Package ‘GPvecchia’

April 14, 2020

Type Package

Title Scalable Gaussian-Process Approximations

Version 0.1.2

Date 2020-04-03

Maintainer Marcin Jurek <marcinjurek1988@gmail.com>

Author Matthias Katzfuss [aut],
  Marcin Jurek [aut, cre],
  Daniel Zilber [aut],
  Wenlong Gong [aut],
  Joe Guinness [ctl],
  Jingjie Zhang [ctl],
  Florian Schaefer [ctl]


License GPL (>= 2)

Imports Rcpp (>= 0.12.16), methods, stats, sparseinv, fields,
  Matrix(>= 1.2.14), parallel, GpGp, FNN

LinkingTo Rcpp, RcppArmadillo, BH

RoxygenNote 6.1.1

Suggests mvtnorm, knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-04-14 10:10:06 UTC
calculate_posterior_VL

Vecchia Laplace extension of GPVecchia for non-Gaussian data

Description

Vecchia Laplace extension of GPVecchia for non-Gaussian data

Usage

calculate_posterior_VL(z, vecchia.approx,
    likelihood_model = c("gaussian", "logistic", "poisson", "gamma",
        "beta", "gamma_alt"), covparms, covmodel = "matern",
    likparms = list(alpha = 2, sigma = sqrt(0.1)), max.iter = 50,
    convg = 1e-06, return_all = FALSE, y_init = NA,
    prior_mean = rep(0, length(z)), verbose = FALSE)
createU

Arguments

z an array of real numbers representing observations
vecchia.approx a vecchia object as generated by vecchia_specify() likelihood_model text describing likelihood model to be used for observations. Can be "gaussian", "logistic", "poisson", "gamma", or "beta"
covparms covariance parameters as a vector
covmodel type of the model covariance or selected elements of the covariance matrix
likparms likelihood parameters for the likelihood_model, as a list. Default values are sqrt(1) for Gaussian noise and 2 for the alpha parameter for Gamma data.
max.iter maximum iterations to perform
convg convergence criteria. End iterations if the Newton step is this small
return_all Return additional posterior covariance terms, TRUE or FALSE
y_init Specify initial guess for posterior mode
prior_mean specify the prior latent mean
verbose if TRUE messages about the posterior estimation will be displayed

Value

multivariate normal posterior parameters calculated by the Vecchia-Laplace approximation

Examples

z=rnorm(10); locs=matrix(1:10,ncol=1); vecchia.approx=vecchia Specify(locs,m=5)
calculate_posterior_VL(z,vecchia.approx,"gaussian",covparms=c(1,2,.5))

createU  
create the sparse triangular U matrix for specific parameters

Description

create the sparse triangular U matrix for specific parameters

Usage

createU(vecchia.approx, covparms, nuggets, covmodel = "matern")

Arguments

vecchia.approx object returned by vecchia_specify
covparms vector of covariance parameters
nuggets nugget variances – if a scalar is provided, variance is assumed constant
covmodel covariance model. currently implemented:
Value

list containing the sparse upper triangular U, plus additional objects required for other functions

Examples

z=rnorm(9); locs=matrix(1:9,ncol=1); vecchia.approx=vecchia_specify(locs,m=5)
U.obj=createU(vecchia.approx,covparms=c(1,2,.5),nuggets=.2)

getMatCov

extract the required elements from the covariance matrix

Description

This function takes the entire covariance matrix and creates a matrix of covariances based on the vecchia approximation object

Usage

getMatCov(V, covariances, factor = FALSE)

Arguments

V the object returned by vecchia_specify
covariances The full covariance matrix or a covariance function
factor True if we are passing a factor of a matrix

Value

matrix of size n x (m+1) with only those elements that are used by the incomplete Cholesky decomposition

getMatCovFromFactorCpp

Calculate the covariance values required by HV for matrix factors passed as sparse matrices

Description

Calculate the covariance values required by HV for matrix factors passed as sparse matrices

Usage

getMatCovFromFactorCpp(F, revNNarray)
GPvecchia

Arguments

- F: factor of a matrix in a sparse format
- revNNarray: array with the neighbourhood structure

Value

- matrix with covariance values

GPvecchia: fast, scalable Gaussian process approximations

Description

The package can be used for parameter inference and prediction for Gaussian and non-Gaussian spatial data using many popular GP approximation methods.

ic0

Incomplete Cholesky decomposition of a sparse matrix passed in the compressed sparse row format

Description

Incomplete Cholesky decomposition of a sparse matrix passed in the compressed sparse row format

Usage

ic0(ptrs, inds, vals)

Arguments

- ptrs: pointers to the beginning of the row
- inds: indices of nonzero elements in a row
- vals: nonzero values

Value

- vector of the values of the incomplete Cholesky factor
MaternFun

*Calculate Matern covariance function*

**Description**

Calculate Matern covariance function

**Usage**

MaternFun(distmat, covparms)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>distmat</td>
<td>A matrix with distances between points</td>
</tr>
<tr>
<td>covparms</td>
<td>A vector with parameters (marg. variance, range, smoothness)</td>
</tr>
</tbody>
</table>

**Value**

A matrix with covariance values corresponding to the distance matrix

---

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**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>locs</td>
<td>A matrix of locations. Each row of locs contains a location, which can be a point in Euclidean space $\mathbb{R}^d$, a point in space-time $\mathbb{R}^d \times T$, 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.</td>
</tr>
<tr>
<td>coordinate</td>
<td>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.</td>
</tr>
</tbody>
</table>
**order_dist_to_point**

**Value**

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

**Examples**

```r
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) )
```

---

**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 $\mathbb{R}^d$, a point in space-time $\mathbb{R}^d \times \mathbb{T}$, 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 $\mathbb{R}^d$.
- `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**

```r
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_exact

Maximum minimum distance ordering

Description

Return the indices of an exact maximum-minimum distance ordering. The first point is chosen as the "center" point, minimizing L2 distance. Dimensions d=2 and d=3 handled separately, dimensions d=1 and d>3 handled similarly. Algorithm is exact and scales quasilinearly.

Usage

order_maxmin_exact(locs)

Arguments

locs Observation locations

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; locs <- cbind(runif(n),runif(n))
ord <- order_maxmin_exact(locs)

order_maxmin_exact_obs_pred

Maximum minimum distance ordering for prediction

Description

Return the indices of an exact maximum-minimum distance ordering. The first point is chosen as the "center" point, minimizing L2 distance. Dimensions d=2 and d=3 handled separately, dimensions d=1 and d>3 handled similarly. Algorithm is exact and scales quasilinearly.

Usage

order_maxmin_exact_obs_pred(locs, locs_pred)

Arguments

locs Observation locations
locs_pred Prediction locations
Value

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

Examples

```r
n=100; locs <- cbind(runif(n),runif(n))
locs_pred = cbind(runif(n), runif(n))
ord <- order_maxmin_exact_obs_pred(locs, locs_pred)
```

Description

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

Usage

```r
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 \( \mathbb{R}^d \), a point in space-time \( \mathbb{R}^d \times T \), 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

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

*Outside-in ordering*

**Description**

Return the ordering of locations decreasing in their distance to the average location. Reverses middleout.

**Usage**

```r
order_outsidein(locs, lonlat = FALSE)
```

**Arguments**

- `locs`: A matrix of locations. Each row of `locs` contains a location, which can be a point in Euclidean space $\mathbb{R}^d$, a point in space-time $\mathbb{R}^d \times T$, 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 farthest from the center.

**Examples**

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

---

**SelInv**

*selected inverse of a sparse matrix*

**Description**

selected inverse of a sparse matrix

**Usage**

```r
SelInv(cholmat)
```

**Arguments**

- `cholmat`: cholesky factor $L$ of a positive definite sparseMatrix $A$
Value

sparse inverse of A, with same sparsity pattern as L

Examples

A=Matrix::sparseMatrix(1:9,1:9,x=4); L=chol(A)
SelInv(L)

descriptions

compute covariance matrix from V.ord Do not run this function for large n or n.p!!!

Usage

V2covmat(preds)

Arguments

preds Object returned by vecchia_prediction()

Value

Covariance matrix at all locations in original order

Examples

z=rnorm(5)
locs=matrix(1:5,ncol=1)
vecchia_specify=function(z,locs,m=5,locs.pred=(1:5)+.5)
V2covmat(vecchia_prediction(vecchia.approx,covparms=c(1,2,.5),nuggets=.2))
vecchia_estimate

estimate mean and covariance parameters using Vecchia

Description

estimate mean and covariance parameters using Vecchia

Usage

vecchia_estimate(data, locs, X, m = 20, covmodel = "matern", theta.ini, output.level = 1, ...)

Arguments

data: data vector of length n
locs: n x d matrix of spatial locations
X: n x p matrix of trend covariates. default is vector of ones (constant trend). set to NULL if data are already detrended
m: number of neighbors for vecchia approximation. default is 20
covmodel: covariance model. default is Matern. see vecchia_likelihood for details.
theta.ini: initial values of covariance parameters. nugget variance must be last.
output.level: passed on to trace in the stats::optim function
...
additional input parameters for vecchia_specify

Value

object containing detrended data z, trend coefficients beta.hat, covariance parameters theta.hat, and other quantities necessary for prediction

Examples

n=10^2; locs=cbind(runif(n),runif(n))
covparms=c(1,1,.5); nuggets=rep(.1,n)
Sigma=exp(-fields::rdist(locs)/covparms[2]) + diag(nuggets)
z=as.numeric(t(chol(Sigma))%*%rnorm(n));
data=z+1
vecchia.est=vecchia_estimate(data,locs,theta.ini=c(covparms,nuggets[1]))
vecchia_laplace_likelihood

Wrapper for VL version of vecchia_likelihood

Description

Wrapper for VL version of vecchia_likelihood

Usage

vecchia_laplace_likelihood(z, vecchia.approx, likelihood_model, covparms,
likparms = list(alpha = 2, sigma = sqrt(0.1)), covmodel = "matern",
max.iter = 50, convg = 1e-05, return_all = FALSE, y_init = NA,
prior_mean = rep(0, length(z)), vecchia.approx.IW = NA)

Arguments

z an array of real numbers representing observations
vecchia.approx a vecchia object as generated by vecchia_specify()
likelihood_model text describing likelihood model to be used for observations
covparms covariance parameters as a vector
likparms likelihood parameters for the likelihood_model, as a list
covmodel describes the covariance model, "matern" by default
max.iter maximum iterations to perform
convg convergence criteria. End iterations if the Newton step is this small
return_all Return additional posterior covariance terms
y_init Specify initial guess for posterior mode
prior_mean specify the prior latent mean
vecchia.approx.IW an optional vecchia approximation object, can reduce computation if method is called repeatedly

Value

(multivariate normal) loglikelihood implied by the Vecchia approximation

Examples

z=rnorm(10); locs=matrix(1:10,ncol=1); vecchia.approx=vecchia_specify(locs,m=5)
vecchia_laplace_likelihood(z,vecchia.approx,"gaussian",covparms=c(1,2,.5))
vecchia_laplace_prediction

*Wrapper for VL version of vecchia_prediction*

**Description**

Wrapper for VL version of vecchia_prediction

**Usage**

```
vecchia_laplace_prediction(vl_posterior, vecchia.approx, covparms,
  pred.mean = 0, var.exact = FALSE, covmodel = "matern",
  return.values = "all")
```

**Arguments**

- `vl_posterior`: a posterior estimate object produced by `calculate_posterior_VL`
- `vecchia.approx`: a vecchia object as generated by `vecchia_specify()`
- `covparms`: covariance parameters as a vector
- `pred.mean`: provides the prior latent mean for the prediction locations
- `var.exact`: should prediction variances be computed exactly, or is a (faster) approximation acceptable
- `covmodel`: covariance model, 'matern' by default.
- `return.values`: either 'mean' only, 'meanvar', 'meanmat', or 'all'

**Value**

(multivariate normal) loglikelihood implied by the Vecchia approximation

**Examples**

```
z=rnorm(10); locs=matrix(1:10,ncol=1); vecchia.approx=vecchia_specify(locs,m=5)
vl_posterior = calculate_posterior_VL(z, vecchia.approx,"gaussian",covparms=c(1,2,.5))
locs.pred=matrix(1:10+.5,ncol=1)
vecchia.approx.pred = vecchia_specify(locs, m=5, locs.pred=locs.pred)
vecchia_laplace_prediction(vl_posterior, vecchia.approx.pred, covparms=c(1,2,.5))
```
vecchia_likelihood

evaluation of the likelihood

Description

evaluation of the likelihood

Usage

vecchia_likelihood(z, vecchia.approx, covparms, nuggets, covmodel = "matern")

Arguments

z the observed data
vecchia.approx a vecchia object as generated by vecchia_specify()
covparms covariance parameters as a vector
nuggets either a single (constant) nugget or a vector of nugget terms for the observations
covmodel covariance model, 'matern' by default

Value

(multivariate normal) loglikelihood implied by the Vecchia approximation

Examples

z=rnorm(5); locs=matrix(1:5,ncol=1); vecchia.approx=vecchia_specify(locs,m=3)
vecchia_likelihood(z,vecchia.approx,covparms=c(1,2,.5),nuggets=.2)

vecchia_lincomb

linear combination of predictions compute the distribution of a linear combination Hy

Description

linear combination of predictions compute the distribution of a linear combination Hy

Usage

vecchia_lincomb(H, U.obj, V.ord, cov.mat = FALSE)
vecchia_pred

**Arguments**

- **H**
  - sparse matrix with \(n\) all columns specifying the linear combination
- **U.obj**
  - \(U\) matrix is the full joint approximated cholesky matrix
- **V.ord**
  - ordered \(V\) matrix from vecchia_prediction() or U2V()
- **cov.mat**
  - logical TRUE or FALSE – should the entire covariance matrix be returned (only do if \(H\) has a small number of rows)

**Value**

Variance of linear combination of predictions.

**Examples**

```r
n=5; z=rnorm(n); locs=matrix(1:n,ncol=1); n.p=5
d < vecchia.approx = vecchia_specify(locs,m=3,locs.pred=locs+.5)
preds=vecchia_prediction(z,vecchia.approx,covparms=c(1,2,.5),nuggets=.2)
H=Matrix::sparseMatrix(i=rep(1,n.p),j=n+(1:n.p),x=1/n.p)
dvecchia_lincomb(H,vecchia.approx,preds$V.ord,cov.mat=TRUE)
```

---

**vecchia_pred**  

*make spatial predictions using Vecchia based on estimated parameters*

**Description**

make spatial predictions using Vecchia based on estimated parameters

**Usage**

```
vecchia_pred(vecchia.est, locs.pred, X.pred, m = 30, ...)
```

**Arguments**

- **vecchia.est**
  - object returned by `vecchia_estimate`
- **locs.pred**
  - \(n.p\) x \(d\) matrix of prediction locations
- **X.pred**
  - \(n.p\) x \(p\) matrix of trend covariates at prediction locations. does not need to be specified if constant or no trend was used in `vecchia_estimate`
- **m**
  - number of neighbors for vecchia approximation. default is 30.
- **...**
  - additional input parameters for `vecchia_specify`

**Value**

object containing prediction means `mean.pred` and variances `var.pred`
Examples

```r
n=10^2; locs=cbind(runif(n),runif(n))
covparms=c(1,.1,.5); nuggets=rep(.1,n)
Sigma=exp(-fields::rdist(locs)/covparms[2])+diag(nuggets)
z=as.numeric(t(chol(Sigma))%*%rnorm(n));
data=z+1
vecchia.est=vecchia_estimate(data,locs,theta.ini=c(covparms,nuggets[1]))
n.p=30^2; grid.oneside=seq(0,1,length=round(sqrt(n.p)))
locs.pred=as.matrix(expand.grid(grid.oneside,grid.oneside))
vecchia.pred=vecchia_pred(vecchia.est,locs.pred)
```

Description

Vecchia prediction

Usage

```r
vecchia_prediction(z, vecchia.approx, covparms, nuggets, var.exact,
covmodel = "matern", return.values = "all")
```

Arguments

- `z` : observed data
- `vecchia.approx` : a vecchia object as generated by vecchia_specify()
- `covparms` : covariance parameters as a vector
- `nuggets` : nugget
- `var.exact` : should prediction variances be computed exactly, or is a (faster) approximation acceptable
- `covmodel` : covariance model, 'matern' by default.
- `return.values` : either 'mean' only, 'meanvar', 'meanmat', or 'all'

Value

posterior mean and variances at observed and unobserved locations; V matrix

Examples

```r
z=rnorm(5); locs=matrix(1:5,ncol=1); vecchia.approx=vecchia_specify(locs,m=3,locs.pred=locs+.5)
vecchia_prediction(z,vecchia.approx,covparms=c(1,2,.5),nuggets=.2)
```
vecchia Specify

Specify a general vecchia approximation

Description

specify the vecchia approximation for later use in likelihood evaluation or prediction. This function does not depend on parameter values, and only has to be run once before repeated likelihood evaluations.

Usage

vecchia_specify(locs, m = -1, ordering, cond.yz, locs.pred, ordering.pred, pred.cond, conditioning, mra.options = NULL, verbose = FALSE)

Arguments

locs nxd matrix of observed locs
m Number of nearby points to condition on
ordering options are ‘coord’ or ‘maxmin’
locs.pred nxd matrix of locations at which to make predictions
ordering.pred options are ‘obspred’ or ‘general’
pred.cond prediction conditioning, options are ‘general’ or ‘independent’
conditioning conditioning on ‘NN’ (nearest neighbor) or ‘firstm’ (fixed set for low rank) or ‘mra’
mra.options Settings for number of levels and neighbors per level
verbose Provide more detail when using MRA calculations. Default is false.

Value

An object that specifies the vecchia approximation for later use in likelihood evaluation or prediction.

Examples

locs=matrix(1:5,ncol=1); vecchia_specify(locs,m=2)
Index

calculate_posterior_VL, 2
createU, 3
getMatCov, 4
getMatCovFromFactorCpp, 4
GPvecchia, 5
GPvecchia-package (GPvecchia), 5
ic0, 5
MaternFun, 6
order_coordinate, 6
order_dist_to_point, 7
order_maxmin_exact, 8
order_maxmin_exact_obs_pred, 8
order_middleout, 9
order_outsidein, 10
SelInv, 10
V2covmat, 11
vecchia_estimate, 12, 16
vecchia_laplace_likelihood, 13
vecchia_laplace_prediction, 14
vecchia_likelihood, 12, 15
vecchia_lincomb, 15
vecchia_pred, 16
vecchia_prediction, 17
vecchia_specify, 3, 12, 16, 18