Package ‘RandomForestsGLS’

October 12, 2022

Type Package

Title Random Forests for Dependent Data

Version 0.1.4

Maintainer Arkajyoti Saha <arkajyotisaha93@gmail.com>

Author Arkajyoti Saha [aut, cre],
   Sumanta Basu [aut],
   Abhirup Datta [aut]

Depends R (>= 3.3.0)

Imports BRISC, parallel, stats, matrixStats, randomForest, pbapply

Suggests knitr, rmarkdown, ggplot2, testthat (>= 2.1.0)


License GPL (>= 2)

URL https://github.com/ArkajyotiSaha/RandomForestsGLS

BugReports https://github.com/ArkajyotiSaha/RandomForestsGLS/issues

Encoding UTF-8

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-04-28 13:30:08 UTC

R topics documented:

 RFGLS_estimate_spatial ......................................................... 2
 RFGLS_estimate_timeseries ...................................................... 5
 RFGLS_predict ................................................................. 8
 RFGLS_predict_spatial ......................................................... 9

Index 12
Function for estimation in spatial data with RF-GLS

Description

The function RFGLS_estimate_spatial fits univariate non-linear spatial regression models for spatial data using RF-GLS in Saha et al. 2020. RFGLS_estimate_spatial uses the sparse Cholesky representation of Vecchia’s likelihood (Vecchia, 1988) developed in Datta et al., 2016 and Saha & Datta, 2018. The fitted Random Forest (RF) model is used later for prediction via the RFGLS_predict and RFGLS_predict_spatial.


Usage

RFGLS_estimate_spatial(coords, y, X, Xtest = NULL, nrnodes = NULL, nthsize = 20, mtry = 1, pinv_choice = 1, n_omp = 1, ntree = 50, h = 1, sigma.sq = 1, tau.sq = 0.1, phi = 5, nu = 0.5, n.neighbors = 15, cov.model = "exponential", search.type = "tree", param_estimate = FALSE, verbose = FALSE)

Arguments

coords an \( n \times 2 \) matrix of the observation coordinates in \( R^2 \) (e.g., easting and northing).
y an \( n \) length vector of response at the observed coordinates.
X an \( n \times p \) matrix of the covariates in the observation coordinates.
Xtest an \( n_{\text{test}} \times p \) matrix of covariates for prediction locations. Its Structure should be identical (including intercept) with that of covariates provided for estimation purpose in \( X \). If NULL, will use \( X \) as \( X_{\text{test}} \). Default value is NULL.
nrnodes the maximum number of nodes a tree can have. Default choice leads to the deepest tree contingent on nthsize. For significantly large \( n \), one needs to bound it for growing shallow trees which trades off efficiency for computation time.
nthsize minimum size of leaf nodes. We recommend not setting this value too small, as that will lead to very deep trees that takes a lot of time to be built and can produce unstable estimates. Default value is 20.
mtry number of variables randomly sampled at each partition as a candidate split
direction. We recommend using the value p/3 where p is the number of variables
in X. Default value is 1.

pinv_choice dictates the choice of method for obtaining the pseudoinverse involved in the
cost function and node representative evaluation. If pinv_choice = 0, SVD is
used (slower but more stable), if pinv_choice = 1, orthogonal decomposition
(faster, may produce unstable results if nthsize is too low) is used. Default
value is 1.

n_omp number of threads to be used, value can be more than 1 if source code is com-
ciled with OpenMP support. Default is 1.

ntree number of trees to be grown. This value should not be too small. Default value
is 50.

h number of core to be used in parallel computing setup for bootstrap samples. If
h = 1, there is no parallelization. Default value is 1.

sigma.sq value of sigma square. Default value is 1.

tau.sq value of tau square. Default value is 0.1.

phi value of phi. Default value is 5.

nu value of nu, only required for matern covariance model. Default value is 0.5.

n.neighbors number of neighbors used in the NNGP. Default value is 15.

cov.model keyword that specifies the covariance function to be used in modelling the spa-
tial dependence structure among the observations. Supported keywords are:
"exponential", "matern", "spherical", and "gaussian" for exponential, matern,
spherical and gaussian covariance function respectively. Default value is "expo-
nential".

search.type keyword that specifies type of nearest neighbor search algorithm to be used.
Supported keywords are: "tree" and "brute". Both of them provide the same
result, though "tree" should be faster. Default value is "tree".

param_estimate if TRUE, using the residuals obtained from fitting a classical RF with default
options and nodesize = nthsize, will estimate the coefficients corresponding
to cov.model from BRISC_estimate with the default options. Default value is FALSE.

verbose if TRUE, model specifications along with information regarding OpenMP support
and progress of the algorithm is printed to the screen. Otherwise, nothing is
printed to the screen. Default value is FALSE.

Value
A list comprising:

P_matrix an n × ntree matrix of zero indexed resamples. t-th column denote the n re-
samples used in the t-th tree.
predicted_matrix an ntest × ntree matrix of predictions. t-th column denote the predictions at
ntest datapoints obtained from the t-th tree.
predicted values at the $n_{test}$ prediction points. Average (rowMeans) of the
treewise predictions in predicted_matrix.

$X$ the matrix $X$.

$y$ the vector $y$.

RFGLS_Object object required for prediction.

Author(s)

Arkajyoti Saha <arkajotisaha93@gmail.com>,
Sumanta Basu <sumbose@cornell.edu>,
Abhirup Datta <abhidatta@jhu.edu>

References


Examples

```r
rmvn <- function(n, mu = 0, V = matrix(1)){
  p <- length(mu)
  if(any(is.na(match(dim(V),p))))
    stop("Dimension not right!")
  D <- chol(V)
  t(matrix(rnorm(n*p), ncol=p)%*%D + rep(mu,rep(n,p)))
}
set.seed(1)
n <- 200
coords <- cbind(runif(n,0,1), runif(n,0,1))
set.seed(2)
x <- as.matrix(rnorm(n),n,1)
sigma.sq = 1
phi = 5
tau.sq = 0.1
```
RFGLS_estimate_timeseries

\begin{verbatim}
D <- as.matrix(dist(coords))
R <- exp(-phi*D)
w <- rmvn(1, rep(0,n), sigma.sq*R)
y <- rnorm(n, 10*sin(pi * x) + w, sqrt(tau.sq))
estimation_result <- RFGLS_estimate_spatial(coords, y, x, ntree = 10)
\end{verbatim}

---

**RFGLS_estimate_timeseries**

*Function for estimation in time-series data with RF-GLS*

**Description**

The function `RFGLS_estimate_spatial` fits univariate non-linear regression models for time-series data using a RF-GLS in Saha et al. 2020. `RFGLS_estimate_spatial` uses the sparse Cholesky representation corresponding to \( AR(q) \) process. The fitted Random Forest (RF) model is used later for prediction via the `RFGLS-predict`.

Some code blocks are borrowed from the R packages: spNNGP: Spatial Regression Models for Large Datasets using Nearest Neighbor Gaussian Processes [https://CRAN.R-project.org/package=spNNGP](https://CRAN.R-project.org/package=spNNGP) and randomForest: Breiman and Cutler’s Random Forests for Classification and Regression [https://CRAN.R-project.org/package=randomForest](https://CRAN.R-project.org/package=randomForest).

**Usage**

\begin{verbatim}
RFGLS_estimate_timeseries(y, X, Xtest = NULL, nrnodes = NULL,
nthsize = 20, mtry = 1,
pinv_choice = 1, n_omp = 1,
ntree = 50, h = 1, lag_params = 0.5,
variance = 1,
param_estimate = FALSE,
verbose = FALSE)
\end{verbatim}

**Arguments**

- **y**: an \( n \) length vector of response at the observed time points.
- **X**: an \( n \times p \) matrix of the covariates in the observation time points.
- **Xtest**: an \( n_{\text{test}} \times p \) matrix of covariates for prediction. Its Structure should be identical (including intercept) with that of covariates provided for estimation purpose in \( X \). If NULL, will use \( X \) as \( X_{\text{test}} \). Default value is NULL.
- **nrnodes**: the maximum number of nodes a tree can have. Default choice leads to the deepest tree contingent on \( nthsize \). For significantly large \( n \), one needs to bound it for growing shallow trees which trades off efficiency for computation time.
**nthsize** minimum size of leaf nodes. We recommend not setting this value too small, as that will lead to very deep trees that takes a lot of time to be built and can produce unstable estimates. Default value is 20.

**mtry** number of variables randomly sampled at each partition as a candidate split direction. We recommend using the value $p/3$ where $p$ is the number of variables in $X$. Default value is 1.

**pinv_choice** dictates the choice of method for obtaining the pseudoinverse involved in the cost function and node representative evaluation. If $\text{pinv}\_\text{choice} = 0$, SVD is used (slower but more stable), if $\text{pinv}\_\text{choice} = 1$, orthogonal decomposition (faster, may produce unstable results if $\text{nthsize}$ is too low) is used. Default value is 1.

**n_omp** number of threads to be used, value can be more than 1 if source code is compiled with OpenMP support. Default is 1.

**ntree** number of trees to be grown. This value should not be too small. Default value is 50.

**h** number of core to be used in parallel computing setup for bootstrap samples. If $h = 1$, there is no parallelization. Default value is 1.

**lag_params** $q$ length vector of AR coefficients. If the parameters need to be estimated from AR($q$) process, should be any numeric vector of length $q$. For notations please see `arima`. Default value is 0.5.

**variance** variance of the white noise in temporal error. The function estimate is not affected by this. Default value is 1.

**param_estimate** if TRUE, using the residuals obtained from fitting a classical RF default options and \text{nodesize} = \text{nthsize}, will estimate the coefficients corresponding to AR($q$) from `arima` with the option, `include\_mean = FALSE`. Default value is FALSE.

**verbose** if TRUE, model specifications along with information regarding OpenMP support and progress of the algorithm is printed to the screen. Otherwise, nothing is printed to the screen. Default value is FALSE.

### Value

A list comprising:

- **P\_matrix** an $n \times n\text{tree}$ matrix of zero indexed resamples. $t$-th column denote the $n$ resamples used in the $t$-th tree.

- **predicted\_matrix** an $n_{test} \times n\text{tree}$ matrix of predictions. $t$-th column denote the predictions at $n_{test}$ datapoints obtained from the $t$-th tree.

- **predicted** predicted values at the $n_{test}$ prediction points. Average (\text{rowMeans}) of the treewise predictions in predicted\_matrix.

- **X** the matrix $X$.

- **y** the vector $y$.

- **RFGLS\_Object** object required for prediction.
Author(s)

Arkajyoti Saha <arkajyotisaha93@gmail.com>,
Sumanta Basu <sumbose@cornell.edu>,
Abhirup Datta <abhidatta@jhu.edu>

References


Examples

```r
rmvn <- function(n, mu = 0, V = matrix(1)) {
  p <- length(mu)
  if(any(is.na(match(dim(V), p))))
    stop("Dimension not right!")
  D <- chol(V)
  t(matrix(rnorm(n*p), ncol=p)%*%D + rep(mu, rep(n,p)))
}
set.seed(2)
n <- 200
x <- as.matrix(rnorm(n), n, 1)
sigma.sq <- 1
rho <- 0.5
set.seed(3)
b <- rho
s <- sqrt(sigma.sq)
eps = arima.sim(list(order = c(1,0,0), ar = b),
    n = n, rand.gen = rnorm, sd = s)
y <- eps + 10*sin(pi * x)
estimation_result <- RFGLS_estimate_timeseries(y, x, ntree = 10)
```
RFGLS_predict  

Prediction of mean function with RF-GLS

Description

The function RFGLS_predict predicts the mean function at a given set of covariates. It uses a fitted RF-GLS model in Saha et al. 2020 to obtain the predictions.

Some code blocks are borrowed from the R package: randomForest: Breiman and Cutler’s Random Forests for Classification and Regression  
https://CRAN.R-project.org/package=randomForest.

Usage

RFGLS_predict(RFGLS_out, Xtest, h = 1, verbose = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFGLS_out</td>
<td>an object obtained from RFGLS_estimateSpatial or RFGLS_estimate_timeseries.</td>
</tr>
<tr>
<td>Xtest</td>
<td>an $n_{test} \times p$ matrix of covariates for prediction. Its Structure should be identical (including intercept) with that of covariates provided for estimation purpose in $X$ in RFGLS_out.</td>
</tr>
<tr>
<td>h</td>
<td>number of core to be used in parallel computing setup for bootstrap samples. If $h = 1$, there is no parallelization. Default value is 1.</td>
</tr>
<tr>
<td>verbose</td>
<td>if TRUE, model specifications along with information regarding OpenMP support and progress of the algorithm is printed to the screen. Otherwise, nothing is printed to the screen. Default value is FALSE.</td>
</tr>
</tbody>
</table>

Value

A list comprising:

- **predicted_matrix**: an $n_{test} \times n_{tree}$ matrix of predictions. $t$-th column denote the predictions at $n_{test}$ datapoints obtained from the $t$-th tree.
- **predicted** preducted values at the $n_{test}$ prediction points. Average (rowMeans) of the treewise predictions in predicted_matrix

Author(s)

Arkajyoti Saha <arkajotisaha93@gmail.com>,  
Sumanta Basu <sumbose@cornell.edu>,  
Abhirup Datta <abhidatta@jhu.edu>
References


Examples

```r
rmvn <- function(n, mu = 0, V = matrix(1)){
  p <- length(mu)
  if(any(is.na(match(dim(V),p))))
    stop("Dimension not right!")
  D <- chol(V)
  t(matrix(rnorm(n*p), ncol=p)%*%D + rep(mu,rep(n,p)))
}

set.seed(2)
n <- 200
x <- as.matrix(rnorm(n),n,1)

sigma.sq <- 1
rho <- 0.5

set.seed(3)
b <- rho
s <- sqrt(sigma.sq)
eps = arima.sim(list(order = c(1,0,0), ar = b),
    n = n, rand.gen = rnorm, sd = s)

y <- eps + 10*sin(pi * x[,1])
estimation_result <- RFGLS_estimate_timeseries(y, x, ntree = 10)
Xtest <- matrix(seq(0,1, by = 1/1000), 1001, 1)
RFGLS_predict <- RFGLS_predict(estimation_result, Xtest)
```

Description

The function RFGLS_predict_spatial performs fast prediction on a set of new locations by combining non-linear mean estimate from a fitted RF-GLS model in Saha et al. 2020 with spatial kriging estimate obtained by using Nearest Neighbor Gaussian Processes (NNGP) (Datta et al., 2016).

Some code blocks are borrowed from the R packages: spNNGP: Spatial Regression Models for Large Datasets using Nearest Neighbor Gaussian Processes
https://CRAN.R-project.org/package=spNNGP and randomForest: Breiman and Cutler’s Random Forests for Classification and Regression
https://CRAN.R-project.org/package=randomForest.

Usage

RFGLS_predict_spatial(RFGLS_out, coords.0, Xtest,
                       h = 1, verbose = FALSE)

Arguments

RFGLS_out an object obtained from RFGLS_estimate_spatial.
coords.0 the spatial coordinates corresponding to prediction locations.
             Its structure should be same as that of coords in BRISC_estimation. Default covariate value is a column of 1 to adjust for the mean (intercept).
Xtest an ntest \times p matrix of covariates for prediction. Its Structure should be identical (including intercept) with that of covariates provided for estimation purpose in X in RFGLS_out.

h number of core to be used in parallel computing setup for bootstrap samples. If h = 1, there is no parallelization. Default value is 1.

verbose if TRUE, model specifications along with information regarding OpenMP support and progress of the algorithm is printed to the screen. Otherwise, nothing is printed to the screen. Default value is FALSE.

Value

A list comprising:

prediction predicted spatial response corresponding to Xtest and coords.0.

Author(s)

Arkajyoti Saha <arkajyotisaha93@gmail.com>,
Sumanta Basu <sumbose@cornell.edu>,
Abhirup Datta <abhidatta@jhu.edu>

References

RFGLS_predict_spatial


Examples

```r
rmvn <- function(n, mu = 0, V = matrix(1)){
  p <- length(mu)
  if(any(is.na(match(dim(V),p))))
    stop("Dimension not right!")
  D <- chol(V)
  t(matrix(rnorm(n*p), ncol=p)%*%D + rep(mu,rep(n,p)))
}

set.seed(1)
n <- 250
coords <- cbind(runif(n,0,1), runif(n,0,1))

set.seed(2)
x <- as.matrix(rnorm(n),n,1)
sigma.sq = 1
phi = 5
tau.sq = 0.1

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

y <- rnorm(n, 10*sin(pi * x) + w, sqrt(tau.sq))
estimation_result <- RFGLS_estimate_spatial(coords[1:200,], y[1:200],
matrix(x[1:200,],200,1), ntree = 10)
prediction_result <- RFGLS_predict_spatial(estimation_result,
coords[201:250,], matrix(x[201:250,],50,1))
```
Index

* model
  RFGLS_estimate_spatial, 2
  RFGLS_estimate_timeseries, 5
  RFGLS_predict, 8
  RFGLS_predict_spatial, 9

RFGLS_estimate_spatial, 2
RFGLS_estimate_timeseries, 5
RFGLS_predict, 8
RFGLS_predict_spatial, 9