Package ‘exageostatr’

December 28, 2022

SystemRequirements GNU Make, GNU CMake, GCC Compiler Suite (C and Fortran), nlopt (>= 2.4.2 http://ab-initio.mit.edu), lapack (https://github.com/xianyi/OpenBLAS/releases), lapacke (https://github.com/xianyi/OpenBLAS/releases), blas (https://github.com/xianyi/OpenBLAS/releases), cblas (https://github.com/xianyi/OpenBLAS/releases), hwloc (>=1.11.5 https://www.open-mpi.org), gsl (>= 2.4 https://ftp.gnu.org)

Version 1.2.0

Date 2022-03-14

Title R Package Demonstrates the R / C Language Interface for Exageostat

Author Sameh Abdulah <sameh.abdulah@kaust.edu.sa>

Maintainer Sameh Abdulah <sameh.abdulah@kaust.edu.sa>

Depends R (>= 3.5.0), assertthat (>= 0.2.1), MASS

Description An R-wrapper for ExaGeoStat: a parallel high performance unified framework for geostatistics on manycore systems. Its abbreviation stands for Exascale Geostatistics. The framework aims at optimizing the likelihood function for a given spatial data to provide an efficient way to predict missing observations. The framework targets many-core systems: clusters of CPUs and GPUs.

License GPL (>= 2)

URL https://www.github.com/ecrc/exageostatr

OS_type unix

RoxygenNote 7.2.2

NeedsCompilation yes

StagedInstall no

Encoding UTF-8

R topics documented:

arg_check_mle 2
arg_check_predict 3
bm 3
bm_comp 4
check_dmetric 5
check_kernel 5
check_theta 6
dst_mle 6
exact_mle 7
exact_mloe_mmom 8
exact_predict 9
exageostat_finalize 9
exageostat_init 10
fisher_general 10
mean_absolute_error 11
mean_squared_logarithmic_error 11
mean_square_error 12
plot.Krig 12
root_mean_squared_error 13
simulate_data_exact 13
simulate_obs_exact 14
splitting_data 15
tlr_mle 15

Index **17**

arg_check_mle *Check the MLE function args*

Description

Check the MLE function args

Usage

```
arg_check_mle(data, dmetric, optimization)
```

Arguments

`data`	A list of x vector (x-dim), y vector (y-dim), and z observation vector
`dmetric`	A string - distance metric - "euclidean" or "great_circle"
`optimization`	A list of opt lb values (clb), opt ub values (cub), tol, max_iters

Value

dmetric as integer

`arg_check_predict` *Check the prediction function args*

Description

Check the prediction function args

Usage

```
arg_check_predict(data_train, data_test, theta, dmetric)
```

Arguments

`data_train` A list of x vector (x-dim), y vector (y-dim), and z observation vector

`data_test` A list of x vector (x-dim) and y vector (y-dim)

`dmetric` A string - distance metric - "euclidean" or "great_circle"

`theta:` list of n parameters

Value

dmetric as integer

`bm` *Benchmark function to run a given FUN on a set of synthetic datasets generated by ExaGeoStatR*

Description

Benchmark function to run a given FUN on a set of synthetic datasets generated by ExaGeoStatR

Usage

```
bm(
  FUN,
  dmetric = "euclidean",
  n = 400,
  min_seed = 0,
  max_seed = 9,
  ts = 320,
  lts = 0,
  ncores = 4,
  ngpus = 0,
  pgrid = 1,
  qgrid = 1
)
```

Arguments

`FUN:` A predefined function to perform both modeling and prediction operations

`dmetric:` A string - distance metric - "euclidean" or "great_circle"

`n:` An integer - numer of spatial locations

`min_seed:` An integer - initial seed to generate the synthetic datasets

`max_seed:` An integer - last seed to generate the synthetic datasets

`ncores:` An integer - numer of CPU cores to use

`ngpus:` An integer - numer of GPUs to use

Value

a list of prediction errors (MSE_vec, RMSE_vec, MAE_vec, MSLE_vec)

`bm_comp` *Running KAUST 2021 competition using the Benchmark function Benchmark function to run a given FUN on KAUST 2021 competition datasets generated by ExaGeoStatR*

Description

Running KAUST 2021 competition using the Benchmark function Benchmark function to run a given FUN on KAUST 2021 competition datasets generated by ExaGeoStatR

Usage

```
bm_comp(FUN, Data_train_list, Data_predict_list)
```

Arguments

`FUN:` A predefined function to perform both modeling and prediction operations

`Data_train_list:`
A list of trainig data

`Data_predict_list:`
A list of predict data

Value

a list of prediction errors (MS, RMSE, MAE, MSLE) and the estimate of the parameters (sigma, beta, nu, nuggets)

`check_dmetric` *Check the distance metric input to be "euclidean" or "great_circle"*

Description

Check the distance metric input to be "euclidean" or "great_circle"

Usage

```
check_dmetric(dmetric)
```

Arguments

`dmetric:`	string

Value

dmetric as integer

`check_kernel` *Check the statistical kernel to be in ("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st")*

Description

Check the statistical kernel to be in ("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st")

Usage

```
check_kernel(kernel)
```

Arguments

`kernel:`	string

Value

kernel as integer

check_theta *Check the statistical parameter vector (theta)*

Description

Check the statistical parameter vector (theta)

Usage

```
check_theta(theta)
```

Arguments

`theta:`	list of n parameters

Value

N/A

dst_mle *Maximum Likelihood Evaluation (MLE) using Diagonal Super-tile (DST) method*

Description

Maximum Likelihood Evaluation (MLE) using Diagonal Super-tile (DST) method

Usage

```
dst_mle(
  data = list(x, y, z),
  kernel = c("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st"),
  dst_band = 2,
  dmetric = c("euclidean", "great_circle"),
  optimization = list(clb = c(0.001, 0.001, 0.001), cub = c(5, 5, 5), tol = 1e-04,
    max_iters = 100)
)
```

Arguments

`data`	A list of x vector (x-dim), y vector (y-dim), and z observation vector
`dst_band`	A number - Diagonal Super-Tile (DST) diagonal thick
`dmetric`	A string - distance metric - "euclidean" or "great_circle"
`optimization`	A list of opt lb (clb), opt ub (cub), tol, max_iters
`kernel:`	string - kernel ("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st")

Value

vector of three values (theta1, theta2, theta3)

Examples

```
seed <- 0 ## Initial seed to generate XY locs.
dmetric <- "euclidean" ## "euclidean" or "great_circle" distance.
n <- 900 ## The number of locations (n must be a square number, n=m^2).
dst_band <- 3 ## Number of used Diagonal Super Tile (DST).
kernel <- "ugsm-s"
theta <- c(1, 0.1, 0.5)                                     #Params vector.
exageostat_init(hardware = list(ncores = 4, ngpus = 0, ts = 320, lts = 0, pgrid = 1, qgrid = 1)) ## Initiate exageosta
data <- simulate_data_exact(kernel, theta, dmetric, n, seed) ## Generate Z observation vector
## Estimate MLE parameters (TLR approximation)
result <- dst_mle(data, kernel, dst_band, dmetric, optimization = list(clb = c(0.001, 0.001, 0.001), cub = c(5, 5, 5
print(result)
exageostat_finalize() ## Finalize exageostat instance
```

`exact_mle` *Maximum Likelihood Evaluation using exact method*

Description

Maximum Likelihood Evaluation using exact method

Usage

```
exact_mle(
  data = list(x, y, z),
  kernel = c("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st"),
  dmetric = c("euclidean", "great_circle"),
  optimization = list(clb = c(0.001, 0.001, 0.001), cub = c(5, 5, 5), tol = 1e-04,
    max_iters = 100)
)
```

Arguments

`data`	A list of x vector (x-dim), y vector (y-dim), and z observation vector
`dmetric`	A string - distance metric - "euclidean" or "great_circle"
`optimization`	A list of opt lb values (clb), opt ub values (cub), tol, max_iters

Value

vector of three values (theta1, theta2, theta3)

Examples

```
seed <- 0 ## Initial seed to generate XY locs.
sigma_sq <- 1 ## Initial variance.
beta <- 0.1 ## Initial range.
nu <- 0.5 ## Initial smoothness.
dmetric <- "euclidean" ## "euclidean" or "great_circle" distance.
n <- 144 ## The number of locations (n must be a square number, n=m^2).
exageostat_init(hardware = list(ncores = 2, ngpus = 0, ts = 320, lts = 0, pgrid = 1, qgrid = 1)) ## Initiate exageosta
data <- simulate_data_exact(sigma_sq, beta, nu, dmetric, n, seed) ## Generate Z observation vector
## Estimate MLE parameters (Exact)
result <- exact_mle(data, dmetric, optimization = list(clb = c(0.001, 0.001, 0.001), cub = c(5, 5, 5), tol = 1e-4, ma
print(result)
exageostat_finalize() ## Finalize exageostat instance
```

---

exact_mloe_mmom *Mean Misspecification of the Mean Square Error (MMOM) and Mean Loss of Efficiency (MLOE) using exact method*

Description

Mean Misspecification of the Mean Square Error (MMOM) and Mean Loss of Efficiency (MLOE) using exact method

Usage

```
exact_mloe_mmom(
  data = list(x_train, y_train, z_train, x_test, y_test),
  kernel = c("ugsm-s", "ugsmn-s", "ugnsm-s", "bgsfm-s", "bgsbm-s", "ugsm-s", "ugsm-st"),
  dmetric = c("euclidean", "great_circle"),
  est_theta,
  true_theta,
  computation = 0
)
```

Arguments

`dmetric`	string - distance metric - "euclidean" or "great_circle"
`data:`	list of training and testing vectors
`kernel:`	string - kernel ("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st")
`est_theta:`	list of n parameters (estimated theta)
`true_theta:`	list of n parameters (true theta)
`computation:`	integer - should be always dense

Value

vector of MLOE/MMOM values

`exact_predict` *Perform prediction on testing data using training data and pre-estimated theta vector*

Description

Perform prediction on testing data using training data and pre-estimated theta vector

Usage

```
exact_predict(
  data_train = list(x, y, z),
  data_test = list(x, y),
 kernel = c("ugsm-s", "ugsmn-s", "ugnsm-s", "bgsfm-s", "bgsbm-s", "ugsm-s", "ugsm-st"),
  dmetric = c("euclidean", "great_circle"),
  theta,
  computation = 0
)
```

Arguments

`dmetric`	string - distance metric - "euclidean" or "great_circle"
`data_train:`	list of training data
`data_test:`	list of testing data
`kernel:`	string - kernel ("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st")
`theta:`	list of n parameters (estimated theta)
`computation:`	integer - computation method

Value

list of predicted values

`exageostat_finalize` *Finalize the current instance of ExaGeoStatR*

Description

Finalize the current instance of ExaGeoStatR

Usage

```
exageostat_finalize()
```

Value

N/A

Examples

```
exageostat_finalize()
```

exageostat_init *Initial an instance of ExaGeoStatR*

Description

Initial an instance of ExaGeoStatR

Usage

```
exageostat_init(
  hardware = list(ncores = 2, ngpus = 0, ts = 320, lts = 0, pgrid = 1, qgrid = 1)
)
```

Arguments

`hardware`	A list of ncores, ngpus, tile size, pgrid, and qgrid

Value

N/A

Examples

```
exageostat_init(hardware = list(ncores = 2, ngpus = 0, ts = 320, lts = 0, pgrid = 1, qgrid = 1))
exageostat_init(hardware = list(ncores = 1, ngpus = 2, ts = 320, lts = 0, pgrid = 1, qgrid = 1))
exageostat_init(hardware = list(ncores = 26, ngpus = 0, ts = 320, lts = 0, pgrid = 3, qgrid = 4))
```

fisher_general *Compute the Fisher information matrix for a given data and theta vector*

Description

Compute the Fisher information matrix for a given data and theta vector

Usage

```
fisher_general(data = list(x, y), theta, dmetric)
```

Arguments

dmetric string - distance metric - "euclidean" or "great_circle"

data: list of data vectors, x, and y

theta: list of n parameters (estimated theta)

Value

list of fisher matrix elements

mean_absolute_error *Mean Absolute Error used as an assessment tool*

Description

Mean Absolute Error used as an assessment tool

Usage

```
mean_absolute_error(y, ypre)
```

Arguments

y<- c(5:20) - vector representing number of true values

ypre<- predict(lm(y ~ x)) - vector denoting values of number of y predicted values.

Value

MAE is the average of the absolute error

mean_squared_logarithmic_error *Mean Absolute Error used as an assessment tool*

Description

Mean Absolute Error used as an assessment tool

Usage

```
mean_squared_logarithmic_error(y, ypre)
```

Arguments

y<- c(5:20) - vector representing number of true values

ypre<- predict(lm(y ~ x)) - vector denoting values of number of y predicted values

Value

MSLE is measure of the ratio between the true and predicted values

mean_square_error *Mean Square Error used as an assessment tool*

Description

Mean Square Error used as an assessment tool

Usage

```
mean_square_error(y, ypre)
```

Arguments

`y<-`	c(5:20) - vector representing number of true values
`ypre<-`	predict(lm(y ~ x)) - vector denoting values of number of y predicted values.

Value

MSE is the is the average squares of the “errors”

plot.Krig *This function plots the the diagnostics and summaries of kriging one thing to note here is that x is Krig or sreg object. need to figure out x."plot.Krig" <- function(x, digits = 4, which = 1:4,*

Description

This function plots the the diagnostics and summaries of kriging one thing to note here is that x is Krig or sreg object. need to figure out x."plot.Krig" <- function(x, digits = 4, which = 1:4,

Usage

```
## S3 method for class 'Krig'
plot(x, digits = 4, which = 1:4, ...)
```

`root_mean_squared_error` *Root Mean Squared Error used as an assessment tool*

Description

Root Mean Squared Error used as an assessment tool

Usage

```
root_mean_squared_error(y, ypre)
```

Arguments

`y<-`	c(5:20) - vector representing number of true values
`ypre<-`	predict(lm(y ~ x)) - vector denoting values of number of y predicted values.

Value

RMSE is the square root of the mean of the square of all of the error

`simulate_data_exact` *Simulate Geospatial data (x, y, z)*

Description

Simulate Geospatial data (x, y, z)

Usage

```
simulate_data_exact(
 kernel = c("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st"),
  theta,
  dmetric = c("euclidean", "great_circle"),
  n,
  seed = 0
)
```

Arguments

`dmetric`	A string - distance metric - "euclidean" or "great_circle"
`n`	A number - data size
`seed`	A number - seed of random generation
`kernel:`	string - kernel ("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st")
`theta:`	list of n parameters (estimated theta)

Value

a list of of three vectors (x, y, z)

Examples

```
seed <- 0 ## Initial seed to generate XY locs.
kernel <- "ugsm-s"
theta <- c(1, 0.1, 0.5)                                        #Params vector.
dmetric <- "euclidean" ## "euclidean" or "great_circle" distance.
n <- 1600 ## The number of locations (n must be a square number, n=m^2).
exageostat_init(hardware = list(ncores = 2, ngpus = 0, ts = 320, lts = 0, pgrid = 1, qgrid = 1)) ## Initiate exageosta
data <- simulate_data_exact(kernel, theta, dmetric, n, seed) ## Generate Z observation vector
data
exageostat_finalize() ## Finalize exageostat instance
```

`simulate_obs_exact` *Simulate Geospatial data given (x, y) locations*

Description

Simulate Geospatial data given (x, y) locations

Usage

```
simulate_obs_exact(
  x,
  y,
 kernel = c("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st"),
  theta,
  dmetric = c("euclidean", "great_circle")
)
```

Arguments

`x:`	A vector (x-dim)
`y:`	A vector (y-dim)
`kernel:`	string - kernel ("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st")
`theta:`	list of n parameters (estimated theta)
`dmetric:`	A string - distance metric - "euclidean" or "great_circle"

Value

a list of three vectors (x, y, z)

Examples

```
kernel <- "ugsm-s"
theta <- c(1, 0.1, 0.5)                                          #Params vector.
dmetric <- "euclidean" ## "euclidean" or "great_circle" distance.
n <- 1600 ## The number of locations (n must be a square number, n=m^2)
x <- rnorm(n, 0, 1) # x measurements of n locations.
y <- rnorm(n, 0, 1) # y measurements of n locations.
exageostat_init(hardware = list(ncores = 2, ngpus = 0, ts = 320, lts = 0, pgrid = 1, qgrid = 1)) ## Initiate exageosta
data <- simulate_obs_exact(x, y, kernel, theta, dmetric) ## Generate Z observation vector based on given locations
data
exageostat_finalize() ## Finalize exageostat instance
```

splitting_data *Spliting data into training and testing datasets*

Usage

```
splitting_data(Datatray, k = 10, n = 400)
```

Arguments

`Datatray:`	the full dataset
`k:`	testing dataset portion k \itemn:number of spatial locations a list of training and testing datasets Spliting data into training and testing datasets

tlr_mle *Maximum Likelihood Evaluation (MLE) using Tile Low-Rank (TLR) method*

Description

Maximum Likelihood Evaluation (MLE) using Tile Low-Rank (TLR) method

Usage

```
tlr_mle(
  data = list(x, y, z),
  kernel = c("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st"),
  tlr_acc = 9,
  tlr_maxrank = 400,
  dmetric = c("euclidean", "great_circle"),
  optimization = list(clb = c(0.001, 0.001, 0.001), cub = c(5, 5, 5), tol = 1e-04,
    max_iters = 100)
)
```

Arguments

`data` A list of x vector (x-dim), y vector (y-dim), and z observation vector

`tlr_acc` A number - TLR accuracy level

`tlr_maxrank` A string - TLR max rank

`dmetric` A string - distance metric - "euclidean" or "great_circle"

`optimization` A list of opt lb values (clb), opt ub values (cub), tol, max_iters

`kernel:` string - kernel ("ugsm-s", "ugsmn-s", "bgsfm-s", "bgspm-s", "tgspm-s", "ugsm-st", "bgsm-st")

Value

vector of three values (theta1, theta2, theta3)

Examples

```
seed <- 0 ## Initial seed to generate XY locs.
kernel <- "ugsm-s"
theta <- c(1, 0.1, 0.5)                                          #Params vector.
dmetric <- "euclidean" ## "euclidean" or "great_circle" distance.
n <- 900 ## The number of locations (n must be a square number, n=m^2).
tlr_acc <- 7 ## Approximation accuracy 10^-(acc)
tlr_maxrank <- 150 ## Max Rank
exageostat_init(hardware = list(ncores = 2, ngpus = 0, ts = 320, lts = 1000, pgrid = 1, qgrid = 1)) ## Initiate exageo
data <- simulate_data_exact(kernel, theta, dmetric, n, seed) ## Generate Z observation vector
## Estimate MLE parameters (TLR approximation)
result <- tlr_mle(data, kernel, tlr_acc, tlr_maxrank, dmetric, optimization = list(clb = c(0.001, 0.001, 0.001), cu
print(result)
exageostat_finalize() ## Finalize exageostat instance
```

Index

arg_check_mle, 2
arg_check_predict, 3

bm, 3
bm_comp, 4

check_dmetric, 5
check_kernel, 5
check_theta, 6

dst_mle, 6

exact_mle, 7
exact_mloe_mmom, 8
exact_predict, 9
exageostat_finalize, 9
exageostat_init, 10

fisher_general, 10

mean_absolute_error, 11
mean_square_error, 12
mean_squared_logarithmic_error, 11

plot.Krig, 12

root_mean_squared_error, 13

simulate_data_exact, 13
simulate_obs_exact, 14
splitting_data, 15

tlr_mle, 15