Package ‘spNNGP’

July 9, 2017

Title Spatial Regression Models using Nearest Neighbor Gaussian Processes

Version 0.1.0

Date 2017-6-14

Author Andrew O. Finley <finleya@msu.edu>, Abhirup Datta <abhidatta@jhu.edu>, Sudipto Banerjee <sudipto@ucla.edu>

Maintainer Andrew Finley <finleya@msu.edu>

Depends R (>= 1.8.0), coda, Formula, RANN

Description Fits Gaussian univariate Bayesian spatial regression models using Nearest Neighbor Gaussian Processes (NNGP).

Encoding UTF-8

URL http://blue.for.msu.edu/software.html

Repository CRAN

NeedsCompilation yes

R topics documented:

`spConjNNGP` *Function for fitting univariate Bayesian conjugate spatial regression models*

Description

The function `spConjNNGP` fits Gaussian univariate Bayesian conjugate spatial regression models using Nearest Neighbor Gaussian Processes (NNGP).

Usage

```
spConjNNGP(formula, data = parent.frame(), coords, n.neighbors = 15,
           theta.alpha, sigma.sq.IG, cov.model = "exponential",
           k.fold, score.rule,
           X.0, coords.0,
           n.omp.threads = 1, 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 `spConjNNGP` is called.

`coords` an $n \times 2$ matrix of the observation coordinates in R^2 (e.g., easting and northing).

`n.neighbors` number of neighbors used in the NNGP.

`theta.alpha` a vector or matrix of parameter values for `phi`, `nu`, and `alpha`, where $\alpha = \tau^2/sigma^2$`nu` is only required if `cov.model` equals `"matern"`. A vector is passed if you want to run the model using one set of parameters. The vector elements must be named and hold values for `phi`, `nu`, and `alpha`. If a matrix is passed, columns must be named and hold values for `phi`, `alpha`, and `nu`. Each row in the matrix defines each set of parameters for which the the model will be run.

`sigma.sq.IG` a vector of length two that holds the hyperparameters, *shape* and *scale* respectively, for the inverse-Gamma prior on σ^2.

`cov.model` a quoted keyword that specifies the covariance function used to model the spatial dependence structure among the observations. Supported covariance model key words are: `"exponential"`, `"matern"`, `"spherical"`, and `"gaussian"`. See below for details.

`k.fold` an optional argument used to specify the number of *k* folds for cross-validation. In k-fold cross-validation, the data specified in `model` is randomly partitioned into *k* equal sized subsamples. Of the *k* subsamples, *k*-1 subsamples are used to fit the model and the remaining *k* samples are used for prediction. The cross-validation process is repeated *k* times (the folds). Root mean squared prediction error (RMSPE) and continuous ranked probability score (CRPS; Gneiting and Raftery, 2007) rules are averaged over the *k* fold prediction results and reported for the parameter set(s) defined by `theta.alpha`. The parameter set that yields the *best* performance based on the scoring rule defined by `score.rule` is used fit the final model that uses all the data and make predictions if `X.0` and `coords.0` are specified. Results from the k-fold cross-validation are returned in the `k.fold.scores` matrix.

`score.rule` a quoted keyword "rmspe" or "crps" that specifies the scoring rule used to select the *best* parameter set, see argument definition for `k.fold` for more details.

`X.0` the design matrix for prediction locations. An intercept should be provided in column if one is specified in `model`.

`coords.0` the spatial coordinates corresponding to `X.0`.

n.omp.threads — a positive integer indicating the number of threads to use for SMP parallel processing. The package must be compiled for OpenMP support. For most Intel-based machines, we recommend setting n.omp.threads to up to the number of hyperthreaded cores.

verbose — if TRUE, model specification and progress of the sampler is printed to the screen. Otherwise, nothing is printed to the screen.

... — currently no additional arguments.

Value

An object of class cNNGP, which is a list comprising:

beta.hat — a matrix of regression coefficient estimates corresponding to parameter set(s) defined in theta.alpha.

theta.alpha.sigmaSq — the theta.alpha matrix with σ^2 estimates appended.

run.time — execution times for building the nearest neighbor index and parameter estimation reported using proc.time().

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

Author(s)

Andrew O. Finley <finleya@msu.edu>,
Abhirup Datta <abhidatta@jhu.edu>,
Sudipto Banerjee <sudipto@ucla.edu>

References

Datta, A., S. Banerjee, A.O. Finley, and A.E. Gelfand. (2016) Hierarchical Nearest-Neighbor Gaussian process models for large geostatistical datasets. Journal of the American Statistical Association, 111:800-812.

Gneiting, T and A.E. Raftery. (2007) Strictly proper scoring rules, prediction, and estimation. Journal of the American Statistical Association, 102:359-378.

Examples

```
## Not run:
rmvn <- function(n, mu=0, V = matrix(1)){
  p <- length(mu)
  if(any(is.na(match(dim(V),p))))
    stop("Dimension problem!")
  D <- chol(V)
  t(matrix(rnorm(n*p), ncol=p)
}

set.seed(1)

n <- 1000
```

```
coords <- cbind(runif(n,0,1), runif(n,0,1))

x <- as.matrix(cbind(1, rnorm(n)))

B <- as.matrix(c(1,5))

sigma.sq <- 5
tau.sq <- 0.1
phi <- 3/0.5

D <- as.matrix(dist(coords))
R <- exp(-phi*D)
w <- rmvn(1, rep(0,n), sigma.sq*R)
y <- rnorm(n, x%*%B + w, sqrt(tau.sq))

sigma.sq.IG <- c(2, sigma.sq)

cov.model <- "matern"

#theta.alpha <- c(3/0.5, tau.sq/sigma.sq, 2)
#names(theta.alpha) <- c("phi", "alpha", "nu")

n.test.sets <- 10
theta.alpha <- cbind(seq(phi,30,length.out=n.test.sets),
                     seq(tau.sq/sigma.sq,5,length.out=n.test.sets),
                     seq(0.5,2,length.out=n.test.set))

colnames(theta.alpha) <- c("phi", "alpha", "nu")

m.1 <- spConjNNGP(y~x-1, coords=coords, n.neighbors = 10,
                  X.0 = x, coords.0 = coords,
                  k.fold = 5, score.rule = "crps",
                  n.omp.threads = 2,
              theta.alpha = theta.alpha, sigma.sq.IG = sigma.sq.IG, cov.model = cov.model)

m.1$beta.hat
m.1$theta.alpha.sigmaSq
m.1$k.fold.scores


## End(Not run)
```

---

spNNGP *Function for fitting univariate Bayesian spatial regression models*

Description

The function spNNGP fits Gaussian univariate Bayesian spatial regression models using Nearest Neighbor Gaussian Processes (NNGP).

Usage

```
spNNGP(formula, data = parent.frame(), coords, method = "response",
n.neighbors = 15,
      starting, tuning, priors, cov.model = "exponential",
      n.samples, n.omp.threads = 1, verbose=TRUE, n.report=100, ...)
```

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 `spNNGP` is called.
`coords`	an $n \times 2$ matrix of the observation coordinates in R^2 (e.g., easting and northing).
`method`	a quoted keyword that specifies the type of NNGP sampling algorithm to use. Supported method key words are: `"response"` and `"sequential"`. See below for details.
`n.neighbors`	number of neighbors used in the NNGP.
`starting`	a list with each tag corresponding to a parameter name. Valid tags are `beta`, `sigma.sq`, `tau.sq`, `phi`, and `nu`. The value portion of each tag is the parameter's starting value.
`tuning`	a list with each tag corresponding to a parameter name. Valid tags are `sigma.sq`, `tau.sq`, `phi`, and `nu`. The value portion of each tag defines the variance of the Metropolis sampler Normal proposal distribution.
`priors`	a list with each tag corresponding to a parameter name. Valid tags are `sigma.sq.ig`, `tau.sq.ig`, `phi.unif`. Variance parameters, `simga.sq` and `tau.sq`, are assumed to follow an inverse-Gamma distribution, whereas the spatial decay `phi` and smoothness `nu` parameters are assumed to follow Uniform distributions. The hyperparameters of the inverse-Gamma are passed as a vector of length two, with the first and second elements corresponding to the *shape* and *scale*, respectively. The hyperparameters of the Uniform are also passed as a vector of length two with the first and second elements corresponding to the lower and upper support, respectively.
`cov.model`	a quoted keyword that specifies the covariance function used to model the spatial dependence structure among the observations. Supported covariance model key words are: `"exponential"`, `"matern"`, `"spherical"`, and `"gaussian"`. See below for details.
`n.samples`	the number of posterior samples to collect.
`n.omp.threads`	a positive integer indicating the number of threads to use for SMP parallel processing. The package must be compiled for OpenMP support. For most Intel-based machines, we recommend setting n.omp.threads to up to the number of hyperthreaded cores.
`verbose`	if `TRUE`, model specification and progress of the sampler is printed to the screen. Otherwise, nothing is printed to the screen.
`n.report`	the interval to report Metropolis sampler acceptance and MCMC progress.
`...`	currently no additional arguments.

Details

Model parameters can be fixed at their `starting` values by setting their `tuning` values to zero.

The *no nugget* model is specified by setting `tau.sq` to zero in the `starting` and `tuning` lists.

Value

An object of class `rNNGP` or `cNNGP` depending on the `method`, which is a list comprising:

`p.beta.samples` a `coda` object of posterior samples for the regression coefficients.

`p.theta.samples` a `coda` object of posterior samples for covariance parameters.

`run.time` execution times for building the nearest neighbor index and MCMC sampler reported using `proc.time()`.

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

Author(s)

Andrew O. Finley <finleya@msu.edu>,
Abhirup Datta <abhidatta@jhu.edu>,
Sudipto Banerjee <sudipto@ucla.edu>

References

Datta, A., S. Banerjee, A.O. Finley, and A.E. Gelfand. (2016) Hierarchical Nearest-Neighbor Gaussian process models for large geostatistical datasets. Journal of the American Statistical Association, 111:800-812.

Examples

```
## Not run:

rmvn <- function(n, mu=0, V = matrix(1)){
  p <- length(mu)
  if(any(is.na(match(dim(V),p))))
    stop("Dimension problem!")
  D <- chol(V)
  t(matrix(rnorm(n*p), ncol=p)
}

set.seed(1)

n <- 5000
coords <- cbind(runif(n,0,1), runif(n,0,1))
X <- as.matrix(cbind(1, rnorm(n)))

B <- as.matrix(c(1,5))
p <- length(B)
```

```
sigma.sq <- 2
tau.sq <- 1
phi <- 3/0.5

D <- as.matrix(dist(coords))
R <- exp(-phi*D)
w <- rmvn(1, rep(0,n), sigma.sq*R)
y <- rnorm(n, X

n.samples <- 2000

starting <- list("phi"=3/0.5, "sigma.sq"=5, "tau.sq"=1)

tuning <- list("phi"=0.1, "sigma.sq"=0.1, "tau.sq"=0.1)

priors <- list("phi.Unif"=c(3/1, 3/0.01), "sigma.sq.IG"=c(2, 1), "tau.sq.IG"=c(2, 1))

cov.model <- "exponential"

n.report <- 500
verbose <- TRUE

m.1 <- spNNGP(y~X-1, coords=coords, starting=starting, n.neighbors=10,
              tuning=tuning, priors=priors, cov.model=cov.model,
              n.samples=n.samples, n.omp.threads=2, verbose=verbose, n.report=n.report)

plot(mcmc(m.1$p.beta.samples), density=FALSE)

plot(mcmc(m.1$p.theta.samples), density=FALSE)


## End(Not run)
```

Index