Package 'sssvcqr'

May 15, 2026

Title Sparse-Smooth Spatially Varying Coefficient Quantile Regression

Version 0.0.4

Author Houjian Hou [aut, cre]

Maintainer Houjian Hou <beidaihe77@qq.com>

Description Implements sparse-smooth spatially varying coefficient quantile regression (SS-SVCQR), combining quantile regression of Koenker and Bassett (1978) <doi:10.2307/1913643>, grouped variable selection of Yuan and Lin (2006) <doi:10.1111/j.1467-9868.2005.00532.x>, graph regularization, and the alternating direction method of multipliers of Boyd et al. (2011) <doi:10.1561/2200000016>. The package provides graph-regularized estimation, spatially blocked cross-validation, prediction, diagnostics, and simulation helpers for global-local spatial quantile regression.

License GPL (>= 3)

URL https://github.com/Stork343/sssvcqr

BugReports https://github.com/Stork343/sssvcqr/issues

Encoding UTF-8

LazyData true

Depends R (>= 4.1)

Imports FNN, igraph, Matrix, methods, stats

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation no

Repository CRAN

Date/Publication 2026-05-15 20:00:02 UTC

Contents

`sssvcqr-package` *Sparse-Smooth Spatially Varying Coefficient Quantile Regression*

Description

Implements sparse-smooth spatially varying coefficient quantile regression (SS-SVCQR). The package separates global baseline effects from spatial deviations, selects global versus local effects with a group penalty, and smooths local deviations with a graph Laplacian.

See Also

`ss_svcqr`, `cv_ss_svcqr`, `simulate_sssvcqr_data`, `selection_recovery_table`

`build_graph_laplacian` *Build a weighted k-nearest-neighbor graph Laplacian*

Description

Constructs a weighted k-nearest-neighbor graph from spatial coordinates using **FNN** nearest-neighbor search and returns sparse adjacency and Laplacian matrices.

Usage

```
build_graph_laplacian(u, k = 10L, normalized = TRUE,
  symmetrize = c("union", "mutual"), sigma = NULL)
```

Arguments

u	Coordinate matrix.
k	Number of nearest neighbors. Values greater than or equal to the number of locations are capped at nrow(u) - 1.
normalized	Use the normalized Laplacian if TRUE; otherwise use the unnormalized Laplacian.
symmetrize	Use the union or mutual k-nearest-neighbor graph.
sigma	Optional positive Gaussian kernel bandwidth.

Details

The graph is built from the directed k-nearest-neighbor edge list rather than from a full pairwise distance matrix. The returned W, L_sym, and L components are sparse **Matrix** objects. For fixed k, graph storage is $O(nk)$ instead of $O(n^2)$. When sigma is omitted, the Gaussian bandwidth is computed from positive k-nearest-neighbor distances; duplicated coordinates are allowed, and zero-distance neighbor edges receive weight one.

Value

A list containing the weight matrix, degree vector, sparse Laplacian, and connected components.

---

cv_ss_svcqr *Spatially blocked cross-validation for SS-SVCQR*

Description

Evaluates a grid of sparsity and smoothness penalties using spatially blocked cross-validation and held-out check loss.

Usage

```
cv_ss_svcqr(y, Z, X, u, tau = 0.5, lambda1_seq, lambda2_seq,
  k_nn = 10L, K_folds = 5L, folds = NULL, adaptive_weights = TRUE,
  lambda1_pilot = NULL, lambda2_pilot = NULL, a_stabilizer = 0.01,
  gamma_power = 1, w = NULL, control = list(max_iter = 500L,
  tol_pri = 1e-04, tol_dual = 0.001), fold_seed = 1L,
  verbose = FALSE, graph_normalized = TRUE,
  graph_symmetrize = c("union", "mutual"), graph_sigma = NULL)
```

Arguments

y, Z, X, u, tau	Model inputs passed to ss_svcqr.
lambda1_seq, lambda2_seq	Penalty grids.
k_nn	Number of nearest neighbors for graph construction.

K_folds	Number of spatial folds when folds is not supplied.
folds	Optional integer fold labels.
adaptive_weights	Whether to estimate adaptive group weights by pilot fits. When TRUE and w is not supplied, pilot weights are estimated within each training fold to avoid validation leakage.
lambda1_pilot, lambda2_pilot	Optional pilot penalties.
a_stabilizer, gamma_power	Adaptive weight parameters.
w	Optional precomputed group weights.
control	ADMM control list.
fold_seed	Random seed for fold construction.
verbose	Print progress messages.
graph_normalized, graph_symmetrize, graph_sigma	Graph construction options.

Value

An object of class sssvcqr_cv with CV losses and the best pair.

---

kkt_sssvcqr *KKT diagnostics for an SS-SVCQR fit*

Description

Computes first-order diagnostic quantities for fitted SS-SVCQR coefficients, including group stationarity residuals, zero-group margins, and centering constraint residuals.

Usage

```
kkt_sssvcqr(y, Z, X, fit, L_sym = NULL, D_vec = NULL,
  components_list = NULL, lambda1 = fit$lambda1, lambda2 = fit$lambda2,
  w = fit$weights, tau = fit$tau)
```

Arguments

y, Z, X	Model inputs used for fitting.
fit	A fitted sssvcqr object.
L_sym, D_vec, components_list	Optional graph quantities. If omitted, they are rebuilt from the fitted object's graph metadata.
lambda1, lambda2, w, tau	Penalty, weight, and quantile settings.

Value

A list with gradient norms, group stationarity residuals, group margins, group norms, degree-weighted centering violations, and the maximum diagnostic violation.

---

lucas_housing_sample *Lucas County housing sample*

Description

A small sample derived from the Lucas County housing data used in the empirical analysis. The data are intended for examples, tutorials, and package-sample replication only, not for reproducing the full empirical results.

Format

A data frame with selected variables:

log_price Log sale price.

log_TLA Log total living area.

log_lotsize Log lot size.

age_scaled House age scaled by 100.

age2_scaled Squared age scaled by 10000.

longitude Longitude.

latitude Latitude.

sale_year Sale year.

Source

Sampled from the project data file `lucas_housing_clean.csv`.

---

make_spatial_folds *Create spatial cross-validation folds*

Description

Assigns observations to spatially coherent folds using k-means clustering or a simple coordinate grid.

Usage

```
make_spatial_folds(u, K = 5L, method = c("kmeans", "grid"), seed = NULL)
```

Arguments

u	Coordinate matrix.
K	Number of folds.
method	Fold construction method.
seed	Optional random seed.

Value

An integer vector of fold labels.

---

plot.sssvcqr *Plot an SS-SVCQR fit*

Description

Displays spatial deviation surfaces, local coefficient surfaces, residual surfaces, or ADMM convergence traces for a fitted SS-SVCQR model. Spatial maps use inverse-distance interpolation on the first two coordinate columns, with observed locations overlaid and a diverging numeric colorbar.

Usage

```
## S3 method for class 'sssvcqr'
plot(x, type = c("deviation", "coefficient", "residual",
  "convergence"), index = 1L, ...)
```

Arguments

x	A fitted sssvcqr object.
type	Plot type. "deviation" displays the estimated spatial deviation delta[, index]; "coefficient" displays the local total coefficient beta_spatial[, index]; "residual" displays response minus fitted conditional quantile; and "convergence" displays ADMM primal and dual residual traces.
index	Index of the candidate local covariate for deviation and local coefficient plots. It is ignored for residual and convergence plots.
...	Additional graphical arguments passed to plot.

Details

For spatial plot types, the horizontal and vertical axes are the first two columns of x$u. These are the coordinates supplied to ss_svcqr when the model was fitted, so they may be raw coordinates or scaled coordinates depending on the user's input. The fitted quantity is rendered as a gridded surface for readability, with observed locations overlaid as small reference marks. Blue colors represent smaller values, white colors are near the middle of the displayed range, and red colors represent larger values. The colorbar reports the numeric scale for the plotted quantity. The convergence plot keeps a line legend for the four ADMM residual traces instead of a colorbar.

Value

The input object, invisibly.

Examples

```
dat <- simulate_sssvcqr_data(n = 20, q = 1, p = 2, seed = 2)
fit <- ss_svcqr(dat$y, dat$Z, dat$X, dat$u,
  lambda1 = 5, lambda2 = 0.1, k_nn = 4,
  control = list(max_iter = 10, warn_nonconvergence = FALSE))
plot(fit, type = "deviation", index = 1)
```

---

predict.sssvcqr *Predict from an SS-SVCQR fit*

Description

Returns fitted quantiles or local coefficient surfaces from an SS-SVCQR model. For new locations, spatial deviations are extrapolated from nearest training locations. If `k > 1`, inverse-distance weights average the `k` nearest fitted deviation vectors.

Usage

```
## S3 method for class 'sssvcqr'
predict(object, Znew = NULL, Xnew = NULL, unew = NULL,
  k = 1L, type = c("response", "coefficients"), ...)
```

Arguments

`object`	A fitted `sssvcqr` object.
`Znew`	New global covariate matrix. Required when the model has global `Z` covariates and `Xnew` is supplied.
`Xnew`	New local-candidate covariate matrix. If omitted, training fitted values or coefficient surfaces are returned.
`unew`	New coordinate matrix. Required for prediction at locations other than the training observations.
`k`	Number of nearest neighbors used to extrapolate spatial deviations.
`type`	Return fitted responses or local coefficient surfaces.
`...`	Unused.

Value

A numeric vector of predicted quantiles or a matrix of local coefficients.

selection_recovery_table *Compare true and estimated spatial-deviation selection*

Description

Builds a compact truth-versus-estimate table for synthetic examples generated by simulate_sssvcqr_data.

Usage

```
selection_recovery_table(fit, truth, tol = 1e-6)
```

Arguments

fit	An sssvcqr fitted object returned by ss_svcqr.
truth	A list containing delta_true and, optionally, active. Objects returned by simulate_sssvcqr_data include these entries.
tol	Non-negative tolerance used to classify an estimated deviation field as selected.

Value

A data frame with the covariate index, covariate name, true active status, true deviation norm, estimated deviation norm, and selected active status.

Examples

```
dat <- simulate_sssvcqr_data(n = 40, q = 1, p = 3, seed = 1)
fit <- ss_svcqr(dat$y, dat$Z, dat$X, dat$u,
  lambda1 = 5, lambda2 = 0.1, k_nn = 6,
  control = list(max_iter = 30, warn_nonconvergence = FALSE))
selection_recovery_table(fit, dat)
```

simulate_sssvcqr_data *Simulate data for SS-SVCQR examples*

Description

Generates synthetic spatial quantile-regression data with known global coefficients and sparse spatial deviation fields. For the default p = 3 design, the first and third deviation fields are active and the second deviation field is exactly zero.

Usage

```
simulate_sssvcqr_data(n = 120, q = 2, p = 3, tau = 0.5,
  noise_sd = 0.4, seed = NULL, graph_k = 8L)
```

Arguments

n	Number of observations.
q	Number of global covariates.
p	Number of candidate spatially varying covariates.
tau	Target quantile level.
noise_sd	Gaussian error scale.
seed	Optional random seed.
graph_k	Graph size used to center true deviations with the same degree-weighted connected-component convention used by the estimator.

Value

A list containing y, Z, X, u, eta, alpha_true, beta_G_true, delta_true, beta_spatial_true, active, tau, and seed. The legacy aliases alpha, beta_G, and delta are retained for backward compatibility.

Examples

```
dat <- simulate_sssvcqr_data(n = 30, q = 1, p = 3, seed = 1)
str(dat)
dat$active
```

---

ss_svcqr *Fit sparse-smooth spatially varying coefficient quantile regression*

Description

Fits an SS-SVCQR model by ADMM for a fixed quantile level. Candidate local coefficients are represented as global baselines plus graph-smoothed spatial deviations, with a group penalty selecting deviations that should be zero.

Usage

```
ss_svcqr(y, Z, X, u, tau = 0.5, lambda1, lambda2, k_nn = 10L,
  w = NULL, control = list(), graph_normalized = TRUE,
  graph_symmetrize = c("union", "mutual"), graph_sigma = NULL)
```

Arguments

y	Numeric response vector.
Z	Matrix of covariates with global effects. Use NULL for none.
X	Matrix of covariates that may have spatially varying effects.
u	Coordinate matrix with one row per observation.
tau	Quantile level in (0, 1).

lambda1	Group sparsity penalty for spatial deviations.
lambda2	Graph Laplacian smoothness penalty.
k_nn	Number of nearest neighbors used to build the graph.
w	Optional non-negative adaptive group weights.
control	List of ADMM controls: max_iter, tol_pri, tol_dual, rho_s, rho_z, ridge, and warn_nonconvergence. Set verbose = TRUE to print residual summaries during fitting.
graph_normalized	Use the normalized graph Laplacian if TRUE.
graph_symmetrize	How to symmetrize the directed k-nearest-neighbor graph.
graph_sigma	Optional Gaussian kernel bandwidth.

Value

An object of class sssvcqr containing global coefficients, spatial deviations, fitted values, residuals, convergence history, and graph metadata.

Examples

```
dat <- simulate_sssvcqr_data(n = 40, q = 1, p = 3, seed = 1)
fit <- ss_svcqr(dat$y, dat$Z, dat$X, dat$u,
  lambda1 = 5, lambda2 = 0.1, k_nn = 5,
  control = list(max_iter = 20, warn_nonconvergence = FALSE))
fit
selection_recovery_table(fit, dat)
head(predict(fit))
```

Index