

Package: SPmodels (via r-universe)

October 22, 2026

Type Package

Title Spatial Models for Research and Teaching

Version 0.1.1

Description Independently implemented spatial modelling methods with a formula-oriented interface, transparent uncertainty, and mathematical derivations for research and teaching. The documented course workflows have regression coverage. This package has not been submitted to CRAN.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.2.0)

Imports graphics, parallel, stats

Suggests sf

Config/roxygen2/version 8.1.1

Repository <https://olatunjijohnson.r-universe.dev>

Date/Publication 2026-10-22 09:00:16 UTC

RemoteUrl <https://github.com/olatunjijohnson/SPmodels>

RemoteRef v0.1.1

RemoteSha 47bbe4a8c3cd83f1c27c974a8dd31df1901df67b

Contents

as.data.frame.spmodels_prediction_target	3
confint.spmodels_fit	3
matern_cor	4
matern_scale_derivative	5
plot.spmodels_cross_validation	5
predict.spmodels_fit	7
simulate.spmodels_fit	8
sp_block_folds	8
sp_buffered_folds	9

sp_calibration	10
sp_control	11
sp_cross_validate	12
sp_diagnostics	13
sp_distance_summary	14
sp_environmental_folds	15
sp_fit	16
sp_fit_neighbor	18
sp_format_table	21
sp_grid	22
sp_group_folds	22
sp_intervention	23
sp_intervention_curve	24
sp_intervention_multiplier	25
sp_latent_mode	26
sp_latent_sample	27
sp_mcmc_control	28
sp_mcml	28
sp_mcml_control	30
sp_mcml_precision	31
sp_mcml_replicates	32
sp_model_table	33
sp_neighbor_covariance	33
sp_nonstationary_cor	35
sp_polygon_area_weights	36
sp_polygon_weights	36
sp_predict_chunked	37
sp_predict_draws	38
sp_predict_draws_neighbor	40
sp_predict_target	42
sp_predict_target_neighbor	43
sp_prediction_table	44
sp_prediction_update	45
sp_predictive_log_score	46
sp_random_slope	47
sp_report	48
sp_sampler_diagnostics	49
sp_simulation_assess	50
sp_simulation_grid	50
sp_simulation_precision	51
sp_simulation_recovery	52
sp_simulation_scenarios	54
sp_spatial_folds	55
sp_target_geometry	56
sp_temporal	56
sp_utm_crs	57
sp_validation_table	58
sp_variogram	58

<code>as.data.frame.spmodels_prediction_target</code>	3
<code>sp_variogram_envelope</code>	59
<code>spacetime_cor</code>	61
Index	62

<code>as.data.frame.spmodels_prediction_target</code>
<i>Convert a prediction-target result to its summary table</i>

Description

Convert a prediction-target result to its summary table

Usage

```
## S3 method for class 'spmodels_prediction_target'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

- `x` An object returned by `sp_predict_target()`.
- `row.names` Optional row names for the returned summary.
- `optional` Passed through for `as.data.frame()` compatibility.
- `...` Reserved for method compatibility.

Value

The target summary data frame stored in `'x$summary'`.

<code>confint.spmodels_fit</code>	<i>Confidence Intervals for Fitted SPmodels Parameters</i>
-----------------------------------	--

Description

Constructs local Wald intervals from the fitted standard errors. Mean coefficients and random intercept-slope covariance entries use the identity scale, positive parameters use a logarithmic scale, and an estimated intervention impact uses the logit scale.

Usage

```
## S3 method for class 'spmodels_fit'
confint(object, parm, level = 0.95, ...)
```

Arguments

object	A fitted <code>spmodels_fit</code> object.
parm	Optional numeric positions or qualified names such as <code>"mean::(Intercept)"</code> or <code>"covariance::scale"</code> .
level	Confidence probability strictly between zero and one.
...	Reserved for method compatibility.

Details

These intervals rely on the fitted local observed-Hessian approximation. A missing standard error produces missing limits rather than a fabricated interval. Fixed parameters are omitted. The intervals do not include Monte Carlo error or repair weak identifiability.

Value

A matrix containing lower and upper limits. Its row names identify the model component and term. Attributes record the level and transformation.

matern_cor	<i>Matérn Correlation</i>
------------	---------------------------

Description

Evaluates the unit-variance Matérn correlation with dimensionless argument distance / scale. This scale convention is explicit; the argument is not described as a practical or effective range.

Usage

```
matern_cor(distance, scale, smoothness)
```

Arguments

distance	A numeric vector, matrix, or array of finite, non-negative distances.
scale	A finite positive numeric scalar in the same units as distance.
smoothness	A finite positive numeric scalar.

Details

At zero distance the function returns one by continuity. Positive arguments are evaluated on the log scale using R's exponentially scaled modified Bessel function. See the Matérn chapter in the package's technical derivations for the assumptions, algorithm, and validation links.

Value

A numeric object with the dimensions and names of distance, containing correlations between zero and one.

Examples

```
d <- as.matrix(dist(c(0, 0.5, 1)))
matern_cor(d, scale = 1, smoothness = 0.5)
```

matern_scale_derivative

Matérn Correlation Derivatives with Respect to Scale

Description

Calculates the first or second derivative of the SPmodels Matérn correlation with respect to scale, holding distance and smoothness fixed.

Usage

```
matern_scale_derivative(distance, scale, smoothness, order = 1L)
```

Arguments

distance	Numeric finite non-negative distances.
scale	One finite positive scale in the distance unit.
smoothness	One finite positive Matérn smoothness.
order	Derivative order, either 1 or 2.

Value

A numeric object preserving the dimensions and names of distance.

plot.spmodels_cross_validation

Display validation and simulation results

Description

Base-graphics views over the unrounded values already stored in SPmodels validation and simulation results. No estimates or summaries are recomputed for display, apart from the displayed Wald endpoints in a recovery plot.

Usage

```
## S3 method for class 'spmmodels_cross_validation'
plot(
  x,
  type = c("observed", "error", "fold"),
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  ...
)

## S3 method for class 'spmmodels_simulation_assessment'
plot(
  x,
  type = c("prediction", "error"),
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  ...
)

## S3 method for class 'spmmodels_simulation_recovery'
plot(
  x,
  parameter = 1L,
  intervals = TRUE,
  xlab = "Simulation replicate",
  ylab = NULL,
  main = NULL,
  ...
)
```

Arguments

<code>x</code>	A cross-validation, predictive-assessment or simulation-recovery object.
<code>type</code>	The requested view for cross-validation or predictive assessment.
<code>parameter</code>	Name or index of one recorded recovery parameter.
<code>intervals</code>	Whether to draw reported Wald intervals when available.
<code>xlab, ylab, main</code>	Plot labels.
<code>...</code>	Further arguments passed to the principal base-graphics call.

Value

The input object, invisibly.

predict.spmodels_fit *Predict from a Fitted SPmodels Model*

Description

Computes conditional Gaussian predictions using universal kriging equations. For binomial and Poisson Laplace fits, computes an approximate Gaussian predictive linear predictor and integrates it through the response link. Prediction uncertainty is conditional on fitted regression and covariance parameters.

Usage

```
## S3 method for class 'spmodels_fit'
predict(
  object,
  newdata = NULL,
  se.fit = FALSE,
  target = c("latent", "observation"),
  type = c("response", "link"),
  unseen = c("error", "marginal"),
  ...
)
```

Arguments

object	A fitted spmodels_fit object.
newdata	Optional data frame containing predictors and coordinates.
se.fit	Whether to return prediction standard errors.
target	Predict the latent spatial process or a future noisy observation-site target. Observation uncertainty includes nugget variance and, on the non-Gaussian response scale, response sampling variation.
type	Return predictions on the response or linear-predictor scale. These coincide for the Gaussian identity link.
unseen	Policy for grouping levels absent during fitting. "error" rejects them. "marginal" integrates a new zero-mean random intercept with its fitted variance and preserves dependence between prediction rows sharing the same new level.
...	Reserved for method compatibility.

Value

A numeric vector, or a list containing fit, se.fit, target, and, for non-Gaussian fits, type.

`simulate.spmodels_fit` *Simulate from a Fitted Spatial Model*

Description

Generates unconditional Gaussian, binomial or Poisson response replicates at the fitted observation sites using the fitted mean, family and covariance parameters. Binomial output contains simulated success counts with the fitted trial totals.

Usage

```
## S3 method for class 'spmodels_fit'
simulate(object, nsim = 1L, seed = NULL, ...)
```

Arguments

<code>object</code>	A fitted <code>spmodels_fit</code> object.
<code>nsim</code>	Positive number of response simulations.
<code>seed</code>	Optional integer seed. The caller's random-number state is restored after seeded simulation.
<code>...</code>	Reserved for method compatibility.

Value

A data frame with one simulated response per column.

`sp_block_folds` *Construct Regular Spatial-Block Folds*

Description

Partitions projected fitted coordinates into axis-aligned rectangular blocks and assigns whole occupied blocks to validation folds. Repeated coordinates and all rows in one block remain together.

Usage

```
sp_block_folds(
  object,
  block_size,
  folds = 5L,
  origin = NULL,
  assignment = c("systematic", "random"),
  seed = NULL
)

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

Arguments

object	A fitted spatial <code>smodels_fit</code> .
block_size	One finite positive block width, or separate widths for the two coordinate axes, in fitted distance units.
folds	Number of validation folds, not exceeding the number of occupied blocks.
origin	Optional two-coordinate block-grid origin. Coordinate-wise minima are used by default.
assignment	Allocate canonically ordered blocks round-robin, or randomly permute whole blocks before balanced allocation.
seed	Optional integer seed local to random allocation.
x	An object returned by <code>sp_block_folds()</code> .
...	Reserved for print-method compatibility.

Details

Systematic assignment follows blocks from top to bottom and left to right, cycling through fold labels. Random assignment permutes whole blocks under a caller-local seed before balanced allocation. This controls the support withheld together; it does not by itself impose a distance buffer.

Value

An integer fold vector in fitted-row order with class `smodels_block_folds`. Occupied-block indices, allocation, block size, origin, units and seed are retained as attributes.

sp_buffered_folds	<i>Construct Buffered Spatial Validation Folds</i>
-------------------	--

Description

Constructs geometry-driven held-out folds and declares a spatial buffer. During cross-validation, rows outside the held-out fold but no farther than the buffer from any test location are excluded from that fold's training fit.

Usage

```
sp_buffered_folds(object, folds = 5L, buffer, seed = NULL, nstart = 20L)

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

Arguments

object	A fitted spatial <code>smodels_fit</code> model.
folds	Number of geometry-driven test folds.
buffer	Finite positive exclusion distance in fitted coordinate units.
seed	Optional local seed for fold construction.
nstart	Positive number of k-means starts.
x	An object returned by <code>sp_buffered_folds()</code> .
...	Reserved for print-method compatibility.

Details

The buffer changes training membership independently for every fold; buffered rows are neither trained on nor scored in that fold. Test rows remain exactly those identified by the fold vector. An over-large buffer that leaves no usable training data fails explicitly.

Value

An integer fold vector carrying the exclusion distance, coordinate units, cluster centres, seed and k-means metadata.

sp_calibration

Assess and display predictive interval calibration

Description

Forms central type-8 empirical predictive intervals at each requested nominal level, evaluates their coverage of the supplied truths and calculates pointwise Wilson uncertainty bounds for the observed proportions.

Usage

```
sp_calibration(
  object,
  truth,
  levels = c(0.5, 0.8, 0.9, 0.95),
  confidence = 0.95
)

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

## S3 method for class 'smodels_calibration'
plot(
  x,
  xlab = "Nominal coverage",
```

```

    ylab = "Observed coverage",
    main = "Predictive interval calibration",
    ...
)

```

Arguments

object	An object returned by <code>sp_predict_draws()</code> .
truth	One finite value for every prediction location.
levels	Central predictive-interval probabilities strictly between zero and one.
confidence	Confidence level for pointwise Wilson bounds around empirical coverage.
x	An object returned by <code>sp_calibration()</code> .
xlab, ylab, main	Plot labels.
...	Further arguments passed to the principal base-graphics call.

Details

The bounds quantify finite-location uncertainty in the empirical coverage proportion. They are not simultaneous bands and do not incorporate dependence between spatial locations.

Value

`sp_calibration()` returns a `spmodels_calibration` object containing the unrounded nominal and observed coverage values, Wilson bounds and the number of locations. The print and plot methods return `x` invisibly.

sp_control

Control Gaussian Spatial Model Fitting

Description

Creates controls for Gaussian geostatistical fitting and shared non-Gaussian covariance settings. The nugget may be fixed independently of the other covariance parameters, including at exactly zero.

Usage

```

sp_control(
  smoothness = 0.5,
  spatial_variance = NULL,
  scale = NULL,
  nugget_variance = NULL,
  random_variance = NULL,
  estimate = TRUE,
  estimate_nugget = NULL,
  distance_units = NULL,

```

```

    maxit = 500L,
    reltol = 1e-8
  )

```

Arguments

smoothness	Fixed positive Matern smoothness.
spatial_variance, scale	Optional positive starting values when estimate = TRUE. Fixed spatial fitting requires both.
nugget_variance	Optional non-negative nugget variance. It is a starting value when estimated and a fixed value when estimate_nugget = FALSE. Supplying zero fixes the nugget at zero unless estimate_nugget = TRUE, which requires a positive start.
random_variance	Optional positive scalar or vector of starting or fixed Gaussian random-intercept variances. Named values must match the grouping variables in random.
estimate	Whether to estimate the spatial variance, scale, and supported additional variance components by maximum likelihood.
estimate_nugget	Optional logical value controlling nugget estimation when estimate = TRUE. The default estimates a missing or positive nugget and fixes a supplied zero nugget. Set it to FALSE to fix any supplied non-negative nugget value.
distance_units	A non-empty description of the units shared by both coordinate axes and used for model distances and covariance scales, such as "km". Numeric coordinates are assumed to use these units already. Projected sf coordinates with a recognised linear CRS unit are converted before fitting.
maxit	Positive maximum number of optimisation iterations.
reltol	Positive optimisation tolerance.

Value

An object of class `spmodels_control`.

sp_cross_validate	<i>Fixed-Fold Cross-Validation for an SPmodels Fit</i>
-------------------	--

Description

Refits the original model after withholding every declared fold, then predicts held-out observations. Reports RMSE, mean absolute error and mean negative log score. Fixed user-supplied fold identifiers are preserved.

Usage

```

sp_cross_validate(
  object,
  folds = 5L,
  seed = NULL,
  unseen = c("error", "marginal")
)

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

```

Arguments

object	A fitted <code>spmodels_fit</code> object retaining aligned source data and controls.
folds	An integer number of balanced randomly assigned folds, or one fixed non-missing fold identifier per fitted observation, or an object from <code>sp_buffered_folds()</code> , <code>sp_block_folds()</code> or <code>sp_environmental_folds()</code> .
seed	Optional local seed used only when folds is a count.
unseen	Policy for held-out grouping levels absent from a training fold. The safe default errors; "marginal" integrates new zero-mean grouped effects.
x	An object returned by <code>sp_cross_validate()</code> .
...	Reserved for print-method compatibility.

Details

Cross-validation currently requires any offset to be written inside the model formula. Random folds are a convenience for reproducible checks; spatially separated folds should normally be supplied for spatial validation. Buffered folds remove near-test rows from training independently for each fold. Group-constrained folds that withhold complete fitted random-effect levels generally require the explicit marginal unseen-level policy.

Value

An object of class `spmodels_cross_validation` containing held-out row predictions, overall metrics, fold metrics, resolved fold identifiers, per-fold training sizes and optional buffer distance.

sp_diagnostics

Fitted Spatial-Model Diagnostics

Description

Provides family-appropriate residual quantities for checking a fitted spatial model. Gaussian decorrelated residuals and non-Gaussian Pearson, deviance or randomized quantile residuals have different interpretations and are labelled separately.

Usage

```
sp_diagnostics(object, seed = NULL)

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

Arguments

object	A fitted spmodels_fit object.
seed	Optional integer seed for non-Gaussian randomized quantile residuals. The caller's random-number state is restored.
x	An object returned by sp_diagnostics().
...	Additional graphical arguments for the residual-index panel.

Value

For Gaussian fits, sp_diagnostics() returns mean-model, conditional and decorrelated residuals. For binomial and Poisson fits it returns response, Pearson, signed deviance and randomized quantile residuals at the fitted latent mode. Coordinates, distance units and family are retained. The plot method invisibly returns its input.

sp_distance_summary *Summarise Pairwise Spatial Distances*

Description

Calculates all unordered Euclidean coordinate-pair distances and reports counts, duplicate pairs, mean, standard deviation and requested type-8 quantiles.

Usage

```
sp_distance_summary(
  data,
  coords = NULL,
  distance_units = NULL,
  duplicates = c("include", "exclude", "error"),
  probs = c(0, 0.25, 0.5, 0.75, 1)
)

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

Arguments

data	A spatial SPmodels fit, projected point sf, or data frame.
coords	A two-column coordinate formula for an ordinary data frame.
distance_units	Explicit coordinate units for an ordinary data frame.
duplicates	Include, exclude, or reject repeated-location pairs.
probs	Finite quantile probabilities between zero and one.
x	A distance-summary object.
...	Reserved for method compatibility.

Value

An object containing retained pair distances, a one-row summary, units and the duplicate policy.

sp_environmental_folds

Construct Environmental-Space Validation Folds

Description

Clusters unique fitted locations in a declared numeric environmental space. All rows at one co-ordinate remain together and must have identical environmental values, preventing exact-location leakage.

Usage

```
sp_environmental_folds(
  object,
  variables,
  folds = 5L,
  scale = TRUE,
  seed = NULL,
  nstart = 20L
)

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

Arguments

object	A fitted spatial spmodels_fit.
variables	Unique character column names or a one-sided additive formula selecting numeric environmental variables from fitted data.
folds	Number of environmental clusters.
scale	Standardize variables before clustering?

seed	Optional integer seed local to k-means.
nstart	Positive number of random k-means starts.
x	An object returned by sp_environmental_folds().
...	Reserved for print-method compatibility.

Details

Variables are standardized by default so measurement units do not arbitrarily determine Euclidean k-means distance. The resulting folds test transfer across environmental regimes; they are not spatial distance buffers and need not be geographically compact.

Value

An integer fold vector in fitted-row order with class `spmodels_environmental_folds`. Environmental centres, variables, scaling values and seed are retained as attributes.

sp_fit	<i>Fit a Geostatistical Regression</i>
--------	--

Description

Fits Gaussian identity-link spatial regression by maximum likelihood, or binomial-logit and Poisson-log latent Gaussian spatial regression by a Laplace approximation. All models use a linear mean and stationary isotropic Matern covariance with fixed smoothness, optionally multiplied by exponential temporal correlation.

Usage

```
sp_fit(
  formula,
  data,
  family = stats::gaussian(),
  coords = NULL,
  spatial = TRUE,
  time = NULL,
  random = NULL,
  intervention = NULL,
  weights = NULL,
  subset = NULL,
  na.action = stats::na.fail,
  offset = NULL,
  control = sp_control(),
  verbose = FALSE,
  random_slope = NULL
)
```

Arguments

formula, data	A model formula and either a data frame or projected point sf object.
family	A Gaussian identity, binomial logit, or Poisson log family.
coords	For a data frame, a one-sided formula selecting exactly two numeric Euclidean coordinate columns. Do not supply this for sf data.
spatial	Fit the spatial model? FALSE fits the independent Gaussian limiting model and does not require coordinates.
random	For any supported spatial family, an optional one-sided additive formula of independent latent Gaussian random-intercept grouping variables, such as <code>~ clinic + survey</code> .
random_slope	An optional grouped random-slope specification made by <code>sp_random_slope()</code> , or a list of additive specifications. Each covariance may be fixed or estimated through constrained Cholesky parameters.
time	Optional separable temporal specification created by <code>sp_temporal()</code> . Every supported spatial family may use a fixed or jointly estimated temporal scale.
intervention	Optional decay-adjusted intervention history created by <code>sp_intervention()</code> for a binomial-logit fit. Its impact parameters may be fixed or estimated by the labelled Laplace route.
weights	Integer binomial trial counts for a proportion response. Gaussian and Poisson weights are not implemented.
subset	Optional row subset following <code>model.frame</code> conventions.
na.action	Missing-value action; the default fails explicitly.
offset	Optional known linear-predictor offset.
control	An object returned by <code>sp_control()</code> .
verbose	Whether to trace optimisation.

Details

Data frames use two numeric Euclidean coordinate columns and require explicit common units. Point sf data must have a known projected CRS; longitude/latitude geometry is rejected rather than silently projected. New sf prediction data must use the fitted CRS. Binomial responses may be Bernoulli values, a two-column successes/failures matrix, or proportions with integer trial weights. Poisson exposure should be supplied through a log offset. Non-Gaussian fits support point and joint prediction; those predictive calculations use a local Gaussian latent approximation at the fitted parameter values. Latent Gaussian random intercepts add one independent effect per observed level of each grouping variable. Prediction is conditional at known levels by default; `unseen = "marginal"` in prediction methods integrates a new zero-mean effect with its fitted variance. Positive-definite grouped random-slope covariance matrices may additionally be supplied or estimated with `sp_random_slope()`; correlated intercept-slope covariance is carried through fitting, prediction, simulation, diagnostics and MCML. Grouped effects are not available for non-spatial fits. A decay-adjusted intervention history may be supplied to a binomial-logit spatial fit with fixed or constrained estimated impact parameters. A `sp_temporal()` specification supplies one finite numeric time per row. The separable correlation is the product of the Matern spatial correlation and $\exp(-\text{abs}(\text{time}_i - \text{time}_j) / \text{temporal_scale})$. New prediction data must contain the declared time variable. Temporal scale may be estimated only when covariance estimation is enabled;

Gaussian ML and non-Gaussian Laplace/MCML fits carry it through prediction, simulation and validation.

Value

An object of class `smodels_fit`. Its standard methods include `coef`, `vcov`, `fitted`, `residuals`, `logLik`, `summary`, `print`, and `predict`.

Examples

```
d <- data.frame(y = c(1, 2, 1.5, 3), x = c(0, 1, 0, 1),
               east = c(0, 1, 0, 1), north = c(0, 0, 1, 1))
fit <- sp_fit(y ~ x, d, coords = ~ east + north,
             control = sp_control(spatial_variance = 1, scale = 1,
                                nugget_variance = 0.2, estimate = FALSE,
                                distance_units = "km"))

coef(fit)

survey <- transform(d, positive = c(0, 1, 0, 1), examined = 1)
binomial_fit <- sp_fit(
  cbind(positive, examined - positive) ~ x, survey,
  family = binomial(), coords = ~ east + north,
  control = sp_control(spatial_variance = 1, scale = 1,
                      nugget_variance = 0.2, estimate = FALSE,
                      distance_units = "km")
)
fitted(binomial_fit)
```

sp_fit_neighbor

Fit an approximate Gaussian nearest-predecessor model

Description

Fits a Gaussian identity-link geostatistical regression by multiplying ordered conditional densities that retain at most neighbors earlier observations. Only covariance blocks of at most that size are factorized. Retaining every predecessor recovers the dense Gaussian likelihood, regression coefficient estimates and coefficient covariance up to numerical precision. The approximation is explicit and never silently replaces `sp_fit()`.

Usage

```
sp_fit_neighbor(
  formula,
  data,
  coords,
  neighbors = 30L,
  ordering = c("input", "maximin"),
  observation_order = NULL,
  subset = NULL,
```

```

    na.action = stats::na.fail,
    offset = NULL,
    control = sp_control(),
    verbose = FALSE
)

## S3 method for class 'spmodels_neighbor_fit'
coef(object, ...)
## S3 method for class 'spmodels_neighbor_fit'
confint(object, parm, level = 0.95, ...)
## S3 method for class 'spmodels_neighbor_fit'
vcov(object, ...)
## S3 method for class 'spmodels_neighbor_fit'
logLik(object, ...)
## S3 method for class 'spmodels_neighbor_fit'
fitted(object, ...)
## S3 method for class 'spmodels_neighbor_fit'
residuals(object, ...)
## S3 method for class 'spmodels_neighbor_fit'
print(x, ...)
## S3 method for class 'spmodels_neighbor_fit'
summary(object, ...)
## S3 method for class 'summary.spmodels_neighbor_fit'
print(x, ...)
## S3 method for class 'spmodels_neighbor_fit'
predict(
  object,
  newdata = NULL,
  se.fit = FALSE,
  target = c("latent", "observation"),
  ...
)

```

Arguments

formula, data	A model formula and data frame.
coords	A one-sided formula selecting exactly two numeric coordinate columns. Coordinates must already be projected.
neighbors	Positive maximum number of earlier observations retained in each conditional density.
ordering	Use input-row order or deterministic maximin order.
observation_order	Optional permutation of observation-row positions. Supplying it overrides ordering.
subset, na.action, offset	Standard model-frame arguments.
control	Settings from <code>sp_control()</code> . The initial approximation supports the Gaussian spatial covariance only, without grouped or temporal covariance components.

verbose	Logical scalar controlling optimization tracing.
object, x	A fitted nearest-predecessor model or its summary.
parm	Optional fitted-parameter names or positions for confidence intervals.
level	Confidence probability strictly between zero and one.
newdata	Optional prediction data containing predictors and point coordinates. With NULL, return sequential training fitted values.
se.fit	Return approximate standard errors?
target	Predict the smooth latent target or a future observation.
...	Reserved for generic compatibility.

Details

The ordering and neighbor budget are part of the statistical approximation and are retained in the fitted object. Maximin ordering starts at row one and then selects the point farthest from the selected set. Prediction conditions each new point on its nearest training observations and does not claim a joint prediction covariance. Fitted-parameter standard errors use local observed curvature when it is positive definite.

This initial route deliberately rejects non-Gaussian, grouped, random-slope and space-time covariance models. Use the ordinary fitters for those models.

Value

sp_fit_neighbor() returns a spmodels_neighbor_fit object. It retains the processing order, predecessor graph, local coefficients and conditional variances. Generic methods return the corresponding coefficients, covariance, likelihood, fitted values, residuals, summaries or local conditional predictions.

Examples

```
d <- data.frame(
  y = c(1.2, 0.8, 1.7, 1.1, 1.5, 0.9),
  z = c(0, 1, 0, 1, 0, 1),
  east = c(0, 1, 2, 0, 1, 2),
  north = c(0, 0, 0, 1, 1, 1)
)
fit <- sp_fit_neighbor(
  y ~ z, d, coords = ~ east + north, neighbors = 2,
  control = sp_control(
    estimate = FALSE, smoothness = 0.5, spatial_variance = 1,
    scale = 1, nugget_variance = 0.1, distance_units = "km"
  )
)
summary(fit)
predict(fit, d[1:2, ], se.fit = TRUE)
```

sp_format_table	<i>Format a stable SPmodels table for presentation</i>
-----------------	--

Description

Creates a presentation-only view over a stable SPmodels table. It never changes the machine-readable values from which the view was produced.

Usage

```
sp_format_table(
  object,
  digits = 3L,
  p_digits = 3L,
  missing = "-",
  detail = c("summary", "fold")
)

## S3 method for class 'spmodels_formatted_table'
print(x, ..., row.names = FALSE)
```

Arguments

object	A data frame or an SPmodels fit, prediction, validation or simulation result supported by the stable table functions.
digits	Decimal places for ordinary numeric columns.
p_digits	Decimal places and lower display threshold for p_value columns.
missing	Character representation of missing values.
detail	Overall or fold detail for cross-validation.
x	A formatted SPmodels table.
...	Further arguments passed to <code>print.data.frame()</code> .
row.names	Display row names?

Value

A character data frame of class `spmodels_formatted_table`. The exact unrounded source remains available in the `raw_table` attribute.

sp_grid

Construct a Regular Prediction Grid

Description

Creates regular point centres within a projected polygon. If point data are supplied without a boundary, their convex hull defines the grid extent. Polygon data use their union. Geographic coordinates and zero-area boundaries fail explicitly.

Usage

```
sp_grid(data, cellsize, boundary = NULL)
```

Arguments

data	A non-empty projected sf object containing points or polygonal geometry.
cellsize	One positive grid spacing or two positive x/y spacings, in the coordinate units of data.
boundary	Optional projected polygon sf or sfc object using the same CRS as data.

Value

A point sf object with stable top-to-bottom, left-to-right grid_id values.

Examples

```
if (requireNamespace("sf", quietly = TRUE)) {
  points <- sf::st_as_sf(data.frame(x = c(0, 2, 0), y = c(0, 0, 2)),
                        coords = c("x", "y"), crs = 3857)
  grid <- sp_grid(points, cellsize = 0.5)
  nrow(grid)
}
```

sp_group_folds

Construct Group-Constrained Validation Folds

Description

Allocates whole observed groups to validation folds, preventing the same group from appearing in both training and held-out rows. Allocation greedily assigns each next group to the fold with the fewest rows.

Usage

```

sp_group_folds(
  object,
  group,
  folds = 5L,
  assignment = c("balanced", "random"),
  seed = NULL
)

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

```

Arguments

object	A fitted spmodels_fit object.
group	Name of one grouping column in the fitted data.
folds	Number of folds, not exceeding the observed group count.
assignment	Deterministic size-first or locally seeded randomized group order, followed by least-loaded allocation.
seed	Optional local seed for randomized assignment.
x	A group-fold object.
...	Reserved for print-method compatibility.

Value

An integer fold vector in fitted-row order. Group names, row counts and allocations are retained in the groups attribute.

sp_intervention	<i>Specify a Decay-Adjusted Intervention History</i>
-----------------	--

Description

Defines a fixed intervention history. Every intervention strictly before a survey multiplies baseline prevalence by $1 - \alpha * \exp(-(survey_time - intervention_time) / decay)$. Same-time interventions are excluded, corresponding to a survey immediately before rollout.

Usage

```

sp_intervention(
  history,
  unit,
  time,
  alpha,
  decay,

```

```

    time_units,
    estimate = FALSE,
    alpha_penalty = NULL,
    treated = NULL
  )

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

Arguments

history	Data frame with one row per unit/intervention time.
unit	Name of the unit-identifying column in history and survey data.
time	Name of the intervention-time column in history and survey-time column in survey data.
alpha	Immediate proportional reduction between zero and one.
decay	Positive decay time in the same units as time.
time_units	Character label for the units shared by time and decay.
estimate	Estimate alpha and decay during Laplace fitting?
alpha_penalty	Optional two positive beta-shape parameters for a normalized alpha penalty.
treated	Optional name of a zero-one history column.
x	An intervention specification.
...	Unused.

Value

An object of class `spmodels_intervention`.

`sp_intervention_curve` *Summarise a Decay-Adjusted Intervention Effect over Time*

Description

Calculates the remaining proportional reduction and one-intervention prevalence multiplier. Fitted models with usable intervention-parameter covariance receive delta-method pointwise intervals.

Usage

```

sp_intervention_curve(object, lag, level = 0.95)

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

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

Arguments

object	An intervention specification or fitted model containing one.
lag	Finite non-negative lags in the intervention time units.
level	Pointwise confidence level strictly between zero and one.
x	An intervention curve.
...	Additional graphical arguments for the plot method; unused by print.

Value

An object of class `smodels_intervention_curve` containing an unrounded values table and metadata.

`sp_intervention_multiplier`

Calculate Decay-Adjusted Intervention Multipliers

Description

Evaluates the intervention-history multiplier for survey or prediction rows.

Usage

```
sp_intervention_multiplier(intervention, data)
```

Arguments

intervention	An object returned by <code>sp_intervention()</code> .
data	A data frame containing the declared unit and survey-time columns.

Value

One multiplier in $[0, 1]$ per row of data.

sp_latent_mode

Calculate a Latent Gaussian Mode

Description

Finds the posterior mode of a latent Gaussian vector for independent binomial-logit or Poisson-log observations. This is a deterministic building block for later Monte Carlo maximum-likelihood fitting and does not itself estimate model parameters.

Usage

```
sp_latent_mode(
  y,
  linear_predictor,
  covariance,
  family = stats::binomial(),
  trials = NULL,
  probability_multiplier = NULL,
  initial = NULL,
  maxit = 100L,
  tolerance = 1e-8
)
```

Arguments

y	Numeric outcomes: binomial successes or Poisson counts.
linear_predictor	Finite fixed linear predictor before the latent vector.
covariance	Finite symmetric positive-definite latent covariance matrix.
family	A binomial-logit or Poisson-log family.
trials	Binomial integer trial counts; defaults to one per outcome.
probability_multiplier	Optional known binomial prevalence multiplier in (0, 1]; not used for Poisson outcomes.
initial	Optional finite initial latent vector.
maxit	Positive maximum Newton iterations.
tolerance	Positive convergence tolerance for the maximum step.

Value

A `spmodels_latent_mode` list containing the mode, inverse local curvature, gradient, iteration information, conditional means and log posterior.

sp_latent_sample	<i>Sample a Non-Gaussian Latent Spatial Vector</i>
------------------	--

Description

Runs a preconditioned Metropolis-adjusted Langevin algorithm for the supplied latent posterior. This samples latent effects conditional on supplied parameters; it is not by itself a fitted parameter model.

Usage

```
sp_latent_sample(
  y,
  linear_predictor,
  covariance,
  family = stats::binomial(),
  trials = NULL,
  initial = NULL,
  control = sp_mcmc_control(),
  seed = NULL
)
```

Arguments

y	Numeric binomial successes or Poisson counts.
linear_predictor	Finite fixed linear predictor before the latent vector.
covariance	Finite symmetric positive-definite latent covariance matrix.
family	A binomial-logit or Poisson-log family.
trials	Binomial integer trial counts; defaults to one per outcome.
initial	Optional finite initial latent vector, or a matrix with one row per chain and one column per latent component.
control	Settings from sp_mcmc_control().
seed	Optional integer seed local to this call.

Value

A `spmodels_latent_sample` object containing pooled draws, a draw-by-component-by-chain array, chain identities, chain-specific acceptance information, initial states, the latent mode and resolved controls.

sp_mcmc_control	<i>Control Latent MCMC Sampling</i>
-----------------	-------------------------------------

Description

Defines explicit iteration and proposal settings for the preconditioned Metropolis-adjusted Langevin sampler.

Usage

```
sp_mcmc_control(
  iterations = 4000L,
  burnin = 1000L,
  thin = 3L,
  step_size = 0.7,
  chains = 1L,
  initial_scale = 2
)
```

Arguments

iterations	Total positive number of MALA iterations.
burnin	Non-negative initial iterations discarded.
thin	Positive retention interval after burn-in.
step_size	Positive MALA step size.
chains	Positive number of independently evolved chains.
initial_scale	Non-negative multiplier for dispersed initial states around the latent mode when multiple chains are requested.

Value

An object of class `spmodels_mcmc_control`.

sp_mcml	<i>Refine a Non-Gaussian Fit by Monte Carlo Maximum Likelihood</i>
---------	--

Description

Replaces the Laplace fitting objective with an importance-sampling estimate of the binomial or Poisson marginal likelihood.

Usage

```
sp_mcml(object, control = sp_mcml_control(), seed = NULL, verbose = FALSE)
```

Arguments

object	A fitted binomial-logit or Poisson-log <code>smodels_fit</code> .
control	Settings returned by <code>sp_mcml_control()</code> .
seed	Optional integer seed local to this call.
verbose	Whether to trace optimization.

Details

The importance distribution is either a scaled Gaussian approximation to the latent posterior at the supplied fit or a normalized same-centre mixture of declared scales. Mixture draws are balanced across components and divided by the complete mixture density. The proposal is normalized, so the returned Monte Carlo log likelihood is absolute rather than known only up to an omitted constant. Common random numbers make optimization deterministic within a call. A small effective sample size or dominant normalized weight indicates that the importance approximation is unreliable. Mixture likelihood MCSE uses its component-stratified variance. With `adapt = TRUE`, an independent pilot supplies importance-weighted moments that are shrunk toward the Laplace moments. The pilot is diagnosed, retained for audit and discarded before a new main sample is drawn. Conditional on the pilot, weighting by the complete normalized adapted density preserves the same marginal likelihood. If pilot ESS is below `ess_warning`, the original moments are retained and the recorded adaptation status explains the fallback. Adaptation is not guaranteed to improve overlap. Downstream latent prediction continues to use local Gaussian conditional calculations at the MCML parameter estimates. If intervention impact and decay were declared estimated, they are re-estimated on logit and log scales and their optional normalized beta penalty is included in the MCML criterion.

Value

An updated `smodels_fit`. The `mcml` component contains the proposal, retained importance draws and weights, effective sample size, maximum normalized weight and an approximate Monte Carlo standard error. When adaptation is requested it also retains the original moments and complete pilot record.

Examples

```
d <- data.frame(y = c(0, 1, 0, 1), x = c(0, 1, 0, 1),
               east = c(0, 1, 0, 1), north = c(0, 0, 1, 1))
fit <- sp_fit(y ~ x, d, family = binomial(), coords = ~ east + north,
             control = sp_control(spatial_variance = 1, scale = 1,
                                nugget_variance = 0.2, estimate = FALSE, distance_units = "km"))
corrected <- sp_mcml(fit,
                    control = sp_mcml_control(nsim = 200, maxit = 20), seed = 1)
corrected$mcml$effective_sample_size
```

sp_mcml_control

*Control Monte Carlo Maximum-Likelihood Correction***Description**

Configures the normalized importance distribution, optional discarded-pilot adaptation and deterministic common-random-number optimization used by `sp_mcml()`.

Usage

```
sp_mcml_control(
  nsim = 2000L,
  maxit = 200L,
  reltol = 1e-06,
  proposal_scale = 1,
  ess_warning = 0.1,
  adapt = FALSE,
  pilot_nsim = 500L,
  adapt_shrinkage = 0.5
)
```

Arguments

<code>nsim</code>	Number of independent importance samples; at least 100.
<code>maxit</code>	Positive maximum number of optimization iterations.
<code>reltol</code>	Positive relative optimization tolerance.
<code>proposal_scale</code>	One or more distinct positive standard-deviation multipliers for the local Gaussian latent-posterior proposal. Multiple values define a normalized deterministic mixture with balanced allocation.
<code>ess_warning</code>	Importance-sampling effective-sample fraction strictly between zero and one below which <code>sp_mcml()</code> warns. When adaptation is requested, a pilot below this threshold retains the original proposal moments.
<code>adapt</code>	Whether to use an independent discarded pilot sample to estimate regularized proposal moments.
<code>pilot_nsim</code>	Number of pilot importance draws; at least 100.
<code>adapt_shrinkage</code>	Number greater than zero and no greater than one. It is the weight assigned to the original Laplace moments; the remaining weight is assigned to the pilot importance-weighted moments.

Value

An object of class `spmodels_mcml_control`.

sp_mcml_precision	<i>Escalate MCML sample size to declared precision targets</i>
-------------------	--

Description

Refits using geometrically increasing independent importance-sample sizes until both declared diagnostic targets are met or max_nsim is reached.

Usage

```
sp_mcml_precision(
  object,
  control = sp_mcml_control(),
  target_relative_mcse = 0.02,
  min_ess_fraction = 0.1,
  max_nsim = 32000L,
  growth = 2,
  seed = NULL,
  refresh = TRUE,
  verbose = FALSE
)
```

Arguments

object	A fitted binomial-logit or Poisson-log <code>spmodels_fit</code> .
control	Initial settings returned by <code>sp_mcml_control()</code> .
target_relative_mcse	Positive target for relative likelihood Monte Carlo standard error.
min_ess_fraction	Required effective-sample fraction strictly between zero and one.
max_nsim	Largest permitted importance-sample count.
growth	Multiplicative sample-size growth greater than one.
seed	Optional integer defining consecutive, call-local attempt seeds.
refresh	Whether an unsuccessful fit supplies the next proposal centre and curvature.
verbose	Logical scalar passed to <code>sp_mcml()</code> .

Value

The final `spmodels_fit`, with an `mcml_precision` component that records every attempted sample size, diagnostic and stopping reason. Adaptive attempts include pilot ESS, maximum weight, relative MCSE and the used/fallback decision.

sp_mcml_replicates	<i>Repeat Monte Carlo maximum-likelihood fits independently</i>
--------------------	---

Description

Runs `sp_mcml()` with independent importance samples. Replication diagnoses Monte Carlo sensitivity; it does not replace scientific sampling uncertainty or combine the fits into a new estimator. The empirical between-replicate covariance measures parameter-level Monte Carlo sensitivity. When every fitted Hessian covariance is usable, the object also reports their mean and its sum with the Monte Carlo covariance; otherwise those sampling and combined entries remain missing.

Usage

```
sp_mcml_replicates(
  object,
  nrep = 3L,
  control = sp_mcml_control(),
  seed = NULL,
  verbose = FALSE
)

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

Arguments

object	A fitted binomial-logit or Poisson-log <code>spmodels_fit</code> .
nrep	Number of independent fits, at least two.
control	Settings returned by <code>sp_mcml_control()</code> .
seed	Optional integer defining consecutive, call-local replicate seeds.
verbose	Logical scalar passed to <code>sp_mcml()</code> .
x	An object returned by <code>sp_mcml_replicates()</code> .
...	Unused.

Value

An object of class `spmodels_mcml_replicates` containing every fit, replicate estimates, between-replicate summaries and likelihood diagnostics. It also includes empirical Monte Carlo, mean fitted-Hessian sampling, and combined parameter covariance matrices on both optimizer and natural model scales. When adaptation is requested, diagnostics also contain pilot ESS, maximum weight, relative MCSE and the used/fallback decision.

sp_model_table	<i>Structured model-estimate table</i>
----------------	--

Description

Converts an SPmodels fit into a machine-readable estimate table without requiring downstream code to inspect internal fit fields.

Usage

```
sp_model_table(object, include_smoothness = TRUE)
```

Arguments

object	A fitted spmodels_fit object.
include_smoothness	Include the fixed Matérn smoothness in the table?

Value

A stable data frame with one row per mean or covariance parameter.

sp_neighbor_covariance	<i>Inspect Nearest-Predecessor Predictive Covariance</i>
------------------------	--

Description

Reconstructs the complete covariance implied by the nearest-predecessor sequential conditionals for diagnostic comparison.

Usage

```
sp_neighbor_covariance(
  object,
  newdata = NULL,
  neighbors = 30L,
  ordering = c("input", "maximin"),
  prediction_order = NULL,
  target = c("latent", "observation"),
  unseen = c("error", "marginal"),
  reference = FALSE
)

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

sp_nonstationary_cor *Non-stationary Matérn correlation*

Description

Construct a valid non-stationary Matérn correlation matrix from a local positive-definite scale matrix at each location. The construction combines local Mahalanobis distances with a determinant adjustment, so locally varying anisotropy and range yield a valid covariance rather than an arbitrary pairwise-distance kernel.

Usage

```
sp_nonstationary_cor(coords, local_covariance, smoothness = 0.5)
```

Arguments

coords	A finite numeric matrix with two columns, one row per location, in common projected Euclidean units.
local_covariance	A numeric 'n x 2 x 2' array, where slice 'i' is the symmetric positive-definite local scale matrix at 'coords[i,]'.
smoothness	A finite positive Matérn smoothness parameter, held fixed across locations.

Details

Each local matrix has units of coordinate-distance squared. An isotropic local matrix ' $\rho^2 * \text{diag}(2)$ ' therefore gives local range ' ρ '.

Value

A symmetric 'n x n' correlation matrix with unit diagonal. The determinant construction is valid for any supplied positive-definite local matrices; this function does not estimate them or fit a regression.

References

Paciorek, C. J. and Schervish, M. J. (2006). Spatial modelling using a new class of nonstationary covariance functions. doi:[10.1002/env.785](https://doi.org/10.1002/env.785).

Examples

```
locations <- cbind(east = c(0, 1, 2), north = c(0, 0.5, 1))
local <- array(0, dim = c(nrow(locations), 2, 2))
for (i in seq_len(nrow(locations))) {
  local[i, , ] <- diag(c(1 + i / 4, 0.8 + i / 5)^2, 2)
}
sp_nonstationary_cor(locations, local, smoothness = 1.5)
```

sp_polygon_area_weights	<i>Construct Exact Fractional Cell-Area Polygon Weights</i>
-------------------------	---

Description

Uses exact projected polygon intersections so a prediction cell can contribute proportionally to more than one target region.

Usage

```
sp_polygon_area_weights(  
  cells,  
  polygons,  
  region,  
  density = NULL,  
  unmatched = c("error", "drop"),  
  empty_regions = c("error", "drop")  
)
```

Arguments

cells	Projected polygon sf prediction cells in draw-column order.
polygons	Projected non-overlapping target polygons in the same CRS.
region	Column name holding non-missing region identifiers.
density	Optional finite non-negative cell values or a column name; weights are intersection area times density.
unmatched	Error for or drop cells with no positive-area intersection.
empty_regions	Error for or record/drop regions with no positive-area cell intersection.

Value

An spmodels_polygon_weights object accepted by sp_predict_target(), retaining cell indices, region groups, raw weights, intersection areas, cell fractions and excluded cells/regions.

sp_polygon_weights	<i>Assign Point Prediction Weights to Polygon Regions</i>
--------------------	---

Description

Uses projected point-in-polygon membership to define regional weights for joint predictive draws. Points assigned to multiple distinct region identifiers fail explicitly. This is centre-point allocation, not an exact cell-area overlay.

Usage

```
sp_polygon_weights(
  points,
  polygons,
  region,
  weight = NULL,
  unmatched = c("error", "drop")
)
```

Arguments

points	A non-empty projected point sf object in the same row order as the prediction draws.
polygons	A non-empty projected polygon sf object using the same CRS.
region	One column name in polygons containing non-missing region identifiers.
weight	Optional numeric point weights or one numeric column name in points. Equal raw weights are used by default.
unmatched	Fail for points outside every polygon, or drop them from subsequent aggregation.

Value

An object of class `spmodels_polygon_weights` containing retained point indices, raw weights, region identifiers, CRS and unmatched indices. It can be passed directly as weights to `sp_predict_target()`.

sp_predict_chunked	<i>Predict a Large Point Set in Bounded-Memory Chunks</i>
--------------------	---

Description

Applies ordinary point prediction to consecutive row chunks and combines results in original order. This is an exact computational reorganization, not a covariance approximation.

Usage

```
sp_predict_chunked(
  object,
  newdata,
  chunk_size = 1000L,
  se.fit = FALSE,
  target = c("latent", "observation"),
  type = c("response", "link"),
  unseen = c("error", "marginal")
)
```

Arguments

object	A fitted <code>spmodels_fit</code> object.
newdata	Non-empty prediction data containing every required predictor, coordinate, time and grouping column.
chunk_size	Positive maximum number of prediction rows per chunk.
se.fit	Return pointwise standard errors as well as fitted values?
target	Predict the latent process or a future observation.
type	Return response- or link-scale predictions.
unseen	Policy for grouping levels absent during fitting.

Details

The complete training-by-prediction cross-covariance matrix is never held in memory. Peak cross-covariance storage is proportional to the number of training rows times `chunk_size`. The function returns pointwise means and optional marginal standard errors; use `sp_predict_draws()` when a joint covariance or dependent draws across all prediction rows are required.

Value

The same numeric vector or pointwise list schema as `predict()`, with attributes `chunk_size` and `chunks`.

Examples

```
d <- data.frame(y = c(1, 2, 1.5, 3), x = c(0, 1, 0, 1),
               east = c(0, 1, 0, 1), north = c(0, 0, 1, 1))
fit <- sp_fit(y ~ x, d, coords = ~ east + north,
             control = sp_control(spatial_variance = 1, scale = 1,
                                nugget_variance = .2, estimate = FALSE,
                                distance_units = "km"))
sp_predict_chunked(fit, d, chunk_size = 2L)
```

sp_predict_draws

Joint Predictive Draws from a Fitted Spatial Model

Description

Draws jointly across prediction locations from the exact conditional Gaussian distribution or the local Gaussian predictive approximation for a binomial-logit or Poisson-log fit. Joint drawing preserves the spatial dependence needed for grid and regional targets.

Usage

```

sp_predict_draws(
  object,
  newdata = NULL,
  nsim = 1000L,
  target = c("latent", "observation"),
  type = c("response", "link"),
  seed = NULL,
  parameter_uncertainty = FALSE,
  unseen = c("error", "marginal")
)

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

```

Arguments

<code>object</code>	A fitted <code>spmodels_fit</code> object.
<code>newdata</code>	Optional prediction data containing all predictors and point coordinates. The fitted locations are used when this is <code>NULL</code> .
<code>nsim</code>	Positive number of joint draws.
<code>target</code>	Draw the smooth latent target or a future observation-site target. The latter includes the independent variance component.
<code>type</code>	Draw on the response or linear-predictor scale. For binomial and Poisson fits, latent response draws are probabilities or intensities; observation response draws are Bernoulli outcomes or Poisson counts.
<code>seed</code>	Optional integer seed local to the call.
<code>parameter_uncertainty</code>	Include a local Gaussian approximation to fitted regression and covariance-parameter uncertainty? This requires a usable fitted Hessian covariance. Gaussian covariance and regression blocks are treated as asymptotically independent. Fitted intervention parameters rebuild the intervention multiplier and latent state for every draw.
<code>unseen</code>	Policy for grouping levels absent during fitting. "error" rejects them; "marginal" draws new zero-mean random intercepts with the fitted variance, shared by rows carrying the same new level.
<code>x</code>	An object returned by <code>sp_predict_draws()</code> .
<code>...</code>	Reserved for print-method compatibility.

Details

For non-Gaussian models, response-scale latent draws transform the joint linear-predictor draw into probabilities or intensities. Observation draws additionally apply Bernoulli or Poisson response sampling. Regression- and covariance-parameter uncertainty is omitted by default. When requested, the returned `parameter_draws` record the regression coefficients, log covariance parameters and any logit-impact/log-decay values used for each predictive draw. Draw-specific intervention multipliers are also retained.

Value

An object of class `smodels_prediction_draws`. Its `draws` component is an `nsim` by number-of-locations matrix. The object also contains the predictive link mean and covariance and target metadata.

Examples

```
d <- data.frame(y = c(1, 2, 1.5, 3), x = c(0, 1, 0, 1),
               east = c(0, 1, 0, 1), north = c(0, 0, 1, 1))
fit <- sp_fit(y ~ x, d, coords = ~ east + north,
             control = sp_control(spatial_variance = 1, scale = 1,
                                nugget_variance = 0.2, estimate = FALSE,
                                distance_units = "km"))
draws <- sp_predict_draws(fit, d[1:2, ], nsim = 10, seed = 1)
dim(draws$draws)
```

`sp_predict_draws_neighbor`

Bounded-Memory Approximate Joint Predictive Draws

Description

Generates joint draws from a nearest-predecessor factorization of the fitted Gaussian predictive distribution without allocating the complete prediction-site covariance matrix.

Usage

```
sp_predict_draws_neighbor(
  object,
  newdata = NULL,
  nsim = 1000L,
  neighbors = 30L,
  ordering = c("input", "maximin"),
  prediction_order = NULL,
  target = c("latent", "observation"),
  type = c("response", "link"),
  seed = NULL,
  parameter_uncertainty = FALSE,
  unseen = c("error", "marginal")
)
```

Arguments

<code>object</code>	A fitted spatial <code>smodels_fit</code> object.
<code>newdata</code>	Optional prediction data containing all predictors and point coordinates. The fitted locations are used when this is <code>NULL</code> .
<code>nsim</code>	Positive number of joint draws.

sp_predict_target	<i>Summarise Targets from Joint Predictive Draws</i>
-------------------	--

Description

Calculates means, standard deviations, quantiles and optional exceedance probabilities from joint predictive draws. Optional grouped weighted means preserve uncertainty because aggregation is performed within every joint draw before summarisation.

Usage

```
sp_predict_target(
  object,
  weights = NULL,
  groups = NULL,
  threshold = NULL,
  probs = c(0.025, 0.5, 0.975)
)

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

Arguments

object	An object returned by <code>sp_predict_draws()</code> .
weights	Optional finite non-negative weight for every prediction location, or an object returned by <code>sp_polygon_weights()</code> . Weights are normalised within each group.
groups	Optional non-missing group identifier for every location. It requires weights; the default is one aggregate target.
threshold	Optional finite scalar used to calculate exceedance probabilities.
probs	Finite quantile probabilities between zero and one.
x	An object returned by <code>sp_predict_target()</code> .
...	Reserved for print-method compatibility.

Value

An object of class `spmodels_prediction_target` containing target-level draws, a summary table, normalised weights, groups, threshold and scale metadata.

sp_predict_target_neighbor

Stream Bounded-Memory Approximate Predictive Targets

Description

Generates nearest-predecessor joint prediction draws and immediately accumulates grouped weighted targets, avoiding storage of a complete location-level draw matrix when only aggregate targets are needed.

Usage

```
sp_predict_target_neighbor(
  object,
  newdata = NULL,
  weights,
  groups = NULL,
  nsim = 1000L,
  neighbors = 30L,
  ordering = c("input", "maximin"),
  prediction_order = NULL,
  target = c("latent", "observation"),
  type = c("response", "link"),
  threshold = NULL,
  probs = c(0.025, 0.5, 0.975),
  seed = NULL,
  unseen = c("error", "marginal")
)
```

Arguments

object	A fitted spatial <code>smodels_fit</code> object.
newdata	Optional point prediction data.
weights	Finite non-negative weights matching prediction rows, or a polygon-weight object.
groups	Optional target identifier per prediction row. It requires numeric weights; the default is one target.
nsim	Positive number of draws.
neighbors	Positive predecessor budget.
ordering	Input or deterministic maximin processing order.
prediction_order	Optional explicit permutation of prediction rows.
target	Latent or future observation-site target.
type	Response or link scale.

threshold	Optional finite target exceedance threshold.
probs	Finite target quantile probabilities in zero–one.
seed	Optional integer seed local to the call.
unseen	Policy for grouping levels absent during fitting.

Details

A point-draw vector is discarded after its final use in the declared neighbor graph. Working draw storage is therefore proportional to the graph’s active width rather than automatically to every prediction location, in addition to the returned number-of-draws by number-of-targets matrix. Numeric and polygon allocation weights are normalized within target. Parameter uncertainty is not included. Observation-scale binomial/Poisson sampling is generated during streaming, so its seeded draws need not equal a post-hoc call that consumes random numbers in a different order.

Value

An `spmodels_prediction_target` containing target-level draws and summaries, sparse normalized weights, the neighbor graph, conditional variances, processing order and peak number of point-draw vectors retained.

Examples

```
d <- data.frame(y = c(1, 2, 1.5, 3), x = c(0, 1, 0, 1),
               east = c(0, 1, 0, 1), north = c(0, 0, 1, 1))
fit <- sp_fit(y ~ x, d, coords = ~ east + north,
             control = sp_control(spatial_variance = 1, scale = 1,
                                nugget_variance = 0.2, estimate = FALSE, distance_units = "km"))
target <- sp_predict_target_neighbor(
  fit, d, weights = rep(1, nrow(d)), nsim = 10,
  neighbors = 2, seed = 1
)
target$summary
```

sp_prediction_table	<i>Structured prediction-summary table</i>
---------------------	--

Description

Summarises joint prediction draws at each location or returns an existing grouped-target summary using a stable downstream schema.

Usage

```
sp_prediction_table(object, threshold = NULL, probs = c(0.025, 0.5, 0.975))
```

Arguments

object	An object returned by an exact or nearest-predecessor prediction-draw or prediction-target function.
threshold	Optional finite exceedance threshold for prediction draws.
probs	Numeric quantile probabilities between zero and one.

Value

A data frame of draw summaries and prediction metadata.

sp_prediction_update *Compare predictions under an updated predictor scenario*

Description

Constructs an explicit prediction scenario without refitting or mutating the fitted model. Coordinates, geometry, row order and unmentioned columns remain unchanged. Formula transformations, factors and offset() expressions are rebuilt through the fitted model's terms and contrast definitions.

Usage

```
sp_prediction_update(
  object,
  newdata,
  changes,
  rows = NULL,
  se.fit = FALSE,
  target = c("latent", "observation"),
  type = c("response", "link")
)

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

Arguments

object	A fitted spmodels_fit object.
newdata	Non-empty prediction data accepted by predict().
changes	A non-empty named list replacing fitted predictor or in-formula offset variables. Each value has length one or the number of selected rows. Named values must exactly match the selected row names.
rows	Rows to update: NULL for all rows, a logical vector, unique integer indices, or unique row names.
se.fit	Include baseline and updated pointwise standard errors.

target	Predict the latent process or a future observation.
type	Return response- or link-scale predictions.
x	An object returned by sp_prediction_update().
...	Reserved for print-method compatibility.

Details

Coordinate variables cannot be changed. Offsets supplied separately to the original fitting call are rejected because they cannot be reconstructed safely from prediction data; write an offset in the model formula for scenario use. The returned differences compare point predictions conditional on the same fitted parameter values and fitted latent state.

Value

An object of class `smodels_prediction_update`. It contains the updated data; baseline and updated design matrices and offsets; row-level baseline, updated and difference values; optional standard errors; changed rows and columns; and preservation checks.

sp_predictive_log_score

Integrated Pointwise Predictive Log Scores

Description

Evaluates Gaussian predictive normal scores or integrates a binomial/Poisson response probability over the approximate Gaussian predictive distribution on the link scale.

Usage

```
sp_predictive_log_score(
  object,
  newdata,
  observed,
  trials = NULL,
  unseen = c("error", "marginal")
)
```

Arguments

object	A fitted <code>smodels_fit</code> model.
newdata	Prediction data containing all required model fields.
observed	One finite observed response per row. Binomial values are success counts.
trials	Optional non-negative integer binomial trial totals. Defaults to one for binomial models and must be omitted otherwise.
unseen	Policy for grouping levels absent from the fitted data.

Details

The non-Gaussian integral is one-dimensional for each prediction row and conditional on fitted parameters. Binomial intervention multipliers are included. A failed or numerically underflowed integral produces an explicit error rather than a clipped score.

Value

A numeric vector of negative log predictive densities.

sp_random_slope	<i>Specify a Grouped Random Slope</i>
-----------------	---------------------------------------

Description

Define a grouped Gaussian coefficient vector whose covariance is supplied explicitly or used as the starting value for constrained estimation. Each group receives an independent copy of that vector. With `intercept = TRUE`, the vector contains a random intercept and a random slope and covariance is therefore a 2 by 2 matrix. This explicit covariance interface keeps assumptions visible in teaching. Estimation is opt-in and retains positive definiteness at every optimizer evaluation.

Usage

```
sp_random_slope(
  group,
  slope,
  covariance,
  intercept = TRUE,
  label = NULL,
  estimate = FALSE
)

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

Arguments

group	Name of the grouping variable.
slope	Name of the numeric random-slope variable.
covariance	Positive-definite covariance matrix for the grouped coefficients. Its order is intercept then slope when <code>intercept = TRUE</code> .
intercept	Include a grouped random intercept as well as the slope?
label	Optional unique label used in summaries.
estimate	Estimate the covariance through a positive-definite Cholesky parameterization? The supplied matrix is then the starting value.
x	An object returned by <code>sp_random_slope()</code> .
...	Reserved for print-method compatibility.

Value

An object for the random_slope argument of sp_fit().

Examples

```
fixed <- sp_random_slope(
  "clinic", "age_centered", matrix(c(0.3, 0.05, 0.05, 0.08), 2L)
)
estimated <- sp_random_slope(
  "clinic", "age_centered", matrix(c(0.3, 0.05, 0.05, 0.08), 2L),
  estimate = TRUE
)
```

sp_report

Assemble an SPmodels analysis report

Description

Assembles model, optional prediction and optional validation sections while keeping presentation formatting separate from all unrounded statistical output.

Usage

```
sp_report(
  object,
  prediction = NULL,
  validation = NULL,
  title = "SPmodels analysis report",
  digits = 3L,
  p_digits = 3L,
  validation_detail = c("summary", "fold")
)

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

Arguments

object	A fitted spmodels_fit object.
prediction	Optional joint-prediction draws or prediction-target result.
validation	Optional cross-validation, predictive-assessment or simulation-recovery result.
title	One non-empty report title.
digits, p_digits	Presentation precision passed to sp_format_table().
validation_detail	Overall or fold-level cross-validation output.
x	An assembled spmodels_report.
...	Reserved for print-method compatibility.

Value

An `spmodels_report` containing analysis metadata, unrounded stable tables and separate formatted views.

`sp_sampler_diagnostics`

Diagnose Retained Latent Samples

Description

Calculates autocorrelation-aware effective sample sizes and Monte Carlo standard errors. With multiple chains it also calculates the maximum of rank-normalized split and folded rank-normalized split R-hat.

Usage

```
sp_sampler_diagnostics(object, component = 1L, max_lag = NULL, chains = NULL)
```

```
## S3 method for class 'spmodels_sampler_diagnostics'
```

```
plot(x, ...)
```

```
## S3 method for class 'spmodels_sampler_diagnostics'
```

```
print(x, ...)
```

Arguments

<code>object</code>	An object returned by <code>sp_latent_sample()</code> .
<code>component</code>	Component index or name selected for trace and ACF output.
<code>max_lag</code>	Positive maximum lag smaller than retained sample size.
<code>chains</code>	Optional distinct chain indices or names. The default uses all available chains.
<code>x</code>	A sampler-diagnostics object.
<code>...</code>	Additional graphical arguments for the trace panel.

Value

A `spmodels_sampler_diagnostics` object containing per-component means, standard deviations, multi-chain effective sample sizes, Monte Carlo standard errors and rank-normalized split R-hat values, plus chain acceptance rates and selected trace/autocorrelation values. Methods invisibly return the object.

`sp_simulation_assess` *Assess Predictive Draws Against Known Values*

Description

Assesses a joint predictive draw ensemble against known simulated truth or held-out observations. All interval and score definitions are calculated directly from the retained draws.

Usage

```
sp_simulation_assess(object, truth, interval = 0.95, threshold = NULL)
```

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

Arguments

<code>object</code>	An object returned by <code>sp_predict_draws()</code> .
<code>truth</code>	One finite value for every prediction location.
<code>interval</code>	Central interval probability strictly between zero and one.
<code>threshold</code>	Optional finite threshold for predicted exceedance probabilities and observed threshold indicators.
<code>x</code>	An object returned by <code>sp_simulation_assess()</code> .
<code>...</code>	Reserved for print-method compatibility.

Value

An object of class `spmodels_simulation_assessment` containing location-level errors, intervals, coverage and empirical CRPS, plus overall bias, RMSE, MAE, coverage, interval width and mean CRPS.

`sp_simulation_grid` *Construct a Cartesian Grid of Simulation-Recovery Scenarios*

Description

Builds a reproducible Cartesian parameter design for `sp_simulation_scenarios()`. Each object retains the fitted design and refitting specification while its generative mean and covariance state are rebuilt at the declared truth.

Usage

```
sp_simulation_grid(object, values, prefix = "scenario")
```

Arguments

object	A fitted <code>spmodels_fit</code> object defining the design, response family and refitting specification.
values	A named list of non-empty finite numeric vectors. Names must identify parameters that the fitted specification estimates. Random-effect variances use names such as <code>random::clinic</code> ; estimated grouped slope covariance entries use their qualified <code>random_slope::model-table</code> names.
prefix	A non-empty prefix for deterministic scenario names.

Details

These are simulation scenario objects, not new analyses of the observed response. Parameters fixed by the model specification cannot be varied because they would not be recoverable on refitting. Variance, scale and decay values must be positive; an estimated intervention alpha must be strictly between zero and one. Every grouped slope covariance combination must remain positive definite.

Value

A named list of generative `spmodels_fit` scenario objects. Its `grid` attribute records the Cartesian product in scenario order.

`sp_simulation_precision`

Precision-Directed Simulation Recovery

Description

Repeats a complete fitted-model recovery experiment at geometrically increasing replicate counts until the Monte Carlo standard error of mean bias passes for every recovered parameter or the exact cap is evaluated.

Usage

```
sp_simulation_precision(
  object,
  initial_nsim = 20L,
  max_nsim = 1000L,
  growth = 2,
  target_mcse,
  min_success = 10L,
  seed = NULL,
  level = 0.95,
  on_error = c("record", "stop"),
  workers = 1L
)

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

Arguments

object	A fitted <code>smodels_fit</code> .
initial_nsim	Initial positive replicate count.
max_nsim	Maximum replicate count, not below the initial count.
growth	Finite multiplicative increase strictly greater than one.
target_mcse	Finite positive target for every recovered parameter's bias Monte Carlo standard error.
min_success	Minimum successful refits required per parameter.
seed	Optional base seed. Attempt seeds advance deterministically.
level	Wald interval probability passed to recovery.
on_error	Record failed refits or stop immediately.
workers	Positive number of replicate workers passed to every complete recovery attempt.
x	An object returned by <code>sp_simulation_precision()</code> .
...	Reserved for print-method compatibility.

Details

For parameter j , the monitored quantity is the empirical standard deviation of successful replicate estimates divided by the square root of their count. Each attempt is a complete independently seeded experiment. Failure at the cap remains explicitly false; criteria are never relaxed to manufacture a pass.

Value

An object containing the final `smodels_simulation_recovery`, every attempted replicate count and seed, parameter-wise achieved MCSE and success count, and the target status.

`sp_simulation_recovery`

Repeated Fitted-Model Simulation and Parameter Recovery

Description

Simulates response datasets from the supplied fitted model and refits the same declared estimation route to every replicate. The summary reports bias, RMSE, empirical standard deviation, mean reported standard error and Wald interval coverage. MCML is repeated when the supplied fit used MCML.

Usage

```

sp_simulation_recovery(
  object,
  nsim = 100L,
  seed = NULL,
  level = 0.95,
  on_error = c("record", "stop"),
  workers = 1L
)

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

```

Arguments

<code>object</code>	A fitted <code>spmodels_fit</code> .
<code>nsim</code>	Positive number of response simulation/refit replicates.
<code>seed</code>	Optional integer seed local to the operation.
<code>level</code>	Wald interval probability strictly between zero and one.
<code>on_error</code>	Record failed refits and continue, or stop on failure. Sequential execution stops immediately; already-dispatched parallel work may finish before the error is returned.
<code>workers</code>	Positive number of replicate workers. Values above one use forked processes on non-Windows systems. MCML recovery then requires an explicit seed.
<code>x</code>	An object returned by <code>sp_simulation_recovery()</code> .
<code>...</code>	Reserved for print-method compatibility.

Details

The fitted parameter values are treated as simulation truth. This is a conditional fitted-model assessment, not evidence of performance across a scientific parameter space. Named responses and two-column binomial responses are supported. External offset arguments must instead be written in the model formula so every refit is self-contained. The recovery vector includes every estimated mean, spatial/nugget covariance, grouped random-intercept variance, temporal-scale and intervention parameter; fixed quantities are not reported as recovered estimates.

Value

An object containing simulated responses, replicate estimates and standard errors, convergence codes, recorded failures, fitted truth and a parameter-level recovery summary.

sp_simulation_scenarios

Run Parameter-Recovery Experiments Across Fitted Scenarios

Description

Runs the existing fitted-model simulation and refitting contract over multiple explicitly named fitted scenarios. Each result remains available for audit; the combined summary adds a scenario column without pooling unlike parameters or replacing individual failures.

Usage

```
sp_simulation_scenarios(
  objects,
  nsim = 100L,
  seed = NULL,
  level = 0.95,
  on_error = c("record", "stop"),
  workers = 1L
)

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

Arguments

objects	A named, non-empty list of fitted spmodels_fit objects. Names are retained as scenario identifiers.
nsim	One positive integer replicate count for every scenario, or one count per scenario. Named values are matched exactly to scenario names.
seed	Optional base integer seed. Successive scenarios use successive deterministic seeds.
level	Wald interval probability strictly between zero and one.
on_error	Record failed refits and continue, or stop immediately.
workers	Positive number of scenario workers. Values above one use forked processes on non-Windows systems; Windows currently requires one.
x	An object returned by sp_simulation_scenarios().
...	Reserved for print-method compatibility.

Details

The scenario models define their own fitted simulation truths and designs. This function orchestrates separate conditional experiments; it does not claim that the supplied scenarios form a scientifically sufficient design. When a base seed is supplied, each scenario is reproducible and the caller's random-number state remains unchanged. Scenario seeds are fixed before dispatch, so sequential and parallel execution are identical.

Value

An object containing every named `smodels_simulation_recovery` result, their row-bound parameter summary, and scenario-level family, replicate, seed and successful-refit metadata.

sp_spatial_folds	<i>Construct Geometry-Driven Spatial Folds</i>
------------------	--

Description

Clusters unique fitted coordinates into spatially compact folds. Repeated coordinate locations always receive the same fold, preventing a physical location from appearing in both training and held-out data. Cluster labels are canonicalized from top to bottom and then left to right. The result can be supplied directly to `sp_cross_validate()`.

Usage

```
sp_spatial_folds(object, folds = 5L, seed = NULL, nstart = 20L)
```

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

Arguments

<code>object</code>	A fitted spatial <code>smodels_fit</code> .
<code>folds</code>	Integer number of spatial folds.
<code>seed</code>	Optional integer seed local to fold construction.
<code>nstart</code>	Positive number of random k-means starts.
<code>x</code>	An object returned by <code>sp_spatial_folds()</code> .
<code>...</code>	Reserved for print-method compatibility.

Details

This is k-means blocking in the fitted projected or declared Euclidean coordinate system. It is not a guarantee of equal fold sizes or a universal design for every scientific dependence structure.

Value

An integer vector in fitted-row order, with class `smodels_spatial_folds` and canonical cluster centres attached.

sp_target_geometry	<i>Attach Prediction-Target Summaries to Spatial Geometry</i>
--------------------	---

Description

Joins calculated point/grid or grouped polygon summaries to geometry using row order or explicit unique identifiers.

Usage

```
sp_target_geometry(object, geometry, id = NULL)

## S3 method for class 'spmodels_target_sf'
plot(x, value = "mean", ...)
```

Arguments

object	An object returned by sp_predict_target().
geometry	A non-empty sf object without empty geometries.
id	Optional geometry column matched exactly to target identifiers.
x	An spmodels_target_sf object.
value	Numeric target-summary column to display.
...	Arguments passed to the ordinary sf plot method.

Value

An sf object containing unrounded prediction-target summaries and audit metadata. The plot method returns it invisibly.

sp_temporal	<i>Specify Survey Time for a Separable Space-Time Model</i>
-------------	---

Description

Declares survey times and the exponential temporal scale for a separable Matern-exponential space-time covariance.

Usage

```
sp_temporal(time, scale, time_units, estimate = FALSE)

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

Arguments

time	One-sided formula selecting one finite numeric time column.
scale	Positive exponential temporal scale or its starting value.
time_units	Non-empty label for the time and scale units.
estimate	Estimate temporal scale jointly with other parameters?
x	A temporal specification.
...	Unused.

Value

An object of class `smodels_temporal`.

sp_utm_crs	<i>Choose a Projected UTM Coordinate Reference System</i>
------------	---

Description

Selects a WGS84 UTM CRS after checking latitude support, antimeridian crossing, standard Norway/Svalbard exceptions, zone boundaries and hemisphere boundaries.

Usage

```
sp_utm_crs(
  data,
  zone_policy = c("error", "centre"),
  hemisphere_policy = c("error", "centre")
)

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

Arguments

data	A non-empty <code>sf</code> or <code>sfc</code> object with known CRS.
zone_policy	Error on multiple zones or use the bounding-box centre.
hemisphere_policy	Error on an equator crossing or use the bounding-box centre.
x	A CRS-choice object.
...	Reserved for method compatibility.

Value

An object containing an `sf::crs`, EPSG code, UTM zone, hemisphere and audit metadata.

sp_validation_table	<i>Structured validation table</i>
---------------------	------------------------------------

Description

Provides a common machine-readable metric schema for SPmodels validation and simulation-assessment results.

Usage

```
sp_validation_table(object, detail = c("summary", "fold"))
```

Arguments

object	An SPmodels cross-validation, simulation-assessment, simulation-recovery, or multi-scenario recovery object.
detail	Return overall summaries or fold summaries for cross-validation.

Value

A long-form data frame with assessment, scope, identifier, metric and value columns.

sp_variogram	<i>Empirical Residual Variograms Before and After Spatial Fitting</i>
--------------	---

Description

Calculates binned pairwise residual semivariances with pair counts. Before fitting, ordinary residuals from `lm`, `glm`, or a non-spatial SPmodels fit can be used to assess whether spatial structure motivates fitting a spatial process. After fitting, conditional residuals can be used to assess remaining spatial structure. These are distinct diagnostic questions. Gaussian spatial-fit results can include the fitted covariance-model variogram. Discrete-family residual variograms deliberately omit a fitted response-scale curve rather than treating link covariance as residual semivariance.

Usage

```
sp_variogram(
  object = NULL,
  bins = 10L,
  max_distance = NULL,
  residual_type = NULL,
  seed = NULL,
  coords = NULL,
  values = NULL,
  data = NULL,
```

```

    distance_units = NULL
)

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

Arguments

object	A fitted spmodels_fit object.
bins	Positive number of equal-width distance bins.
max_distance	Optional positive maximum pair distance.
residual_type	For Gaussian fits, use "mean" or "conditional" residuals. For binomial/Poisson fits, use "randomized_quantile", "pearson" or "deviance". The family-specific first choice is the default.
seed	Optional integer seed local to randomized-quantile residuals.
coords	For an ordinary lm/glm or direct residual vector, either two numeric coordinate columns or a one-sided coordinate formula.
values	Numeric residual values when object = NULL.
data	Data used to evaluate a coordinate formula for an ordinary model. Supply the complete source data when coordinates are not in the model formula.
distance_units	Optional target unit for distances. Spatial and non-spatial SPmodels fits are converted from their stored units; projected sf coordinates are converted when their CRS units are recognised. Numeric coordinate columns are assumed already to use this unit.
x	An object returned by sp_variogram().
...	Additional graphical arguments passed to plot.

Value

sp_variogram() returns an object of class spmodels_variogram containing binned empirical semivariances and, for Gaussian fits, the fitted model semivariance. The plot method invisibly returns its input.

sp_variogram_envelope *Monte Carlo Envelope for a Residual Variogram*

Description

Builds a pointwise residual-variogram envelope and a global curve-discrepancy test from residual permutations or fitted-model parametric simulations. Pre-fit permutation envelopes are supported for ordinary model residuals, supplied residuals and non-spatial SPmodels fits.

Usage

```

sp_variogram_envelope(
  object = NULL,
  nsim = 999L,
  method = c("permutation", "parametric"),
  level = 0.95,
  bins = 10L,
  max_distance = NULL,
  residual_type = NULL,
  seed = NULL,
  coords = NULL,
  values = NULL,
  data = NULL,
  distance_units = NULL
)

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

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

```

Arguments

<code>object</code>	A spatial or non-spatial <code>spmodels_fit</code> , an <code>lm/glm</code> object, or <code>NULL</code> when supplying <code>coords</code> and <code>values</code> . The permutation method supports pre-fit inputs; the parametric method requires a fitted spatial model.
<code>nsim</code>	Positive number of Monte Carlo curves.
<code>method</code>	Permute residuals across fixed locations, or simulate from the fitted model while keeping fitted parameters fixed.
<code>level</code>	Pointwise envelope probability strictly between zero and one.
<code>bins</code>	Positive number of equal-width distance bins.
<code>max_distance</code>	Optional positive largest pair distance.
<code>residual_type</code>	Passed to <code>sp_variogram()</code> : Gaussian mean or conditional residuals, or binomial/Poisson randomized-quantile, Pearson or deviance residuals.
<code>seed</code>	Optional integer seed local to this call.
<code>coords</code>	Two numeric coordinate columns, a one-sided coordinate formula, or projected <code>sf</code> data, passed to <code>sp_variogram()</code> .
<code>values</code>	Numeric residual values when <code>object = NULL</code> .
<code>data</code>	Data used to evaluate a coordinate formula for an ordinary model.
<code>distance_units</code>	Optional target unit for distances. Stored model or projected <code>sf</code> coordinates are converted when their source units are known; numeric coordinate values are assumed already in this unit.
<code>x</code>	A variogram-envelope object.
<code>...</code>	Additional graphical arguments.

Details

For a pre-fit diagnostic, the permutation null is exchangeability of the selected ordinary-model residual values across fixed locations. This provides the week 7 teaching comparison before fitting a spatial process. Gaussian parametric curves keep fitted parameters fixed. Binomial and Poisson parametric curves simulate a response, repeat the declared Laplace or Monte Carlo ML fitting route, and calculate the selected residuals from that refit. This includes fitting variability but is more computationally expensive.

Value

An object containing observed and simulated variogram curves, pointwise limits, standardized global statistics and a Monte Carlo p-value. Discrete parametric envelopes additionally retain every refit's convergence code and captured warning messages.

spacetime_cor

Separable Matern-Exponential Space-Time Correlation

Description

Multiplies stationary Matern spatial correlation by stationary exponential temporal correlation.

Usage

```
spacetime_cor(distance, time_lag, spatial_scale, smoothness, temporal_scale)
```

Arguments

distance	Non-negative finite spatial distances.
time_lag	Non-negative finite time lags with dimensions matching distance.
spatial_scale	Positive Matern spatial scale.
smoothness	Positive Matern smoothness.
temporal_scale	Positive exponential temporal scale in the same units as time_lag.

Value

Correlations with dimensions and names inherited from distance.

Index

`as.data.frame.spmodels_prediction_target`,
3

`coef.spmodels_fit(sp_fit)`, 16
`coef.spmodels_neighbor_fit`
 (`sp_fit_neighbor`), 18
`confint.spmodels_fit`, 3
`confint.spmodels_neighbor_fit`
 (`sp_fit_neighbor`), 18

`fitted.spmodels_fit(sp_fit)`, 16
`fitted.spmodels_neighbor_fit`
 (`sp_fit_neighbor`), 18

`logLik.spmodels_fit(sp_fit)`, 16
`logLik.spmodels_neighbor_fit`
 (`sp_fit_neighbor`), 18

`matern_cor`, 4
`matern_scale_derivative`, 5

`plot.spmodels_calibration`
 (`sp_calibration`), 10
`plot.spmodels_cross_validation`, 5
`plot.spmodels_diagnostics`
 (`sp_diagnostics`), 13
`plot.spmodels_intervention_curve`
 (`sp_intervention_curve`), 24
`plot.spmodels_sampler_diagnostics`
 (`sp_sampler_diagnostics`), 49
`plot.spmodels_simulation_assessment`
 (`plot.spmodels_cross_validation`),
 5
`plot.spmodels_simulation_recovery`
 (`plot.spmodels_cross_validation`),
 5
`plot.spmodels_target_sf`
 (`sp_target_geometry`), 56
`plot.spmodels_variogram(sp_variogram)`,
 58

`plot.spmodels_variogram_envelope`
 (`sp_variogram_envelope`), 59
`predict.spmodels_fit`, 7
`predict.spmodels_neighbor_fit`
 (`sp_fit_neighbor`), 18
`print.spmodels_block_folds`
 (`sp_block_folds`), 8
`print.spmodels_buffered_folds`
 (`sp_buffered_folds`), 9
`print.spmodels_calibration`
 (`sp_calibration`), 10
`print.spmodels_cross_validation`
 (`sp_cross_validate`), 12
`print.spmodels_crs_choice(sp_utm_crs)`,
 57
`print.spmodels_distance_summary`
 (`sp_distance_summary`), 14
`print.spmodels_environmental_folds`
 (`sp_environmental_folds`), 15
`print.spmodels_fit(sp_fit)`, 16
`print.spmodels_formatted_table`
 (`sp_format_table`), 21
`print.spmodels_group_folds`
 (`sp_group_folds`), 22
`print.spmodels_intervention`
 (`sp_intervention`), 23
`print.spmodels_intervention_curve`
 (`sp_intervention_curve`), 24
`print.spmodels_mcml_control`
 (`sp_mcml_control`), 30
`print.spmodels_mcml_replicates`
 (`sp_mcml_replicates`), 32
`print.spmodels_neighbor_covariance`
 (`sp_neighbor_covariance`), 33
`print.spmodels_neighbor_fit`
 (`sp_fit_neighbor`), 18
`print.spmodels_prediction_draws`
 (`sp_predict_draws`), 38
`print.spmodels_prediction_target`

(sp_predict_target), 42
 print.spmodels_prediction_update
 (sp_prediction_update), 45
 print.spmodels_random_slope
 (sp_random_slope), 47
 print.spmodels_report(sp_report), 48
 print.spmodels_sampler_diagnostics
 (sp_sampler_diagnostics), 49
 print.spmodels_simulation_assessment
 (sp_simulation_assess), 50
 print.spmodels_simulation_precision
 (sp_simulation_precision), 51
 print.spmodels_simulation_recovery
 (sp_simulation_recovery), 52
 print.spmodels_simulation_scenarios
 (sp_simulation_scenarios), 54
 print.spmodels_spatial_folds
 (sp_spatial_folds), 55
 print.spmodels_temporal(sp_temporal),
 56
 print.spmodels_variogram_envelope
 (sp_variogram_envelope), 59
 print.summary.spmodels_fit(sp_fit), 16
 print.summary.spmodels_neighbor_fit
 (sp_fit_neighbor), 18

 residuals.spmodels_fit(sp_fit), 16
 residuals.spmodels_neighbor_fit
 (sp_fit_neighbor), 18

 simulate.spmodels_fit, 8
 sp_block_folds, 8
 sp_buffered_folds, 9
 sp_calibration, 10
 sp_control, 11
 sp_cross_validate, 12
 sp_diagnostics, 13
 sp_distance_summary, 14
 sp_environmental_folds, 15
 sp_fit, 16
 sp_fit_neighbor, 18
 sp_format_table, 21
 sp_grid, 22
 sp_group_folds, 22
 sp_intervention, 23
 sp_intervention_curve, 24
 sp_intervention_multiplier, 25
 sp_latent_mode, 26
 sp_latent_sample, 27

 sp_mcmc_control, 28
 sp_mcml, 28
 sp_mcml_control, 30
 sp_mcml_precision, 31
 sp_mcml_replicates, 32
 sp_model_table, 33
 sp_neighbor_covariance, 33
 sp_nonstationary_cor, 35
 sp_polygon_area_weights, 36
 sp_polygon_weights, 36
 sp_predict_chunked, 37
 sp_predict_draws, 38
 sp_predict_draws_neighbor, 40
 sp_predict_target, 42
 sp_predict_target_neighbor, 43
 sp_prediction_table, 44
 sp_prediction_update, 45
 sp_predictive_log_score, 46
 sp_random_slope, 47
 sp_report, 48
 sp_sampler_diagnostics, 49
 sp_simulation_assess, 50
 sp_simulation_grid, 50
 sp_simulation_precision, 51
 sp_simulation_recovery, 52
 sp_simulation_scenarios, 54
 sp_spatial_folds, 55
 sp_target_geometry, 56
 sp_temporal, 56
 sp_utm_crs, 57
 sp_validation_table, 58
 sp_variogram, 58
 sp_variogram_envelope, 59
 spacetime_cor, 61
 summary.spmodels_fit(sp_fit), 16
 summary.spmodels_neighbor_fit
 (sp_fit_neighbor), 18

 vcov.spmodels_fit(sp_fit), 16
 vcov.spmodels_neighbor_fit
 (sp_fit_neighbor), 18