Package 'GpGp'

February 19, 2018

Type Package

Title Fast Gaussian Process Computation Using Vecchia's Approximation

Version 0.1.0

Date 2018-02-18

Maintainer Joseph Guinness <joeguinness@gmail.com>

Description Functions for reordering
input locations, finding ordered nearest neighbors
(with help from 'FNN' package), grouping operations,
approximate likelihood evaluations, profile likelihoods,
Gaussian process predictions, and conditional simulations.
Covariance functions for spatial and spatial-temporal data
on Euclidean domains and spheres are provided. The original
approximation is due to Vecchia (1988)
<http://www.jstor.org/stable/2345768>, and the reordering and
grouping methods are from Guinness (2018)
<doi:10.1080/00401706.2018.1437476>.

Depends R (>= 2.10)

License MIT + file LICENSE

Imports Rcpp (>= 0.12.13), FNN

Suggests fields, knitr, rmarkdown, testthat

LinkingTo Rcpp

RoxygenNote 6.0.1

VignetteBuilder knitr

LazyData true

NeedsCompilation yes

Author Joseph Guinness [aut, cre],
Matthias Katzfuss [aut]

Repository CRAN

Date/Publication 2018-02-19 10:14:03 UTC

R topics documented:

`cond_sim` *Conditional Simulation using Vecchia's approximation*

Description

With the prediction locations ordered after the observation locations, an approximation for the inverse Cholesky of the covariance matrix is computed, and standard formulas are applied to obtain a conditional simulation.

Usage

```
cond_sim(covparms, covfun_name = "matern_isotropic", y_obs, locs_obs,
  locs_pred, X_obs, X_pred, beta, m = 60, nsims = 1, reorder = TRUE)
```

Arguments

covparms	Covariance parameters
covfun_name	Name of covariance function
y_obs	Observations associated with locs_obs
locs_obs	observation locations
locs_pred	prediction locations
X_obs	Design matrix for observations
X_pred	Design matrix for predictions
beta	Linear mean parameters
m	Number of nearest neighbors to use. Larger m gives better approximations.
nsims	Number of conditional simulations to return.
reorder	TRUE/FALSE for whether reordering should be done. This should generally be kept at TRUE, unless testing out the effect of reordering.

---

fast_Gp_sim *Approximate GP simulation*

Description

Calculates an approximation to the inverse Cholesky factor of the covariance matrix using Vecchia's approximation, then the simulation is produced by solving a linear system with a vector of uncorrelated standard normals

Usage

```
fast_Gp_sim(covparms, covfun_name = "matern_isotropic", locs, m = 30)
```

Arguments

covparms	A vector of covariance parameters appropriate for the specified covariance function
covfun_name	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
locs	matrix of locations. Row i of locs specifies the location of element i of y, and so the length of y should equal the number of rows of locs.
m	Number of nearest neighbors to use in approximation

Value

vector of simulated values

Examples

```
locs <- as.matrix( expand.grid( (1:100)/100, (1:100)/100 ) )
y <- fast_Gp_sim(c(4,0.2,0.5,0), "matern_isotropic",  locs, 30 )
fields::image.plot( matrix(y,100,100) )
```

---

fast_Gp_sim_Linv *Approximate GP simulation with specified Linverse*

Description

In situations where we want to do many gaussian process simulations from the same model, we can compute Linverse once and reuse it, rather than recomputing for each identical simulation. This function also allows the user to input the vector of standard normals z.

Usage

```
fast_Gp_sim_Linv(Linv, NNarray, z = NULL)
```

Arguments

Linv	Matrix containing the entries of Linverse, usually the output from vecchia_Linv.
NNarray	Matrix of nearest neighbor indices, usually the output from find_ordered_nn
z	Optional vector of standard normals. If not specified, these are computed within the function.

Value

vector of simulated values

Examples

```
locs <- as.matrix( expand.grid( (1:100)/100, (1:100)/100 ) )
ord <- order_maxmin(locs)
locsord <- locs[ord,]
m <- 10
NNarray <- find_ordered_nn(locsord,m)
covparms <- c(2, 0.2, 1, 0)
Linv <- vecchia_Linv( covparms, "matern_isotropic", locsord, NNarray )
y <- fast_Gp_sim_Linv(Linv,NNarray)
y[ord] <- y
fields::image.plot( matrix(y,100,100) )
```

find_ordered_nn *Find ordered nearest neighbors.*

Description

Given a matrix of reordered locations, find the m nearest neighbors to each location, subject to the neighbors coming previously in the ordering. The algorithm uses the kdtree algorithm in the FNN package, adapted to the setting where the nearest neighbors must come from previous in the ordering.

Usage

```
find_ordered_nn(locs, m, lonlat = FALSE, space_time = FALSE,
  st_scale = NULL)
```

Arguments

locs	A matrix of locations. Each row of locs contains a location, which can be a point in Euclidean space R^d, a point in space-time R^d x T, a longitude and latitude (in degrees) giving a point on the sphere, or a longitude, latitude, and time giving a point in the sphere-time domain.
m	Number of neighbors to return
lonlat	TRUE/FALSE whether locations are longitudes and latitudes.
space_time	TRUE if locations are euclidean space-time locations, FALSE otherwise. If set to TRUE, temporal dimension is ignored.
st_scale	two-vector giving the amount by which the spatial and temporal coordinates are scaled. If NULL, the function uses the locations to automatically select a scaling. If set to FALSE, temporal dimension treated as another spatial dimension (not recommended).

Value

An matrix containing the indices of the neighbors. Row i of the returned matrix contains the indices of the nearest m locations to the i'th location. Indices are ordered within a row to be increasing in distance. By convention, we consider a location to neighbor itself, so the first entry of row i is i, the second entry is the index of the nearest location, and so on. Because each location neighbors itself, the returned matrix has m+1 columns.

Examples

```
locs <- as.matrix( expand.grid( (1:40)/40, (1:40)/40 ) )     # grid of locations
ord <- order_maxmin(locs)                                     # calculate an ordering
locsord <- locs[ord,]                                        # reorder locations
m <- 20
NNarray <- find_ordered_nn(locsord,20)             # find ordered nearest 20 neighbors
ind <- 100
# plot all locations in gray, first ind locations in black,
```

```
# ind location with magenta circle, m neighhbors with blue circle
plot( locs[,1], locs[,2], pch = 16, col = "gray" )
points( locsord[1:ind,1], locsord[1:ind,2], pch = 16 )
points( locsord[ind,1], locsord[ind,2], col = "magenta", cex = 1.5 )
points( locsord[NNarray[ind,2:(m+1)],1], locsord[NNarray[ind,2:(m+1)],2], col = "blue", cex = 1.5 )
```

`find_ordered_nn_brute` *Naive brute force nearest neighbor finder*

Description

Naive brute force nearest neighbor finder

Usage

```
find_ordered_nn_brute(locs, m)
```

Arguments

`locs`	matrix of locations
`m`	number of neighbors

Value

integer vector giving ordering

`fit_model` *Estimate mean and covariance parameters*

Description

Given a response, set of locations, (optionally) a design matrix, and a specified covariance function, return the maximum approximate likelihood estimates, using Vecchia's likelihood approximation.

Usage

```
fit_model(y, locs, X = NULL, covfun_name = "matern_isotropic",
  silent = FALSE, group = TRUE, reorder = TRUE)
```

Arguments

y	response vector
locs	matrix of locations. Each row is a single spatial or spatial-temporal location. If using one of the "matern_sphere" covariance functions, the locations should be longitudes and latitudes (in that order) in degrees.
X	design matrix. Each row contains covariates for the corresponding observation in y. If not specified, the function sets X to be a matrix with a single column of ones, that is, a constant mean function.
covfun_name	string name of a covariance function. Currently supported are "matern_isotropic", "matern_sphere", and "matern_sphere_time".
silent	TRUE/FALSE for whether to print some information during fitting.
group	TRUE/FALSE for whether to use the grouped version of the approximation (Guinness, 2018) or not. The grouped version is used by default.
reorder	TRUE/FALSE indicating whether maxmin ordering should be used (TRUE) or whether no reordering should be done before fitting (FALSE).

Details

The fit_model is a user-friendly model fitting function that automatically performs many of the auxiliary tasks needed for using Vecchia's approximation, including reordering, computing nearest neighbors, grouping, and optimization. Optimization proceeds in several steps, using increasingly accurate versions of the approximation. The first step using 5 neighbors, then 15 neighbors, and the last step uses 30 neighbors. The actual number of neighbors in the grouped version is guaranteed to be larger, though depends on the ordering and the configuration of the locations. We recommend always using group = TRUE since the grouping is guaranteed to improve the approximation.

The Jason-3 windspeed vignette is a useful source for a use-case of the fit_model function for data on sphere. The example below shows a very small example with a simulated dataset in 2d.

Value

A list object containing covariance parameter estimates, mean parameter estimates, and covariance matrix for mean parameter estimates.

Examples

```
n1 <- 10
n2 <- 10
n <- n1*n2
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
covparms <- c(2,0.1,1/2,0)
y <- 7 + fast_Gp_sim(covparms, "matern_isotropic", locs)
X <- as.matrix( rep(1,n) )
fit <- fit_model(y, locs, X, "matern_isotropic")
fit
```

GpGp *GpGp: Fast Gaussian Process Computing.*

Description

Vecchia's (1988) Gaussian process approximation has emerged among its competitors as a leader in computational scalability and accuracy. This package includes implementations of the original approximation, as well as several updates to it, including the reordered and grouped versions of the approximation outlined in Guinness (2018).

Details

The main functions of the package are `fit_model`, and `predictions`. `fit_model` returns maximum likelihood estimates for covariance parameters and linear mean parameters. The user is expected to store the response in a vector `y` and the locations–either spatial or spatial-temporal–in a matrix `locs`, which contains in each row the spatial or spatial-temporal location of the corresponding response. The user is also expected to select a covariance function and specify it with a string. Currently supported functions are `"matern_isotropic"` for spatial data, `"matern_sphere"` for spatial data over the globe, and `"matern_sphere_time"` for spatial-temporal data over the globe. For the `sphere` the coordinates should be expressed in longitudes and latitudes. If there are covariates, they can be expressed via a design matrix `X`, each row containing the covariates corresponding to the same row in `locs`.

For `predictions`, the user should specify prediction locations `locs_pred` and a prediction design matrix `X_pred`.

The vignettes are intended to be helpful for getting a sense of how these functions work. The windspeed vignette can be a guide for spatial and spatial-temporal data over the globe.

For Gaussian process researchers, the package also provides access to likelihood and profile likelihood functions, reordering functions, nearest neighbor-finding functions, grouping (partitioning) functions, and approximate simulation functions. We think that there is potential for improving the grouping algorithms, and scope for improving the approximations by tailoring the choice of ordering and the grouping algorithm to suit one another.

`group_obs` *Automatic grouping (partitioning) of locations*

Description

Take in an array of nearest neighbors, and automatically partition the array into groups that share neighbors. This is helpful to speed the computations and improve their accuracy. The function returns a list, with each list element containing one or several rows of NNarray. The algorithm attempts to find groupings such that observations within a group share many common neighbors.

Usage

```
group_obs(NNarray, exponent = 2)
```

Arguments

NNarray	Matrix of nearest neighbor indices, usually the result of find_ordered_nn.
exponent	Within the algorithm, two groups are merged if the number of unique neighbors raised to the exponent power is less than the sum of the unique numbers raised to the exponent power from the two groups.

Value

A list with elements defining the grouping. The list entries are:

- all_inds: vector of all indices of all blocks.
- last_ind_of_block: The ith entry tells us the location in all_inds of the last index of the ith block. Thus the length of last_ind_of_block is the number of blocks, and last_ind_of_block can be used to chop all_inds up into blocks.
- global_resp_inds: The ith entry tells us the index of the ith response, as ordered in all_inds.
- local_resp_inds: The ith entry tells us the location within the block of the response index.
- last_resp_of_block: The ith entry tells us the location within local_resp_inds and global_resp_inds of the last index of the ith block. last_resp_of_block is to global_resp_inds and local_resp_inds as last_ind_of_block is to all_inds.

Examples

```
locs <- matrix( runif(200), 100, 2 )   # generate random locations
ord <- order_maxmin(locs)              # calculate an ordering
locsord <- locs[ord,]                  # reorder locations
m <- 10
NNarray <- find_ordered_nn(locsord,m)  # m nearest neighbor indices
NNlist2 <- group_obs(NNarray)          # join blocks if joining reduces squares
NNlist3 <- group_obs(NNarray,3)        # join blocks if joining reduces cubes
object.size(NNarray)
object.size(NNlist2)
object.size(NNlist3)
mean( NNlist2[["local_resp_inds"]] - 1 )   # average number of neighbors (exponent 2)
mean( NNlist3[["local_resp_inds"]] - 1 )   # average number of neighbors (exponent 3)

all_inds <- NNlist2$all_inds
last_ind_of_block <- NNlist2$last_ind_of_block
inds_of_block_2 <- all_inds[ (last_ind_of_block[1] + 1):last_ind_of_block[2] ]

local_resp_inds <- NNlist2$local_resp_inds
global_resp_inds <- NNlist2$global_resp_inds
last_resp_of_block <- NNlist2$last_resp_of_block
local_resp_of_block_2 <- local_resp_inds[(last_resp_of_block[1]+1):last_resp_of_block[2]]

global_resp_of_block_2 <- global_resp_inds[(last_resp_of_block[1]+1):last_resp_of_block[2]]
inds_of_block_2[local_resp_of_block_2]
# these last two should be the same
```

jason3 *Windspeed measurements from Jason-3 Satellite*

Description

A dataset containing lightly preprocessed windspeed values from the Jason-3 satellite. Observations near clouds and ice have been removed, and the data have been aggregated (averaged) over 10 second intervals. Jason-3 reports windspeeds over the ocean only. The data are from a six day period between August 4 and 9 of 2016.

Usage

```
jason3
```

Format

A data frame with 18973 rows and 4 columns

windspeed wind speed, in maters per second

lon longitude in degrees between 0 and 360

lat latitude in degrees between -90 and 90

time time in seconds from midnight August 4

Source

https://www.nodc.noaa.gov/SatelliteData/jason/

Linv_mult *Multiply approximate inverse Cholesky by a vector*

Description

Vecchia's approximation implies a sparse approximation to the inverse Cholesky factor of the covariance matrix. This function returns the result of multiplying that matrix by a vector.

Usage

```
Linv_mult(Linv, z, NNarray)
```

Arguments

Linv	Entries of the sparse inverse Cholesky factor, usually the output from vecchia_Linv.
z	the vector to be multiplied
NNarray	A matrix of indices, usually the output from find_ordered_nn. Row i contains the indices of the observations that observation i conditions on. By convention, the first element of row i is i.

Value

the product of the sparse inverse Cholesky factor with a vector

Examples

```
n <- 2000
locs <- matrix( runif(2*n), n, 2 )
covparms <- c(2, 0.2, 0.75, 0.1)
ord <- order_maxmin(locs)
NNarray <- find_ordered_nn(locs,20)
Linv <- vecchia_Linv( covparms, "matern_isotropic", locs, NNarray )
z1 <- rnorm(n)
y <- fast_Gp_sim_Linv(Linv,NNarray,z1)
z2 <- Linv_mult(Linv, y, NNarray)
print( sum( (z1-z2)^2 ) )
```

Linv_mult_grouped *Multiply approximate inverse Cholesky by a vector*

Description

Vecchia's approximation implies a sparse approximation to the inverse Cholesky factor of the covariance matrix. This function returns the result of multiplying that matrix by a vector.

Usage

```
Linv_mult_grouped(Linv, z, NNlist)
```

Arguments

Linv	Entries of the sparse inverse Cholesky factor, usually the output from vecchiaLinv.
z	the vector to be multiplied
NNlist	A list with grouped neighbor information. Usually the output from group_obs(NNarray).

Value

the product of the sparse inverse Cholesky factor with a vector

Examples

```
n <- 2000
locs <- matrix( runif(2*n), n, 2 )
covparms <- c(2, 0.2, 0.75, 0.1)
ord <- order_maxmin(locs)
NNarray <- find_ordered_nn(locs,20)
Linv <- vecchia_Linv( covparms, "matern_isotropic", locs, NNarray )
z1 <- rnorm(n)
y <- fast_Gp_sim_Linv(Linv,NNarray,z1)
z2 <- Linv_mult(Linv, y, NNarray)
print( sum( (z1-z2)^2 ) )
```

L_mult *Multiply approximate Cholesky by a vector*

Description

Vecchia's approximation implies a sparse approximation to the inverse Cholesky factor of the covariance matrix. This function returns the result of multiplying the inverse of that matrix by a vector (i.e. an approximation to the Cholesky factor).

Usage

```
L_mult(Linv, z, NNarray)
```

Arguments

Linv	Entries of the sparse inverse Cholesky factor, usually the output from vecchia_Linv.
z	the vector to be multiplied
NNarray	A matrix of indices, usually the output from find_ordered_nn. Row i contains the indices of the observations that observation i conditions on. By convention, the first element of row i is i.

Value

the product of the Cholesky factor with a vector

Examples

```
n <- 2000
locs <- matrix( runif(2*n), n, 2 )
covparms <- c(2, 0.2, 0.75, 0.1)
ord <- order_maxmin(locs)
NNarray <- find_ordered_nn(locs,20)
Linv <- vecchia_Linv( covparms, "matern_isotropic", locs, NNarray )
z <- rnorm(n)
y1 <- fast_Gp_sim_Linv(Linv,NNarray,z)
y2 <- L_mult(Linv, z, NNarray)
print( sum( (y1-y2)^2 ) )
```

matern_isotropic *Isotropic Matern covariance function*

Description

From a matrix of locations and covariance parameters of the form (variance, range, smoothness, nugget), return the square matrix of all pairwise covariances.

Usage

```
matern_isotropic(covparms, locs)
```

Arguments

covparms	A vector giving positive-valued covariance parameters in the form (variance, range, smoothness, nugget)
locs	A matrix with n rows and d columns. Each row of locs gives a point in R^d.

Value

A matrix with n rows and n columns, with the i,j entry containing the covariance between observations at `locs[i,]` and `locs[j,]`.

Parameterization

The covariance parameter vector is (variance, range, smoothness, nugget) = $(\sigma^2, \alpha, \nu, \tau^2)$, and the covariance function is parameterized as

$$M(x,y) = \sigma^2 2^{1-\nu}/\Gamma(\nu)(||x-y||/\alpha)^\nu K_\nu(||x-y||/\alpha)$$

The nugget value $\sigma^2\tau^2$ is added to the diagonal of the covariance matrix. NOTE: the nugget is $\sigma^2\tau^2$, not τ^2. The reason for this choice is for simpler profiling of σ^2.

matern_space_time *Space-time Matern-covariance function*

Description

From a matrix of locations and times and a vector covariance parameters of the form (variance, spatial range, temporal range, smoothness, nugget), return the square matrix of all pairwise covariances.

Usage

```
matern_space_time(covparms, locstime)
```

Arguments

covparms	A vector giving positive-valued covariance parameters in the form (variance, spatial range, temporal range, smoothness, nugget)
locstime	A matrix with n rows and d+1 columns. The first d columns give a location in R^d, and the last column gives a time. Each row corresponds to a space-time location.

Value

A matrix with n rows and n columns, with the i,j entry containing the covariance between observations at locstime[i,] and locstime[j,].

---

matern_sphere *Isotropic Matern covariance function on sphere*

Description

From a matrix of longitudes and latitudes and a vector covariance parameters of the form (variance, range, smoothness, nugget), return the square matrix of all pairwise covariances.

Usage

```
matern_sphere(covparms, lonlat)
```

Arguments

covparms	A vector giving positive-valued covariance parameters in the form (variance, range, smoothness, nugget)
lonlat	A matrix with n rows and one column with longitudes in (-180,180) and one column of latitudes in (-90,90). Each row of locs describes a point on the sphere.

Value

A matrix with n rows and n columns, with the i,j entry containing the covariance between observations at lonlat[i,] and lonlat[j,].

Matern on Sphere Domain

The function first calculates the (x,y,z) 3D coordinates, and then inputs the resulting locations into maternIsotropic. This means that we construct covariances on the sphere by embedding the sphere in a 3D space. There has been some concern expressed in the literature that such embeddings may produce distortions. The source and nature of such distortions has never been articulated, and to date, no such distortions have been documented. Guinness and Fuentes (2016) argue that 3D embeddings produce reasonable models for data on spheres.

`matern_sphere_time` *Isotropic Matern covariance function on sphere-time*

Description

From a matrix of longitudes, latitudes, and times and a vector covariance parameters of the form (variance, spatial range, temporal range, smoothness, nugget), return the square matrix of all pairwise covariances.

Usage

```
matern_sphere_time(covparms, lonlattime)
```

Arguments

`covparms`	A vector giving positive-valued covariance parameters in the form (variance, spatial range, temporal range, smoothness, nugget)
`lonlattime`	A matrix with `n` rows and one column with longitudes in (-180,180), one column of latitudes in (-90,90), and one column of times. Each row of locs describes a point on the sphere-time domain.

Value

A matrix with `n` rows and `n` columns, with the i,j entry containing the covariance between observations at `lonlattime[i,]` and `lonlattime[j,]`.

Matern on Sphere-Time Domain

The function first calculates the (x,y,z) 3D spatial coordinates, and scales the spatial coordinates by the spatial range and the temporal coordinates by the temporal range. Then the scaled coordinates are input into `maternIsotropic`. This means that we construct covariances on the sphere-time by embedding the sphere-time domain in a 4D space, with a different range parameter for the three spatial dimensions versus the one temporal dimension. There has been some concern expressed in the literature that embedding points on the sphere into a 3D domain may cause distortions. The source and nature of such distortions has never been articulated, and to date, no such distortions have been documented. Guinness and Fuentes (2016) argue that 3D embeddings produce reasonable models for data on spheres.

order_coordinate *Sorted coordinate ordering*

Description

Return the ordering of locations sorted along one of the coordinates or the sum of multiple coordinates

Usage

```
order_coordinate(locs, coordinate)
```

Arguments

locs	A matrix of locations. Each row of locs contains a location, which can be a point in Euclidean space R^d, a point in space-time R^d x T, a longitude and latitude (in degrees) giving a point on the sphere, or a longitude, latitude, and time giving a point in the sphere-time domain.
coordinate	integer or vector of integers in 1,...,d. If a single integer, coordinates are ordered along that coordinate. If multiple integers, coordinates are ordered according to the sum of specified coordinate values. For example, when d=2, coordinate = c(1,2) orders from bottom left to top right.

Value

A vector of indices giving the ordering, i.e. the first element of this vector is the index of the first location.

Examples

```
n <- 100             # Number of locations
d <- 2               # dimension of domain
locs <- matrix( runif(n*d), n, d )
ord1 <- order_coordinate(locs, 1 )
ord12 <- order_coordinate(locs, c(1,2) )
```

order_dist_to_point *Distance to specified point ordering*

Description

Return the ordering of locations increasing in their distance to some specified location

Usage

```
order_dist_to_point(locs, loc0, lonlat = FALSE)
```

Arguments

locs	A matrix of locations. Each row of locs contains a location, which can be a point in Euclidean space R^d, a point in space-time R^d x T, a longitude and latitude (in degrees) giving a point on the sphere, or a longitude, latitude, and time giving a point in the sphere-time domain.
loc0	A vector containing a single location in R^d.
lonlat	TRUE/FALSE whether locations are longitudes and latitudes.

Value

A vector of indices giving the ordering, i.e. the first element of this vector is the index of the location nearest to loc0.

Examples

```
n <- 100             # Number of locations
d <- 2               # dimension of domain
locs <- matrix( runif(n*d), n, d )
loc0 <- c(1/2,1/2)
ord <- order_dist_to_point(locs,loc0)
```

---

order_maxmin *Maximum minimum distance ordering*

Description

Return the indices of an approximation to the maximum minimum distance ordering. A point in the center is chosen first, and then each successive point is chosen to maximize the minimum distance to previously selected points

Usage

```
order_maxmin(locs, lonlat = FALSE, space_time = FALSE, st_scale = NULL)
```

Arguments

locs	A matrix of locations. Each row of locs contains a location, which can be a point in Euclidean space R^d, a point in space-time R^d x T, a longitude and latitude (in degrees) giving a point on the sphere, or a longitude, latitude, and time giving a point in the sphere-time domain.
lonlat	TRUE/FALSE whether locations are longitudes and latitudes.
space_time	TRUE if locations are euclidean space-time locations, FALSE otherwise. If set to TRUE, temporal dimension is ignored.
st_scale	two-vector giving the amount by which the spatial and temporal coordinates are scaled. If NULL, the function uses the locations to automatically select a scaling. If set to FALSE, temporal dimension treated as another spatial dimension (not recommended).

Value

A vector of indices giving the ordering, i.e. the first element of this vector is the index of the first location.

Examples

```
# planar coordinates
nvec <- c(50,50)
locs <- as.matrix( expand.grid( 1:nvec[1]/nvec[1], 1:nvec[2]/nvec[2] ) )
ord <- order_maxmin(locs)
par(mfrow=c(1,3))
plot( locs[ord[1:100],1], locs[ord[1:100],2], xlim = c(0,1), ylim = c(0,1) )
plot( locs[ord[1:300],1], locs[ord[1:300],2], xlim = c(0,1), ylim = c(0,1) )
plot( locs[ord[1:900],1], locs[ord[1:900],2], xlim = c(0,1), ylim = c(0,1) )

# longitude/latitude coordinates (sphere)
latvals <- seq(-80, 80, length.out = 40 )
lonvals <- seq( 0, 360, length.out = 81 )[1:80]
locs <- as.matrix( expand.grid( lonvals, latvals ) )
ord <- order_maxmin(locs, lonlat = TRUE)
par(mfrow=c(1,3))
plot( locs[ord[1:100],1], locs[ord[1:100],2], xlim = c(0,360), ylim = c(-90,90) )
plot( locs[ord[1:300],1], locs[ord[1:300],2], xlim = c(0,360), ylim = c(-90,90) )
plot( locs[ord[1:900],1], locs[ord[1:900],2], xlim = c(0,360), ylim = c(-90,90) )
```

---

order_middleout *Middle-out ordering*

Description

Return the ordering of locations increasing in their distance to the average location

Usage

```
order_middleout(locs, lonlat = FALSE)
```

Arguments

locs	A matrix of locations. Each row of locs contains a location, which can be a point in Euclidean space R^d, a point in space-time R^d x T, a longitude and latitude (in degrees) giving a point on the sphere, or a longitude, latitude, and time giving a point in the sphere-time domain.
lonlat	TRUE/FALSE whether locations are longitudes and latitudes.

Value

A vector of indices giving the ordering, i.e. the first element of this vector is the index of the location nearest the center.

Examples

```
n <- 100             # Number of locations
d <- 2               # dimension of domain
locs <- matrix( runif(n*d), n, d )
ord <- order_middleout(locs)
```

`predictions` *Compute Gaussian process predictions using Vecchia's approximations*

Description

With the prediction locations ordered after the observation locations, an approximation for the inverse Cholesky of the covariance matrix is computed, and standard formulas are applied to obtain the conditional expectation.

Usage

```
predictions(covparms, covfun_name = "matern_isotropic", y_obs, locs_obs,
  locs_pred, X_obs, X_pred, beta, m = 60, reorder = TRUE)
```

Arguments

`covparms`	Covariance parameters
`covfun_name`	Name of covariance function
`y_obs`	Observations associated with locs_obs
`locs_obs`	observation locations
`locs_pred`	prediction locations
`X_obs`	Design matrix for observations
`X_pred`	Design matrix for predictions
`beta`	Linear mean parameters
`m`	Number of nearest neighbors to use
`reorder`	TRUE/FALSE for whether reordering should be done. This should generally be kept at TRUE, unless testing out the effect of reordering.

proflik_mean *Profile likelihood (profiling out mean only)*

Description

The profile likelihood is the maximum likelihood over a subset of the parameters, given specified values of the remaining parameters. In Gaussian process models, we can usually profile out linear mean parameters and an overall variance (scale) parameter.

Usage

```
proflik_mean(parms, covfun_name = "matern_isotropic", y, X, locs, NNarray,
  return_parms = FALSE)
```

Arguments

parms	All parameters. The specific meaning of each parameter depends on covfun_name.
covfun_name	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
y	vector of response values
X	design matrix, each column of X is a single covariate
locs	matrix of locations. Row i of locs specifies the location of element i of y, and so the length of y should equal the number of rows of locs.
NNarray	A matrix of indices, usually the output from find_ordered_nn. Row i contains the indices of the observations that observation i conditions on. By convention, the first element of row i is i.
return_parms	flag for whether the function should return the loglikelihood only (return_parms = FALSE) or to return both the loglikelihood and all of the parameter values, including mean vector and variance parameter (return_parms = TRUE). Usually, we do the optimization using return_parms = FALSE and then collect the parameter estimates with another call with return_parms = TRUE.

Details

It is important that the ordering of y and locs correspond to the ordering in NNarray. See example below.

Value

Either the loglikelihood only (if return_parms = FALSE) or a list containing the loglikelihood, parameter values, and covariance matrix for linear mean parameters (if return_parms = TRUE).

Examples

```
n1 <- 50
n2 <- 50             # size of grid of locations
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
n <- nrow(locs)
covparms = c(3,0.1,1,0)    # variance, range, smoothness, nugget
X = as.matrix( rep(1,n) )  # design matrix

# simulated response
y <- 2*X[,1] + fast_Gp_sim(covparms, "matern_isotropic", locs, m = 30)

ord <- order_maxmin(locs)         # ordering of locations
yord <- y[ord]                    # reordered response
Xord <- as.matrix( X[ord,] )      # reordered design matrix
locsord <- locs[ord,]             # reordered locations
NNarray <- find_ordered_nn(locsord, m = 30)     # nearest neighbor indices

# loglikelihood at true values of parameters
vecchia_loglik( covparms, "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNarray )
# profile out mean only (likelihood larger than vecchia_loglik)
proflik_mean( covparms[1:4], "matern_isotropic",
    yord, Xord, locsord, NNarray, return_parms = FALSE)
# profile out variance (likelihood larger than vecchia_loglik)
proflik_variance( covparms[2:4], "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNarray, return_parms = FALSE)
# profile out mean and variance (likelihood largest)
proflik_mean_variance( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNarray, return_parms = FALSE)
# get all parameter values
proflik_mean_variance( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNarray, return_parms = TRUE)
```

`proflik_mean_grouped` *Grouped Version of Profile Likelihood (profiling out mean only)*

Description

The profile likelihood is the maximum likelihood over a subset of the parameters, given specified values of the remaining parameters. In Gaussian process models, we can usually profile out linear mean parameters and an overall variance (scale) parameter.

Usage

```
proflik_mean_grouped(parms, covfun_name = "matern_isotropic", y, X, locs,
  NNlist, return_parms = FALSE)
```

Arguments

parms	All parameters. The specific meaning of each parameter depends on covfun_name.
covfun_name	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
y	vector of response values
X	design matrix, each column of X is a single covariate
locs	matrix of locations. Row i of locs specifies the location of element i of y, and so the length of y should equal the number of rows of locs.
NNlist	List object for grouped version of Vecchia's likelihood. Usually the result of group_obs(NNarray).
return_parms	flag for whether the function should return the loglikelihood only (return_parms = FALSE) or to return both the loglikelihood and all of the parameter values, including mean vector and variance parameter (return_parms = TRUE). Usually, we do the optimization using return_parms = FALSE and then collect the parameter estimates with another call with return_parms = TRUE.

Details

It is important that the ordering of y and locs correspond to the ordering in NNarray. See example below.

Value

Either the loglikelihood only (if return_parms = FALSE) or a list containing the loglikelihood, parameter values, and covariance matrix for linear mean parameters (if return_parms = TRUE).

Examples

```
n1 <- 50
n2 <- 50             # size of grid of locations
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
n <- nrow(locs)
covparms = c(3,0.1,1,0)    # variance, range, smoothness, nugget
X = as.matrix( rep(1,n) )  # design matrix

# simulated response
y <- 2*X[,1] + fast_Gp_sim(covparms, "matern_isotropic", locs, m = 30)

ord <- order_maxmin(locs)         # ordering of locations
yord <- y[ord]                    # reordered response
Xord <- as.matrix( X[ord,] )      # reordered design matrix
locsord <- locs[ord,]             # reordered locations
NNarray <- find_ordered_nn(locsord, m = 30)     # nearest neighbor indices
NNlist <- group_obs(NNarray)
```

```
# loglikelihood at true values of parameters
vecchia_loglik_grouped( covparms, "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNlist )
# profile out mean only (likelihood larger than vecchia_loglik)
proflik_mean_grouped( covparms[1:4], "matern_isotropic",
    yord, Xord, locsord, NNlist, return_parms = FALSE)
# profile out variance (likelihood larger than vecchia_loglik)
proflik_variance_grouped( covparms[2:4], "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNlist, return_parms = FALSE)
# profile out mean and variance (likelihood largest)
proflik_mean_variance_grouped( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNlist, return_parms = FALSE)
# get all parameter values
proflik_mean_variance_grouped( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNlist, return_parms = TRUE)
```

`proflik_mean_variance` *Profile likelihood (profiling out mean and variance)*

Description

The profile likelihood is the maximum likelihood over a subset of the parameters, given specified values of the remaining parameters. In Gaussian process models, we can usually profile out linear mean parameters and an overall variance (scale) parameter.

Usage

```
proflik_mean_variance(subparms, covfun_name = "matern_isotropic", y, X, locs,
  NNarray, return_parms = FALSE)
```

Arguments

`subparms`	All parameters except for variance parameter. The specific meaning of each parameter depends on `covfun_name`.
`covfun_name`	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
`y`	vector of response values
`X`	design matrix, each column of X is a single covariate
`locs`	matrix of locations. Row `i` of locs specifies the location of element `i` of `y`, and so the length of `y` should equal the number of rows of `locs`.

NNarray	A matrix of indices, usually the output from find_ordered_nn. Row i contains the indices of the observations that observation i conditions on. By convention, the first element of row i is i.
return_parms	flag for whether the function should return the loglikelihood only (return_parms = FALSE) or to return both the loglikelihood and all of the parameter values, including mean vector and variance parameter (return_parms = TRUE). Usually, we do the optimization using return_parms = FALSE and then collect the parameter estimates with another call with return_parms = TRUE.

Details

It is important that the ordering of y and locs correspond to the ordering in NNarray. See example below.

Value

Either the loglikelihood only (if return_parms = FALSE) or a list containing the loglikelihood, parameter values, and covariance matrix for linear mean parameters (if return_parms = TRUE).

Examples

```
n1 <- 50
n2 <- 50             # size of grid of locations
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
n <- nrow(locs)
covparms = c(3,0.1,1,0)    # variance, range, smoothness, nugget
X = as.matrix( rep(1,n) )  # design matrix

# simulated response
y <- 2*X[,1] + fast_Gp_sim(covparms, "matern_isotropic", locs, m = 30)

ord <- order_maxmin(locs)         # ordering of locations
yord <- y[ord]                    # reordered response
Xord <- as.matrix( X[ord,] )      # reordered design matrix
locsord <- locs[ord,]             # reordered locations
NNarray <- find_ordered_nn(locsord, m = 30)     # nearest neighbor indices

# loglikelihood at true values of parameters
vecchia_loglik( covparms, "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNarray )
# profile out mean only (likelihood larger than vecchia_loglik)
proflik_mean( covparms[1:4], "matern_isotropic",
    yord, Xord, locsord, NNarray, return_parms = FALSE)
# profile out variance (likelihood larger than vecchia_loglik)
proflik_variance( covparms[2:4], "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNarray, return_parms = FALSE)
# profile out mean and variance (likelihood largest)
proflik_mean_variance( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNarray, return_parms = FALSE)
# get all parameter values
proflik_mean_variance( covparms[2:4], "matern_isotropic",
```

```
yord, Xord, locsord, NNarray, return_parms = TRUE)
```

`proflik_mean_variance_grouped` *Grouped Version of Profile Likelihood (profiling out mean and variance)*

Description

The profile likelihood is the maximum likelihood over a subset of the parameters, given specified values of the remaining parameters. In Gaussian process models, we can usually profile out linear mean parameters and an overall variance (scale) parameter.

Usage

```
proflik_mean_variance_grouped(subparms, covfun_name = "matern_isotropic", y,
  X, locs, NNlist, return_parms = FALSE)
```

Arguments

`subparms`	All parameters except for variance parameter. The specific meaning of each parameter depends on `covfun_name`.
`covfun_name`	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
`y`	vector of response values
`X`	design matrix, each column of X is a single covariate
`locs`	matrix of locations. Row `i` of locs specifies the location of element `i` of `y`, and so the length of `y` should equal the number of rows of `locs`.
`NNlist`	List object for grouped version of Vecchia's likelihood. Usually the result of `group_obs(NNarray)`.
`return_parms`	flag for whether the function should return the loglikelihood only (`return_parms = FALSE`) or to return both the loglikelihood and all of the parameter values, including mean vector and variance parameter (`return_parms = TRUE`). Usually, we do the optimization using `return_parms = FALSE` and then collect the parameter estimates with another call with `return_parms = TRUE`.

Details

It is important that the ordering of `y` and `locs` correspond to the ordering in `NNarray`. See example below.

Value

Either the loglikelihood only (if `return_parms = FALSE`) or a list containing the loglikelihood, parameter values, and covariance matrix for linear mean parameters (if `return_parms = TRUE`).

Examples

```
n1 <- 50
n2 <- 50             # size of grid of locations
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
n <- nrow(locs)
covparms = c(3,0.1,1,0)    # variance, range, smoothness, nugget
X = as.matrix( rep(1,n) )  # design matrix

# simulated response
y <- 2*X[,1] + fast_Gp_sim(covparms, "matern_isotropic", locs, m = 30)

ord <- order_maxmin(locs)         # ordering of locations
yord <- y[ord]                    # reordered response
Xord <- as.matrix( X[ord,] )      # reordered design matrix
locsord <- locs[ord,]             # reordered locations
NNarray <- find_ordered_nn(locsord, m = 30)     # nearest neighbor indices
NNlist <- group_obs(NNarray)

# loglikelihood at true values of parameters
vecchia_loglik_grouped( covparms, "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNlist )
# profile out mean only (likelihood larger than vecchia_loglik)
proflik_mean_grouped( covparms[1:4], "matern_isotropic",
    yord, Xord, locsord, NNlist, return_parms = FALSE)
# profile out variance (likelihood larger than vecchia_loglik)
proflik_variance_grouped( covparms[2:4], "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNlist, return_parms = FALSE)
# profile out mean and variance (likelihood largest)
proflik_mean_variance_grouped( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNlist, return_parms = FALSE)
# get all parameter values
proflik_mean_variance_grouped( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNlist, return_parms = TRUE)
```

---

proflik_variance *Profile likelihood (profiling out variance in mean-zero model)*

Description

The profile likelihood is the maximum likelihood over a subset of the parameters, given specified values of the remaining parameters. In Gaussian process models, we can usually profile out linear mean parameters and an overall variance (scale) parameter.

Usage

```
proflik_variance(subparms, covfun_name = "matern_isotropic", y, locs, NNarray,
  return_parms = FALSE)
```

Arguments

subparms	All parameters except for variance parameter. The specific meaning of each parameter depends on covfun_name.
covfun_name	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
y	vector of response values
locs	matrix of locations. Row i of locs specifies the location of element i of y, and so the length of y should equal the number of rows of locs.
NNarray	A matrix of indices, usually the output from find_ordered_nn. Row i contains the indices of the observations that observation i conditions on. By convention, the first element of row i is i.
return_parms	flag for whether the function should return the loglikelihood only (return_parms = FALSE) or to return both the loglikelihood and all of the parameter values, including mean vector and variance parameter (return_parms = TRUE). Usually, we do the optimization using return_parms = FALSE and then collect the parameter estimates with another call with return_parms = TRUE.

Details

It is important that the ordering of y and locs correspond to the ordering in NNarray. See example below.

Value

Either the loglikelihood only (if return_parms = FALSE) or a list containing the loglikelihood, parameter values, and covariance matrix for linear mean parameters (if return_parms = TRUE).

Examples

```
n1 <- 50
n2 <- 50             # size of grid of locations
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
n <- nrow(locs)
covparms = c(3,0.1,1,0)    # variance, range, smoothness, nugget
X = as.matrix( rep(1,n) )  # design matrix

# simulated response
y <- 2*X[,1] + fast_Gp_sim(covparms, "matern_isotropic", locs, m = 30)
```

```
ord <- order_maxmin(locs)         # ordering of locations
yord <- y[ord]                    # reordered response
Xord <- as.matrix( X[ord,] )      # reordered design matrix
locsord <- locs[ord,]             # reordered locations
NNarray <- find_ordered_nn(locsord, m = 30)     # nearest neighbor indices

# loglikelihood at true values of parameters
vecchia_loglik( covparms, "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNarray )
# profile out mean only (likelihood larger than vecchia_loglik)
proflik_mean( covparms[1:4], "matern_isotropic",
    yord, Xord, locsord, NNarray, return_parms = FALSE)
# profile out variance (likelihood larger than vecchia_loglik)
proflik_variance( covparms[2:4], "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNarray, return_parms = FALSE)
# profile out mean and variance (likelihood largest)
proflik_mean_variance( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNarray, return_parms = FALSE)
# get all parameter values
proflik_mean_variance( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNarray, return_parms = TRUE)
```

`proflik_variance_grouped` *Grouped Version of Profile Likelihood (profiling out variance only)*

Description

The profile likelihood is the maximum likelihood over a subset of the parameters, given specified values of the remaining parameters. In Gaussian process models, we can usually profile out linear mean parameters and an overall variance (scale) parameter.

Usage

```
proflik_variance_grouped(subparms, covfun_name = "matern_isotropic", y, locs,
  NNlist, return_parms = FALSE)
```

Arguments

`subparms`	All parameters except for variance parameter. The specific meaning of each parameter depends on `covfun_name`.
`covfun_name`	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.

y	vector of response values
locs	matrix of locations. Row i of locs specifies the location of element i of y, and so the length of y should equal the number of rows of locs.
NNlist	List object for grouped version of Vecchia's likelihood. Usually the result of group_obs(NNarray).
return_parms	flag for whether the function should return the loglikelihood only (return_parms = FALSE) or to return both the loglikelihood and all of the parameter values, including mean vector and variance parameter (return_parms = TRUE). Usually, we do the optimization using return_parms = FALSE and then collect the parameter estimates with another call with return_parms = TRUE.

Details

It is important that the ordering of y and locs correspond to the ordering in NNarray. See example below.

Value

Either the loglikelihood only (if return_parms = FALSE) or a list containing the loglikelihood, parameter values, and covariance matrix for linear mean parameters (if return_parms = TRUE).

Examples

```
n1 <- 50
n2 <- 50             # size of grid of locations
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
n <- nrow(locs)
covparms = c(3,0.1,1,0)    # variance, range, smoothness, nugget
X = as.matrix( rep(1,n) )  # design matrix

# simulated response
y <- 2*X[,1] + fast_Gp_sim(covparms, "matern_isotropic", locs, m = 30)

ord <- order_maxmin(locs)         # ordering of locations
yord <- y[ord]                    # reordered response
Xord <- as.matrix( X[ord,] )      # reordered design matrix
locsord <- locs[ord,]             # reordered locations
NNarray <- find_ordered_nn(locsord, m = 30)     # nearest neighbor indices
NNlist <- group_obs(NNarray)

# loglikelihood at true values of parameters
vecchia_loglik_grouped( covparms, "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNlist )
# profile out mean only (likelihood larger than vecchia_loglik)
proflik_mean_grouped( covparms[1:4], "matern_isotropic",
    yord, Xord, locsord, NNlist, return_parms = FALSE)
# profile out variance (likelihood larger than vecchia_loglik)
proflik_variance_grouped( covparms[2:4], "matern_isotropic",
    yord - 2*Xord[,1], locsord, NNlist, return_parms = FALSE)
# profile out mean and variance (likelihood largest)
proflik_mean_variance_grouped( covparms[2:4], "matern_isotropic",
```

```
    yord, Xord, locsord, NNlist, return_parms = FALSE)
# get all parameter values
proflik_mean_variance_grouped( covparms[2:4], "matern_isotropic",
    yord, Xord, locsord, NNlist, return_parms = TRUE)
```

---

vecchia_Linv *Inverse Cholesky factor implied by Vecchia's approximation*

Description

This function returns the entries of the sparse approximation to the Cholesky factor implied by Vecchia's (1988) approximation to the Gaussian loglikelihood. The approximation modifies the ordered conditional specification of the joint density; rather than each term in the product conditioning on all previous observations, each term conditions on a small subset of previous observations.

Usage

```
vecchia_Linv(covparms, covfun_name, locs, NNarray)
```

Arguments

covparms	A vector of covariance parameters appropriate for the specified covariance function
covfun_name	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
locs	matrix of locations. Row i of locs specifies the location of element i of y, and so the length of y should equal the number of rows of locs.
NNarray	A matrix of indices, usually the output from find_ordered_nn. Row i contains the indices of the observations that observation i conditions on. By convention, the first element of row i is i.

Value

the Gaussian loglikelihood

Examples

```
n1 <- 40
n2 <- 40
n <- n1*n2
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
```

```
covparms <- c(2, 0.2, 0.75, 0)
ord <- order_maxmin(locs)
NNarray <- find_ordered_nn(locs,20)
Linv <- vecchia_Linv( covparms, "matern_isotropic", locs, NNarray )
```

`vecchia_Linv_grouped` *Inverse Cholesky factor implied by Vecchia's approximation*

Description

This function returns the entries of the sparse approximation to the Cholesky factor implied by Vecchia's (1988) approximation to the Gaussian loglikelihood. The approximation modifies the ordered conditional specification of the joint density; rather than each term in the product conditioning on all previous observations, each term conditions on a small subset of previous observations.

Usage

```
vecchia_Linv_grouped(covparms, covfun_name, locs, NNlist)
```

Arguments

`covparms`	A vector of covariance parameters appropriate for the specified covariance function
`covfun_name`	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
`locs`	matrix of locations. Row `i` of locs specifies the location of element `i` of `y`, and so the length of `y` should equal the number of rows of `locs`.
`NNlist`	A list with grouped neighbor information. Usually the output from `group_obs(NNarray)`.

Value

the Gaussian loglikelihood

Examples

```
n1 <- 40
n2 <- 40
n <- n1*n2
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
covparms <- c(2, 0.2, 0.75, 0)
ord <- order_maxmin(locs)
NNarray <- find_ordered_nn(locs,20)
Linv <- vecchia_Linv( covparms, "matern_isotropic", locs, NNarray )
```

`vecchia_loglik` *Vecchia's approximation to the Gaussian loglikelihood*

Description

This function returns Vecchia's (1988) approximation to the Gaussian loglikelihood. The approximation modifies the ordered conditional specification of the joint density; rather than each term in the product conditioning on all previous observations, each term conditions on a small subset of previous observations.

Usage

```
vecchia_loglik(covparms, covfun_name, y, locs, NNarray)
```

Arguments

`covparms`	A vector of covariance parameters appropriate for the specified covariance function
`covfun_name`	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
`y`	vector of response values
`locs`	matrix of locations. Row `i` of locs specifies the location of element `i` of `y`, and so the length of `y` should equal the number of rows of `locs`.
`NNarray`	A matrix of indices, usually the output from `find_ordered_nn`. Row `i` contains the indices of the observations that observation `i` conditions on. By convention, the first element of row `i` is `i`.

Value

the Gaussian loglikelihood

Examples

```
n1 <- 40
n2 <- 40
n <- n1*n2
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
covparms <- c(2, 0.2, 0.75, 0)
y <- fast_Gp_sim(covparms, "matern_isotropic", locs, 50 )
ord <- order_maxmin(locs)
NNarray <- find_ordered_nn(locs,20)
loglik <- vecchia_loglik( covparms, "matern_isotropic", y, locs, NNarray )
```

vecchia_loglik_grouped *Grouped Vecchia's approximation to the Gaussian loglikelihood*

Description

This function returns the grouped version (Guinness, 2018) of Vecchia's (1988) approximation to the Gaussian loglikelihood. The approximation modifies the ordered conditional specification of the joint density; rather than each term in the product conditioning on all previous observations, each term conditions on a small subset of previous observations.

Usage

```
vecchia_loglik_grouped(covparms, covfun_name, y, locs, NNlist)
```

Arguments

covparms	A vector of covariance parameters appropriate for the specified covariance function
covfun_name	One of "matern_isotropic", "matern_space_time", "matern_sphere", or "matern_sphere_time". "matern_isotropic" and "matern_sphere" have four covariance parameters, (variance, range, smoothness, nugget), while "matern_space_time" and "matern_sphere_time" have five, (variance, spatial range, temporal range, smoothness, nugget). For more details, see the documentation for each of the covariance functions by typing, for example, ?matern_isotropic or ?matern_sphere_time.
y	vector of response values
locs	matrix of locations. Row i of locs specifies the location of element i of y, and so the length of y should equal the number of rows of locs.
NNlist	A list with grouped neighbor information. Usually the output from group_obs(NNarray).

Value

grouped version of Vecchia's approximation to the Gaussian loglikelihood

Examples

```
n1 <- 40
n2 <- 40
n <- n1*n2
locs <- as.matrix( expand.grid( (1:n1)/n1, (1:n2)/n2 ) )
covparms <- c(2, 0.2, 0.75, 0)
y <- fast_Gp_sim(covparms, "matern_isotropic", locs, 50 )
ord <- order_maxmin(locs)
NNarray <- find_ordered_nn(locs,20)
NNlist <- group_obs(NNarray)
loglik <- vecchia_loglik_grouped( covparms, "matern_isotropic", y, locs, NNlist )
```

Index