Package 'spNNGP'

July 12, 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).

License GPL (>= 2)

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 $\alpha = \tau^2/\sigma^2$ and nu is only required if cov.model="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 a 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 the first 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 vector or matrix with σ^2 estimates appended.
k.fold.scores	results from the k-fold cross-validation if k.fold is specified.
y.0.hat	prediction if X.0 and coords.0 are specified.
y.0.var.hat	prediction variance if X.0 and coords.0 are specified.
run.time	execution time 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)%*%D + rep(mu,rep(n,p)))
```

```
}

##Make some data
set.seed(1)
n <- 2000
coords <- cbind(runif(n,0,1), runif(n,0,1))

x <- cbind(1, rnorm(n))

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

sigma.sq <- 5
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%*%B + w, sqrt(tau.sq))

ho <- sample(1:n, 1000)

y.ho <- y[ho]
x.ho <- x[ho,,drop=FALSE]
w.ho <- w[ho]
coords.ho <- coords[ho,]

y <- y[-ho]
x <- x[-ho,,drop=FALSE]
w <- w[-ho,,drop=FALSE]
coords <- coords[-ho,]

##Fit a Conjugate NNGP model and predict for the holdout
sigma.sq.IG <- c(2, sigma.sq)

cov.model <- "exponential"

g <- 10
theta.alpha <- cbind(seq(phi,30,length.out=g), seq(tau.sq/sigma.sq,5,length.out=g))

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

m.c <- spConjNNGP(y~x-1, coords=coords, n.neighbors = 10,
                  X.0 = x.ho, coords.0 = coords.ho,
                  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.c$beta.hat
m.c$theta.alpha.sigmaSq
m.c$k.fold.scores

plot(m.c$y0.hat, y.ho)
```

```
## 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 NNGP sampling algorithm. 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 sigma.sq, tau.sq, phi, and nu. nu needs to be specified if cov.model="matern". 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. If method="sequential" then only phi and nu need to be specified. 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 sNNGP 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 time 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)%*%D + rep(mu,rep(n,p)))
}

##Make some data
set.seed(1)
n <- 2000
coords <- cbind(runif(n,0,1), runif(n,0,1))

x <- cbind(1, rnorm(n))

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

sigma.sq <- 5
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%*%B + w, sqrt(tau.sq))

ho <- sample(1:n, 1000)

y.ho <- y[ho]
x.ho <- x[ho,,drop=FALSE]
w.ho <- w[ho]
coords.ho <- coords[ho,]

y <- y[-ho]
x <- x[-ho,,drop=FALSE]
w <- w[-ho,,drop=FALSE]
coords <- coords[-ho,]

##Fit a Response and Sequential NNGP model
n.samples <- 1000

starting <- list("phi"=phi, "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, 5), "tau.sq.IG"=c(2, 1))

cov.model <- "exponential"
```

```
n.report <- 500
verbose <- TRUE

m.s <- spNNGP(y~x-1, coords=coords, starting=starting, method="sequential", 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)

round(summary(m.s$p.beta.samples)$quantiles[,c(3,1,5)],2)
round(summary(m.s$p.theta.samples)$quantiles[,c(3,1,5)],2)

m.r <- spNNGP(y~x-1, coords=coords, starting=starting, method="response", 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)

round(summary(m.r$p.beta.samples)$quantiles[,c(3,1,5)],2)
round(summary(m.r$p.theta.samples)$quantiles[,c(3,1,5)],2)


## End(Not run)
```

---

spPredict *Function for prediction at new locations using* spNNGP *models.*

Description

The function spPredict collects posterior predictive samples for a set of new locations given a spNNGP object.

Usage

```
spPredict(sp.obj, X.0, coords.0, start=1, end, thin=1,
n.omp.threads = 1, verbose=TRUE, n.report=100, ...)
```

Arguments

sp.obj	an object returned by spNNGP.
X.0	the design matrix for prediction locations. An intercept should be provided in the first column if one is specified in sp.obj model.
coords.0	the spatial coordinates corresponding to X.0.
start	specifies the first sample included in the composition sampling.
end	specifies the last sample included in the composition. The default is to use all posterior samples in sp.obj.
thin	a sample thinning factor. The default of 1 considers all samples between start and end. For example, if thin = 10 then 1 in 10 samples are considered between start and end.

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 sampling progress.
...	currently no additional arguments.

Value

A list comprising:

p.y.0	a matrix that holds the response variable posterior predictive samples where rows are location corresponding to coords.0 and columns are samples.
p.w.0	a matrix that holds the random effect posterior predictive samples where rows are location corresponding to coords.0 and columns are samples. This is only returned if spNNGP method = "sequential".
run.time	execution time 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)%*%D + rep(mu,rep(n,p)))
}

##Make some data
set.seed(1)
n <- 2000
```

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

x <- cbind(1, rnorm(n))

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

sigma.sq <- 5
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%*%B + w, sqrt(tau.sq))

ho <- sample(1:n, 1000)

y.ho <- y[ho]
x.ho <- x[ho,,drop=FALSE]
w.ho <- w[ho]
coords.ho <- coords[ho,]

y <- y[-ho]
x <- x[-ho,,drop=FALSE]
w <- w[-ho,,drop=FALSE]
coords <- coords[-ho,]

##Fit a Response and Sequential NNGP model
n.samples <- 1000

starting <- list("phi"=phi, "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, 5), "tau.sq.IG"=c(2, 1))

cov.model <- "exponential"

n.report <- 500
verbose <- TRUE

m.s <- spNNGP(y~x-1, coords=coords, starting=starting, method="sequential", 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)

round(summary(m.s$p.beta.samples)$quantiles[,c(3,1,5)],2)
round(summary(m.s$p.theta.samples)$quantiles[,c(3,1,5)],2)

m.r <- spNNGP(y~x-1, coords=coords, starting=starting, method="response", 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)

round(summary(m.r$p.beta.samples)$quantiles[,c(3,1,5)],2)
```

```
round(summary(m.r$p.theta.samples)$quantiles[,c(3,1,5)],2)

##Prediction for holdout data
p.s <- spPredict(m.s, X.0 = x.ho, coords.0 = coords.ho, n.omp.threads=2)

plot(apply(p.s$p.y.0, 1, mean), y.ho)
plot(apply(p.s$p.w.0, 1, mean), w.ho)

p.r <- spPredict(m.r, X.0 = x.ho, coords.0 = coords.ho, n.omp.threads=2)

plot(apply(p.r$p.y.0, 1, mean), y.ho)

## End(Not run)
```

Index