density, average temperature).
x	Matrix of disaggregate-level covariates (n x K).
prop_weight	Vector of disaggregate-level proportional allocation weights (length n) used to distribute the aggregate-level response across the disaggregate-level units. When Y_type="mean", prop_weight should corresponding to the denominator of the intensive response variable. Examples include residential land area for population downscaling, population for morbidity downscaling, and NULL (= rep(1, n)) for temperature downscaling.
coords	Matrix of disaggregate-level coordinates (n x 2).
agg_id	Area ID for each disaggregate-level unit (length n).
train_rat	Ratio of the aggregate-level units used for model training (default 0.75) in the holdout validation.
id_train	Optional. If specified, the corresponding aggregate-level units are used as training units. Otherwise, training units are chosen based on 'train_rat'.
alpha	Decay ratio of the kernel bandwidth in the coarse-to-fine training (default: 0.9). Values closer to one make the optimization more stringent but increase computation time.
kernel	Kernel type for modeling spatial dependence. '"exp"' for the exponential kernel (default) and '"gau"' for the Gaussian kernel.
rel_tol	Relative improvement threshold for validation SSE (default 1e-4). At each scale, the spatial process is retained only if validation SSE improves by more than rel_tol; otherwise a stopping counter is incremented, and learning stops once 5 consecutive scales fail to improve. Larger values stop earlier, whereas smaller values allow finer scales to be selected.
seed	Random seed used for the training/validation split when 'id_train' is not supplied. Default is '123'. Set to 'NULL' to allow a different split at each call (useful for assessing split sensitivity).

Value

A list with the following elements:

sse_hv Final sum-of-squared error (SSE) for validation samples.

sse_hv_all SSEs obtained at each learning step.

id_train ID of training aggregate-level units.

other Other internally used output objects.

Author(s)

Daisuke Murakami

References

Murakami, D., Chun, Y., Yoshida, T., & Seya, H. (2026). Scalable coarse-to-fine spatial downscaling. *ArXiv preprint*.

See Also

cf_downscale, cf_lm_hv

---

cf_glm *Coarse-to-fine spatial generalized linear mixed models (CF-GLMMs)*

Description

Scalable prediction, regression, and multiscale analysis via CF-GLMMs.

Usage

```
cf_glm(
  y,
  x = NULL,
  coords,
  offset = NULL,
  x0 = NULL,
  coords0 = NULL,
  offset0 = NULL,
  mod_hv,
  robust_se = TRUE
)
```

Arguments

y	Vector of response variables (N x 1), including continuous, count, and binary responses following an exponential family distribution.
x	Matrix of covariates (N x K).
coords	Matrix of 2-dimensional point coordinates (N x 2).
offset	Optional. Vector of offset variables (N x 1) included in the linear predictor, consistent with glm.
x0	Optional. Matrix of covariates at prediction sites (N0 x K).
coords0	Optional. Matrix of 2-dimensional point coordinates at prediction sites (N0 x 2).
offset0	Optional. Vector of offset variables at prediction sites (N0 x 1)

mod_hv	Output object of the cf_glm_hv function.
robust_se	If TRUE (default), coefficient standard errors and predictive uncertainty are computed using a cluster-robust sandwich estimator accounting for local spatial correlation. Set FALSE to use naive SEs (not recommended).

Value

A list with the following elements:

beta Regression coefficients, their standard errors, and the lower and upper limits of the 95 percent confidence intervals.

sd_summary Standard deviation of the regression term (xb), spatial process (spatial_scale1, spatial_scale2,...), additional learning, and residuals.

e_summary Holdout validation accuracy evaluated on the validation samples: R-squared (validation_Pseudo-R2), root mean squared error (validation_RMSE), and mean absolute error (validation_MAE).

pred Predictive means and standard deviations (sample sites). The spatial-process contribution to the predictive SD is rescaled by a holdout-calibrated factor (stored as other$tau) estimated on the validation samples.

pred0 Predictive means and standard deviations (prediction sites).

pred_q Predictive quantiles on the response scale at the sample sites. A data frame whose columns q0.005, q0.025, q0.05, q0.1, ..., q0.9, q0.95, q0.975, q0.995 give the corresponding quantile levels, obtained by Gaussian approximation on the link scale followed by inverse-link transformation.

pred0_q Predictive quantiles on the response scale at the prediction sites. Column structure is identical to pred_q. NULL when prediction sites are not supplied.

bands Bandwidth values for each scale. The i-th bandwidth corresponds to the i-th column of the Z matrix.

Z Predictive mean of the spatial process at each scale (sample sites; list).

Z_sd Predictive standard deviation of the spatial process at each scale (sample sites; list).

Z0 Predictive mean of the spatial process at each scale (prediction sites; list).

Z0_sd Predictive standard deviation of the spatial process at each scale (prediction sites; list).

other Other internally used output objects.

Author(s)

Daisuke Murakami

References

Murakami, D., Comber, A., Yoshida, T., Tsutsumida, N., Brunsdon, C., & Nakaya, T. (2025). Coarse-to-fine spatial GLMMs for scalable prediction and multiscale analysis. *ArXiv preprint*, 2605.01157. https://doi.org/10.48550/arXiv.2605.01157

See Also

cf_glm_hv, sp_scalewise

Examples

```
################ Example 1: Count data modeling/Disease mapping/smoothing
set.seed(1234)
require( CARBayesdata )
require( sf )
data(pollutionhealthdata)
data(GGHB.IZ)

### Data
dat      <- pollutionhealthdata[pollutionhealthdata$year==2011,]
y        <- dat[,"observed"]             # count data
x        <- dat[,c("pm10","jsa","price")]
offset   <- log(dat[,"expected"])
coords   <- st_coordinates(st_centroid(GGHB.IZ))

### Holdout validation optimizing the number of spatial scales
mod_hv   <- cf_glm_hv(y = y, x = x, offset=offset, coords = coords, family=poisson())

### Spatial modeling and prediction
mod      <- cf_glm(y = y, x = x, coords = coords, mod_hv = mod_hv)
mod

### Mapping predictive mean and standard deviations (SD)
GGHB.IZ$y      <- y
GGHB.IZ$pred   <- mod$pred$pred
GGHB.IZ$pred_sd<- mod$pred$pred_sd
plot(GGHB.IZ[,c("pred")],lwd=0.2,axes=TRUE, key.pos=4,nbreaks=50)   # Predictive mean
plot(GGHB.IZ[,c("pred_sd")],lwd=0.2,axes=TRUE, key.pos=4,nbreaks=50)# Predictive SD

### Multiscale spatial pattern/feature extraction
mod_s1      <- sp_scalewise(mod,bw_range=c(4000,Inf)) # Large scale (4000 <= bandwidth)
mod_s2      <- sp_scalewise(mod,bw_range=c(0,4000))   # Small scale (bandwidth <= 4000)
GGHB.IZ$z1  <- mod_s1$pred$pred
GGHB.IZ$z2  <- mod_s2$pred$pred
plot(GGHB.IZ[,c("z1","z2")],lwd=0.2,axes=TRUE,key.pos=4, nbreaks=50)# Extracted features


################ Example 2: Binary data modeling/spatial prediction
set.seed(1234)
require(sp); require(sf)
data(meuse)
data(meuse.grid)

### Data
y        <- ifelse(meuse$ffreq==1, 1, 0 )# binary data
coords   <- meuse[,c("x","y")]
x        <- meuse["dist"]

### Data at prediction sites
coords0  <- meuse.grid[,c("x","y")]
x0       <- meuse.grid["dist"]
```

```
### Holdout validation optimizing the number of spatial scales
mod_hv   <- cf_glm_hv(y = y, x = x, coords = coords, family=binomial())

### Spatial modeling and prediction
mod      <- cf_glm(y = y, x=x, coords = coords, x0=x0, coords0 = coords0,
                   mod_hv = mod_hv)
mod

### Mapping predictive mean and standard deviations (SD)
meuse.grid$pred   <- mod$pred0$pred
meuse.grid$pred_sd<- mod$pred0$pred_sd
meuse.grid_sf     <- st_as_sf(meuse.grid, coords = c("x","y"))
plot(meuse.grid_sf[,"pred"], pch = 15, cex = 0.8, nbreaks = 20)   # Predictive mean
plot(meuse.grid_sf[,"pred_sd"], pch = 15, cex = 0.8, nbreaks = 20)# Predictive SD

### Multiscale spatial pattern/feature extraction
mod_s1<- sp_scalewise(mod,bw_range=c(1000,Inf)) # Large scale (1000 <= bandwidth)
mod_s2<- sp_scalewise(mod,bw_range=c(0,1000))   # Small scale (0 <= bandwidth <= 1000)
meuse.grid_sf$z1    <- mod_s1$pred0$pred
meuse.grid_sf$z2    <- mod_s2$pred0$pred
plot(meuse.grid_sf[,c("z1","z2")], pch = 15,
     cex = 0.5, nbreaks = 20,axes=TRUE) # Predictive means
```

`cf_glm_hv` *Holdout validation for coarse-to-fine spatial generalized linear mixed models (CF-GLMMs)*

Description

Trains CF-GLMMs and selects the number of spatial scales through sequential holdout validation.

Usage

```
cf_glm_hv(
  y,
  x = NULL,
  coords,
  offset = NULL,
  train_rat = 0.75,
  id_train = NULL,
  alpha = 0.9,
  kernel = "exp",
  family = gaussian(),
  seed = 1234
)
```

Arguments

y	Vector of response variables (N x 1) including continuous, count, and binary responses, following an exponential family distribution.
x	Matrix of covariates (N x K).
coords	Matrix of 2-dimensional point coordinates (N x 2).
offset	Optional. Vector of offset variables (N x 1) included in the linear predictor, consistent with glm.
train_rat	Training sample ratio (default: 0.75). For small to moderate samples (N <= 30000), samples closest to the k-means centers are used for validation samples to stabilize training. For larger samples, training samples are drawn at random.
id_train	Optional. ID indicating training samples. If specified, the corresponding samples are used as training samples. Otherwise, training samples are chosen based on 'train_rat'.
alpha	Decay ratio of the kernel bandwidth in the coarse-to-fine training (default: 0.9). Values closer to one make the optimization more stringent but increase computation time.
kernel	Kernel type for modeling spatial dependence. '"exp"' for the exponential kernel (default) and '"gau"' for the Gaussian kernel.
family	Error distribution and link function specification, consistent with the 'family' argument of glm.
seed	Random seed used for the training/validation split when 'id_train' is not supplied. Default is '1234'. Set to 'NULL' to allow a different split at each call (useful for assessing split sensitivity).

Value

A list with the following elements:

loss_hv Final deviance loss for validation samples.

loss_hv_all Deviance losses obtained at each learning step.

id_train ID of training samples.

other Other internally used output objects.

Author(s)

Daisuke Murakami

References

Murakami, D., Comber, A., Yoshida, T., Tsutsumida, N., Brunsdon, C., & Nakaya, T. (2025). Coarse-to-fine spatial GLMMs for scalable prediction and multiscale analysis. *ArXiv preprint*, 2605.01157. https://doi.org/10.48550/arXiv.2605.01157

See Also

cf_glm

cf_lm *Coarse-to-fine spatial modeling (CFSM) for Gaussian response*

Description

Scalable prediction, regression, and multiscale analysis via Gaussian CFSM.

Usage

```
cf_lm(y, x = NULL, coords, x0 = NULL, coords0 = NULL, mod_hv, robust_se = TRUE)
```

Arguments

y	Vector of response variables (N x 1).
x	Matrix of covariates (N x K).
coords	Matrix of 2-dimensional point coordinates (N x 2).
x0	Optional. Matrix of covariates at prediction sites (N0 x K).
coords0	Optional. Matrix of 2-dimensional point coordinates at prediction sites (N0 x 2).
mod_hv	Output object of the cf_lm_hv function.
robust_se	If TRUE (default), coefficient standard errors and predictive uncertainty are computed using a cluster-robust sandwich estimator accounting for local spatial correlation. Set FALSE to use naive SEs (not recommended).

Value

A list with the following elements:

beta Regression coefficients, their standard errors, and the lower and upper limits of the 95 percent confidence intervals.

sd_summary Standard deviation of the regression term (xb), spatial processes (spatial_scale1, spatial_scale2,...), additional learned components (effective if 'cf_lm_hv/add_learn' is not 'none'), and residuals.

e_summary Holdout validation accuracy evaluated on the validation samples: R-squared (validation_R2), root mean squared error (validation_RMSE), and mean absolute error (validation_MAE).

pred Predictive means and standard deviations (sample sites). When no additional learner is active, the spatial-process contribution to the predictive SD is rescaled by a holdout-calibrated factor (stored as other$tau) estimated on the validation samples.

pred0 Predictive means and standard deviations (prediction sites).

pred_q Predictive quantiles at the sample sites (data.frame with columns q0.005, q0.025, ..., q0.975, q0.995). With add_learn = "rf"/"lightgbm" active, the combined predictive distribution is calibrated by total conformalized quantile regression (CQR) on the validation samples; otherwise the quantiles are Gaussian about the predictive mean using the (tau-calibrated) pred_sd. pred_sd is a Gaussian-equivalent summary of these quantiles.

pred0_q Predictive quantiles at the prediction sites; identical column structure to `pred_q`. `NULL` when prediction sites are not supplied.

bands Bandwidth values for each scale. The i-th bandwidth corresponding to the i-th column of the Z matrix.

Z Predictive means of the single-scale processes at each scale, corresponding to each bandwidth value (sample sites; list).

Z_sd Predictive standard deviation of the spatial processes at each scale (sample sites; list).

Z0 Predictive mean of the spatial process at each scale (prediction sites; list).

Z0_sd Predictive standard deviation of the spatial process at each bandwidth (prediction sites; list).

other Other internally used output objects.

Author(s)

Daisuke Murakami

References

Murakami, D., Comber, A., Yoshida, T., Tsutsumida, N., Brunsdon, C., & Nakaya, T. (2026). Coarse-to-fine spatial modeling: A scalable, machine-learning-compatible framework. *Geographical Analysis*, 58(2), e70034. https://onlinelibrary.wiley.com/doi/10.1111/gean.70034

See Also

`cf_glm`, `cf_lm_hv`, `sp_scalewise`

Examples

```
set.seed(123)
require(sp); require(sf)
data(meuse)
data(meuse.grid)

### Data
y        <- log(meuse[,"zinc"])
coords   <- meuse[,c("x","y")]
x        <- data.frame(dist   = meuse[,"dist"],
                       ffreq2 = as.integer(meuse$ffreq == 2),
                       ffreq3 = as.integer(meuse$ffreq == 3))

### Data at prediction sites
coords0  <- meuse.grid[,c("x","y")]
x0       <- data.frame(dist   = meuse.grid[,"dist"],
                       ffreq2 = as.integer(meuse.grid$ffreq == 2),
                       ffreq3 = as.integer(meuse.grid$ffreq == 3))

### Holdout validation optimizing the number of spatial scales
mod_hv   <- cf_lm_hv(y = y, x = x, coords = coords, add_learn = "none")

### Spatial modeling and prediction
```

```
mod      <- cf_lm(y = y, x = x, x0 = x0, coords = coords, coords0 = coords0,
                  mod_hv = mod_hv)
mod

### Mapping predictive mean and standard deviations (SD)
meuse.grid$pred   <- mod$pred0$pred
meuse.grid$pred_sd<- mod$pred0$pred_sd
meuse.grid_sf     <- st_as_sf(meuse.grid, coords = c("x","y"))
plot(meuse.grid_sf[,"pred"], pch = 15, cex = 0.5, nbreaks = 20)   # Predictive mean
plot(meuse.grid_sf[,"pred_sd"], pch = 15, cex = 0.5, nbreaks = 20)# Predictive SD

### Multiscale spatial pattern/feature extraction
mod_s1<- sp_scalewise(mod,bw_range=c(1000,Inf)) # Large scale (1000 <= bandwidth)
mod_s2<- sp_scalewise(mod,bw_range=c(500,1000)) # Middle scale (500 <= bandwidth <= 1000)
mod_s3<- sp_scalewise(mod,bw_range=c(0,500))    # Small scale (bandwidth <= 500)
z1    <- mod_s1$pred0$pred                      # Predictive mean
z2    <- mod_s2$pred0$pred
z3    <- mod_s3$pred0$pred
z1_sd <- mod_s1$pred0$pred_sd                   # Predictive SD
z2_sd <- mod_s2$pred0$pred_sd
z3_sd <- mod_s3$pred0$pred_sd
meuse.grid_sf3  <- cbind(meuse.grid_sf, z1, z2, z3, z1_sd, z2_sd, z3_sd)
plot(meuse.grid_sf3[,c("z1","z2","z3")], pch = 15,
     cex = 0.5, nbreaks = 20,key.pos=4,axes=TRUE) # Predictive means
plot(meuse.grid_sf3[,c("z1_sd","z2_sd","z3_sd")], pch = 15,
     cex = 0.5, nbreaks = 20,key.pos=4,axes=TRUE) # Predictive SD
```

`cf_lm_hv` *Holdout validation for the Gaussian coarse-to-fine spatial modeling (CFSM)*

Description

Trains the CFSM-based Gaussian spatial regression and selects the number of spatial scales through sequential holdout validation.

Usage

```
cf_lm_hv(
  y,
  x = NULL,
  coords,
  train_rat = 0.75,
  id_train = NULL,
  alpha = 0.9,
  kernel = "exp",
  add_learn = "none",
  seed = 123
)
```

Arguments

y	Vector of response variables (N x 1).
x	Matrix of covariates (N x K).
coords	Matrix of 2-dimensional point coordinates (N x 2).
train_rat	Training sample ratio (default: 0.75). For small to moderate samples (N <= 30000), samples closest to the k-means centers are used for validation samples to stabilize training. For larger samples, training samples are drawn at random.
id_train	Optional. ID indicating training samples. If specified, the corresponding samples are used as training samples. Otherwise, training samples are chosen based on 'train_rat'.
alpha	Decay ratio of the kernel bandwidth in the coarse-to-fine training (default: 0.9). Values closer to one make the optimization more stringent but increase computation time.
kernel	Kernel type for modeling spatial dependence. '"exp"' for the exponential kernel (default) and '"gau"' for the Gaussian kernel.
add_learn	Additional learner trained on the residuals to capture non-linear patterns and/or higher-order interactions. '"rf"' uses a random forest (**ranger**) and '"lightgbm"' uses LightGBM (**lightgbm**); both are tuned by minimizing validation SSE. For '"lightgbm"', the predictive quantiles are conformalized on the validation split so that their uncertainty is calibrated. Default is '"none"', meaning no additional training.
seed	Random seed used for the training/validation split when 'id_train' is not supplied. Default is '123'. Set to 'NULL' to allow a different split at each call (useful for assessing split sensitivity).

Value

A list with the following elements:

sse_hv Final sum-of-squared error (SSE) for validation samples.

sse_hv_all SSEs obtained at each learning step.

id_train ID of training samples.

other Other internally used output objects.

Author(s)

Daisuke Murakami

References

Murakami, D., Comber, A., Yoshida, T., Tsutsumida, N., Brunsdon, C., & Nakaya, T. (2026). Coarse-to-fine spatial modeling: A scalable, machine-learning-compatible framework. *Geographical Analysis*, 58(2), e70034. https://onlinelibrary.wiley.com/doi/10.1111/gean.70034

See Also

cf_lm

sp_scalewise *Extract scale-wise spatial processes*

Description

Evaluate mean and variance of the spatial process with bandwidth values within a pre-specified range

Usage

```
sp_scalewise(mod, bw_range = c(0, Inf))
```

Arguments

mod	Output object from the cf_lm or cf_glm function.
bw_range	Range of bandwidth values of the simulated spatial processes. For example, if bw_range = c(10, 20), spatial processes with bandwidths between 10 and 20 are synthesized and simulated. The default is c(0, Inf), which synthesizes all scales.

Value

A list with the following elements:

pred Means and standard deviations of the spatial process (sample sites).

pred0 Means and standard deviations of the spatial process (prediction sites). NULL when mod was fitted without prediction sites.

Author(s)

Daisuke Murakami

See Also

cf_lm, cf_glm

Index