Package 'spStack'

July 15, 2025

Type Package

Version 1.1.1

Title Bayesian Geostatistics Using Predictive Stacking

Description Fits Bayesian hierarchical spatial and spatial-temporal process models for point-referenced Gaussian, Poisson, binomial, and binary data using stacking of predictive densities. It involves sampling from analytically available posterior distributions conditional upon candidate values of the spatial process parameters and, subsequently assimilate inference from these individual posterior distributions using Bayesian predictive stacking. Our algorithm is highly parallelizable and hence, much faster than traditional Markov chain Monte Carlo algorithms while delivering competitive predictive performance. See Zhang, Tang, and Banerjee (2025) <doi:10.48550/arXiv.2304.12414>, and, Pan, Zhang, Bradley, and Banerjee (2025) <doi:10.48550/arXiv.2406.04655> for details.

Imports CVXR, future, future.apply, ggplot2, MBA, rstudioapi

NeedsCompilation yes

License GPL-3

Encoding UTF-8

LazyData true

Suggests dplyr, ggpubr, knitr, patchwork, rmarkdown, spelling, testthat (>= 3.0.0), tidyr

Config/testthat/edition 3

RoxygenNote 7.3.2

Depends R (>= 3.5)

VignetteBuilder knitr

URL https://span-18.github.io/spStack-dev/

BugReports https://github.com/SPan-18/spStack-dev/issues

Language en-US

Author Soumyakanti Pan [aut, cre] (ORCID: <https://orcid.org/0009-0005-9889-7112>), Sudipto Banerjee [aut]

Maintainer Soumyakanti Pan <span18@ucla.edu>

Repository CRAN

Date/Publication 2025-07-14 22:20:02 UTC

Contents

`spStack-package` *spStack: Bayesian Geostatistics Using Predictive Stacking*

Description

This package delivers functions to fit Bayesian hierarchical spatial process models for point-referenced Gaussian, Poisson, binomial, and binary data using stacking of predictive densities. It involves sampling from analytically available posterior distributions conditional upon some candidate values of the spatial process parameters for both Gaussian response model as well as non-Gaussian responses, and, subsequently assimilate inference from these individual posterior distributions using Bayesian predictive stacking. Our algorithm is highly parallelizable and hence, much faster than traditional Markov chain Monte Carlo algorithms while delivering competitive predictive performance.

In context of inference for spatial point-referenced data, Bayesian hierarchical models involve latent spatial processes characterized by spatial process parameters, which besides lacking substantive relevance in scientific contexts, are also weakly identified and hence, impedes convergence of MCMC

algorithms. This motivates us to build methodology that involves fast sampling from posterior distributions conditioned on a grid of the weakly identified model parameters and combine the inference by stacking of predictive densities (Yao *et. al* 2018). We exploit the Bayesian conjugate linear modeling framework for the Gaussian case (Zhang, Tang and Banerjee 2024) and the generalized conjugate multivariate distribution theory (Pan, Zhang, Bradley and Banerjee 2024) to analytically derive the individual posterior distributions.

Details

Package:	spStack
Type:	Package
Version:	1.1.0
License:	GPL-3

Accepts a formula, e.g., `y~x1+x2`, for most regression models accompanied by candidate values of spatial process parameters, and returns posterior samples of the regression coefficients and the latent spatial random effects. Posterior inference or prediction of any quantity of interest proceed from these samples. Main functions are -
`spLMexact()`
`spGLMexact()`
`spLMstack()`
`spGLMstack()`

Author(s)

Maintainer: Soumyakanti Pan <span18@ucla.edu> (ORCID)

Authors:

- Sudipto Banerjee <sudipto@ucla.edu>

References

Zhang L, Tang W, Banerjee S (2025). "Bayesian Geostatistics Using Predictive Stacking." doi:10.48550/arXiv.2304.12414.

Pan S, Zhang L, Bradley JR, Banerjee S (2025). "Bayesian Inference for Spatial-temporal Non-Gaussian Data Using Predictive Stacking." doi:10.48550/arXiv.2406.04655.

Yao Y, Vehtari A, Simpson D, Gelman A (2018). "Using Stacking to Average Bayesian Predictive Distributions (with Discussion)." *Bayesian Analysis*, **13**(3), 917-1007. doi:10.1214/17BA1091.

See Also

Useful links:

- https://span-18.github.io/spStack-dev/
- Report bugs at https://github.com/SPan-18/spStack-dev/issues

`candidateModels` *Create a collection of candidate models for stacking*

Description

Creates an object of class `'candidateModels'` that contains a list of candidate models for stacking. The function takes a list of candidate values for each model parameter and returns a list of possible combinations of these values based on either simple aggregation or Cartesian product of indivdual candidate values.

Usage

```
candidateModels(params_list, aggregation = "simple")
```

Arguments

`params_list`	a list of candidate values for each model parameter. See examples for details.
`aggregation`	a character string specifying the type of aggregation to be used. Options are `'simple'` and `'cartesian'`. Default is `'simple'`.

Value

an object of class `'candidateModels'`

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

See Also

`stvcGLMstack()`

Examples

```
m1 <- candidateModels(list(phi_s = c(1, 1), phi_t = c(1, 2)), "simple")
m1
m2 <- candidateModels(list(phi_s = c(1, 1), phi_t = c(1, 2)), "cartesian")
m2
m3 <- candidateModels(list(phi_s = list(c(1, 1), c(1, 2)),
                           phi_t = list(c(1, 3), c(2, 3)),
                           boundary = c(0.5, 0.75)),
                      "simple")
```

`cholUpdate` *Different Cholesky factor updates*

Description

Provides functions that implements different types of updates of a Cholesky factor that includes rank-one update, single row/column deletion update and a block deletion update.

Usage

```
cholUpdateRankOne(A, v, alpha, beta, lower = TRUE)

cholUpdateDel(A, del.index, lower = TRUE)

cholUpdateDelBlock(A, del.start, del.end, lower = TRUE)
```

Arguments

`A`	an $n \times n$ triangular matrix
`v`	an $n \times 1$ matrix/vector
`alpha`	scalar; if not supplied, default is 1
`beta`	scalar; if not supplied, default is 1
`lower`	logical; if `A` is lower-triangular or not
`del.index`	an integer from 1 to n indicating the row/column to be deleted
`del.start`	an integer from 1 to n indicating the first row/column of a block to be deleted, must be at least 1 less than `del.end`
`del.end`	an integer from 1 to n indicating the last row/column of a block to be deleted, must be at least 1 more than `del.start`

Details

Suppose $B = AA^\top$ is a $n \times n$ matrix with A being its lower-triangular Cholesky factor. Then rank-one update corresponds to finding the Cholesky factor of the matrix $C = \alpha B + \beta vv^\top$ for some $\alpha, \beta \in \mathbb{R}$ given A (see, Krause and Igel 2015). Similarly, single row/column deletion update corresponds to finding the Cholesky factor of the $(n-1) \times (n-1)$ matrix B_i which is obtained by removing the i-th row and column of B, given A for some $i - 1, \ldots, n$. Lastly, block deletion corresponds to finding the Cholesky factor of the $(n - n_k) \times (n - n_k)$ matrix B_I for a subset I of $\{1, \ldots, n\}$ containing n_k consecutive indices, given the factor A.

Value

An $m \times m$ lower-triangular `matrix` with $m = n$ in case of `cholUpdateRankOne()`, $m = n - 1$ in case of `cholUpdateDel()`, and, $m = n - n_k$ in case of `cholUpdateDelBlock()` where n_k is the size of the block removed.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

References

Oswin Krause and Christian Igel. 2015. "A More Efficient Rank-one Covariance Matrix Update for Evolution Strategies". In *Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII* (FOGA '15). Association for Computing Machinery, New York, NY, USA, 129-136. doi:10.1145/2725494.2725496.

Examples

```
n <- 10
A <- matrix(rnorm(n^2), n, n)
A <- crossprod(A)
cholA <- chol(A)

## Rank-1 update
v <- 1:n
APlusvvT <- A + tcrossprod(v)
cholA1 <- t(chol(APlusvvT))
cholA2 <- cholUpdateRankOne(cholA, v, lower = FALSE)
print(all(abs(cholA1 - cholA2) < 1E-9))

## Single Row-deletion update
ind <- 2
A1 <- A[-ind, -ind]
cholA1 <- t(chol(A1))
cholA2 <- cholUpdateDel(cholA, del.index = ind, lower = FALSE)
print(all(abs(cholA1 - cholA2) < 1E-9))

## Block-deletion update
start_ind <- 2
end_ind <- 6
del_ind <- c(start_ind:end_ind)
A1 <- A[-del_ind, -del_ind]
cholA1 <- t(chol(A1))
cholA2 <- cholUpdateDelBlock(cholA, start_ind, end_ind, lower = FALSE)
print(all(abs(cholA1 - cholA2) < 1E-9))
```

---

get_stacking_weights *Optimal stacking weights*

Description

Obtains optimal stacking weights given leave-one-out predictive densities for each candidate model.

Usage

```
get_stacking_weights(log_loopd, solver = "ECOS")
```

Arguments

log_loopd	an $n \times M$ matrix with i-th row containing the leave-one-out predictive densities for the i-th data point for the M candidate models.
solver	specifies the solver to use for obtaining optimal weights. Default is "ECOS". Internally calls CVXR::psolve().

Value

A list of length 2.

weights optimal stacking weights as a numeric vector of length M

status solver status, returns "optimal" if solver succeeded.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

References

Yao Y, Vehtari A, Simpson D, Gelman A (2018). "Using Stacking to Average Bayesian Predictive Distributions (with Discussion)." *Bayesian Analysis*, **13**(3), 917-1007. doi:10.1214/17BA1091.

See Also

CVXR::psolve(), spLMstack(), spGLMstack()

Examples

```
set.seed(1234)
data(simGaussian)
dat <- simGaussian[1:100, ]

mod1 <- spLMstack(y ~ x1, data = dat,
                  coords = as.matrix(dat[, c("s1", "s2")]),
                  cor.fn = "matern",
                  params.list = list(phi = c(1.5, 3),
                                     nu = c(0.5, 1),
                                     noise_sp_ratio = c(1)),
                  n.samples = 1000, loopd.method = "exact",
                  parallel = FALSE, solver = "ECOS", verbose = TRUE)

loopd_mat <- do.call('cbind', mod1$loopd)
w_hat <- get_stacking_weights(loopd_mat)
print(round(w_hat$weights, 4))
print(w_hat$status)
```

iDist *Calculate distance matrix*

Description

Computes the inter-site Euclidean distance matrix for one or two sets of points.

Usage

```
iDist(coords.1, coords.2, ...)
```

Arguments

coords.1	an $n \times p$ matrix with each row corresponding to a point in p-dimensional space.
coords.2	an $m \times p$ matrix with each row corresponding to a point in p dimensional space. If this is missing then coords.1 is used.
...	currently no additional arguments.

Value

The $n \times n$ or $n \times m$ inter-site Euclidean distance matrix.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

Examples

```
n <- 10
p1 <- cbind(runif(n),runif(n))
m <- 5
p2 <- cbind(runif(m),runif(m))
D <- iDist(p1, p2)
```

posteriorPredict *Prediction of latent process at new spatial or temporal locations*

Description

A function to sample from the posterior predictive distribution of the latent spatial or spatial-temporal process.

Usage

```
posteriorPredict(mod_out, coords_new, covars_new, joint = FALSE, nBinom_new)
```

Arguments

mod_out	an object returned by any model fit under fixed hyperparameters or using predictive stacking, i.e., spLMexact(), spLMstack(), spGLMexact(), spGLMstack(), stvcGLMexact(), or stvcGLMstack().
coords_new	a list of new spatial or spatial-temporal coordinates at which the latent process, the mean, and the response is to be predicted.
covars_new	a list of new covariates at the new spatial or spatial-temporal coordinates. See examples for the structure of this list.
joint	a logical value indicating whether to return the joint posterior predictive samples of the latent process at the new locations or times. Defaults to FALSE.
nBinom_new	a vector of the number of trials for each new prediction location or time. Only required if the model family is "binomial". Defaults to a vector of ones, indicating one trial for each new prediction.

Value

A modified object with the class name preceeded by the identifier pp separated by a .. For example, if input is of class spLMstack, then the output of this prediction function would be pp.spLMstack. The entry with the tag samples is updated and will include samples from the posterior predictive distribution of the latent process, the mean, and the response at the new locations or times. An entry with the tag prediction is added and contains the new coordinates and covariates, and whether the joint posterior predictive samples were requested.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

See Also

spLMexact(), spLMstack(), spGLMexact(), spGLMstack(), stvcGLMexact(), stvcGLMstack()

Examples

```
set.seed(1234)
# training and test data sizes
n_train <- 100
n_pred <- 10

# Example 1: Spatial linear model
# load and split data into training and prediction sets
data(simGaussian)
dat <- simGaussian
dat_train <- dat[1:n_train, ]
dat_pred <- dat[n_train + 1:n_pred, ]

# fit a spatial linear model using predictive stacking
mod1 <- spLMstack(y ~ x1, data = dat_train,
                  coords = as.matrix(dat_train[, c("s1", "s2")]),
```

```
                    cor.fn = "matern",
                    params.list = list(phi = c(1.5, 3, 5), nu = c(0.75, 1.25),
                                       noise_sp_ratio = c(0.5, 1, 2)),
                    n.samples = 1000, loopd.method = "psis",
                    parallel = FALSE, solver = "ECOS", verbose = TRUE)

# prepare new coordinates and covariates for prediction
sp_pred <- as.matrix(dat_pred[, c("s1", "s2")])
X_new <- as.matrix(cbind(rep(1, n_pred), dat_pred$x1))

# carry out posterior prediction
mod.pred <- posteriorPredict(mod1, coords_new = sp_pred, covars_new = X_new,
                             joint = TRUE)

# sample from the stacked posterior and posterior predictive distribution
post_samps <- stackedSampler(mod.pred)

# analyze posterior samples
postpred_z <- post_samps$z.pred
post_z_summ <- t(apply(postpred_z, 1, function(x) quantile(x, c(0.025, 0.5, 0.975))))
z_combn <- data.frame(z = dat_pred$z_true, zL = post_z_summ[, 1],
                      zM = post_z_summ[, 2], zU = post_z_summ[, 3])
library(ggplot2)
ggplot(data = z_combn, aes(x = z)) +
  geom_errorbar(aes(ymin = zL, ymax = zU), width = 0.05, alpha = 0.15, color = "skyblue") +
  geom_point(aes(y = zM), size = 0.25, color = "darkblue", alpha = 0.5) +
  geom_abline(slope = 1, intercept = 0, color = "red", linetype = "solid") +
  xlab("True z1") + ylab("Posterior of z1") + theme_bw() +
  theme(panel.background = element_blank(), aspect.ratio = 1)
```

`recoverGLMscale` *Recover posterior samples of scale parameters of spatial/spatial-temporal generalized linear models*

Description

A function to recover posterior samples of scale parameters that were marginalized out during model fit. This is only applicable for spatial or, spatial-temporal generalized linear models. This function applies on outputs of functions that fits a spatial/spatial-temporal generalized linear model, such as `spGLMexact()`, `spGLMstack()`, `stvcGLMexact()`, and `stvcGLMstack()`.

Usage

```
recoverGLMscale(mod_out)
```

Arguments

`mod_out`	an object returned by a fitting a spatial or spatial-temporal GLM.

Value

An object of the same class as input, and updates the list tagged `samples` with the posterior samples of the scale parameters. The new tags are `sigmasq.beta` and `z.scale`.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

See Also

`spGLMexact()`, `spGLMstack()`, `stvcGLMexact()`, `stvcGLMstack()`

Examples

```
set.seed(1234)
data("simPoisson")
dat <- simPoisson[1:100, ]
mod1 <- spGLMstack(y ~ x1, data = dat, family = "poisson",
                   coords = as.matrix(dat[, c("s1", "s2")]), cor.fn = "matern",
                   params.list = list(phi = c(3, 5, 7), nu = c(0.5, 1.5),
                                      boundary = c(0.5)),
                   n.samples = 100,
                   loopd.controls = list(method = "CV", CV.K = 10, nMC = 500),
                   verbose = TRUE)

# Recover posterior samples of scale parameters
mod1.1 <- recoverGLMscale(mod1)

# sample from the stacked posterior distribution
post_samps <- stackedSampler(mod1.1)
```

---

simBinary *Synthetic point-referenced binary data*

Description

Dataset of size 500, with a binary response variable indexed by spatial coordinates sampled uniformly from the unit square. The model includes one covariate and spatial random effects induced by a Matérn covariogram.

Usage

```
data(simBinary)
```

Format

a `data.frame` object.

`s1, s2` 2-D coordinates; latitude and longitude.

`x1` a covariate sampled from the standard normal distribution.

`y` response vector (0/1).

`z_true` true spatial random effects that generated the data.

Details

With $n = 500$, the binary data is simulated using

$$y(s_i) \sim \text{Bernoulli}(\pi(s_i)), i = 1, \ldots, n,$$
$$\pi(s_i) = \text{ilogit}(x(s_i)^\top \beta + z(s_i))$$

where the function ilogit refers to the inverse-logit function, the spatial effects $z \sim N(0, \sigma^2 R)$ with R being a $n \times n$ correlation matrix given by the Matérn covariogram

$$R(s, s') = \frac{(\phi|s - s'|)^\nu}{\Gamma(\nu)2^{\nu-1}} K_\nu(\phi|s - s'|),$$

where ϕ is the spatial decay parameter and ν the spatial smoothness parameter. We have sampled the data with $\beta = (0.5, -0.5)$, $\phi = 5$, $\nu = 0.5$, and $\sigma^2 = 0.4$. This data can be generated with the code as given in the example below.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

See Also

simGaussian, simPoisson, simBinom

Examples

```
set.seed(1729)
n <- 500
beta <- c(0.5, -0.5)
phi0 <- 5
nu0 <- 0.5
spParams <- c(phi0, nu0)
spvar <- 0.4
sim1 <- sim_spData(n = n, beta = beta, cor.fn = "matern",
                   spParams = spParams, spvar = spvar, deltasq = deltasq,
                   family = "binary")
plot1 <- surfaceplot(sim1, coords_name = c("s1", "s2"), var_name = "z_true")

library(ggplot2)
plot2 <- ggplot(sim1, aes(x = s1, y = s2)) +
```

```
  geom_point(aes(color = factor(y)), alpha = 0.75) +
  scale_color_manual(values = c("red", "blue"), labels = c("0", "1")) +
  guides(alpha = 'none') +
  theme_bw() +
  theme(axis.ticks = element_line(linewidth = 0.25),
        panel.background = element_blank(),
        panel.grid = element_blank(),
        legend.title = element_text(size = 10, hjust = 0.25),
        legend.box.just = "center", aspect.ratio = 1)
```

`simBinom` *Synthetic point-referenced binomial count data*

Description

Dataset of size 500, with a binomial response variable indexed by spatial coordinates sampled uniformly from the unit square. The model includes one covariate and spatial random effects induced by a Matérn covariogram. The number of trials at each location is sampled from a Poisson distribution with mean 20.

Usage

```
data(simBinom)
```

Format

a `data.frame` object.

`s1, s2` 2-D coordinates; latitude and longitude.

`x1` a covariate sampled from the standard normal distribution.

`y` response vector.

`n_trials` Number of trials at that location.

`z_true` true spatial random effects that generated the data.

Details

With $n = 500$, the count data is simulated using

$$y(s_i) \sim \text{Binomial}(m(s_i), \pi(s_i)), i = 1, \ldots, n,$$
$$\pi(s_i) = \text{ilogit}(x(s_i)^\top \beta + z(s_i))$$

where the function ilogit refers to the inverse-logit function, the number of trials $m(s_i)$ is sampled from a Poisson distribution with mean 20, the spatial effects $z \sim N(0, \sigma^2 R)$ with R being a $n \times n$ correlation matrix given by the Matérn covariogram

$$R(s, s') = \frac{(\phi|s - s'|)^\nu}{\Gamma(\nu)2^{\nu-1}} K_\nu(\phi|s - s'|),$$

where ϕ is the spatial decay parameter and ν the spatial smoothness parameter. We have sampled the data with $\beta = (0.5, -0.5)$, $\phi = 3$, $\nu = 0.5$, and $\sigma^2 = 0.4$. This data can be generated with the code as given in the example below.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

See Also

simGaussian, simPoisson, simBinary

Examples

```
set.seed(1729)
n <- 500
beta <- c(0.5, -0.5)
phi0 <- 3
nu0 <- 0.5
spParams <- c(phi0, nu0)
spvar <- 0.4
sim1 <- sim_spData(n = n, beta = beta, cor.fn = "matern",
                   spParams = spParams, spvar = spvar, deltasq = deltasq,
                   n_binom = rpois(n, 20),
                   family = "binomial")
plot1 <- surfaceplot(sim1, coords_name = c("s1", "s2"), var_name = "z_true")

library(ggplot2)
plot2 <- ggplot(sim1, aes(x = s1, y = s2)) +
  geom_point(aes(color = y), alpha = 0.75) +
  scale_color_distiller(palette = "RdYlGn", direction = -1,
                        label = function(x) sprintf("%.0f", x)) +
  guides(alpha = 'none') +
  theme_bw() +
  theme(axis.ticks = element_line(linewidth = 0.25),
        panel.background = element_blank(),
        panel.grid = element_blank(),
        legend.title = element_text(size = 10, hjust = 0.25),
        legend.box.just = "center", aspect.ratio = 1)
```

`simGaussian` *Synthetic point-referenced Gaussian data*

Description

Dataset of size 500 with a Gaussian response variable, simulated with spatial coordinates sampled uniformly from the unit square. The model includes one covariate and spatial random effects induced by a Matérn covariogram.

Usage

```
data(simGaussian)
```

Format

a `data.frame` object.

`s1, s2` 2-D coordinates; latitude and longitude.

`x1` a covariate sampled from the standard normal distribution.

`y` response vector.

`z_true` true spatial random effects that generated the data.

Details

The data is generated using the model

$$y = X\beta + z + \epsilon,$$

where the spatial effects $z \sim N(0, \sigma^2 R)$ is independent of the measurement error $\epsilon \sim N(0, \delta^2\sigma^2 I_n)$ with δ^2 being the noise-to-spatial variance ratio and R being a $n \times n$ correlation matrix given by the Matérn covariogram

$$R(s, s') = \frac{(\phi|s - s'|)^\nu}{\Gamma(\nu)2^{\nu-1}} K_\nu(\phi|s - s'|),$$

where ϕ is the spatial decay parameter and ν the spatial smoothness parameter. We have sampled the data with $\beta = (2, 5)$, $\phi = 2$, $\nu = 0.5$, $\delta^2 = 1$ and $\sigma^2 = 0.4$. This data can be generated with the code as given in the example.

Author(s)

Soumyakanti Pan span18@ucla.edu

See Also

simPoisson, simBinom, simBinary

Examples

```
set.seed(1729)
n <- 500
beta <- c(2, 5)
phi0 <- 2
nu0 <- 0.5
spParams <- c(phi0, nu0)
spvar <- 0.4
deltasq <- 1
sim1 <- sim_spData(n = n, beta = beta, cor.fn = "matern",
                   spParams = spParams, spvar = spvar, deltasq = deltasq,
                   family = "gaussian")
plot1 <- surfaceplot(sim1, coords_name = c("s1", "s2"), var_name = "z_true",
                     mark_points = TRUE)
plot1

library(ggplot2)
plot2 <- ggplot(sim1, aes(x = s1, y = s2)) +
```

```
  geom_point(aes(color = y), alpha = 0.75) +
  scale_color_distiller(palette = "RdYlGn", direction = -1,
                        label = function(x) sprintf("%.0f", x)) +
  guides(alpha = 'none') + theme_bw() +
  theme(axis.ticks = element_line(linewidth = 0.25),
        panel.background = element_blank(), panel.grid = element_blank(),
        legend.title = element_text(size = 10, hjust = 0.25),
        legend.box.just = "center", aspect.ratio = 1)
plot2
```

`simPoisson` *Synthetic point-referenced Poisson count data*

Description

Dataset of size 500, with a Poisson distributed response variable indexed by spatial coordinates sampled uniformly from the unit square. The model includes one covariate and spatial random effects induced by a Matérn covariogram.

Usage

```
data(simPoisson)
```

Format

a `data.frame` object.

`s1, s2` 2-D coordinates; latitude and longitude.

`x1` a covariate sampled from the standard normal distribution.

`y` response vector.

`z_true` true spatial random effects that generated the data.

Details

With $n = 500$, the count data is simulated using

$$y(s_i) \sim \text{Poisson}(\lambda(s_i)), i = 1, \ldots, n,$$
$$\log \lambda(s_i) = x(s_i)^\top \beta + z(s_i)$$

where the spatial effects $z \sim N(0, \sigma^2 R)$ with R being a $n \times n$ correlation matrix given by the Matérn covariogram

$$R(s, s') = \frac{(\phi|s - s'|)^\nu}{\Gamma(\nu)2^{\nu-1}} K_\nu(\phi|s - s'|),$$

where ϕ is the spatial decay parameter and ν the spatial smoothness parameter. We have sampled the data with $\beta = (2, -0.5)$, $\phi = 5$, $\nu = 0.5$, and $\sigma^2 = 0.4$. This data can be generated with the code as given in the example below.

See Also

simGaussian, simBinom, simBinary

Examples

```
set.seed(1729)
n <- 500
beta <- c(2, -0.5)
phi0 <- 5
nu0 <- 0.5
spParams <- c(phi0, nu0)
spvar <- 0.4
sim1 <- sim_spData(n = n, beta = beta, cor.fn = "matern",
                   spParams = spParams, spvar = spvar, deltasq = deltasq,
                   family = "poisson")

# Plot an interpolated spatial surface of the true random spatial effects
plot1 <- surfaceplot(sim1, coords_name = c("s1", "s2"), var_name = "z_true")

# Plot the simulated count data
library(ggplot2)
plot2 <- ggplot(sim1, aes(x = s1, y = s2)) +
  geom_point(aes(color = y), alpha = 0.75) +
  scale_color_distiller(palette = "RdYlGn", direction = -1,
                        label = function(x) sprintf("%.0f", x)) +
  guides(alpha = 'none') + theme_bw() +
  theme(axis.ticks = element_line(linewidth = 0.25),
        panel.background = element_blank(), panel.grid = element_blank(),
        legend.title = element_text(size = 10, hjust = 0.25),
        legend.box.just = "center", aspect.ratio = 1)
```

`sim_spData` *Simulate spatial data on unit square*

Description

Generates synthetic spatial data of different types where the spatial co-ordinates are sampled uniformly on an unit square. Different types include point-referenced Gaussian, Poisson, binomial and binary data. The design includes an intercept and fixed covariates sampled from a standard normal distribution.

Usage

```
sim_spData(n, beta, cor.fn, spParams, spvar, deltasq, family, n_binom)
```

Arguments

n	sample size.
beta	a p-dimensional vector of fixed effects.
cor.fn	a quoted keyword that specifies the correlation function used to model the spatial dependence structure among the observations. Supported covariance model key words are: 'exponential' and 'matern'.
spParams	a numeric vector containing spatial process parameters - e.g., spatial decay and smoothness.
spvar	value of spatial variance parameter.
deltasq	value of noise-to-spatial variance ratio.
family	a character specifying the distribution of the response as a member of the exponential family. Valid inputs are 'gaussian', 'poisson', 'binary', and 'binomial'.
n_binom	necessary only when family = 'binomial'. Must be a vector of length n that will specify the number of trials for each observation. If it is of length 1, then that value is considered to be the common value for the number of trials for all n observations.

Value

a data.frame object containing the columns -

s1, s2 2D-coordinates in unit square

x1, x2, ... covariates, not including intercept

y response

n_trials present only when binomial data is generated

z_true true spatial effects with which the data is generated

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

Examples

```
set.seed(1729)
n <- 10
beta <- c(2, 5)
phi0 <- 2
nu0 <- 0.5
spParams <- c(phi0, nu0)
spvar <- 0.4
deltasq <- 1
sim1 <- sim_spData(n = n, beta = beta, cor.fn = "matern",
                   spParams = spParams, spvar = spvar, deltasq = deltasq,
                   family = "gaussian")
```

`sim_stvcPoisson` *Synthetic point-referenced spatial-temporal Poisson count data simulated using spatially-temporally varying coefficients*

Description

Dataset of size 500, with a Poisson distributed response variable indexed by spatial and temporal coordinates sampled uniformly from the unit square. The model includes an intercept and a covariate with spatially and temporally varying coefficients, and spatial-temporal random effects induced by a Matérn covariogram.

Usage

```
data(sim_stvcPoisson)
```

Format

a `data.frame` object.

`s1, s2` 2-D coordinates; latitude and longitude.

`t_coords` temporal coordinates.

`x1` a covariate sampled from the standard normal distribution.

`y` response vector.

`z1_true` true spatial-temporal random effect associated with the intercept.

`z2_true` true spatial-temporal random effect associated with the covariate.

Details

With $n = 500$, the count data is simulated using

$$y(s_i) \sim \text{Poisson}(\lambda(s_i)), i = 1, \ldots, n,$$
$$\log \lambda(s_i) = x(s_i)^\top \beta + x(s_i)^\top z(s_i),$$

where the spatial-temporal random effects $z(s) = (z_1(s), z_2(s))^\top$ with independent processes $z_j(s) \sim GP(0, \sigma_j^2 R(\cdot, \cdot; \theta_j))$ for $j = 1, 2$, and with $R(\cdot, \cdot; \theta_j)$ given by

$$R((s,t),(s',t');\theta_j) = \frac{1}{\phi_{tj}|t-t'|^2+1} \exp\left(-\frac{\phi_{sj}\|s-s'\|}{\sqrt{1+\phi_{tj}|t-t'|^2}}\right),$$

where ϕ_s is the spatial decay parameter, ϕ_t is the temporal decay parameter. We have sampled the data with $\beta = (2, -0.5)$, $\phi_{s1} = 2$, $\phi_{s2} = 3$, $\phi_{t1} = 2$, $\phi_{t2} = 4$, $\sigma_1^2 = \sigma_2^2 = 1$. This data can be generated with the code as given in the example below.

See Also

simPoisson, simGaussian, simBinom, simBinary

Examples

```
rmvn <- function(n, mu = 0, V = matrix(1)) {
p <- length(mu)
  if (any(is.na(match(dim(V), p))))
    stop("error: dimension mismatch.")
  D <- chol(V)
  t(matrix(rnorm(n * p), ncol = p) %*% D + rep(mu, rep(n, p)))
}
set.seed(1726)
n <- 500
beta <- c(2, -0.5)
p <- length(beta)
X <- cbind(rep(1, n), sapply(1:(p - 1), function(x) rnorm(n)))
X_tilde <- X
phi_s <- c(2, 3)
phi_t <- c(2, 4)
S <- data.frame(s1 = runif(n, 0, 1), s2 = runif(n, 0, 1))
Tm <- runif(n)
dist_S <- as.matrix(dist(as.matrix(S)))
dist_T <- as.matrix(dist(as.matrix(Tm)))
Vz1 <- 1/(1 + phi_t[1] * dist_T^2) * exp(- (phi_s[1] * dist_S) / sqrt(1 + phi_t[1] * dist_T^2))
Vz2 <- 1/(1 + phi_t[2] * dist_T^2) * exp(- (phi_s[2] * dist_S) / sqrt(1 + phi_t[2] * dist_T^2))
z1 <- rmvn(1, rep(0, n), Vz1)
z2 <- rmvn(1, rep(0, n), Vz2)
muFixed <- X %*% beta
muSpT <- X_tilde[, 1] * z1 + X_tilde[, 2] * z2
mu <- muFixed + muSpT
y <- sapply(1:n, function(x) rpois(n = 1, lambda = exp(mu[x])))
dat <- cbind(S, Tm, X[, -1], y, z1, z2)
names(dat) <- c("s1", "s2", "t_coords", paste("x", 1:(p - 1), sep = ""), "y", "z1_true", "z2_true")
```

---

spGLMexact *Univariate Bayesian spatial generalized linear model*

Description

Fits a Bayesian spatial generalized linear model with fixed values of spatial process parameters and some auxiliary model parameters. The output contains posterior samples of the fixed effects, spatial random effects and, if required, finds leave-one-out predictive densities.

Usage

```
spGLMexact(
  formula,
  data = parent.frame(),
  family,
  coords,
  cor.fn,
  priors,
```

```
  spParams,
  boundary = 0.5,
  n.samples,
  loopd = FALSE,
  loopd.method = "exact",
  CV.K = 10,
  loopd.nMC = 500,
  verbose = TRUE,
  ...
)
```

Arguments

formula	a symbolic description of the regression model to be fit. See example below.
data	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which spGLMexact is called.
family	Specifies the distribution of the response as a member of the exponential family. Supported options are 'poisson', 'binomial' and 'binary'.
coords	an $n \times 2$ matrix of the observation coordinates in $\mathbb{R}^2$ (e.g., easting and northing).
cor.fn	a quoted keyword that specifies the correlation function used to model the spatial dependence structure among the observations. Supported covariance model key words are: 'exponential' and 'matern'. See below for details.
priors	(optional) a list with each tag corresponding to a hyperparameter name and containing hyperprior details. Valid tags include V.beta, nu.beta, nu.z and sigmaSq.xi. Values of nu.beta and nu.z must be at least 2.1. If not supplied, uses defaults.
spParams	fixed values of spatial process parameters.
boundary	Specifies the boundary adjustment parameter. Must be a real number between 0 and 1. Default is 0.5.
n.samples	number of posterior samples to be generated.
loopd	logical. If loopd=TRUE, returns leave-one-out predictive densities, using method as given by loopd.method. Default is FALSE.
loopd.method	character. Ignored if loopd=FALSE. If loopd=TRUE, valid inputs are 'exact', 'CV' and 'PSIS'. The option 'exact' corresponds to exact leave-one-out predictive densities which requires computation almost equivalent to fitting the model n times. The options 'CV' and 'PSIS' are faster and they implement K-fold cross validation and Pareto-smoothed importance sampling to find approximate leave-one-out predictive densities (Vehtari *et al.* 2017).
CV.K	An integer between 10 and 20. Considered only if loopd.method='CV'. Default is 10 (as recommended in Vehtari *et. al* 2017).
loopd.nMC	Number of Monte Carlo samples to be used to evaluate leave-one-out predictive densities when loopd.method is set to either 'exact' or 'CV'.
verbose	logical. If verbose = TRUE, prints model description.
...	currently no additional argument.

Details

With this function, we fit a Bayesian hierarchical spatial generalized linear model by sampling exactly from the joint posterior distribution utilizing the generalized conjugate multivariate distribution theory (Bradley and Clinch 2024). Suppose $\chi = (s_1, \ldots, s_n)$ denotes the n spatial locations the response y is observed. Let $y(s)$ be the outcome at location s endowed with a probability law from the natural exponential family, which we denote by

$$y(s) \sim \mathrm{EF}(x(s)^\top \beta + z(s); b, \psi)$$

for some positive parameter $b > 0$ and unit log partition function ψ. We consider the following response models based on the input supplied to the argument `family`.

`'poisson'` It considers point-referenced Poisson responses $y(s) \sim \mathrm{Poisson}(e^{x(s)^\top \beta + z(s)})$. Here, $b = 1$ and $\psi(t) = e^t$.

`'binomial'` It considers point-referenced binomial counts $y(s) \sim \mathrm{Binomial}(m(s), \pi(s))$ where, $m(s)$ denotes the total number of trials and probability of success $\pi(s) = \mathrm{ilogit}(x(s)^\top \beta + z(s))$ at location s. Here, $b = m(s)$ and $\psi(t) = \log(1 + e^t)$.

`'binary'` It considers point-referenced binary data (0 or, 1) i.e., $y(s) \sim \mathrm{Bernoulli}(\pi(s))$, where probability of success $\pi(s) = \mathrm{ilogit}(x(s)^\top \beta + z(s))$ at location s. Here, $b = 1$ and $\psi(t) = \log(1 + e^t)$.

The hierarchical model is given as

$$\begin{aligned}
y(s_i) \mid \beta, z, \xi &\sim EF(x(s_i)^\top \beta + z(s_i) + \xi_i - \mu_i; b_i, \psi_y), i = 1, \ldots, n \\
\xi \mid \beta, z, \sigma^2_\xi, \alpha_\epsilon &\sim \mathrm{GCM_c}(\cdots), \\
\beta \mid \sigma^2_\beta &\sim N(0, \sigma^2_\beta V_\beta), \quad \sigma^2_\beta \sim \mathrm{IG}(\nu_\beta/2, \nu_\beta/2) \\
z \mid \sigma^2_z &\sim N(0, \sigma^2_z R(\chi; \phi, \nu)), \quad \sigma^2_z \sim \mathrm{IG}(\nu_z/2, \nu_z/2),
\end{aligned}$$

where $\mu = (\mu_1, \ldots, \mu_n)^\top$ denotes the discrepancy parameter. We fix the spatial process parameters ϕ and ν and the hyperparameters V_β, ν_β, ν_z and σ^2_ξ. The term ξ is known as the fine-scale variation term which is given a conditional generalized conjugate multivariate distribution as prior. For more details, see Pan *et al.* 2024. Default values for V_β, ν_β, ν_z, σ^2_ξ are diagonal with each diagonal element 100, 2.1, 2.1 and 0.1 respectively.

Value

An object of class `spGLMexact`, which is a list with the following tags -

priors details of the priors used, containing the values of the boundary adjustment parameter (`boundary`), the variance parameter of the fine-scale variation term (`simasq.xi`) and others.

samples a list of length 3, containing posterior samples of fixed effects (`beta`), spatial effects (`z`) and the fine-scale variation term (`xi`).

loopd If `loopd=TRUE`, contains leave-one-out predictive densities.

model.params Values of the fixed parameters that includes `phi` (spatial decay), `nu` (spatial smoothness).

The return object might include additional data that can be used for subsequent prediction and/or model fit evaluation.

Author(s)

Soumyakanti Pan span18@ucla.edu

References

Bradley JR, Clinch M (2024). "Generating Independent Replicates Directly from the Posterior Distribution for a Class of Spatial Hierarchical Models." *Journal of Computational and Graphical Statistics*, **0**(0), 1-17. doi:10.1080/10618600.2024.2365728.

Pan S, Zhang L, Bradley JR, Banerjee S (2024). "Bayesian Inference for Spatial-temporal Non-Gaussian Data Using Predictive Stacking." doi:10.48550/arXiv.2406.04655.

Vehtari A, Gelman A, Gabry J (2017). "Practical Bayesian Model Evaluation Using Leave-One-out Cross-Validation and WAIC." *Statistics and Computing*, **27**(5), 1413-1432. ISSN 0960-3174. doi:10.1007/s1122201696964.

See Also

spLMexact()

Examples

```
# Example 1: Analyze spatial poisson count data
data(simPoisson)
dat <- simPoisson[1:10, ]
mod1 <- spGLMexact(y ~ x1, data = dat, family = "poisson",
                   coords = as.matrix(dat[, c("s1", "s2")]),
                   cor.fn = "matern",
                   spParams = list(phi = 4, nu = 0.4),
                   n.samples = 100, verbose = TRUE)

# summarize posterior samples
post_beta <- mod1$samples$beta
print(t(apply(post_beta, 1, function(x) quantile(x, c(0.025, 0.5, 0.975)))))

# Example 2: Analyze spatial binomial count data
data(simBinom)
dat <- simBinom[1:10, ]
mod2 <- spGLMexact(cbind(y, n_trials) ~ x1, data = dat, family = "binomial",
                   coords = as.matrix(dat[, c("s1", "s2")]),
                   cor.fn = "matern",
                   spParams = list(phi = 3, nu = 0.4),
                   n.samples = 100, verbose = TRUE)

# summarize posterior samples
post_beta <- mod2$samples$beta
print(t(apply(post_beta, 1, function(x) quantile(x, c(0.025, 0.5, 0.975)))))

# Example 3: Analyze spatial binary data
data(simBinary)
dat <- simBinary[1:10, ]
mod3 <- spGLMexact(y ~ x1, data = dat, family = "binary",
```

```
                    coords = as.matrix(dat[, c("s1", "s2")]),
                    cor.fn = "matern",
                    spParams = list(phi = 4, nu = 0.4),
                    n.samples = 100, verbose = TRUE)

# summarize posterior samples
post_beta <- mod3$samples$beta
print(t(apply(post_beta, 1, function(x) quantile(x, c(0.025, 0.5, 0.975)))))
```

---

spGLMstack *Bayesian spatial generalized linear model using predictive stacking*

Description

Fits Bayesian spatial generalized linear model on a collection of candidate models constructed based on some candidate values of some model parameters specified by the user and subsequently combines inference by stacking predictive densities. See Pan, Zhang, Bradley, and Banerjee (2024) for more details.

Usage

```
spGLMstack(
  formula,
  data = parent.frame(),
  family,
  coords,
  cor.fn,
  priors,
  params.list,
  n.samples,
  loopd.controls,
  parallel = FALSE,
  solver = "ECOS",
  verbose = TRUE,
  ...
)
```

Arguments

formula	a symbolic description of the regression model to be fit. See example below.
data	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which spLMstack is called.
family	Specifies the distribution of the response as a member of the exponential family. Supported options are 'poisson', 'binomial' and 'binary'.
coords	an $n \times 2$ matrix of the observation coordinates in $\mathbb{R}^2$ (e.g., easting and northing).

`cor.fn`	a quoted keyword that specifies the correlation function used to model the spatial dependence structure among the observations. Supported covariance model key words are: `'exponential'` and `'matern'`. See below for details.
`priors`	(optional) a list with each tag corresponding to a parameter name and containing prior details. Valid tags include `V.beta`, `nu.beta`, `nu.z` and `sigmaSq.xi`.
`params.list`	a list containing candidate values of spatial process parameters for the `cor.fn` used, and, the boundary parameter.
`n.samples`	number of posterior samples to be generated.
`loopd.controls`	a list with details on how leave-one-out predictive densities (LOO-PD) are to be calculated. Valid tags include `method`, `CV.K` and `nMC`. The tag `method` can be either `'exact'` or `'CV'`. If sample size is more than 100, then the default is `'CV'` with `CV.K` equal to its default value 10 (Gelman *et al.* 2024). The tag `nMC` decides how many Monte Carlo samples will be used to evaluate the leave-one-out predictive densities, which must be at least 500 (default).
`parallel`	logical. If `parallel=FALSE`, the parallelization plan, if set up by the user, is ignored. If `parallel=TRUE`, the function inherits the parallelization plan that is set by the user via the function `future::plan()` only. Depending on the parallel backend available, users may choose their own plan. More details are available at `https://cran.R-project.org/package=future`.
`solver`	(optional) Specifies the name of the solver that will be used to obtain optimal stacking weights for each candidate model. Default is `'ECOS'`. Users can use other solvers supported by the CVXR-package package.
`verbose`	logical. If `TRUE`, prints model-specific optimal stacking weights.
`...`	currently no additional argument.

Details

Instead of assigning a prior on the process parameters ϕ and ν, the boundary adjustment parameter ϵ, we consider a set of candidate models based on some candidate values of these parameters supplied by the user. Suppose the set of candidate models is $\mathcal{M} = \{M_1, \ldots, M_G\}$. Then for each $g = 1, \ldots, G$, we sample from the posterior distribution $p(\sigma^2, \beta, z \mid y, M_g)$ under the model M_g and find leave-one-out predictive densities $p(y_i \mid y_{-i}, M_g)$. Then we solve the optimization problem

$$\max_{w_1,\ldots,w_G} \frac{1}{n} \sum_{i=1}^{n} \log \sum_{g=1}^{G} w_g p(y_i \mid y_{-i}, M_g)$$

$$\text{subject to} \quad w_g \geq 0, \sum_{g=1}^{G} w_g = 1$$

to find the optimal stacking weights $\hat{w}_1, \ldots, \hat{w}_G$.

Value

An object of class `spGLMstack`, which is a list including the following tags -

`family` the distribution of the responses as indicated in the function call

`samples` a list of length equal to total number of candidate models with each entry corresponding to a list of length 3, containing posterior samples of fixed effects (`beta`), spatial effects (`z`) and fine-scale variation term (`xi`) for that particular model.

`loopd` a list of length equal to total number of candidate models with each entry containing leave-one-out predictive densities under that particular model.

`loopd.method` a list containing details of the algorithm used for calculation of leave-one-out predictive densities.

`n.models` number of candidate models that are fit.

`candidate.models` a matrix with `n_model` rows with each row containing details of the model parameters and its optimal weight.

`stacking.weights` a numeric vector of length equal to the number of candidate models storing the optimal stacking weights.

`run.time` a `proc_time` object with runtime details.

`solver.status` solver status as returned by the optimization routine.

The return object might include additional data that is useful for subsequent prediction, model fit evaluation and other utilities.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

References

Pan S, Zhang L, Bradley JR, Banerjee S (2024). "Bayesian Inference for Spatial-temporal Non-Gaussian Data Using Predictive Stacking." doi:10.48550/arXiv.2406.04655.

Vehtari A, Simpson D, Gelman A, Yao Y, Gabry J (2024). "Pareto Smoothed Importance Sampling." *Journal of Machine Learning Research*, **25**(72), 1-58. URL https://jmlr.org/papers/v25/19-556.html.

See Also

`spGLMexact()`, `spLMstack()`

Examples

```
set.seed(1234)
data("simPoisson")
dat <- simPoisson[1:100,]
mod1 <- spGLMstack(y ~ x1, data = dat, family = "poisson",
                   coords = as.matrix(dat[, c("s1", "s2")]), cor.fn = "matern",
                  params.list = list(phi = c(3, 7, 10), nu = c(0.25, 0.5, 1.5),
                                     boundary = c(0.5, 0.6)),
                  n.samples = 1000,
                  loopd.controls = list(method = "CV", CV.K = 10, nMC = 1000),
                  parallel = TRUE, solver = "ECOS", verbose = TRUE)
```

```
# print(mod1$solver.status)
# print(mod1$run.time)

post_samps <- stackedSampler(mod1)
post_beta <- post_samps$beta
print(t(apply(post_beta, 1, function(x) quantile(x, c(0.025, 0.5, 0.975)))))

post_z <- post_samps$z
post_z_summ <- t(apply(post_z, 1, function(x) quantile(x, c(0.025, 0.5, 0.975))))

z_combn <- data.frame(z = dat$z_true,
                      zL = post_z_summ[, 1],
                      zM = post_z_summ[, 2],
                      zU = post_z_summ[, 3])

library(ggplot2)
plot_z <- ggplot(data = z_combn, aes(x = z)) +
 geom_errorbar(aes(ymin = zL, ymax = zU),
               width = 0.05, alpha = 0.15,
               color = "skyblue") +
 geom_point(aes(y = zM), size = 0.25,
            color = "darkblue", alpha = 0.5) +
 geom_abline(slope = 1, intercept = 0,
             color = "red", linetype = "solid") +
 xlab("True z") + ylab("Posterior of z") +
 theme_bw() +
 theme(panel.background = element_blank(),
       aspect.ratio = 1)
```

---

spLMexact *Univariate Bayesian spatial linear model*

Description

Fits a Bayesian spatial linear model with spatial process parameters and the noise-to-spatial variance ratio fixed to a value supplied by the user. The output contains posterior samples of the fixed effects, variance parameter, spatial random effects and, if required, leave-one-out predictive densities.

Usage

```
spLMexact(
  formula,
  data = parent.frame(),
  coords,
  cor.fn,
  priors,
  spParams,
  noise_sp_ratio,
```

```
  n.samples,
  loopd = FALSE,
  loopd.method = "exact",
  verbose = TRUE,
  ...
)
```

Arguments

formula	a symbolic description of the regression model to be fit. See example below.
data	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which spLMexact is called.
coords	an $n \times 2$ matrix of the observation coordinates in $\mathbb{R}^2$ (e.g., easting and northing).
cor.fn	a quoted keyword that specifies the correlation function used to model the spatial dependence structure among the observations. Supported covariance model key words are: 'exponential' and 'matern'. See below for details.
priors	a list with each tag corresponding to a parameter name and containing prior details.
spParams	fixed value of spatial process parameters.
noise_sp_ratio	noise-to-spatial variance ratio.
n.samples	number of posterior samples to be generated.
loopd	logical. If loopd=TRUE, returns leave-one-out predictive densities, using method as given by loopd.method. Default is FALSE.
loopd.method	character. Ignored if loopd=FALSE. If loopd=TRUE, valid inputs are 'exact' and 'PSIS'. The option 'exact' corresponds to exact leave-one-out predictive densities which requires computation almost equivalent to fitting the model n times. The option 'PSIS' is faster and finds approximate leave-one-out predictive densities using Pareto-smoothed importance sampling (Gelman *et al.* 2024).
verbose	logical. If verbose = TRUE, prints model description.
...	currently no additional argument.

Details

Suppose $\chi = (s_1, \ldots, s_n)$ denotes the n spatial locations the response y is observed. With this function, we fit a conjugate Bayesian hierarchical spatial model

$$y \mid z, \beta, \sigma^2 \sim N(X\beta + z, \delta^2\sigma^2 I_n), \quad z \mid \sigma^2 \sim N(0, \sigma^2 R(\chi; \phi, \nu)),$$
$$\beta \mid \sigma^2 \sim N(\mu_\beta, \sigma^2 V_\beta), \quad \sigma^2 \sim \text{IG}(a_\sigma, b_\sigma)$$

where we fix the spatial process parameters ϕ and ν, the noise-to-spatial variance ratio δ^2 and the hyperparameters μ_β, V_β, a_σ and b_σ. We utilize a composition sampling strategy to sample the model parameters from their joint posterior distribution which can be written as

$$p(\sigma^2, \beta, z \mid y) = p(\sigma^2 \mid y) \times p(\beta \mid \sigma^2, y) \times p(z \mid \beta, \sigma^2, y).$$

We proceed by first sampling σ^2 from its marginal posterior, then given the samples of σ^2, we sample β and subsequently, we sample z conditioned on the posterior samples of β and σ^2 (Banerjee 2020).

Value

An object of class `spLMexact`, which is a list with the following tags -

samples a list of length 3, containing posterior samples of fixed effects (`beta`), variance parameter (`sigmaSq`), spatial effects (`z`).

loopd If `loopd=TRUE`, contains leave-one-out predictive densities.

model.params Values of the fixed parameters that includes `phi` (spatial decay), `nu` (spatial smoothness) and `noise_sp_ratio` (noise-to-spatial variance ratio).

The return object might include additional data used for subsequent prediction and/or model fit evaluation.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

References

Banerjee S (2020). "Modeling massive spatial datasets using a conjugate Bayesian linear modeling framework." *Spatial Statistics*, **37**, 100417. ISSN 2211-6753. doi:10.1016/j.spasta.2020.100417.

Vehtari A, Simpson D, Gelman A, Yao Y, Gabry J (2024). "Pareto Smoothed Importance Sampling." *Journal of Machine Learning Research*, **25**(72), 1-58. URL https://jmlr.org/papers/v25/19-556.html.

See Also

`spLMstack()`

Examples

```
# load data
data(simGaussian)
dat <- simGaussian[1:100, ]

# setup prior list
muBeta <- c(0, 0)
VBeta <- cbind(c(1.0, 0.0), c(0.0, 1.0))
sigmaSqIGa <- 2
sigmaSqIGb <- 0.1
prior_list <- list(beta.norm = list(muBeta, VBeta),
                   sigma.sq.ig = c(sigmaSqIGa, sigmaSqIGb))

# supply fixed values of model parameters
phi0 <- 3
nu0 <- 0.75
noise.sp.ratio <- 0.8

mod1 <- spLMexact(y ~ x1, data = dat,
                  coords = as.matrix(dat[, c("s1", "s2")]),
```

```
                  cor.fn = "matern",
                  priors = prior_list,
                  spParams = list(phi = phi0, nu = nu0),
                  noise_sp_ratio = noise.sp.ratio,
                  n.samples = 100,
                  loopd = TRUE, loopd.method = "exact")

beta.post <- mod1$samples$beta
z.post.median <- apply(mod1$samples$z, 1, median)
dat$z.post.median <- z.post.median
plot1 <- surfaceplot(dat, coords_name = c("s1", "s2"),
                     var_name = "z_true")
plot2 <- surfaceplot(dat, coords_name = c("s1", "s2"),
                     var_name = "z.post.median")
plot1
plot2
```

`spLMstack` *Bayesian spatial linear model using predictive stacking*

Description

Fits Bayesian spatial linear model on a collection of candidate models constructed based on some candidate values of some model parameters specified by the user and subsequently combines inference by stacking predictive densities. See Zhang, Tang and Banerjee (2024) for more details.

Usage

```
spLMstack(
  formula,
  data = parent.frame(),
  coords,
  cor.fn,
  priors,
  params.list,
  n.samples,
  loopd.method,
  parallel = FALSE,
  solver = "ECOS",
  verbose = TRUE,
  ...
)
```

Arguments

`formula`	a symbolic description of the regression model to be fit. See example below.
`data`	an optional data frame containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `spLMstack` is called.

coords	an $n \times 2$ matrix of the observation coordinates in $\mathbb{R}^2$ (e.g., easting and northing).
cor.fn	a quoted keyword that specifies the correlation function used to model the spatial dependence structure among the observations. Supported covariance model key words are: 'exponential' and 'matern'. See below for details.
priors	a list with each tag corresponding to a parameter name and containing prior details. If not supplied, uses defaults.
params.list	a list containing candidate values of spatial process parameters for the cor.fn used, and, noise-to-spatial variance ratio.
n.samples	number of posterior samples to be generated.
loopd.method	character. Valid inputs are 'exact' and 'PSIS'. The option 'exact' corresponds to exact leave-one-out predictive densities. The option 'PSIS' is faster, as it finds approximate leave-one-out predictive densities using Pareto-smoothed importance sampling (Gelman *et al.* 2024).
parallel	logical. If parallel=FALSE, the parallelization plan, if set up by the user, is ignored. If parallel=TRUE, the function inherits the parallelization plan that is set by the user via the function future::plan() only. Depending on the parallel backend available, users may choose their own plan. More details are available at https://cran.R-project.org/package=future.
solver	(optional) Specifies the name of the solver that will be used to obtain optimal stacking weights for each candidate model. Default is "ECOS". Users can use other solvers supported by the CVXR-package package.
verbose	logical. If TRUE, prints model-specific optimal stacking weights.
...	currently no additional argument.

Details

Instead of assigning a prior on the process parameters ϕ and ν, noise-to-spatial variance ratio δ^2, we consider a set of candidate models based on some candidate values of these parameters supplied by the user. Suppose the set of candidate models is $\mathcal{M} = \{M_1, \ldots, M_G\}$. Then for each $g = 1, \ldots, G$, we sample from the posterior distribution $p(\sigma^2, \beta, z \mid y, M_g)$ under the model M_g and find leave-one-out predictive densities $p(y_i \mid y_{-i}, M_g)$. Then we solve the optimization problem

$$\max_{w_1,\ldots,w_G} \frac{1}{n} \sum_{i=1}^{n} \log \sum_{g=1}^{G} w_g p(y_i \mid y_{-i}, M_g)$$

$$\text{subject to} \quad w_g \geq 0, \sum_{g=1}^{G} w_g = 1$$

to find the optimal stacking weights $\hat{w}_1, \ldots, \hat{w}_G$.

Value

An object of class spLMstack, which is a list including the following tags -

samples a list of length equal to total number of candidate models with each entry corresponding to a list of length 3, containing posterior samples of fixed effects (beta), variance parameter (sigmaSq), spatial effects (z) for that model.

`loopd` a list of length equal to total number of candidate models with each entry containing leave-one-out predictive densities under that particular model.

`n.models` number of candidate models that are fit.

`candidate.models` a matrix with `n_model` rows with each row containing details of the model parameters and its optimal weight.

`stacking.weights` a numeric vector of length equal to the number of candidate models storing the optimal stacking weights.

`run.time` a `proc_time` object with runtime details.

`solver.status` solver status as returned by the optimization routine.

The return object might include additional data that is useful for subsequent prediction, model fit evaluation and other utilities.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

References

Vehtari A, Simpson D, Gelman A, Yao Y, Gabry J (2024). "Pareto Smoothed Importance Sampling." *Journal of Machine Learning Research*, **25**(72), 1-58. URL https://jmlr.org/papers/v25/19-556.html.

Zhang L, Tang W, Banerjee S (2024). "Bayesian Geostatistics Using Predictive Stacking." doi:10.48550/arXiv.2304.12414.

See Also

`spLMexact()`, `spGLMstack()`

Examples

```
set.seed(1234)
# load data and work with first 100 rows
data(simGaussian)
dat <- simGaussian[1:100, ]

# setup prior list
muBeta <- c(0, 0)
VBeta <- cbind(c(1.0, 0.0), c(0.0, 1.0))
sigmaSqIGa <- 2
sigmaSqIGb <- 2
prior_list <- list(beta.norm = list(muBeta, VBeta),
                   sigma.sq.ig = c(sigmaSqIGa, sigmaSqIGb))

mod1 <- spLMstack(y ~ x1, data = dat,
                  coords = as.matrix(dat[, c("s1", "s2")]),
                  cor.fn = "matern",
                  priors = prior_list,
```

```
                  params.list = list(phi = c(1.5, 3),
                                     nu = c(0.5, 1),
                                     noise_sp_ratio = c(1)),
                  n.samples = 1000, loopd.method = "exact",
                  parallel = FALSE, solver = "ECOS", verbose = TRUE)

post_samps <- stackedSampler(mod1)
post_beta <- post_samps$beta
print(t(apply(post_beta, 1, function(x) quantile(x, c(0.025, 0.5, 0.975)))))

post_z <- post_samps$z
post_z_summ <- t(apply(post_z, 1,
                       function(x) quantile(x, c(0.025, 0.5, 0.975))))

z_combn <- data.frame(z = dat$z_true,
                      zL = post_z_summ[, 1],
                      zM = post_z_summ[, 2],
                      zU = post_z_summ[, 3])

library(ggplot2)
plot1 <- ggplot(data = z_combn, aes(x = z)) +
  geom_point(aes(y = zM), size = 0.25,
             color = "darkblue", alpha = 0.5) +
  geom_errorbar(aes(ymin = zL, ymax = zU),
                width = 0.05, alpha = 0.15) +
  geom_abline(slope = 1, intercept = 0,
              color = "red", linetype = "solid") +
  xlab("True z") + ylab("Stacked posterior of z") +
  theme_bw() +
  theme(panel.background = element_blank(),
        aspect.ratio = 1)
```

`stackedSampler` *Sample from the stacked posterior distribution*

Description

A helper function to sample from the stacked posterior distribution to obtain final posterior samples that can be used for subsequent analysis. This function applies on outputs of functions `spLMstack()` and `spGLMstack()`.

Usage

```
stackedSampler(mod_out, n.samples)
```

Arguments

`mod_out`	an object that is an output of a model fit or a prediction task, i.e., the class should be either `spLMstack`, 'pp.spLMstack', `spGLMstack`, `pp.spGLMstack`, `stvcGLMexact`, or `pp.stvcGLMexact`.

`n.samples` (optional) If missing, inherits the number of posterior samples from the original output. Otherwise, it specifies number of posterior samples to draw from the stacked posterior. If it exceeds the number of posterior draws used in the original function, then a message is thrown and the samples are obtained by resampling. We recommended running the original model fit/prediction with enough samples.

Details

After obtaining the optimal stacking weights $\hat{w}_1, \ldots, \hat{w}_G$, posterior inference of quantities of interest subsequently proceed from the *stacked* posterior,

$$\tilde{p}(\cdot \mid y) = \sum_{g=1}^{G} \hat{w}_g p(\cdot \mid y, M_g),$$

where $\mathcal{M} = \{M_1, \ldots, M_g\}$ is the collection of candidate models.

Value

An object of class `stacked_posterior`, which is a list that includes the following tags -

beta samples of the fixed effect from the stacked joint posterior.

z samples of the spatial random effects from the stacked joint posterior.

The list may also include other scale parameters corresponding to the model.

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

See Also

`spLMstack()`, `spGLMstack()`

Examples

```
set.seed(1234)
data(simGaussian)
dat <- simGaussian[1:100, ]

mod1 <- spLMstack(y ~ x1, data = dat,
                  coords = as.matrix(dat[, c("s1", "s2")]),
                  cor.fn = "matern",
                  params.list = list(phi = c(1.5, 3),
                                     nu = c(0.5, 1),
                                     noise_sp_ratio = c(1)),
                  n.samples = 1000, loopd.method = "exact",
                  parallel = FALSE, solver = "ECOS", verbose = TRUE)
print(mod1$solver.status)
print(mod1$run.time)
```

```
post_samps <- stackedSampler(mod1)
post_beta <- post_samps$beta
print(t(apply(post_beta, 1, function(x) quantile(x, c(0.025, 0.5, 0.975)))))
```

`stvcGLMexact` *Bayesian spatially-temporally varying generalized linear model*

Description

Fits a Bayesian generalized linear model with spatially-temporally varying coefficients under fixed values of spatial-temporal process parameters and some auxiliary model parameters. The output contains posterior samples of the fixed effects, spatial-temporal random effects and, if required, finds leave-one-out predictive densities.

Usage

```
stvcGLMexact(
  formula,
  data = parent.frame(),
  family,
  sp_coords,
  time_coords,
  cor.fn,
  process.type,
  sptParams,
  priors,
  boundary = 0.5,
  n.samples,
  loopd = FALSE,
  loopd.method = "exact",
  CV.K = 10,
  loopd.nMC = 500,
  verbose = TRUE,
  ...
)
```

Arguments

`formula`	a symbolic description of the regression model to be fit. Variables in parenthesis are assigned spatially-temporally varying coefficients. See examples.
`data`	an optional data frame containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `stvcGLMexact` is called.
`family`	Specifies the distribution of the response as a member of the exponential family. Supported options are `'poisson'`, `'binomial'` and `'binary'`.

sp_coords	an $n \times 2$ matrix of the observation spatial coordinates in $\mathbb{R}^2$ (e.g., easting and northing).
time_coords	an $n \times 1$ matrix of the observation temporal coordinates in $\mathcal{T} \subseteq [0, \infty)$.
cor.fn	a quoted keyword that specifies the correlation function used to model the spatial-temporal dependence structure among the observations. Supported covariance model key words are: 'gneiting-decay' (Gneiting and Guttorp 2010). See below for details.
process.type	a quoted keyword specifying the model for the spatial-temporal process. Supported keywords are 'independent' which indicates independent processes for each varying coefficients characterized by different process parameters, independent.shared implies independent processes for the varying coefficients that shares common process parameters, and multivariate implies correlated processes for the varying coefficients modeled by a multivariate Gaussian process with an inverse-Wishart prior on the correlation matrix. The input for sptParams and priors must be given accordingly.
sptParams	fixed values of spatial-temporal process parameters in usually a list of length 2. If cor.fn='gneiting-decay', then it is a list of length 2 with tags phi_s and phi_t. If process.type='independent', then phi_s and phi_t contain fixed values of the r spatial-temporal processes, otherwise they will contain scalars. See examples below.
priors	(optional) a list with each tag corresponding to a hyperparameter name and containing hyperprior details. Valid tags include V.beta, nu.beta, nu.z, sigmaSq.xi and IW.scale. Values of nu.beta and nu.z must be at least 2.1. If not supplied, uses defaults.
boundary	Specifies the boundary adjustment parameter. Must be a real number between 0 and 1. Default is 0.5.
n.samples	number of posterior samples to be generated.
loopd	logical. If loopd=TRUE, returns leave-one-out predictive densities, using method as given by loopd.method. Default is FALSE.
loopd.method	character. Ignored if loopd=FALSE. If loopd=TRUE, valid inputs are 'exact', 'CV'. The option 'exact' corresponds to exact leave-one-out predictive densities which requires computation almost equivalent to fitting the model n times. The options 'CV' is faster as it implements K-fold cross validation to find approximate leave-one-out predictive densities (Vehtari *et al.* 2017).
CV.K	An integer between 10 and 20. Considered only if loopd.method='CV'. Default is 10 (as recommended in Vehtari *et. al* 2017).
loopd.nMC	Number of Monte Carlo samples to be used to evaluate leave-one-out predictive densities when loopd.method is set to either 'exact' or 'CV'.
verbose	logical. If verbose = TRUE, prints model description.
...	currently no additional argument.

Details

With this function, we fit a Bayesian hierarchical spatially-temporally varying generalized linear model by sampling exactly from the joint posterior distribution utilizing the generalized conjugate

multivariate distribution theory (Bradley and Clinch 2024). Suppose $\chi = (\ell_1, \ldots, \ell_n)$ denotes the n spatial-temporal co-ordinates in $\mathcal{L} = \mathcal{S} \times \mathcal{T}$, the response y is observed. Let $y(\ell)$ be the outcome at the co-ordinate ℓ endowed with a probability law from the natural exponential family, which we denote by

$$y(\ell) \sim \mathrm{EF}(x(\ell)^\top \beta + \tilde{x}(\ell)^\top z(\ell); b(\ell), \psi)$$

for some positive parameter $b(\ell) > 0$ and unit log partition function ψ. Here, $\tilde{x}(\ell)$ denotes covariates with spatially-temporally varying coefficients We consider the following response models based on the input supplied to the argument `family`.

`'poisson'` It considers point-referenced Poisson responses $y(\ell) \sim \mathrm{Poisson}(e^{x(\ell)^\top \beta + \tilde{x}(\ell)^\top z(\ell)})$. Here, $b(\ell) = 1$ and $\psi(t) = e^t$.

`'binomial'` It considers point-referenced binomial counts $y(\ell) \sim \mathrm{Binomial}(m(\ell), \pi(\ell))$ where, $m(\ell)$ denotes the total number of trials and probability of success $\pi(\ell) = \mathrm{ilogit}(x(\ell)^\top \beta + \tilde{x}(\ell)^\top z(\ell))$ at spatial-temporal co-ordinate ℓ. Here, $b = m(\ell)$ and $\psi(t) = \log(1 + e^t)$.

`'binary'` It considers point-referenced binary data (0 or, 1) i.e., $y(\ell) \sim \mathrm{Bernoulli}(\pi(\ell))$, where probability of success $\pi(\ell) = \mathrm{ilogit}(x(\ell)^\top \beta + \tilde{x}(\ell)^\top z(\ell))$ at spatial-temporal co-ordinate ℓ. Here, $b(\ell) = 1$ and $\psi(t) = \log(1 + e^t)$.

The hierarchical model is given as

$$\begin{aligned}
y(\ell_i) \mid \beta, z, \xi &\sim EF(x(\ell_i)^\top \beta + \tilde{x}(\ell_i)^\top z(s_i) + \xi_i - \mu_i; b_i, \psi_y), i = 1, \ldots, n \\
\xi \mid \beta, z, \sigma^2_\xi, \alpha_\epsilon &\sim \mathrm{GCM_c}(\cdots), \\
\beta \mid \sigma^2_\beta &\sim N(0, \sigma^2_\beta V_\beta), \quad \sigma^2_\beta \sim \mathrm{IG}(\nu_\beta/2, \nu_\beta/2) \\
z_j \mid \sigma^2_{z_j} &\sim N(0, \sigma^2_{z_j} R(\chi; \phi_s, \phi_t)), \quad \sigma^2_{z_j} \sim \mathrm{IG}(\nu_z/2, \nu_z/2), j = 1, \ldots, r
\end{aligned}$$

where $\mu = (\mu_1, \ldots, \mu_n)^\top$ denotes the discrepancy parameter. We fix the spatial-temporal process parameters ϕ_s and ϕ_t and the hyperparameters V_β, ν_β, ν_z and σ^2_ξ. The term ξ is known as the fine-scale variation term which is given a conditional generalized conjugate multivariate distribution as prior. For more details, see Pan *et al.* 2024. Default values for V_β, ν_β, ν_z, σ^2_ξ are diagonal with each diagonal element 100, 2.1, 2.1 and 0.1 respectively.

Value

An object of class `stvcGLMexact`, which is a list with the following tags -

priors details of the priors used, containing the values of the boundary adjustment parameter (`boundary`), the variance parameter of the fine-scale variation term (`simasq.xi`) and others.

samples a list of length 3, containing posterior samples of fixed effects (`beta`), spatial-temporal effects (`z`) and the fine-scale variation term (`xi`). The element with tag `z` will again be a list of length r, each containing posterior samples of the spatial-temporal random effects corresponding to each varying coefficient.

loopd If `loopd=TRUE`, contains leave-one-out predictive densities.

model.params Values of the fixed parameters that includes `phi_s` (spatial decay), `phi_t` (temporal smoothness).

The return object might include additional data that can be used for subsequent prediction and/or model fit evaluation.

Author(s)

Soumyakanti Pan span18@ucla.edu

References

Bradley JR, Clinch M (2024). "Generating Independent Replicates Directly from the Posterior Distribution for a Class of Spatial Hierarchical Models." *Journal of Computational and Graphical Statistics*, **0**(0), 1-17. doi:10.1080/10618600.2024.2365728.

T. Gneiting and P. Guttorp (2010). "Continuous-parameter spatio-temporal processes." In *A.E. Gelfand, P.J. Diggle, M. Fuentes, and P Guttorp, editors, Handbook of Spatial Statistics*, Chapman & Hall CRC Handbooks of Modern Statistical Methods, p 427–436. Taylor and Francis.

Pan S, Zhang L, Bradley JR, Banerjee S (2024). "Bayesian Inference for Spatial-temporal Non-Gaussian Data Using Predictive Stacking." doi:10.48550/arXiv.2406.04655.

Vehtari A, Gelman A, Gabry J (2017). "Practical Bayesian Model Evaluation Using Leave-One-out Cross-Validation and WAIC." *Statistics and Computing*, **27**(5), 1413-1432. ISSN 0960-3174. doi:10.1007/s1122201696964.

See Also

`spGLMexact()`

Examples

```
data("sim_stvcPoisson")
dat <- sim_stvcPoisson[1:100, ]

# Fit a spatial-temporal varying coefficient Poisson GLM
mod1 <- stvcGLMexact(y ~ x1 + (x1), data = dat, family = "poisson",
                     sp_coords = as.matrix(dat[, c("s1", "s2")]),
                     time_coords = as.matrix(dat[, "t_coords"]),
                     cor.fn = "gneiting-decay",
                     process.type = "multivariate",
                     sptParams = list(phi_s = 1, phi_t = 1),
                     verbose = FALSE, n.samples = 100)
```

`stvcGLMstack` *Bayesian spatially-temporally varying coefficients generalized linear model using predictive stacking*

Description

Fits Bayesian spatial-temporal generalized linear model with spatially-temporally varying coefficients on a collection of candidate models constructed based on some candidate values of some model parameters specified by the user and subsequently combines inference by stacking predictive densities. See Pan, Zhang, Bradley, and Banerjee (2024) for more details.

Usage

```
stvcGLMstack(
  formula,
  data = parent.frame(),
  family,
  sp_coords,
  time_coords,
  cor.fn,
  process.type,
  priors,
  candidate.models,
  n.samples,
  loopd.controls,
  parallel = FALSE,
  solver = "ECOS",
  verbose = TRUE,
  ...
)
```

Arguments

formula	a symbolic description of the regression model to be fit. Variables in parenthesis are assigned spatially-temporally varying coefficients. See examples.
data	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which stvcGLMstack is called.
family	Specifies the distribution of the response as a member of the exponential family. Supported options are 'poisson', 'binomial' and 'binary'.
sp_coords	an $n \times 2$ matrix of the observation spatial coordinates in $\mathbb{R}^2$ (e.g., easting and northing).
time_coords	an $n \times 1$ matrix of the observation temporal coordinates in $\mathcal{T} \subseteq [0, \infty)$.
cor.fn	a quoted keyword that specifies the correlation function used to model the spatial-temporal dependence structure among the observations. Supported covariance model key words are: 'gneiting-decay' (Gneiting and Guttorp 2010). See below for details.
process.type	a quoted keyword specifying the model for the spatial-temporal process. Supported keywords are 'independent' which indicates independent processes for each varying coefficients characterized by different process parameters, independent.shared implies independent processes for the varying coefficients that shares common process parameters, and multivariate implies correlated processes for the varying coefficients modeled by a multivariate Gaussian process with an inverse-Wishart prior on the correlation matrix. The input for sptParams and priors must be given accordingly.
priors	(optional) a list with each tag corresponding to a hyperparameter name and containing hyperprior details. Valid tags include V.beta, nu.beta, nu.z, sigmaSq.xi and IW.scale. Values of nu.beta and nu.z must be at least 2.1. If not supplied, uses defaults.

`candidate.models`
: an object of class `candidateModels` containing a list of candidate models for stacking. See `candidateModels()` for details.

`n.samples`
: number of samples to be drawn from the posterior distribution.

`loopd.controls`
: a list with details on how leave-one-out predictive densities (LOO-PD) are to be calculated. Valid tags include `method`, `CV.K` and `nMC`. The tag `method` can be either `'exact'` or `'CV'`. If sample size is more than 100, then the default is `'CV'` with `CV.K` equal to its default value 10 (Gelman *et al.* 2024). The tag `nMC` decides how many Monte Carlo samples will be used to evaluate the leave-one-out predictive densities, which must be at least 500 (default).

`parallel`
: logical. If `parallel=FALSE`, the parallelization plan, if set up by the user, is ignored. If `parallel=TRUE`, the function inherits the parallelization plan that is set by the user via the function `future::plan()` only. Depending on the parallel backend available, users may choose their own plan. More details are available at `https://cran.R-project.org/package=future`.

`solver`
: (optional) Specifies the name of the solver that will be used to obtain optimal stacking weights for each candidate model. Default is `'ECOS'`. Users can use other solvers supported by the CVXR-package package.

`verbose`
: logical. If `TRUE`, prints model-specific optimal stacking weights.

`...`
: currently no additional argument.

Value

An object of class `stvcGLMstack`, which is a list including the following tags -

`samples` a list of length equal to total number of candidate models with each entry corresponding to a list of length 3, containing posterior samples of fixed effects (`beta`), spatial effects (`z`), and fine scale variation `xi` for that model.

`loopd` a list of length equal to total number of candidate models with each entry containing leave-one-out predictive densities under that particular model.

`n.models` number of candidate models that are fit.

`candidate.models` a list of length `n_model` rows with each entry containing details of the model parameters.

`stacking.weights` a numeric vector of length equal to the number of candidate models storing the optimal stacking weights.

`run.time` a `proc_time` object with runtime details.

`solver.status` solver status as returned by the optimization routine.

This object can be further used to recover posterior samples of the scale parameters in the model, and subsequrently, to make predictions at new locations or times using the function `posteriorPredict()`.

Examples

```
set.seed(1234)
data("sim_stvcPoisson")
dat <- sim_stvcPoisson[1:100, ]
```

```
# create list of candidate models (multivariate)
mod.list2 <- candidateModels(list(phi_s = list(2, 3),
                                  phi_t = list(1, 2),
                                  boundary = c(0.5, 0.75)), "cartesian")

# fit a spatial-temporal varying coefficient model using predictive stacking
mod1 <- stvcGLMstack(y ~ x1 + (x1), data = dat, family = "poisson",
                     sp_coords = as.matrix(dat[, c("s1", "s2")]),
                     time_coords = as.matrix(dat[, "t_coords"]),
                     cor.fn = "gneiting-decay",
                     process.type = "multivariate",
                     candidate.models = mod.list2,
                     loopd.controls = list(method = "CV", CV.K = 10, nMC = 500),
                     n.samples = 500)
```

`surfaceplot` *Make a surface plot*

Description

Make a surface plot

Usage

```
surfaceplot(tab, coords_name, var_name, h = 8, col.pal, mark_points = FALSE)
```

Arguments

`tab`	a data-frame containing spatial co-ordinates and the variable to plot
`coords_name`	name of the two columns that contains the co-ordinates of the points
`var_name`	name of the column containing the variable to be plotted
`h`	integer; (optional) controls smoothness of the spatial interpolation as appearing in the `MBA::mba.surf()` function. Default is 8.
`col.pal`	Optional; color palette, preferably divergent, use `colorRampPalette` function from `grDevices`. Default is 'RdYlBu'.
`mark_points`	Logical; if `TRUE`, the input points are marked. Default is `FALSE`.

Value

a `ggplot` object containing the surface plot

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

Examples

```
data(simGaussian)
plot1 <- surfaceplot(simGaussian, coords_name = c("s1", "s2"),
                     var_name = "z_true")
plot1

# try your favourite color palette
col.br <- colorRampPalette(c("blue", "white", "red"))
col.br.pal <- col.br(100)
plot2 <- surfaceplot(simGaussian, coords_name = c("s1", "s2"),
                     var_name = "z_true", col.pal = col.br.pal)
plot2
```

---

surfaceplot2 *Make two side-by-side surface plots*

Description

Make two side-by-side surface plots, particularly useful towards a comparative study of two spatial surfaces.

Usage

```
surfaceplot2(
  tab,
  coords_name,
  var1_name,
  var2_name,
  h = 8,
  col.pal,
  mark_points = FALSE
)
```

Arguments

tab	a data-frame containing spatial co-ordinates and the variables to plot
coords_name	name of the two columns that contains the co-ordinates of the points
var1_name	name of the column containing the first variable to be plotted
var2_name	name of the column containing the second variable to be plotted
h	integer; (optional) controls smoothness of the spatial interpolation as appearing in the MBA::mba.surf() function. Default is 8.
col.pal	Optional; color palette, preferably divergent, use colorRampPalette function from grDevices. Default is 'RdYlBu'.
mark_points	Logical; if TRUE, the input points are marked. Default is FALSE.

Value

a list containing two ggplot objects

Author(s)

Soumyakanti Pan span18@ucla.edu,
Sudipto Banerjee sudipto@ucla.edu

Examples

```
data(simGaussian)
plots_2 <- surfaceplot2(simGaussian, coords_name = c("s1", "s2"),
                        var1_name = "z_true", var2_name = "y")
plots_2
```

Index