Package 'mgwrsar'

November 5, 2026

Type Package

Title GWR, Mixed GWR with Spatial Autocorrelation and Multiscale GWR/GTWR (Top-Down Scale Approaches)

Version 1.4.4

Date 2026-10-28

Maintainer Ghislain Geniaux <ghislain.geniaux@inrae.fr>

Description Provides methods for Geographically Weighted Regression with spatial autocorrelation (Geniaux and Martinetti 2017) <doi:10.1016/j.regsciurbeco.2017.04.001>. Implements Multiscale Geographically Weighted Regression with Top-Down Scale approaches (Geniaux 2026) <doi:10.1007/s10109-025-00481-4>.

License GPL (>= 2)

Depends R (>= 3.5.0), sp, Matrix

Imports Rcpp, ggplot2, sf, knitr, methods, doParallel, foreach, nabor, mapview, rlang, dplyr, gridExtra, grid, mboost, mgcv, caret, stringr, plotly, RhpcBLASctl, magrittr, lifecycle

Suggests rmarkdown

VignetteBuilder knitr

LinkingTo Rcpp, RcppEigen, RcppArmadillo

Config/notCompileAttributes true

NeedsCompilation yes

Encoding UTF-8

SystemRequirements C++17 (OpenMP optional; enable with MGWRSAR_OPENMP=1)

Author Ghislain Geniaux [aut, cre], Davide Martinetti [aut], César Martinez [aut]

Config/roxygen2/version 8.1.0

Repository CRAN

Date/Publication 2026-11-05 09:10:02 UTC

Contents

as_gtwr . . . 2
atds_gwr . . . 4
bw_gdt2hwb . . . 4
bw_hwb2gdt . . . 5
coef,mgwrsar-method . . . 6
find_TP . . . 6
fitted,mgwrsar-method . . . 8
golden_search_2d_bandwidth . . . 8
golden_search_bandwidth . . . 9
gtwr-class . . . 11
gtwr_HWB2010 . . . 11
kernel_matW . . . 13
make_unique_by_structure . . . 15
MGWRSAR . . . 16
mgwrsar-class . . . 19
mgwrsar_bootstrap_test . . . 21
mgwrsar_bootstrap_test_all . . . 22
multiscale_gwr . . . 22
mydata . . . 24
mydatasf . . . 25
normW . . . 25
plot,mgwrsar,missing-method . . . 26
plot_effect . . . 27
predict,mgwrsar-method . . . 28
reord_D . . . 30
reord_M_R . . . 30
residuals,mgwrsar-method . . . 31
search_bandwidths . . . 31
simu_multiscale . . . 33
summary,gtwr-method . . . 35
summary,mgwrsar-method . . . 36
summary_Matrix . . . 36
TDS_MGWR . . . 37

Index **42**

`as_gtwr` *Coerce a calibrated Type='GDT' model into Huang's parameterisation*

Description

Bandwidth calibration for GTWR goes through `search_bandwidths`, the entry point of the package for space-time kernels, which optimises on the `(h_S, h_T)` axes. This turns its `best_model` into an object of class `gtwr-class`, so that the result reads in the (h_{ST}, τ) parameterisation of Huang et al. (2010) and inherits the dedicated `summary`.

Usage

```
as_gtwr(object)
```

Arguments

object	a model of class mgwrsar-class fitted with Type = 'GDT', typically search_bandwidths(...)$best_mo

Details

The model is checked against the HWB2010 contract: Type = 'GDT', a gaussian kernel on both axes, fixed (non-adaptive) bandwidths and a purely multiplicative kernel (alpha = 1). Anything else is rejected, because the separability that Huang's equation 12 relies on no longer holds.

Value

An object of class gtwr-class. The w_tail slot is NA for a coerced model: the kNN screening diagnostic needs the distance matrices, which the fitted object does not carry. Refit with gtwr_HWB2010 if you need it.

See Also

gtwr_HWB2010, search_bandwidths, bw_gdt2hwb

Examples

```
library(mgwrsar)
simu <- simu_multiscale(n = 1000, myseed = 1, type = 'GG2024',
                        config_beta = 'spatiotemp_old', config_snr = 0.9)
dat <- simu$mydata

res <- search_bandwidths(
  formula = as.formula('Y ~ X1 + X2 + X3'), data = dat,
  coords = simu$coords, kernels = c('gauss', 'gauss'), Model = 'GWR',
  control = list(Z = dat$time, criterion = 'AICc', Type = 'GDT',
                 adaptive = c(FALSE, FALSE), alpha = 1),
  hs_range = c(0, 1.4), ht_range = c(0, 1),
  n_seq = 12, n_rounds = 3, refine = FALSE)

m <- as_gtwr(res$best_model)
summary(m)
bw_gdt2hwb(res$minimum[1], res$minimum[2])
```

atds_gwr *atds_gwr Top-Down Scaling approach of GWR*

Description

This function performs a Geographically Weighted Regression (GWR) using a top-down scaling approach, adjusting GWR coefficients with a progressively decreasing bandwidth as long as the AICc criterion improves.

Usage

```
atds_gwr(formula,data,coords,kernels='triangle',fixed_vars=NULL,
control_tds=list(nns=30),control=list(adaptive=TRUE,verbose=FALSE))
```

Arguments

formula	a formula.
data	a dataframe.
coords	default NULL, a dataframe or a matrix with coordinates
kernels	A vector containing the kernel types. Possible types: triangle ("triangle"), bisquare ("bisq"), tricube ("tcub"), epanechnikov ("epane").
fixed_vars	a vector with the names of spatiallay constant coefficient for mixed model. All other variables present in formula are supposed to be spatially varying. If empty or NULL (default), all variables in formula are supposed to be spatially varying.
control_tds	list of extra control arguments for TDS_MGWR model - see TDS_MGWR Help
control	list of extra control arguments for MGWRSAR wrapper - see MGWRSAR Help

See Also

TDS_MGWR, gwr_multiscale, MGWRSAR, golden_search_bandwidth.

bw_gdt2hwb *Convert an mgwrsar (h_S, h_T) pair back into Huang's (h_ST, tau)*

Description

Inverse of bw_hwb2gdt: reads the optimum returned by search_bandwidths in the parameterisation of Huang et al. (2010), with $h_{ST} = \sqrt{2}h_S$ and $\tau = (h_S/h_T)^2$.

Usage

```
bw_gdt2hwb(h_s, h_t)
```

Arguments

h_s	numeric scalar, spatial bandwidth on the mgwrsar scale.
h_t	numeric scalar, temporal bandwidth on the mgwrsar scale.

Value

A numeric vector `c(h_st, tau)`.

See Also

bw_hwb2gdt, as_gtwr, search_bandwidths

Examples

```
bw_gdt2hwb(h_s = 0.06, h_t = 0.4)         # -> c(0.0849, 0.0225)
```

---

bw_hwb2gdt *Convert Huang's (h_ST, tau) into the mgwrsar (h_S, h_T) pair*

Description

Huang et al. (2010) weight observation j around i with $\alpha_{ij} = \exp\{-[d_S^2 + \tau d_T^2]/h_{ST}^2\}$ (their equations 11-12, with the scale convention $\lambda = 1$, $\tau = \mu/\lambda$). The gaussian kernel of mgwrsar is $\exp\{-\frac{1}{2}(d/h)^2\}$, hence the factor $\sqrt{2}$. Matching the two expressions term by term gives $h_S = h_{ST}/\sqrt{2}$ and $h_T = h_S/\sqrt{\tau}$.

Usage

```
bw_hwb2gdt(h_st, tau)
```

Arguments

h_st	numeric scalar, Huang's single spatio-temporal bandwidth.
tau	numeric scalar in `[0, Inf]`, Huang's ratio μ/λ.

Details

The two degenerate cases are the special cases of Huang's equation 10 rather than limits of the τ parameterisation: `tau = 0` is $\mu = 0$ (pure GWR, flat temporal kernel) and `tau = Inf` is $\lambda = 0$ (pure TWR, flat spatial kernel), in which case `h_st` is read on the temporal axis.

Use it to turn a (h_{ST}, τ) pair into the `hs_range` and `ht_range` that `search_bandwidths` expects, and `bw_gdt2hwb` to read its result back.

Value

A numeric vector `c(h_S, h_T)`.

See Also

bw_gdt2hwb, gtwr_HWB2010, search_bandwidths

Examples

```
bw_hwb2gdt(h_st = 0.0849, tau = 0.0225)   # -> c(0.06, 0.4)
```

`coef,mgwrsar-method` *coef for mgwrsar model*

Description

coef for mgwrsar model

Usage

```
## S4 method for signature 'mgwrsar'
coef(object, ...)
```

Arguments

`object`	A model of class `mgwrsar-class`.
`...`	coef parameters forwarded.

Value

A named list with a matrix of varying coefficients and a vector or non varying coefficients.

`find_TP` *Search of a suitable set of target points. find_TP is a wrapper function that identifies a set of target points based on spatial smoothed OLS residuals.*

Description

Search of a suitable set of target points. find_TP is a wrapper function that identifies a set of target points based on spatial smoothed OLS residuals.

Usage

```
find_TP(formula, data,coords,kt,ks=16,Wtp=NULL,type='residuals',
model_residuals=NULL,verbose=0,prev_TP=NULL,nTP=NULL)
```

Arguments

formula	a formula
data	a dataframe or a spatial dataframe (SP package)
coords	a dataframe or a matrix with coordinates, not required if data is a spatial dataframe
kt	the minimum number of first neighbors with lower (resp.higer) absolute value of the smoothed residuals.
ks	the number of first neighbors for computing the smoothed residuals, default 16.
Wtp	a precomputed matrix of weights, default NULL.
type	method for choosing TP, could be 'residuals', ' kdtree','random', default 'residuals'
model_residuals	(optional) a vector of residuals.
verbose	verbose mode, default FALSE.
prev_TP	index of already used TP (version length(kt)>1), default NULL.
nTP	numbeer of target points for random choice of target points, default NULL.

Details

find_TP is a wrapper function that identifies a set of target points, based on spatial smoothed residuals by default. If no vector of residuals are provided, OLS residuals are computed. The function first computes the smooth of model residuals using a Shepard's kernel with ks neighbors (default 16). Then it identifies local maxima (resp. minima) that fits the requirement of having at least kt neighbors with lower (resp.higer) absolute value of the smoothed residuals. As kt increases the number of target points decreases.

Value

find_TP returns an index vector of Target Points set.

Examples

```
library(mgwrsar)
## loading data example
data(mydata)
coords=as.matrix(mydata[,c("x","y")])
TP=find_TP(formula = 'Y_gwr~X1+X2+X3', data =mydata,coords=coords,kt=6,
type='residuals')
# only 60 targets points are used
length(TP)

model_GWR_tp<-MGWRSAR(formula = 'Y_gwr~X1+X2+X3', data = mydata,
coords=coords, fixed_vars=NULL,kernels=c('gauss'),  H=0.03, Model = 'GWR',
control=list(SE=TRUE,TP=TP,ks=12))
summary(model_GWR_tp@Betav)
```

`fitted,mgwrsar-method` *fitted for mgwrsar model*

Description

fitted for mgwrsar model

Usage

```
## S4 method for signature 'mgwrsar'
fitted(object, ...)
```

Arguments

`object`	A model of class `mgwrsar-class`.
`...`	fitted parameters forwarded.

Value

A vector of fitted values.

`golden_search_2d_bandwidth` *Optimization of 2D Bandwidths (Spatial and Temporal) using Golden Section Search*

Description

This function optimizes spatial and temporal bandwidths simultaneously using a 2D Golden Section Search approach. It is typically used internally by `search_bandwidths` when `refine = TRUE` for spatio-temporal models (Type 'GDT').

Usage

```
golden_search_2d_bandwidth(formula, data, coords, fixed_vars, kernels, Model,
                                  control, lower.bound.space, upper.bound.space,
                                   lower.bound.time, upper.bound.time,
                                   tolerance_s = 1e-06, tolerance_t = 1e-06,
                                   max_iter = 10)
```

Arguments

formula	A formula object.
data	A data frame containing the variables.
coords	A matrix of coordinates (spatial indices).
fixed_vars	Vector of names of variables with fixed coefficients.
kernels	Vector of kernel types (e.g., c('gauss', 'gauss')).
Model	Character string specifying the model type (e.g., 'GWR').
control	List of control parameters.
lower.bound.space	Numeric. Lower bound for the spatial bandwidth search.
upper.bound.space	Numeric. Upper bound for the spatial bandwidth search.
lower.bound.time	Numeric. Lower bound for the temporal bandwidth search.
upper.bound.time	Numeric. Upper bound for the temporal bandwidth search.
tolerance_s	Numeric. Convergence tolerance for the spatial dimension. Default is 1e-6.
tolerance_t	Numeric. Convergence tolerance for the temporal dimension. Default is 1e-6.
max_iter	Integer. Maximum number of iterations for the optimization loop. Default is 10.

Value

A list containing:

minimum	Vector of optimized bandwidths c(h_spatial, h_temporal).
objective	Final AICc value.
model	The fitted mgwrsar model object.

See Also

golden_search_bandwidth, search_bandwidths

---

golden_search_bandwidth

Golden search bandwidth (deprecated)

Description

'golden_search_bandwidth()' is deprecated. Please use [search_bandwidths()], which provides a unified interface for spatial and spatio-temporal bandwidth search and can run golden-search refinement internally.

This wrapper preserves backward compatibility by translating the historical arguments ('lower.bound', 'upper.bound', 'Ht') to the new interface.

Usage

```
golden_search_bandwidth(
  formula,
  Ht = NULL,
  data,
  coords,
  fixed_vars = NULL,
  kernels,
  Model = "GWR",
  control,
  lower.bound,
  upper.bound,
  tolerance = 1e-06,
  ncore = 1,
  show_progress = TRUE
)
```

Arguments

formula	A formula object specifying the model (e.g., y ~ x1 + x2).
Ht	Optional temporal bandwidth (used only for legacy calls). If not 'NULL', it is mapped to 'ht_range = c(Ht, Ht)' (i.e., fixed temporal bandwidth) when 'n_rounds = 0'.
data	A data frame containing the variables in the model.
coords	A matrix or data frame of coordinates (2 columns for spatial, 1 for temporal, or more for GDT).
fixed_vars	A character vector indicating the names of variables with spatially stationary (fixed) coefficients. Default is NULL (all coefficients are varying).
kernels	A character vector specifying the kernel types for spatial and temporal components (e.g., c("gauss", "gauss")).
Model	A character string specifying the model type. Options include "GWR", "MGWR", "OLS", "SAR", etc. Default is "GWR".
control	A named list of extra control arguments passed to the MGWRSAR function (e.g., adaptive, NN, Z).
lower.bound, upper.bound	Numeric bounds for the spatial bandwidth search; mapped to 'hs_range'.
tolerance	Numeric tolerance; mapped to 'tol' when relevant.
ncore	Number of cores to use.
show_progress	Logical; forwarded to [search_bandwidths()].

Value

A list returned by [search_bandwidths()] (see its Value section).

gtwr-class *Class "gtwr"*

Description

S4 class for the GTWR model of Huang, Wu and Barry (2010), produced by `gtwr_HWB2010`. It extends `mgwrsar-class` and inherits all of its methods; the extra slots only record the model in Huang's own parameterisation, so that a table produced from this object speaks the language of the paper rather than the internal `(h_S, h_T)` pair.

Slots

`h_st` numeric. Huang's single spatio-temporal bandwidth.

`tau` numeric. Huang's scale ratio mu/lambda, with lambda = 1.

`causal` logical. FALSE for the symmetric temporal kernel of Huang et al. (2010), TRUE for the past-only variant.

`w_tail` numeric. Largest kernel weight still carried by the outer ring of the kNN pre-selection; near zero means the screening does not truncate the gaussian tail.

See Also

gtwr_HWB2010

gtwr_HWB2010 *Geographically and Temporally Weighted Regression (Huang, Wu and Barry 2010)*

Description

Calibrates the GTWR model of Huang, Wu and Barry (2010) in its own parameterisation: a single spatio-temporal bandwidth `h_st` and a scale ratio `tau`. The estimator is the gaussian, fixed-bandwidth special case of the space-time kernel of `MGWRSAR` (`Type = 'GDT'`), reached through the exact change of variables $h_S = h_{ST}/\sqrt{2}$, $h_T = h_S/\sqrt{\tau}$.

Usage

```
gtwr_HWB2010(
  formula,
  data,
  coords,
  time,
  h_st,
  tau,
  fixed_vars = NULL,
  control = list()
)
```

Arguments

formula	a formula.
data	a data.frame.
coords	a n x 2 matrix or data.frame of spatial coordinates. The time index is passed through time, never as a third column.
time	a numeric vector of length n holding the time index. Objects of class Date, POSIXct or POSIXlt are accepted only when control$t.units is given (any unit understood by difftime); they are then converted to elapsed time from the earliest observation, expressed in that unit.
h_st	the spatio-temporal bandwidth of Huang et al. (2010), on the scale of the combined distance d_{ST}.
tau	the ratio μ/λ of Huang et al. (2010), with the convention $\lambda = 1$. tau = 0 gives a plain GWR and tau = Inf a plain TWR.
fixed_vars	a character vector of covariates with a constant coefficient. NULL (default) gives the model of Huang et al. (2010); anything else gives a mixed GTWR, which is an extension of this package and not part of HWB2010.
control	list of extra control arguments forwarded to MGWRSAR, plus the keys described below.

Details

Three control keys are consumed by this function and are not forwarded:

causal Logical, default FALSE. FALSE uses a symmetric temporal kernel, which is the model of Huang et al. (2010). TRUE switches to a past-only temporal kernel ('gauss_past'), an extension in the spirit of Wu et al. (2014), not HWB2010.

tail_tol Numeric, default 0.01. Threshold above which a warning reports that the kNN pre-selection may be truncating the tail of the gaussian kernel. Only checked when control$NN < n.

t.units Character, required when time is a date or time object, ignored otherwise.

Setting control$NN below n activates the kNN pre-selection of prep_d, which ranks neighbours in a joint space-time metric standardised by sample standard deviations. That ranking does not depend on tau, so with screening on the retained neighbourhood is not the one implied by the calibrated tau: in particular tau = 0 then no longer reproduces a plain GWR exactly. Since the gaussian kernel has no compact support, NN = n (the default) is the faithful setting, and w_tail is reported so that any screening can be checked.

The keys Type, kernels, adaptive, alpha and Z are set by this function and supplying them raises an error. In particular the kernel is locked to gaussian, because the separability $W^{ST} = W^S \odot W^T$ that Huang's equation 12 relies on holds for no other kernel, and adaptive bandwidths are refused because mgwrsar adapts each dimension separately whereas Huang ranks neighbours in the joint space-time metric. Use MGWRSAR(Type = 'GDT') directly for those variants.

Bandwidths are not calibrated here. Use search_bandwidths, the calibration entry point of the package for Type = 'GDT', with hs_range/ht_range obtained from bw_hwb2gdt, then read its optimum back with bw_gdt2hwb or turn its best_model into a gtwr with as_gtwr.

Value

An object of class gtwr-class, which extends mgwrsar-class and therefore supports summary, coef, fitted, residuals, plot and predict. Out-of-sample prediction requires method_pred in c('model','tWtp_model') and newdata_coords with columns (x, y, t).

References

Huang, B., Wu, B. and Barry, M. (2010). Geographically and temporally weighted regression for modeling spatio-temporal variation in house prices. International Journal of Geographical Information Science, 24(3), 383-401.

Wu, B., Li, R. and Huang, B. (2014). A geographically and temporally weighted autoregressive model with application to housing prices. International Journal of Geographical Information Science, 28(6), 1186-1204.

See Also

MGWRSAR, search_bandwidths, as_gtwr, bw_hwb2gdt, predict.mgwrsar

Examples

```
library(mgwrsar)
data(mydata)
coords <- as.matrix(mydata[, c("x", "y")])
mydata$time <- rep(1:10, length.out = nrow(mydata))

## coords are in metres here, hence a bandwidth of a few kilometres;
## tau rescales the time axis so that one period weighs like h_st/sqrt(tau)
m <- gtwr_HWB2010(formula = 'Y_gwr ~ X1 + X2 + X3', data = mydata,
                  coords = coords, time = mydata$time,
                  h_st = 4000, tau = 1e6,
                  control = list(SE = TRUE))
summary(m)

## out-of-sample prediction on (x, y, t)
newc <- cbind(coords[1:10, ], mydata$time[1:10])
predict(m, newdata = mydata[1:10, ], newdata_coords = newc,
        method_pred = 'model')
```

---

kernel_matW *kernel_matW A function that returns a sparse weight matrix based computed with a specified kernel (gauss,bisq,tcub,epane,rectangle,triangle) considering coordinates provides in S and a given bandwidth. If NN<nrow(S) only NN firts neighbours are considered. If Type!='GD' then S should have additional columns and several kernels and bandwidths should be be specified by the user.*

Description

kernel_matW A function that returns a sparse weight matrix based computed with a specified kernel (gauss,bisq,tcub,epane,rectangle,triangle) considering coordinates provides in S and a given bandwidth. If NN<nrow(S) only NN firts neighbours are considered. If Type!='GD' then S should have additional columns and several kernels and bandwidths should be be specified by the user.

Usage

```
kernel_matW(H,kernels,coords,NN,TP=NULL,Type='GD',adaptive=FALSE,
diagnull=TRUE,alpha=1,dists=NULL,indexG=NULL,extrapol=FALSE,QP=NULL,K=0)
```

Arguments

H	A vector of bandwidths
kernels	A vector of kernel types
coords	A matrix with variables used in kernel (reference)
NN	Number of spatial Neighbours for kernels computations
TP	A vector with index of target points
Type	Type of Genelarized kernel product ('GD' only spatial,'GDC' spatial + a categorical variable,'GDX' spatial + a continuous variable, 'GDT' spatial + a time index, and other combinations 'GDXXC','GDTX',...)
adaptive	A vector of boolean to choose adaptive version for each kernel
diagnull	Zero on diagonal, default FALSE
alpha	Numeric exponent for the generalized kernel product, default 1.
dists	A precomputed list of distance matrices (output of prep_d), default NULL.
indexG	A precomputed matrix of neighbor indices (output of prep_d), default NULL.
extrapol	Logical. If TRUE, compute weights for extrapolation (out-of-sample), default FALSE.
QP	A matrix with variables used in kernel (neighbors), default NULL (if NULL coord_j=coord_i).
K	Integer, number of folds for block computation, default 0 (no blocking).

Value

A sparse Matrix of weights (dgCMatrix).

Examples

```
library(mgwrsar)
## loading data example
data(mydata)
coords=as.matrix(mydata[,c("x","y")])
## Creating a spatial weight matrix (sparce dgCMatrix) of 4 nearest neighbors with 0 in diagonal
W=kernel_matW(H=4,kernels='rectangle',coords=coords,NN=4,adaptive=TRUE,diagnull=TRUE)
```

`make_unique_by_structure` *Ensure Uniqueness of Coordinates or Values in 1D, 2D, or 3D Structures*

Description

This function enforces uniqueness in numeric data structures of dimension one, two, or three by applying minimal perturbations to duplicated values. It is primarily intended to avoid degeneracy issues (e.g., duplicated coordinates or time stamps) in spatial and spatio-temporal modeling.

Usage

```
make_unique_by_structure(M)
```

Arguments

`M`	A numeric vector, matrix, or data frame with 1, 2, or 3 columns. For matrices or data frames, columns are interpreted as spatial (`x`, `y`) and optional temporal (`t`) coordinates.

Details

Depending on the dimensionality of the input, uniqueness is enforced as follows:

- **1D**: duplicated values are slightly perturbed to ensure uniqueness;
- **2D**: duplicated (x, y) coordinate pairs are resolved by perturbing the second coordinate;
- **3D**: duplicated (x, y) pairs and duplicated temporal coordinates are handled separately, ensuring uniqueness in both space and time.

The perturbation magnitude is proportional to machine precision and the scale of the input data, ensuring numerical stability while preserving the original structure as closely as possible.

This function is designed as a low-level utility for preprocessing spatial or spatio-temporal inputs prior to kernel-based local regression or distance computations, where duplicated locations or timestamps may cause numerical singularities.

Value

If `M` is a vector, a numeric vector of the same length is returned. If `M` is a matrix or data frame, a numeric matrix with the same dimensions is returned, with standardized column names: `"t"` (1D), `"x"`, `"y"` (2D), or `"x"`, `"y"`, `"t"` (3D).

MGWRSAR *Estimation of linear and local linear model with spatial autocorrelation model (mgwrsar).*

Description

MGWRSAR is is a wrapper function for estimating linear and local linear models with spatial autocorrelation (SAR models with spatially varying coefficients).

Usage

```
MGWRSAR(formula, data, coords, fixed_vars = NULL, kernels, H,
Model = "GWR", control = list())
```

Arguments

formula	a formula.
data	a dataframe or a spatial dataframe (sp package).
coords	default NULL, a dataframe or a matrix with coordinates, not required if data is a spatial dataframe.
fixed_vars	a vector with the names of spatiallay constant coefficient for mixed model. All other variables present in formula are supposed to be spatially varying. If empty or NULL (default), all variables in formula are supposed to be spatially varying.
kernels	A vector containing the kernel types. Possible types: rectangle ("rectangle"), bisquare ("bisq"), tricube ("tcub"), epanechnikov ("epane"), gaussian ("gauss")) .
H	vector containing the bandwidth parameters for the kernel functions.
Model	character containing the type of model: Possible values are "OLS", "SAR", "GWR" (default), "MGWR" , "MGWRSAR_0_0_kv","MGWRSAR_1_0_kv", "MGWRSAR_0_kc_kv", "MGWRSAR_1_kc_kv", "MGWRSAR_1_kc_0". See Details for more explanation.
control	list of extra control arguments for MGWRSAR wrapper - see Details below

Details

Z A matrix of variables for genralized kernel product, default NULL.

W A row-standardized spatial weight matrix for Spatial Aurocorrelation, default NULL.

Type Verbose mode, default FALSE.

adaptive A vector of boolean to choose adaptive version for each kernel.

kernel_w The type of kernel for computing W, default NULL.

h_w The bandwidth value for computing W, default 0.

Method Estimation method for computing the models with Spatial Dependence. '2SLS' or 'B2SLS', default '2SLS'.

TP Avector of target points, default NULL.

ncore Number of CPU core for parallel computation, default 1

isgcv If TRUE, compute a LOOCV criteria, default FALSE.

verbose Verbose mode, default FALSE.

Value

MGWRSAR returns an object of class mgwrsar with at least the following components:

Betav matrix of coefficients of dim(n,kv) x kv.

Betac vector of coefficients of length kc.

Model The sum of square residuals.

Y The dependent variable.

XC The explanatory variables with constant coefficients.

XV The explanatory variables with varying coefficients.

X The explanatory variables.

W The spatial weight matrix for spatial dependence.

isgcv if gcv has been computed.

edf The estimated degrees of freedom.

formula The formula.

data The dataframe used for computation.

Method The type of model.

coords The spatial coordinates of observations.

H The bandwidth vector.

fixed_vars The names of constant coefficients.

kernels The kernel vector.

SSR The sum of square residuals.

residuals The vector of residuals.

fit the vector of fitted values.

sev local standard error of parameters: the variance of each local weighted fit, given the local residual variance.

get_ts Boolean, if trace of hat matrix Tr(S) should be stored.

NN Maximum number of neighbors for weights computation

MGWRSAR is is a wrapper function for estimating linear and local linear model with spatial autocorrelation that allows to estimate the following models : $y = \beta_c X_c + \epsilon_i$ (OLS)

$y = \beta_v(u_i, v_i) X_v + \epsilon_i$ (GWR)

$y = \beta_c X_c + \beta_v(u_i, v_i) X_v + \epsilon_i$ (MGWR)

$y = \lambda W y + \beta_c X_c + \epsilon_i$ (MGWR-SAR(0,k,0))

$y = \lambda W y + \beta_v(u_i, v_i) X_v + \epsilon_i$ (MGWR-SAR(0,0,k))

$y = \lambda Wy + \beta_c X_c + \beta_v(u_i, v_i)X_v + \epsilon_i$ (MGWR-SAR(0,k_c,k_v))

$y = \lambda(u_i, v_i)Wy + \beta_c X_c + \epsilon_i$ (MGWR-SAR(1,k,0))

$y = \lambda(u_i, v_i)Wy + \beta_v(u_i, v_i)X_v + \epsilon_i$ (MGWR-SAR(1,0,k))

$y = \lambda(u_i, v_i)Wy + \beta_c X_c + \beta_v(u_i, v_i)X_v + \epsilon_i$ (MGWR-SAR(1,k_c,k_v))

When model imply spatial autocorrelation, a row normalized spatial weight matrix must be provided. 2SLS and Best 2SLS method can be used. When model imply local regression, a bandwidth and a kernel type must be provided. Optimal bandwidth can be estimated using goldens_search_bandwid function. When model imply mixed local regression, the names of stationary covariates must be provided.

#' In addition to the ability of considering spatial autocorrelation in GWR/MGWR like models, MGWRSAR function introduces several useful technics for estimating local regression with space coordinates:

- it uses RCCP and RCCPeigen code that speed up computation and allows parallel computings;
- it allows to drop out variables with not enough local variance in local regression, which allows to consider dummies in GWR/MGWR framework without trouble.
- it allows to drop out local outliers in local regression.
- it allows to consider additional variable for kernel, including time (asymetric kernel) and categorical variables (see Li and Racine 2010). Experimental version.

References

Geniaux, G. and Martinetti, D. (2017). A new method for dealing simultaneously with spatial autocorrelation and spatial heterogeneity in regression models. Regional Science and Urban Economics. (https://doi.org/10.1016/j.regsciurbeco.2017.04.001)

McMillen, D. and Soppelsa, M. E. (2015). A conditionally parametric probit model of microdata land use in chicago. Journal of Regional Science, 55(3):391-415.

Loader, C. (1999). Local regression and likelihood, volume 47. springer New York.

Franke, R. and Nielson, G. (1980). Smooth interpolation of large sets of scattered data. International journal for numerical methods in engineering, 15(11):1691-1704.

See Also

golden_search_bandwidth, summary, plot, predict, kernel_matW

Examples

```
library(mgwrsar)
## loading data example
data(mydata)
coords=as.matrix(mydata[,c("x","y")])
## Creating a spatial weight matrix (sparce dgCMatrix)
## of 4 nearest neighbors with 0 in diagonal
W=kernel_matW(H=4,kernels='rectangle',coords=coords,NN=4,adaptive=TRUE,
diagnull=TRUE)
mgwrsar_0_kc_kv<-MGWRSAR(formula = 'Y_mgwrsar_0_kc_kv~X1+X2+X3', data = mydata,
coords=coords, fixed_vars='X2',kernels=c('gauss'),H=20, Model = 'MGWRSAR_0_kc_kv',
```

```
control=list(SE=FALSE,adaptive=TRUE,W=W,ncore=1))
summary(mgwrsar_0_kc_kv)
```

`mgwrsar-class` *Class of mgwrsar Model.*

Description

A S4 class to represent the results of MGWRSAR, TDS-MGWR and related spatial models.

Slots

`Betav` matrix. The estimated varying coefficients, dimension (n, kv).

`Betac` numeric. The estimated constant coefficients, length kc.

`Model` character. The type of model (e.g., "GWR", "MGWR", "SAR", "tds_mgwr").

`fixed_vars` character. A vector with names of spatially constant covariates.

`Y` numeric. The dependent variable.

`XC` matrix. The explanatory variables with constant coefficients.

`XV` matrix. The explanatory variables with varying coefficients.

`X` matrix. All explanatory variables.

`W` Matrix. The spatial weight matrix for spatial dependence (row-standardized).

`isgcv` logical. Indicates if Leave-One-Out Cross-Validation (LOOCV) has been computed.

`edf` numeric. The estimated effective degrees of freedom.

`edf_k` numeric. The estimated effective degrees of freedom of univariate models.

`formula` formula. The model formula.

`data` data.frame. The dataframe used for computation.

`Method` character. The estimation technique for spatial dependence models ('2SLS' or 'B2SLS'). Default is '2SLS'.

`coords` matrix. The spatial coordinates of observations.

`H` numeric. The bandwidth vector (spatial).

`Ht` numeric. The bandwidth vector (temporal), if applicable.

`kernels` character. The kernel type(s) used (e.g., 'gauss', 'bisq').

`adaptive` logical. Indicates if an adaptive kernel (nearest neighbors) was used.

`Type` character. The type of Generalized Kernel Product ('GD' for spatial, 'GDT' for spatio-temporal).

`TP` numeric. Indices of target points (if a subset was used).

`SSRtp` numeric. The residual sum of squares calculated only on target points.

`SSR` numeric. The total residual sum of squares.

`residuals` numeric. The vector of residuals.

`fit` numeric. The vector of fitted values.

`pred` numeric. The vector of predicted values (out-of-sample).

`sev` matrix. Local standard errors of varying coefficients: the variance of each local weighted fit, given the local residual variance (conditional on the fixed coefficients in a mixed model), or, for `TDS_MGWR()` with `get_AIC = TRUE`, the backfitting variance of Yu et al. (2020).

`se` numeric. Standard errors of constant coefficients.

`tS` numeric. The trace of the Hat matrix (effective number of parameters).

`Shat` matrix. The Hat matrix (or approximation).

`R_k` list. List of partial Hat matrices by covariate (for MGWR inference).

`h_w` numeric. The bandwidth value used for computing the spatial weight matrix W. Default is 0.

`kernel_w` character. The kernel type used for computing W. Default is NULL.

`RMSE` numeric. Root Mean Square Error (on training data).

`RMSEtp` numeric. Root Mean Square Error computed on target points.

`CV` numeric. Leave-One-Out Cross-Validation score.

`AIC` numeric. Akaike Information Criterion.

`AICc` numeric. Corrected Akaike Information Criterion.

`AICctp` numeric. Corrected AIC for target points.

`BIC` numeric. Bayesian Information Criterion.

`R2` numeric. R-squared.

`R2_adj` numeric. Adjusted R-squared.

`get_ts` logical. Indicates if the trace of the Hat matrix (Tr(S)) was stored.

`NN` numeric. The maximum number of neighbors used for weight computation (truncation parameter).

`ncore` numeric. Number of CPU cores used.

`mycall` call. The original function call.

`ctime` numeric. Computation time in seconds.

`HRMSE` matrix. History of RMSE values (for iterative algorithms like backfitting).

`HBETA` list. History of estimated Beta coefficients at each iteration.

`loglik` numeric. Log-likelihood value.

`G` list. List containing neighboring indices and distances (knn object).

`V` numeric. Sequence of spatial bandwidths tested (for TDS algorithms).

`Vt` numeric. Sequence of temporal bandwidths tested (for TDS algorithms).

`max_dist` numeric. maximum distance possible with this kernel.

`max_dist_t` numeric. maximum temporal distance with this kernel.

`Z` numeric. Temporal or auxiliary variable for GDT kernel type.

`TS` numeric. Diagonal elements of the Hat Matrix.

`alpha` numeric. Ratio parameter for GDT kernels (balancing space and time).

`HM` matrix. Matrix of optimal bandwidths per covariate (for TDS).

`HKmin` numeric. Minimum allowed bandwidth per covariate (for TDS).

`HKMIN` list. List of minimum bandwidths per covariate for spatio-temporal models (TDS).

`saturation` list. TDS calibration: the saturation filter of the candidate fits (`ratio`, number of candidates `excluded`, of component `updates` with an exclusion, and of updates with every candidate excluded, `all_excluded`).

`isolated_idx` numeric. Indices of observations without sufficient neighbors.

`my_crs` ANY. Coordinate Reference System (CRS) information.

`mgwrsar_bootstrap_test` *A bootstrap test for Betas for mgwrsar class model.*

Description

A bootstrap test for Betas for mgwrsar class model.

Usage

```
mgwrsar_bootstrap_test(x0,x1,B=100,ncore=1,type='standard',eps='H1',
df='H1',focal='median',D=NULL,verbose=FALSE)
```

Arguments

`x0`	The H0 mgwrsar model
`x1`	The H1 mgwrsar model
`B`	number of bootstrap repetitions, default 100
`ncore`	number of cores
`type`	type of bootstap : 'wild','Rademacher','spatial' or 'standard' (default)
`eps`	Hypothesis under wich residuals are simulated, 'H0' or 'H1' (default)
`df`	Hypothesis under wich degree of freedom is estimated.
`focal`	see sample_stat help
`D`	A matrix of distance
`verbose`	default FAlSE

Value

The value of the statistics test and a p ratio.

See Also

mgwrsar_bootstrap_test_all

mgwrsar_bootstrap_test_all *A bootstrap test for testing nullity of all Betas for mgwrsar class model,*

Description

A bootstrap test for testing nullity of all Betas for mgwrsar class model,

Usage

```
mgwrsar_bootstrap_test_all(model,B=100,ncore=1,type='standard')
```

Arguments

model	A mgwrsar model
B	number of bootstrap replications, default 100
ncore	number of cores for parallelization.
type	type of boostrap ('spatial','wild','random')

Value

a matrix with statistical test values and p ratios

See Also

mgwrsar_bootstrap_test

multiscale_gwr *Multiscale Geographically Weighted Regression (MGWR)*

Description

This function estimates a Multiscale Geographically Weighted Regression (MGWR) model based on the proposition of Fotheringham et al. (2017). Unlike standard GWR where a single bandwidth is used for all covariates, MGWR allows for covariate-specific bandwidths. It uses a backfitting algorithm to iteratively estimate the optimal bandwidth and coefficients for each explanatory variable.

Usage

```
multiscale_gwr(formula, data, coords, kernels = 'bisq',
                      control_mgwr = list(), control = list())
```

Arguments

formula	A formula object specifying the model (e.g., y ~ x1 + x2).
data	A data frame containing the variables in the model.
coords	A matrix or data frame of coordinates (2 columns for spatial, 3 for spatio-temporal).
kernels	A character string specifying the kernel type. Options include 'bisq' (default), 'gauss', 'triangle', 'tricube', 'rectangle'.
control_mgwr	A named list of control parameters specific to the MGWR backfitting algorithm. See 'Details' for available components.
control	A named list of standard control arguments passed to the internal GWR estimation steps. See 'Details' for available components.

Details

Components for control_mgwr**:**

init Character. The type of model used for initialization. Options are 'GWR' (default) or 'lm' (OLS).

maxiter Integer. Maximum number of backfitting iterations. Default is 20.

tolerance Numeric. Convergence threshold based on the change in RMSE or bandwidths. Default is 1e-6.

nstable Integer. Number of consecutive iterations where bandwidths must remain stable to declare convergence. Default is 6.

H0 Numeric vector. Optional initial bandwidths for each covariate. If NULL, they are initialized via GWR or global search.

get_AIC Logical. If TRUE, calculates the corrected Akaike Information Criterion (AICc) at the end. Default is FALSE.

verbose Logical. If TRUE, prints progress information during backfitting. Default is FALSE.

Components for control**:**

adaptive Logical. If TRUE (default), uses an adaptive bandwidth (k-nearest neighbors). If FALSE, uses a fixed distance bandwidth.

Type Character. The type of spatial weighting. 'GD' (Geographical Distance, default) or 'GDT' (Geo-Temporal).

NN Integer. Maximum number of neighbors for matrix truncation (speeds up computation). Default is nrow(data).

ncore Integer. Number of cores to use for parallel computation.

isgcv Logical. If TRUE, computes Leave-One-Out Cross-Validation scores. Default is FALSE.

Value

An object of class `mgwrsar` containing:

`Betav`	Matrix of estimated spatially varying coefficients.
`H`	Vector of final optimal bandwidths for each covariate.
`RMSE`	Root Mean Square Error of the final model.
`residuals`	Vector of residuals.
`fitted.values`	Vector of fitted values.
`AICc`	Corrected AIC (if `get_AIC = TRUE`).
`R2`	R-squared of the model.

References

Fotheringham, A. S., Yang, W., & Kang, W. (2017). Multiscale geographically weighted regression (MGWR). *Annals of the American Association of Geographers*, 107(6), 1247-1265.

See Also

`MGWRSAR`, `TDS_MGWR`, `golden_search_bandwidth`

`mydata` *mydata is a simulated data set of a mgwrsar model*

Description

mydata is a simulated data set of a mgwrsar model

Format

A data frames with 1000 rows 22 variables and a matrix of coordinates with two columns

Author(s)

Ghislain Geniaux and Davide Martinetti <ghislain.geniaux@inrae.fr>

References

doi:10.1016/j.regsciurbeco.2017.04.001

mydatasf — *mydataf is a Simple Feature object with real estate data in south of France.*

Description

mydataf is a Simple Feature object with real estate data in south of France.

Format

A sf object with 1403 rows, 5 columns

Author(s)

Ghislain Geniaux <ghislain.geniaux@inrea.fr>

References

https://www.data.gouv.fr/fr/datasets/demandes-de-valeurs-foncieres/

normW — *Row Normalization of Sparse Matrix*

Description

Row-normalizes a sparse matrix (dgCMatrix) or dense matrix so that each row sums to 1. Rows with zero sums are left unchanged.

Usage

```
normW(x)
```

Arguments

x	A matrix or dgCMatrix class sparse matrix.

Value

A row-normalized matrix of the same class as the input.

plot,mgwrsar,missing-method
Plot method for mgwrsar model

Description

Maps the results of an mgwrsar model using an interactive Plotly map. Supports plotting coefficients, t-statistics, residuals, and fitted values. Automatically switches between a geographic map (if CRS is present) and a scatter plot.

Visualizes the results of an MGWRSAR model (coefficients, t-statistics, residuals, or fitted values) using interactive plots via the `plotly` package.

If a Coordinate Reference System (CRS) is provided (either via the `crs` argument or stored in the model object), the function generates an interactive map. Otherwise, it generates a standard scatter plot of the values against the coordinates.

Usage

```
## S4 method for signature 'mgwrsar,missing'
plot(
  x,
  y,
  type = "coef",
  var = NULL,
  crs = NULL,
  mypalette = "RdYlGn",
  opacity = 0.8,
  size = 5,
  title = NULL,
  show_legend = TRUE,
  n_time_steps = 10,
  ...
)

## S3 method for class 'mgwrsar'
plot(
  x,
  type = "coef",
  var = NULL,
  crs = NULL,
  mypalette = "RdYlGn",
  size = 5,
  opacity = 0.8,
  title = NULL,
  show_legend = TRUE,
  n_time_steps = 10,
  ...
)
```

Arguments

x	An object of class mgwrsar.
y	missing (not used).
type	Character. The type of output to plot. Options are: • 'coef': Spatially varying coefficients (default). • 't_coef': t-statistics of the coefficients (visualized with a significance threshold of 1.96). • 'residuals': Model residuals. • 'fitted': Fitted values.
var	Character. The name of the variable (covariate) to plot. Required if type is 'coef' or 't_coef'.
crs	Numeric or character. The Coordinate Reference System (e.g., EPSG code) of the coordinates. If NULL, the function attempts to retrieve the CRS from x@my_crs.
mypalette	Character. The color palette to use for the points (e.g., "RdYlGn", "Viridis"). Default is "RdYlGn".
opacity	Numeric. The opacity of the markers, between 0 and 1. Default is 0.8.
size	Numeric. The size of the markers. Default is 5.
title	Character. A custom title for the plot. If NULL, a default title is automatically generated.
show_legend	Logical. Whether to display the legend. Default is TRUE.
n_time_steps	time step for spatio-temporal model (default 10).
...	Additional arguments passed to the internal plot function.

Value

An interactive Plotly object (Mapbox or Scatter).

A plotly object representing the interactive plot (or map).

See Also

MGWRSAR

---

plot_effect *plot_effect plot_effect is a function that plots the effect of a variable X_k with spatially varying coefficient, i.e X_k * Beta_k(u_i,v_i) for comparing the magnitude of effects of between variables.*

Description

plot_effect plot_effect is a function that plots the effect of a variable X_k with spatially varying coefficient, i.e X_k * Beta_k(u_i,v_i) for comparing the magnitude of effects of between variables.

Usage

```
plot_effect(model,sampling=TRUE,nsample=2000,nsample_max=5000,title='')
```

Arguments

model	a model of mgwrsar class with some spatially varying coefficients.
sampling	Bolean, if nrow(model@Betav)> nsample_max a sample of size nsample is randomly selected, default TRUE.
nsample	integer, size of the sample if sampling is TRUE, default 2000.
nsample_max	integer, size max to engage sampling if sampling is TRUE, default 5000.
title	a title for the plot.

Value

A side-effect function that displays a ggplot2 plot. Returns the plot object invisibly.

Examples

```
library(mgwrsar)
## loading data example
data(mydata)
coords=as.matrix(mydata[,c("x","y")])
## Creating a spatial weight matrix (sparce dgCMatrix)
## of 8 nearest neighbors with 0 in diagonal
model_GWR0<-MGWRSAR(formula = 'Y_gwr~X1+X2+X3', data = mydata,coords=coords,
fixed_vars=NULL,kernels=c('gauss'),H=0.13, Model = 'GWR',control=list(SE=TRUE))
plot_effect(model_GWR0)
```

---

predict,mgwrsar-method

predict method for mgwrsar model

Description

predict method for mgwrsar model

Usage

```
## S4 method for signature 'mgwrsar'
predict(
  object,
  newdata,
  newdata_coords,
  W = NULL,
  type = "BPN",
  h_w = 100,
```

```
  kernel_w = "rectangle",
  maxobs = 4000,
  beta_proj = FALSE,
  method_pred = "TP",
  k_extra = 8,
  exposant = 8,
  ...
)
```

Arguments

object	A model of class mgwrsar-class.
newdata	a matrix or data.frame of new data.
newdata_coords	a matrix of new coordinates, and eventually other variables if a General Kernel Product is used.
W	the spatial weight matrix for models with spatial autocorrelation.
type	Type for BLUP estimator, default "BPN". If NULL use predictions without spatial bias correction.
h_w	A bandwidth value for the spatial weight matrix
kernel_w	kernel type for the spatial weight matrix. Possible types: rectangle ("rectangle"), bisquare ("bisq"), tricube ("tcub"), epanechnikov ("epane"), gaussian ("gauss")) .
maxobs	maximum number of observations for exact calculation of solve(I- rho*W), default maxobs=4000.
beta_proj	A boolean, if TRUE the function then return a two elements list(Y_predicted,Beta_proj_out)
method_pred	If method_pred = 'TP' (default) prediction is done by recomputing a MGWRSAR model with new-data as target points, else if method_pred in ('tWtp_model','model','shepard') a matrix for projecting estimated betas is used (see details).
k_extra	number of neighboors for local parameter extrapolation if shepard kernel is used, default 8.
exposant	shapenig parameter for tds_mgtwr model, default 6.
...	predict parameters forwarded.

Details

if method_pred ='tWtp_model', the weighting matrix for prediction is based on the expected weights of outsample data if they were had been added to insample data to estimate the corresponding MGWRSAR (see Geniaux 2022 for further detail), if method_pred ='shepard'a shepard kernel with k_extra neighbours (default 8) is used and if method_pred='kernel_model' the same kernel and number of neighbors as for computing the MGWRSAR model is used.

Value

A vector of predictions if beta_proj is FALSE or a list with a vector named Y_predicted and a matrix named Beta_proj_out.

reord_D *reord_D*

Description

Reorders the elements of a distance matrix row by row according to a corresponding index matrix.

Usage

```
reord_D(M, idxG)
```

Arguments

M	A numeric matrix to be reordered.
idxG	An integer matrix of the same number of rows as ‘M‘, where each row contains column indices indicating the new ordering for that row.

Value

A reordered numeric matrix with the same number of rows as ‘M‘ and columns equal to the maximum index in ‘idxG‘.

reord_M_R *reord_M*

Description

Reorders the elements of a matrix row by row according to a corresponding index matrix. Each row of the index matrix specifies the new column order for the corresponding row of the input matrix.

Usage

```
reord_M_R(M, idxG)
```

Arguments

M	A numeric matrix to be reordered.
idxG	An integer matrix of the same number of rows as ‘M‘, where each row contains column indices indicating the new ordering for that row.

Details

This function performs a scatter-like reordering: for each row ‘i‘, the values of ‘M[i,]‘ are placed in the positions specified by ‘idxG[i,]‘. It is particularly useful for spatial or neighborhood-based rearrangements where each observation has its own local indexing of neighbors.

Value

A reordered numeric matrix with the same number of rows as ‘M‘ and columns equal to the maximum index in ‘idxG‘.

Examples

```
M <- matrix(1:9, nrow = 3, byrow = TRUE)
idxG <- matrix(c(3, 1, 2, 1, 2, 3, 2, 3, 1), nrow = 3, byrow = TRUE)
reord_M_R(M, idxG)
```

---

residuals,mgwrsar-method *residuals for mgwrsar model*

Description

residuals for mgwrsar model

Usage

```
## S4 method for signature 'mgwrsar'
residuals(object, ...)
```

Arguments

object	A model of class mgwrsar-class.
...	residuals parameters forwarded.

Value

A vector of residuals.

---

search_bandwidths *Bandwidth Selection via Multi-round Grid Search based on AICc*

Description

This function selects the optimal bandwidths (spatial and/or temporal) for GWR or MGWR models by minimizing the Corrected Akaike Information Criterion (AICc). It uses a multi-round grid search approach (coarse-to-fine) to efficiently narrow down the optimal parameter space before optionally applying a golden section search for final refinement.

Usage

```
search_bandwidths(
  formula,
  data,
  coords,
  fixed_vars = NULL,
  kernels = c("gauss", "gauss"),
  Model = "GWR",
  control = list(),
  hs_range = NULL,
  ht_range = NULL,
  n_seq = 10,
  ncore = 1,
  n_rounds = 0,
  refine = TRUE,
  verbose = FALSE,
  show_progress = FALSE,
  tol = NULL,
  parallel_method = "auto"
)
```

Arguments

formula	A formula object specifying the model (e.g., y ~ x1 + x2).
data	A data frame containing the variables in the model.
coords	A matrix or data frame of coordinates (2 columns for spatial, 1 for temporal, or more for GDT).
fixed_vars	A character vector indicating the names of variables with spatially stationary (fixed) coefficients. Default is NULL (all coefficients are varying).
kernels	A character vector specifying the kernel types for spatial and temporal components (e.g., c("gauss", "gauss")).
Model	A character string specifying the model type. Options include "GWR", "MGWR", "OLS", "SAR", etc. Default is "GWR".
control	A named list of extra control arguments passed to the MGWRSAR function (e.g., adaptive, NN, Z).
hs_range	A numeric vector of length 2 defining the lower and upper bounds for the spatial bandwidth search.
ht_range	A numeric vector of length 2 defining the lower and upper bounds for the temporal bandwidth search. Set to NULL for spatial-only models.
n_seq	An integer specifying the number of bandwidth candidates to test per dimension in each round.
ncore	An integer specifying the number of CPU cores to use for parallel processing. Default is 1. Examples/documentation should keep this at 1.
n_rounds	An integer specifying the number of grid search rounds (zooming steps). Default is 3.

refine	Logical. If TRUE, a final optimization step using golden section search is performed around the best candidate found. Default is FALSE.
verbose	Logical. If TRUE, prints progress messages to the console.
show_progress	Logical. If TRUE, displays a progress bar during computation.
tol	A numeric vector of length 2 (or 1) specifying the tolerance for spatial and temporal bandwidths. If NULL, it is calculated automatically based on the range.
parallel_method	Parallelization method ("auto", "fork", "socket"); "auto" selects the best available backend depending on the OS, other values run sequentially.

Details

The function performs a grid search over n_rounds. In the first round, it tests n_seq candidates linearly or geometrically spaced within the provided ranges. In subsequent rounds, the search range is narrowed around the best candidate from the previous round.

Value

A list containing:

mode	The mode of optimization ("spatial_only" or "spatio-temporal").
results	A list of data frames containing the results (bandwidths and AICc) for each round.
best	A data frame row corresponding to the best parameter combination found.
refined	The result of the refinement step (if refine = TRUE).
best_model	The final mgwrsar model object fitted with the optimal bandwidths.
ctime	The total computation time.

See Also

MGWRSAR

---

simu_multiscale *Simulate Data Generating Processes (DGP) for Multiscale GWR*

Description

The simu_multiscale function generates synthetic datasets with spatially varying coefficients based on Data Generating Processes (DGP) proposed in the literature. It supports formulations from Fotheringham et al. (2017), Gao et al. (2021), and Geniaux (2024). It also allows for the introduction of spatial autocorrelation (SAR) in the response variable.

Usage

```
simu_multiscale(
  n = 1000,
  myseed = 1,
  type = "GG2024",
  constant = NULL,
  nuls = NULL,
  lambda = NULL,
  NN = 4,
  config_beta = "default",
  config_snr = 0.7,
  config_eps = "normal"
)
```

Arguments

n	Integer. The number of observations to simulate. Default is 1000.
myseed	Integer. Random seed for reproducibility. Default is 1.
type	Character. The type of DGP to use. Options are 'FT2017', 'Gao2021', or 'GG2024' (default).
constant	Vector. Indices or names of explanatory variables that should have constant (spatially stationary) coefficients. Default is NULL (all coefficients vary).
nuls	Vector. Indices of explanatory variables that should have a null effect (coefficient = 0). Default is NULL.
lambda	Numeric. The spatial autoregressive parameter (rho) if a SAR process is desired. Default is NULL (no spatial autocorrelation).
NN	Integer. The number of nearest neighbors used to construct the spatial weight matrix W if lambda is provided. Default is 4.
config_beta	Character. Configuration of the spatial pattern for Beta coefficients (e.g., 'default').
config_snr	Numeric. The desired Signal-to-Noise Ratio (SNR). Default is 0.7.
config_eps	Character. The distribution of the error term. Options are 'normal' (default), 'unif', or 'Chi2'.

Value

A named list containing:

mydata A data frame with the simulated response variable y and explanatory variables.

coords A matrix of spatial coordinates.

Betav A matrix of the true spatially varying coefficients.

W The spatial weight matrix (if lambda is not NULL).

References

Fotheringham, A. S., Yang, W., & Kang, W. (2017). Multiscale geographically weighted regression (MGWR). *Annals of the American Association of Geographers*, 107(6), 1247-1265.

Gao, S., Mei, C. L., Xu, Q. X., & Wang, N. (2021). Non-iterative multiscale estimation for spatial autoregressive geographically weighted regression models. *Entropy*, 25(2), 320.

Geniaux, G. (2026). Top-down scale approaches for multiscale gwr with locally adaptive bandwidths. *Journal of Geographical Systems*: 1–50.

Examples

```
library(mgwrsar)
library(ggplot2)
library(gridExtra)
library(grid)

# Simulate data using Geniaux (2024) DGP
simu <- simu_multiscale(n = 1000, type = 'GG2024', config_snr = 0.7)
mydata <- simu$mydata
coords <- simu$coords

# Visualizing the spatial patterns of the coefficients
p1 <- ggplot(mydata, aes(x, y, col = Beta1)) + geom_point() + scale_color_viridis_c()
p2 <- ggplot(mydata, aes(x, y, col = Beta2)) + geom_point() + scale_color_viridis_c()
p3 <- ggplot(mydata, aes(x, y, col = Beta3)) + geom_point() + scale_color_viridis_c()
p4 <- ggplot(mydata, aes(x, y, col = Beta4)) + geom_point() + scale_color_viridis_c()

grid.arrange(p1, p2, p3, p4, nrow = 2, ncol = 2,
             top = textGrob("DGP Geniaux (2024)", gp = gpar(fontsize = 20, font = 3)))
```

---

summary,gtwr-method *summary for gtwr model*

Description

Prints the bandwidths in Huang's parameterisation and their mgwrsar equivalents side by side, then the standard `mgwrsar` summary.

Usage

```
## S4 method for signature 'gtwr'
summary(object, ...)
```

Arguments

`object`	A model of class `gtwr-class`.
`...`	summary parameters forwarded to the mgwrsar method.

Value

The model, invisibly.

`summary,mgwrsar-method` *summary for mgwrsar model*

Description

summary for mgwrsar model

Usage

```
## S4 method for signature 'mgwrsar'
summary(object, ...)
```

Arguments

`object`	A model of class `mgwrsar-class`.
`...`	summary parameters forwarded. In particular, `fdr_method` selects the multiple-testing correction for local coefficients: `"BY"` (default, Benjamini-Yekutieli, valid under arbitrary dependence) or `"spatial_BY"`, a bandwidth-calibrated variant that replaces the worst-case dependence penalty by the effective number of local tests $m_{eff,k} = tr(R_k)$ implied by each covariate's bandwidth (interpolates between BY and Benjamini-Hochberg).

Value

A summary object.

`summary_Matrix` *Summary of a Sparse Matrix*

Description

Produces a summary of a sparse matrix object showing its non-zero elements in triplet form (row, column, value).

Usage

```
summary_Matrix(object, ...)
```

Arguments

`object`	A sparse matrix object (inheriting from `Matrix`).
`...`	Additional arguments (currently unused).

Value

A data frame with columns `i`, `j`, `x` representing the row indices, column indices, and values of non-zero elements.

`TDS_MGWR` *Top-Down Scale (TDS) and Adaptive Top-Down Scale (ATDS) Estimation for MGWR*

Description

This function implements the "Top-Down Scale" (TDS) methodology for estimating Multiscale Geographically Weighted Regression (MGWR) models. Unlike classical backfitting approaches that fully optimize bandwidths at each iteration, TDS uses a pre-defined sequence of decreasing bandwidths to efficiently identify the optimal spatial scale for each covariate.

The function supports two main algorithms:

- **'tds_mgwr'**: A backfitting algorithm that selects a unique optimal bandwidth for each covariate from a decreasing sequence.
- **'atds_mgwr'**: Extends 'tds_mgwr' with a second "boosting" stage (Adaptive TDS). It refines estimates by allowing bandwidths to vary locally, capturing complex spatial patterns (e.g., simultaneous large-scale trends and local variations).

Usage

```
TDS_MGWR(formula, data, coords, Model = 'tds_mgwr',
                    kernels = 'gauss', fixed_vars = NULL, Ht = NULL,
                control_tds = list(nns = 20, get_AIC = FALSE, init_model = "OLS"),
                    control = list(adaptive = TRUE))
```

Arguments

`formula`	A formula object specifying the model (e.g., `y ~ x1 + x2`).
`data`	A data frame containing the variables in the model.
`coords`	A matrix or data frame of coordinates (2 columns for spatial).
`Model`	A character string specifying the algorithm. Options: • `'tds_mgwr'` (default): Top-Down Scale MGWR (Stage 1 only). • `'atds_mgwr'`: Adaptive Top-Down Scale MGWR (Stage 1 + Stage 2 boosting). • `'atds_gwr'`: Univariate Adaptive Top-Down Scale GWR.
`kernels`	A character string or vector specifying the kernel type (e.g., `'triangle'`, `'bisq'`, `'gauss'`). Default is `'triangle'`.
`fixed_vars`	A character vector indicating the names of variables with spatially stationary (fixed) coefficients. Default is `NULL`.

`Ht` Numeric. Optional bandwidth for the second dimension (time) if using spatio-temporal models (Type 'GDT').

`control_tds` A named list of control parameters specific to the TDS algorithm:

`nns` Integer. Number of bandwidth steps in the decreasing sequence (default 20, should be less than n/8 and maxit/2).

`get_AIC` Logical. If `TRUE`, computes AICc (slower). Default `FALSE` (except for `'atds_mgwr'`).

`init_model` Character. Initial model type to start backfitting: `'OLS'` (default), `'GWR'`, `'GTWR'`, or `'known'`.

`ncore` Integer. Number of cores for parallelization. Default 1.

`tol` Numeric. Convergence tolerance. Default 0.001.

`maxit` Numeric. Maximum number of iteration.

`nrounds` Integer. Number of boosting rounds for Stage 2 (only for `'atds_mgwr'`). Default 3.

`H`, `Ht` Numeric. Known bandwidths instead of a search: `H` fixes every spatial bandwidth, `Ht` (`Type = 'GDT'` only) every temporal one. Each is an unnamed vector with one value per varying coefficient, in model-matrix order (`Intercept` first); a value at or above the largest bandwidth (e.g. `Inf`) is a global bandwidth. An axis is either fully fixed or fully searched (no `NA`, no named subset): `H` alone still searches the temporal bandwidths and conversely, `H` and `Ht` together freeze the search entirely, in which case the backfitting is iterated to its fixed point (every sweep kept, stop on `tol`) so that the returned model and its criterion depend on the bandwidths only. Units follow `control$adaptive`: numbers of neighbours (rounded) for an adaptive kernel, distances otherwise. Lower `tol` (e.g. `1e-6`) when the criterion at given bandwidths is to be read precisely.

`V` Numeric. User-supplied spatial bandwidth sequence, always in numbers of neighbours (converted to distances when the spatial kernel is not adaptive). Any order; `first_nn` is added on top.

`V_dist` Numeric. Fixed spatial kernel only: user-supplied spatial bandwidth sequence directly in distances (any order; the largest distance is added on top, and the grid is taken as is, without the panel extension). Overrides `V`, `minv` and `panel_floor`.

`panel_floor` Numeric in (0, 1]. Fixed spatial kernel with repeated locations (panel data) only. The neighbour-count grid cannot go below the distance to the first site, although the own observations of a site identify a smaller bandwidth; the grid is therefore extended geometrically below its floor down to `panel_floor` times that distance (ratio 0.8 for a Gaussian kernel, 0.9 otherwise), provided every site has at least two observations. Default 0.2 for a Gaussian kernel (its bandwidth is a standard deviation) and 0.5 for compact-support kernels; 1 keeps the grid unchanged.

`first_nn` Integer. Top of the spatial bandwidth sequence, in numbers of neighbours. Default `nrow(data)`.

`trunc_gauss` Numeric or `NULL`. Fixed gaussian spatial kernel only: when set to c, each bandwidth h of the descent is evaluated on the neighbours within distance ch (relative weight $e^{-c^2/2}$ beyond: 1.5e-8 for $c = 6$, 1.3e-14 for

$c = 8$), which shrinks the weight matrices at small scales. `NULL` (default) keeps every neighbour. Fixed compact kernels are always truncated exactly (zero weights dropped), as adaptive ones already were.

`minv` Integer. Floor of the spatial bandwidth sequence, in numbers of neighbours: the grid stops at the median distance to the `minv`-th neighbour (or at `minv` neighbours for an adaptive kernel). Default: the identifiability floor, 2 for a Gaussian kernel and `K + 1` for compact-support kernels. Raising it above that floor truncates the descent: a coefficient whose optimal bandwidth is smaller stops on the grid floor with a censored bandwidth and non-converged coefficients, so a warning is issued at setup and again when a bandwidth ends on the floor.

`check_pairs` Logical. If `TRUE`, each varying coefficient other than the intercept receives an identifiability floor before the search: the smallest bandwidth of the grid (`Type = 'GD'`, slot `HKmin`) or, for each spatial bandwidth of the grid, the smallest temporal one (`Type = 'GDT'`, slot `HKMIN`, one matrix of (hs, ht) pairs per coefficient) at which the local weighted variance of the covariate exceeds `control$min_var_ratio` times its global variance at a share `control$min_var_q` of the sites; the search then never goes below that floor. A covariate that never reaches the threshold on the grid is kept at the global bandwidth(s), with a warning naming it. The intercept takes the smallest floor of the covariates. The local moments are computed natively, on `ncore` threads when the package is built with OpenMP (`MGWRSAR_OPENMP=1`), and the grid is searched by bisection (the ratio is taken as non-decreasing in each bandwidth). Default `FALSE`.

`ordering` Character. Order in which the varying coefficients are updated within a sweep, the intercept always first: `'Importance'` (default), by decreasing (or increasing, see `importance_order`) standard deviation of the fitted contribution at the previous sweep; `'cyclic'`, the order of the formula at every sweep; `'random'`, a new permutation at every sweep, reproducible through the random-number seed.

`update_last` Character vector or `NULL` (default). Names of varying coefficients updated last in every sweep, whatever the ordering, in their order of appearance; the intercept is always updated first. Meant for covariates that are deterministic functions of the kernel coordinates (a time index, a day of year, a year of sale), whose component would otherwise pre-empt, as a global drift, the temporal variation that the other components can capture through their own temporal bandwidths.

`importance_order` Character. Direction of the importance-driven update order (`ordering = 'Importance'`): `'decreasing'` (default) updates first the component whose fitted contribution $\hat{\beta}_k x_k$ has the largest standard deviation over the observations, `'increasing'` updates it last. The intercept is always updated first.

`saturation_ratio` Numeric in [0, 1). Safeguard against near-interpolating local fits: at each component update, a candidate fit whose residual variance is below `saturation_ratio` times the residual variance of the current complete fit is excluded from the search; if every candidate is excluded, the one with the smallest AICc is kept with a warning. A ratio of variances, so the rule does not depend on the unit of the response nor on `get_AIC` (until 1.4.2

it compared the AICc of a candidate with half of a reference AICc). The counts are returned in the slot `saturation`. Default `exp(-2)`; 0 disables it.

`polish` Integer. Maximum number of polishing sweeps run after the search has stopped, whatever its stopping rule: the backfitting is iterated with every bandwidth fixed at the retained state (no search, no patience rule, every sweep kept) until the relative change of the in-sample RMSE between two sweeps falls below `tol`, or that many sweeps have been run, so that the returned coefficients, projection matrices and standard errors are the fixed point of the backfitting at these bandwidths, to `tol`. The bandwidths are unchanged. Default 3; 0 returns the state at which the search stopped.

`retained_state` Character. Which sweep of the backfitting is returned: `'terminal'` (default), the state that met the stopping rule, as in MGWR backfitting, where the sweeps continue from the current state whatever the change of the in-sample RMSE and only the stopping rule reads that change; or `'best'`, the sweep with the smallest in-sample RMSE, the behaviour of versions 1.3.1 to 1.4.2. The sweeps, the stopping rules and the patience counter are identical in both cases; only the returned state (coefficients, bandwidths, projection matrices, standard errors) differs, and only when a sweep raised the RMSE. The slots `HBETA` and `HRMSE` hold every sweep up to the returned one.

`TRUEBETA` Numeric matrix. Diagnostic lever for simulations: the true coefficients, one row per observation and one column per coefficient of the model matrix (`Intercept` first). When given, the slot `HRMSE` of the returned model holds, for the starting model (row 1) and each kept sweep, the RMSE of every coefficient, their mean, the spatial bandwidth of the last varying coefficient, the AICc (`get_AIC = TRUE`) and the residual RMSE. Default `NULL`.

`control` A named list of standard control arguments passed to the internal `MGWRSAR` calls:

`adaptive` Logical or Vector. `TRUE` for adaptive bandwidth (nearest neighbors), `FALSE` for fixed distance. Can be a vector of length 2 for space/time; a single value applies to both. Default `c(TRUE, FALSE)`. With an adaptive temporal kernel the temporal bandwidths are searched on a sequence of numbers of neighbours built like the spatial one.

`Type` Character. Spatial weighting type: `'GD'` (Spatial only) or `'GDT'` (Space-Time).

`NN` Integer. Maximum number of neighbors for distance matrix computation (truncation). Default is `nrow(data)`.

`min_var_ratio`, `min_var_q` Numeric. Local variance criterion of `control_tds$check_pairs` (also read by the penalised golden-section search): a bandwidth is admissible for a covariate when the ratio of its local weighted variance to its global variance exceeds `min_var_ratio` (default 0.05) at the quantile `min_var_q` (default 0.10) of the sites.

Details

The TDS algorithm works in two stages:

1. **Stage 1 (Backfitting):** Starts with a global model (OLS) or a simple GWR. It iteratively updates the bandwidth for each covariate by testing values from a decreasing sequence. This

avoids the "yo-yo" effect of standard backfitting and converges faster.

2. **Stage 2 (Boosting - atds_mgwr only):** Uses the residuals from Stage 1 to iteratively refine coefficients. This stage allows the effective bandwidth to adapt locally, improving accuracy for covariates with spatially heterogeneous scales.

Value

An object of class `mgwrsar` containing:

`Betav`	Matrix of spatially varying coefficients.
`H`	Vector of optimal bandwidths found for each covariate.
`RMSE`	Root Mean Square Error of the final model.
`AICc`	Corrected Akaike Information Criterion (if requested).
`fitted.values`	Vector of fitted values.
`residuals`	Vector of residuals.

References

Geniaux, G. (2024). Top-Down Scale Approaches for Multiscale GWR with Locally Adaptive Bandwidths. *Springer Nature*.

See Also

`MGWRSAR`, `golden_search_bandwidth`

Index

as_gtwr, 2, *12*
atds_gwr, 4

bw_gdt2hwb, 4, *5*, *12*
bw_hwb2gdt, *4*, 5, *12*

coef,mgwrsar-method, 6

difftime, *12*

find_TP, 6
fitted,mgwrsar-method, 8

golden_search_2d_bandwidth, 8
golden_search_bandwidth, *9*, 9, *24*, *41*
gtwr-class, 11
gtwr_HWB2010, *3*, *11*, 11

kernel_matW, 13

make_unique_by_structure, 15
MGWRSAR, *11*, *12*, 16, *24*, *27*, *33*, *41*
mgwrsar-class, 19
mgwrsar_bootstrap_test, 21
mgwrsar_bootstrap_test_all, 22
multiscale_gwr, 22
mydata, 24
mydatasf, 25

normW, 25

plot,mgwrsar,missing-method, 26
plot.mgwrsar
        *(*plot,mgwrsar,missing-method*)*, 26
plot_effect, 27
predict,mgwrsar-method, 28

reord_D, 30
reord_M_R, 30
residuals,mgwrsar-method, 31

search_bandwidths, *2*, *4*, *5*, *9*, *12*, 31
simu_multiscale, 33
summary,gtwr-method, 35
summary,mgwrsar-method, 36
summary_Matrix, 36

TDS_MGWR, *24*, 37