slm.lat{slm}

Spatial Linear Geostatistical Models

Description

This function fits spatial linear models for lattice data. It can estimate fixed effects in the linear model, make predictions for a set of unsampled locations, estimate contrasts, make spatially-smoothed maps, and do finite population block kriging. It returns a variety of other linear model objects such as R^2, AIC, cross-validation statistics, and residuals.

Usage

```
slm.lat(formula, data = data, ndef.adj, ndef.num,
        CorModel = "CAR", EstMeth = "REML",
        use.row.standard = TRUE, use.distance = FALSE,
        weight.col = NULL, xcol = NULL, ycol = NULL,
        create.smooth = FALSE, est.con.mat = NULL, alpha = 0,
        est.con.label = NULL, use.FPBK = FALSE, FPBK.col = NULL)
```

Arguments

formula	a symbolic description of the model to be fit. The details of model specification are given below.
data	a data frame containing the variables in the model.
ndef.adj	a vector listing the ID numbers of the neighbors for each location (this is a sparse representation of the full adjacency matrix for the study region
ndef.num	A vector of length *N* (the total number of locations) giving the number of neighbors n_i for each location.
CorModel	The spatial correlation model. The model name should be quoted; e.g. CorModel = "CAR" (this is the default). The following options are available: "SAR", "MA".
EstMeth	Type of estimation method. The default is restricted maximum likelihood (REML). The other option is maximum likelihood (ML). The method should be quoted; e.g. EstMeth = "ML".
use.row. standard	If TRUE (the default), then rows of the **C** matrix are standardized after building an neighborhood incidence (0/1) or the distance matrix (see next).
use.distance	if TRUE, then the **C** matrix is based on distance. xcol and ycol must then be specified, and Euclidean distance will be computed from these. The default is FALSE. The **C** matrix values are determined by exp(−distance^alpha)/*c* where *c* is a scaling constant, and distance is only calculated for neighbors of a location, otherwise values are 0. Alpha values can be specified as given below.

`weight.col`	A column of weights to be applied to each location in the covariance matrix. An example is the sample size for counts such as disease. The default is NULL.
`xcol`	the name of the *x*-coordinate in the data set. This will be used to compute Euclidean distance for the distance-based **C** matrix. The default is NULL.
`ycol`	the name of the *y*-coordinate in the data set. This will be used to compute Euclidean distance for the distance-based **C** matrix. The default is NULL.
`create.smooth`	if TRUE, this will created a smoothed prediction map. Smoothed maps are created by removing a location and then predicting it using the conditional expectation for that location.
`est.con.mat`	a matrix where the rows consist of contrasts or other linear combinations of the fixed effect parameters. The function will test for estimability.
`est.con.label`	a vector of character labels for each row of the est.con.mat.
`alpha`	The power to be used in the distance model for the **C** matrix. The **C** matrix values are determined by exp(–distance^alpha)/*c* where *c* is a scaling constant. The default is 0.
`use.FPBK`	use finite population block kriging. The default is NULL.
`FPBK.col`	the column containing the weights in the data frame. If it is typed in directly, it should be quoted; e.g., FPBK.col = "bt".

Details

Models for `slm.geo` are specified symbolically. A typical model has the form `response ~ terms` where `response` is the (numeric) response vector and `terms` is a series of terms which specifies a linear predictor for `response`. A terms specification of the form `first + second` indicates all the terms in `first` together with all the terms in `second` with duplicates removed. A specification of the form `first:second` indicates the set of terms obtained by taking the interactions of all terms in `first` with all terms in `second`. The specification `first*second` indicates the *cross* of `first` and `second`. This is the same as `first + second + first:second`.
Missing values for the response variable are automatically predicted.

Value

`slm.lat` returns a list that may contain some of the following components:

`sample.size`	the number of rows in the data frame.
`obs.sample.size`	the observed sample size; the number of rows in the data frame minus the number of missing values.
`missing.sample.size`	the number of missing values.
`CorModel`	the correlation model
`EstMeth`	the estimation method
`fixed.effects.estimates`	a data frame with the fixed effects estimates. The data frame lists all fixed effects in the model, and if they are factors, the levels of the factors, then it gives the estimate(s) for that effect, the standard error, the degrees of freedom, a t-value, and the probability of the t-value under a null hypothesis that the effect is 0.
`typeIII.hypoth`	the type III sums of squares test-of-hypothesis for each effect.
`R2g`	a generalized R^2 value.
`cov.fix.eff`	the covariance matrix of the fixed effect estimates.
`theta`	the spatial covariance parameter estimates.
`rho.restrict`	the limits of the spatial autocorrelation parameter rho.
`m2LL`	-2 time the log likelihood.
`AIC`	AIC, Akaike's Information Criteria
`AICc`	AIC with a small sample correction.
`BIC`	Bayesian Information Criteria, also known as Schwarz's criteria.
`cv.stats`	crossvalidation statistics. This list object is a data frame of a single row with the following values: bias, sum(z - pred)/n, where z are observed and pred are predicted values; std.bias, sum((z - pred)/sqrt(var))/n, where var is the estimated prediction variance; RMSPE, sqrt(sum((z - pred)^2)/n); std.MSPE, sum((z - pred)^2/var)/n; cov.80, proportion of times the 80% prediction interval covered the observed value; cov.90, proportion of times the 90% prediction interval covered the observed value; cov.95, proportion of times the 95% prediction interval covered the observed value.
`data`	The data frame returned with fitted values; see below. If there are missing values for the response variable, they are predicted if all covariates are available. If the smoothed = TRUE option is chosen, a column called smooth and smooth.se are added.
`fit`	fitted values. The output is a data frame with the following columns: the data data-frame; fit, the design matrix time the estimated fixed effects; resid, the observed values minus the fit; resid.stand, standardized residuals; resid.student, studentized residuals; leverage; CooksD; cv.pred, crossvalidation predictions; cv.se, estimated prediction standard errors from crossvalidation; cv.resid, the observed values minus the crossvalidation predictions; cv.stndr, cv.resid divided by cv.se.

FPBK	Finite population block kriging identified by the weights in FPBK.col. The object is a dataframe with one row and two columns: FPBK.est and FPBK.se.
Estimates. Contrasts	a data frame containing contrasts and estimates identified by est.con.mat and est.con.label. The data frame contains the following columns: label, the labels identified by est.con.label; estimable, a logical condition for whether or not each row in est.con.mat is estimable; estimate, the estimate for each row in est.con.mat; std.err, the estimated standard error for each row in est.con.mat; df, the degrees of freedom; t.value, the t-value; and prob.t, the probability of t.value under the null hypothesis that the true value is 0.
V	the estimated covariance matrix.
Vi	inverse of the estimated covariance matrix.
X	The design matrix

Examples

```
# --------------A SMOOTHED MAPPING AND REGRESSION EXAMPLE
#------------------- USING DISTANCE AND WEIGHTS

slm.lat.out <- slm.lat(Free.Tukey ~  nwbir74.ft,
        data = Cdata1,
        ndef.adj = Cadj, ndef.num = Cnum,
        CorModel = "CAR", EstMeth = "REML",
        use.distance = TRUE, xcol = "x", ycol = "y",
        weight.col = "bir74",
        use.row.standard = FALSE, alpha = 1)
slm.lat.out$sample.size
slm.lat.out$CorModel
slm.lat.out$EstMeth
slm.lat.out$fixed.effects.estimates
slm.lat.out$typeIII.hypoth
slm.lat.out$R2g
slm.lat.out$theta
slm.lat.out$rho.restrict
slm.lat.out$m2LL
slm.lat.out$AIC
slm.lat.out$AICc
slm.lat.out$BIC
slm.lat.out$cv.stats
data.out <- slm.lat.out$data
data.out[,c("smooth", smooth.se")]

# --------- A SPATIAL DESIGNED EXPERIMENT EXAMPLE

# CREATE A SET OF CONTRASTS

est.con.mat <- c(-1, .5, .5, 0, 0)
est.con.mat <- rbind(est.con.mat, c(-1, 0, 0, .5, .5))
```

```
est.con.mat <- rbind(est.con.mat, c(0, -.5, -.5, .5, .5))
est.con.mat <- rbind(est.con.mat, c(0, 1, -1, 0, 0))
est.con.mat <- rbind(est.con.mat, c(0, 0, 0, 1, -1))
est.con.lab <- c("con1", "con2", "con3", "con4", "con5")

# FIT A SPATIAL LINEAR LATTICE MODEL

slm.lat.out <- slm.lat(z ~ trt - 1,
        data = data,
        ndef.adj = ndef.out$adj, ndef.num = ndef.out$num,
        est.con.mat = est.con.mat, est.con.label = est.con.lab,
        CorModel = "CAR", EstMeth = "REML")
slm.lat.out$sample.size
slm.lat.out$CorModel
slm.lat.out$EstMeth
slm.lat.out$fixed.effects.estimates
slm.lat.out$typeIII.hypoth
slm.lat.out$R2g
slm.lat.out$theta
slm.lat.out$rho.restrict
slm.lat.out$m2LL
slm.lat.out$AIC
slm.lat.out$AICc
slm.lat.out$BIC
slm.lat.out$cv.stats
slm.lat.out$Estimates.Contrasts

# --------- A FINITE POPULATION BLOCK KRIGING EXAMPLE

slm.lat.out <- slm.lat(TOTAL ~ STRAT,
        data = data,
        ndef.adj = ndef.out$adj, ndef.num = ndef.out$num,
        CorModel = "CAR", EstMeth = "REML",
        use.distance = TRUE, xcol = xcol, ycol = ycol,
        use.row.standard = FALSE, alpha = 1,
        use.FPBK = TRUE, FPBK.col = "FPBK.locs")
slm.lat.out$sample.size
slm.lat.out$obs.sample.size
slm.lat.out$missing.sample.size
slm.lat.out$CorModel
slm.lat.out$EstMeth
slm.lat.out$fixed.effects.estimates
slm.lat.out$typeIII.hypoth
slm.lat.out$R2g
slm.lat.out$theta
slm.lat.out$rho.restrict
slm.lat.out$m2LL
slm.lat.out$AIC
slm.lat.out$AICc
slm.lat.out$BIC
slm.lat.out$cv.stats
slm.lat.out$FPBK
```