Package 'BayesMRA'

August 18, 2020

Type Package

Title Bayesian Multi-Resolution Gaussian Process Approximations

Version 1.0.0

Date 2020-08-11

Description Software for fitting sparse Bayesian multi-resolution spatial models using Markov Chain Monte Carlo.

License GPL (>= 3)

Depends R (>= 3.5.0)

Imports fields, igraph, Matrix, mvnfast, Rcpp (>= 1.0.4.6), spam

RoxygenNote 7.1.0

Suggests knitr, pkgdown, rmarkdown, testthat (>= 2.1.0), covr

URL https://github.com/jtipton25/BayesMRA

BugReports https://github.com/jtipton25/BayesMRA/issues

VignetteBuilder knitr

Encoding UTF-8

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Author John Tipton [aut, cre]

Maintainer John Tipton <jrtipton@uark.edu>

Repository CRAN

Date/Publication 2020-08-18 09:52:11 UTC

R topics documented:

`BayesMRA` *BayesMRA*

Description

Software for fitting sparse multi-resolution spatial models

Author(s)

John Tipton

`make_Q_alpha_2d` *Generate CAR precision matrix*

Description

A function for setting up a conditional autoregressive (CAR) or simultaneous autoregressive (SAR) precision matrix for use as a prior in Bayesian models

Usage

```
make_Q_alpha_2d(n_dims, phi, use_spam = TRUE, prec_model = "CAR")
```

Arguments

`n_dims`	is a vector of length M that are the dimensions of the CAR/SAR matrix at each resolution
`phi`	is a vector of length M with each element between -1 and 1 that defines the strength of the autoregressive process. Typically this will be set to 1 for use as a prior in penalized Bayesian models
`use_spam`	is a boolean flag to determine whether the output is a list of spam matrix objects (`use_spam = TRUE`) or a an $n \times n$ sparse Matrix of class "dgCMatrix" `use_spam = FALSE`(see Matrix package for details)
`prec_model`	is a string that takes the values "CAR" or "SAR" and defines the graphical structure for the precision matrix.

Value

a list of $n \times n$ sparse spam matrices or Matrix matrices of class "dgCMatrix" (see Matrix package for details)

Examples

```
n_dims <- c(4, 8)
phi <- c(0.8, 0.9)
Q_alpha <- make_Q_alpha_2d(n_dims, phi)
## plot the precision matrix structure at each resolution
layout(matrix(1:2, 1, 2))
spam::display(Q_alpha[[1]])
spam::display(Q_alpha[[2]])
```

---

make_Q_alpha_tau2 *Title*

Description

Title

Usage

```
make_Q_alpha_tau2(Q_alpha, tau2, use_spam = TRUE)
```

Arguments

Q_alpha	a list of length M composed of matrices that are the correlation structure of the CAR prior on beta.
tau2	a vector of length M that contains the CAR prior precision matrices.
use_spam	a boolean that determines if the output matrix is of class "spam" (use_spam = TRUE) or of class "dgCMatrix" (use_spam = FALSE; see Matrix package for details).

Value

A sparse block diagonal matrix representing the precision matrices for all of the resolutions of the random effects.

Examples

```
n_dims <- c(4, 8)
phi <- c(0.8, 0.9)
tau2 <- c(3, 4)
Q_alpha <- make_Q_alpha_2d(n_dims, phi)
Q_alpha_tau2 <- make_Q_alpha_tau2(Q_alpha, tau2)
## plot the full precision matrix structure
```

```
spam::display(Q_alpha_tau2)
```

mcmc_mra *Bayesian Multi-resolution Spatial Regression*

Description

this function runs Markov Chain Monte Carlo to estimate the Bayesian multi-resolution spatial regression model.

Usage

```
mcmc_mra(
  y,
  X,
  locs,
  params,
  priors = NULL,
  M = 4,
  n_neighbors = 68,
  n_coarse_grid = 10,
  n_padding = 5L,
  n_cores = 1L,
  inits = NULL,
  config = NULL,
  verbose = FALSE,
  use_spam = TRUE,
  n_chain = 1
)
```

Arguments

`y` is a n vector of Gaussian data.

`X` is a $n \times p$ matrix of fixed effects (like latitude, elevation, etc)

`locs` is a $n \times 2$ matrix of observation locations.

`params` is a list of parameter settings. The list `params` must contain the following values:

- `n_adapt`: A positive integer number of adaptive MCMC iterations.
- `n_mcmc`: A positive integer number of total MCMC iterations post adaptation.
- `n_thin`: A positive integer number of MCMC iterations per saved sample.
- `n_message`: A positive integer number of frequency of iterations to output a progress message. For example, `n_message = 50` outputs progress messages every 50 iterations.

`priors` is the list of prior settings.

M	The number of resolutions.
n_neighbors	The expected number of neighbors for each interior basis function. This determines the basis radius parameter.
n_coarse_grid	The number of basis functions in one direction (e.g. n_coarse_grid = 10 results in a 10×10 course grid which is further extended by the number of additional padding basis functions given by n_padding.
n_padding	The number of additional boundary points to add on each boundary. For example, n_padding = 5 will add 5 boundary knots to the both the left and right side of the grid).
n_cores	is the number of cores for parallel computation using openMP.
inits	is the list of initial values if the user wishes to specify initial values. If these values are not specified, then the initial values will be randomly sampled from the prior.
config	is the list of configuration values if the user wishes to specify initial values. If these values are not specified, then default a configuration will be used.
verbose	Should verbose output be printed? Typically this is only useful for troubleshooting.
use_spam	is a boolean flag to determine whether the output is a list of spam matrix objects (use_spam = TRUE) or a an $n \times n$ sparse Matrix of class "dgCMatrix" use_spam = FALSE (see spam and Matrix packages for details).
n_chain	is the MCMC chain id. The default is 1.

Examples

```
set.seed(111)
## genereate the locations
locs <- matrix(runif(20), 10, 2)
## generate some covariates and regression coefficients
X <- cbind(1, matrix(rnorm(30), 10, 3))
beta <- rnorm(ncol(X))

## simulate the MRA process
M <- 2
MRA <- mra_wendland_2d(locs, M = M, n_coarse_grid = 4)
W <- do.call(cbind, MRA$W)

n_dims   <- rep(NA, length(MRA$W))
dims_idx <- NULL
for (i in 1:M) {
    n_dims[i] <- ncol(MRA$W[[i]])
    dims_idx  <- c(dims_idx, rep(i, n_dims[i]))
}
## set up the process precision matrices
Q_alpha <- make_Q_alpha_2d(sqrt(n_dims), c(0.9, 0.8))
Q_alpha_tau2 <- make_Q_alpha_tau2(Q_alpha, tau2 = c(2, 4))

## add in constraints so each resolution has random effects that sum to 0
A_constraint <- sapply(1:M, function(i){
```

```
    tmp = rep(0, sum(n_dims))
    tmp[dims_idx == i] <- 1
    return(tmp)
})
a_constraint <- rep(0, M)
alpha <- as.vector(spam::rmvnorm.prec.const(
    n = 1,
    mu = rep(0, nrow(W)),
    Q = Q_alpha_tau2,
    A = t(A_constraint),
    a = a_constraint))
## define the data
y <-  as.vector(X %*% beta + W %*% alpha + rnorm(10))

## define the params for MCMC fitting
params <- list(
    n_mcmc    = 5,
    n_adapt   = 5,
    n_thin    = 1,
    n_message = 5)

## define the model priors
priors <- list(
    alpha_tau2   = 1,
    beta_tau2    = 1,
    alpha_sigma2 = 1,
    beta_sigma2  = 1,
    mu_beta      = rep(0, ncol(X)),
    Sigma_beta   = 5 * diag(ncol(X)))

## Fit the MRA model using MCMC
out     <- mcmc_mra(
    y            = y,
    X            = X,
    locs         = locs,
    params       = params,
    priors       = priors,
    M            = 2,
    n_coarse_grid = 4,
    n_cores      = 1L,
    verbose      = FALSE
)
```

`mra_wendland_2d` *Code to construct the mutli-resolution sparse basis function representation for fitting spatial processes*

Description

Code to construct the mutli-resolution sparse basis function representation for fitting spatial processes

Usage

```
mra_wendland_2d(
  locs,
  M = 4,
  n_coarse_grid = 10,
  n_padding = 5L,
  n_neighbors = 68,
  use_spam = TRUE
)
```

Arguments

`locs`	The location variables in 2 dimensions over which to construct the basis function representation
`M`	The number of resolutions.
`n_coarse_grid`	The number of basis functions in one direction (e.g. `n_coarse_grid = 10` results in a 10×10 course grid which is further extended by the number of additional padding basis functions given by `n_padding`.
`n_padding`	The number of additional boundary points to add on each boundary. For example, n_padding = 5 will add 5 boundary knots to the both the left and right side of the grid).
`n_neighbors`	The expected number of neighbors for each interior basis function. This determines the basis radius parameter.
`use_spam`	is a boolean flag to determine whether the output is a list of `spam::spam` matrix objects (`use_spam = TRUE`) or a an $n \times n$ sparse Matrix of class `Matrix::dgCMatrix` `use_spam = FALSE` (see spam and Matrix packages for details).

Value

A list of objects including the MRA knots locations `locs_grid`, the Wendland basis representation matrix `W` at the observed locations, the basis radius `radius`, the numbers of resolutions `M`, the number of expected neighbors in the interior of each grid `n_neighbors`, the number of interior basis functions in one direction `n_coarse_grid`, the number of additional padding basis functions given by `n_padding`, and the setting `use_spam` which determines whether the MRA output uses the `spam` format.

Examples

```
set.seed(111)
locs <- matrix(runif(20), 10, 2)
MRA <- mra_wendland_2d(locs, M = 2, n_coarse_grid = 4)
## plot the MRA grid at different resolutions
layout(matrix(1:2, 1, 2))
```

```
plot(MRA$locs_grid[[1]])
plot(MRA$locs_grid[[2]])
```

`mra_wendland_2d_pred` *Code to construct the mutli-resolution sparse basis function representation for fitting spatial processes*

Description

Code to construct the mutli-resolution sparse basis function representation for fitting spatial processes

Usage

```
mra_wendland_2d_pred(locs, locs_pred, MRA, use_spam = TRUE)
```

Arguments

`locs`	The location variables in 2 dimensions over which to construct the basis function representation in the fitting stage.
`locs_pred`	The location variables in 2 dimensions over which to construct the basis function representation in the prediction stage.
`MRA`	The multi-resolution basis expansion at the observed locations. This object is the output of mra_wendland-2d() and is of class "mra_wendland_2d".
`use_spam`	is a boolean flag to determine whether the output is a list of spam matrix objects (`use_spam = TRUE`) or a an $n \times n$ sparse Matrix of class "dgCMatrix" `use_spam = FALSE` (see spam and Matrix packages for details).

Value

A list of objects including the MRA knots locations `locs_grid`, the Wendland basis representation matrix `W_pred` at the prediction locations, and the basis radius `radius`

Examples

```
set.seed(111)
locs <- matrix(runif(20), 10, 2)
locs_pred <- matrix(runif(20), 10, 2)
MRA <- mra_wendland_2d(locs, M = 2, n_coarse_grid = 4)
MRA_pred <- mra_wendland_2d_pred(locs, locs_pred, MRA)

## plot the MRA prediction grid at different resolutions
layout(matrix(1:2, 1, 2))
plot(MRA_pred$locs_grid[[1]])
plot(MRA_pred$locs_grid[[2]])
```

---

rmvn_arma	*A function for sampling from conditional multivariate normal distributions with mean A^-1b and covariance matrix A^-1.*

Description

A function for sampling from conditional multivariate normal distributions with mean A^-1b and covariance matrix A^-1.

Usage

```
rmvn_arma(A, b)
```

Arguments

A	`A` A $d \times d$ `matrix` for the Gaussian full conditional distribution precision matrix.
b	`b` A d `vector` for the Gaussian full conditional distribution mean.

Examples

```
set.seed(111)
A <- diag(4)
b <- rnorm(4)
sample <- rmvn_arma(A, b)
```

---

rmvn_arma_chol	*A function for sampling from conditional multivariate normal distributions with mean A^-1b and covariance matrix A^-1.*

Description

A function for sampling from conditional multivariate normal distributions with mean A^-1b and covariance matrix A^-1.

Usage

```
rmvn_arma_chol(A_chol, b)
```

Arguments

A_chol	`A` A $d \times d$ `matrix` for the Gaussian full conditional distribution precision matrix Cholesky factor.
b	`b` A d `vector` for the Gaussian full conditional distribution mean.

Examples

```
set.seed(111)
A <- diag(4)
A_chol <- chol(A)
b <- rnorm(4)
sample <- rmvn_arma_chol(A_chol, b)
```

---

rmvn_arma_scalar — *A function for sampling from conditional multivariate normal distributions with mean A^-1b and covariance matrix A^-1.*

Description

A function for sampling from conditional multivariate normal distributions with mean A^-1b and covariance matrix A^-1.

Usage

```
rmvn_arma_scalar(a, b)
```

Arguments

a	a A scalar for the Gaussian full conditional distribution precision.
b	b A d vector for the Gaussian full conditional distribution mean.

Examples

```
set.seed(111)
a <- 4
b <- rnorm(1)
sample <- rmvn_arma_scalar(a, b)
```

---

wendland_basis — *calculate the Wendland basis function*

Description

calculate the Wendland basis function

Usage

```
wendland_basis(d, radius)
```

Arguments

d	The distance over which to calculate the Wendland basis
radius	The effective radius over which the Wendland basis is defined

Value

The output of the Wendland basis applied to the distance d for a given radius radius.

Examples

```
layout(matrix(1:2, 1, 2))
curve(wendland_basis(sqrt(x^2), radius = 1), from = -2, to = 2)
curve(wendland_basis(sqrt(x^2), radius = 2), from = -2, to = 2)
```

Index