

Package ‘rlikkriging’

July 12, 2026

Type Package

Title Kriging Models using the 'libKriging' Library

Version 1.1-1

Date 2026-07-12

Maintainer Yann Richet <yann.richet@asn.fr>

Description Interface to 'libKriging' 'C++' library <<https://github.com/libKriging>> that should provide most standard Kriging / Gaussian process regression features (like in 'DiceKriging', 'kergp' or 'RobustGaSP' packages). 'libKriging' relies on Armadillo linear algebra library (Apache 2 license) by Conrad Sanderson, 'lbfgsb_cpp' is a 'C++' port around by Pascal Have of 'lbfgsb' library (BSD-3 license) by Ciyou Zhu, Richard Byrd, Jorge Nocedal and Jose Luis Morales used for hyperparameters optimization.

License Apache License (>= 2)

Encoding UTF-8

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 4.2)

Imports Rcpp (>= 1.0.12), methods, DiceKriging

Suggests testthat, utils, foreach, roxygen2, RobustGaSP

SystemRequirements GNU make, cmake (>= 3.2.0), gcc

URL <https://github.com/libKriging>

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Yann Richet [aut, cre] (ORCID: <<https://orcid.org/0000-0002-5677-8458>>),
Pascal Havé [aut],
Yves Deville [aut],
Conrad Sanderson [ctb],
Ciyou Zhu [ctb],
Richard Byrd [ctb],
Jorge Nocedal [ctb],
Jose Luis Morales [ctb],
Mike Smith [ctb]

Repository CRAN

Date/Publication 2026-07-12 15:20:02 UTC

Contents

activation	5
activation.MLPKriging	5
as.km	6
as.km.Kriging	6
as.list.Kriging	7
beta	8
beta.MLPKriging	8
beta.WarpKriging	9
centerX	9
centerX.MLPKriging	10
centerX.WarpKriging	10
centerY	11
centerY.MLPKriging	11
centerY.WarpKriging	12
classKriging	12
classMLPKriging	13
classNestedKriging	13
classWarpKriging	14
copy	14
copy.Kriging	15
copy.MLPKriging	15
copy.WarpKriging	16
covMat	17
covMat.Kriging	17
feature_dim	18
feature_dim.MLPKriging	18
feature_dim.WarpKriging	19
fit	19
fit.Kriging	20
fit.MLPKriging	21
fit.WarpKriging	22
F_	23
F_.MLPKriging	24
F_.WarpKriging	24
hidden_dims	25
hidden_dims.MLPKriging	25
is_fitted	26
is_fitted.MLPKriging	26
is_fitted.WarpKriging	27
kernel	27
kernel.WarpKriging	28
KM	28

KM-class	30
Kriging	31
leaveOneOut	33
leaveOneOut.Kriging	33
leaveOneOutFun	34
leaveOneOutFun.Kriging	34
leaveOneOutVec	35
leaveOneOutVec.Kriging	36
load	37
load.Kriging	38
load.MLPKriging	39
load.WarpKriging	39
logLikelihood	40
logLikelihood.Kriging	40
logLikelihood.WarpKriging	41
logLikelihoodFun	41
logLikelihoodFun.Kriging	42
logLikelihoodFun.MLPKriging	43
logLikelihoodFun.WarpKriging	43
logMargPost	44
logMargPost.Kriging	44
logMargPostFun	45
logMargPostFun.Kriging	46
M	47
M.MLPKriging	47
M.WarpKriging	48
MLPKriging	48
NestedKriging	49
NoiseKM	51
normalize	51
normalize.MLPKriging	52
normalize.WarpKriging	52
NuggetKM	53
predict,KM-method	53
predict.Kriging	55
predict.MLPKriging	56
predict.NestedKriging	57
predict.WarpKriging	57
print.Kriging	58
print.NestedKriging	59
regmodel	60
regmodel.MLPKriging	60
regmodel.WarpKriging	61
save	61
save.Kriging	62
save.MLPKriging	63
save.WarpKriging	63
scaleX	64

scaleX.MLPKriging	64
scaleX.WarpKriging	65
scaleY	65
scaleY.MLPKriging	66
scaleY.WarpKriging	66
sigma2	67
sigma2.WarpKriging	67
simulate,KM-method	68
simulate.Kriging	69
simulate.MLPKriging	70
simulate.WarpKriging	71
theta	72
theta.WarpKriging	72
T_	73
T_.MLPKriging	73
T_.WarpKriging	74
update,KM-method	74
update.Kriging	76
update.MLPKriging	77
update.WarpKriging	78
update_simulate	79
update_simulate.Kriging	79
update_simulate.MLPKriging	80
update_simulate.WarpKriging	81
warping	82
warping.WarpKriging	82
WarpKriging	83
warp_affine	84
warp_boxcox	84
warp_categorical	85
warp_knots	85
warp_kumaraswamy	86
warp_mlp	86
warp_neural_mono	87
warp_none	87
warp_ordinal	88
X	88
X.MLPKriging	89
y	89
y.MLPKriging	90
z	90
z.MLPKriging	91
z.WarpKriging	91

activation	<i>Get activation function name</i>
------------	-------------------------------------

Description

Get activation function name

Usage

activation(object, ...)

Arguments

- | | |
|--------|--|
| object | A Kriging/MLPKriging/WarpKriging model object. |
| ... | Unused. |

activation.MLPKriging	<i>Get activation function for an MLPKriging model</i>
-----------------------	--

Description

Get activation function for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
activation(object, ...)
```

Arguments

- | | |
|--------|--|
| object | A Kriging/MLPKriging/WarpKriging model object. |
| ... | Unused. |

as.km	<i>Coerce an Object into a km Object</i>
-------	--

Description

Coerce an object into an object with S4 class "km" from the **DiceKriging** package.

Usage

```
as.km(x, ...)
```

Arguments

x	Object to be coerced.
...	Further arguments for methods.

Details

Such a coercion is typically used to compare the performance of the methods implemented in the current **rlikkriging** package to those which are available in the **DiceKriging** package.

Value

An object with S4 class "km".

as.km.Kriging	<i>Coerce a Kriging object into the "km" class of the DiceKriging package.</i>
---------------	---

Description

Coerce a Kriging object into the "km" class of the **DiceKriging** package.

Usage

```
## S3 method for class 'Kriging'
as.km(x, .call = NULL, ...)
```

Arguments

x	An object with S3 class "Kriging".
.call	Force the call slot to be filled in the returned km object.
...	Not used.

Value

An object of having the S4 class "KM" which extends the "km" class of the **DiceKriging** package and contains an extra Kriging slot.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, "matern3_2")
print(k)

k_km <- as.km(k)
print(k_km)
```

as.list.Kriging

Coerce a Kriging Object into a List

Description

Coerce a Kriging Object into a List

Usage

```
## S3 method for class 'Kriging'
as.list(x, ...)
```

Arguments

x	An object with class "Kriging".
...	Ignored

Value

A list with its elements copying the content of the Kriging object fields: kernel, optim, objective, theta (vector of ranges), sigma2 (variance), X, centerX, scaleX, y, centerY, scaleY, regmodel, F, T, M, z, beta.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x ) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, kernel = "matern3_2")

l <- as.list(k)
cat(paste0(names(l), " = " , l, collapse = "\n"))
```

beta	<i>Get trend coefficients beta</i>
------	------------------------------------

Description

Get trend coefficients beta

Usage

```
beta(object, ...)
```

Arguments

- object A Kriging/MLPKriging/WarpKriging model object.
- ... Unused.

beta.MLPKriging	<i>Get trend coefficients beta for an MLPKriging model</i>
-----------------	--

Description

Get trend coefficients beta for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'
beta(object, ...)
```

Arguments

- object A Kriging/MLPKriging/WarpKriging model object.
- ... Unused.

beta.WarpKriging	<i>Get trend coefficients beta for a WarpKriging model</i>
------------------	--

Description

Get trend coefficients beta for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
beta(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

centerX	<i>Get input centering vector</i>
---------	-----------------------------------

Description

Get input centering vector

Usage

```
centerX(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

centerX.MLPKriging	<i>Get input centering vector for an MLPKriging model</i>
--------------------	---

Description

Get input centering vector for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
centerX(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

centerX.WarpKriging	<i>Get input centering vector for a WarpKriging model</i>
---------------------	---

Description

Get input centering vector for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
centerX(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

centerY	<i>Get output centering value</i>
---------	-----------------------------------

Description

Get output centering value

Usage

```
centerY(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

centerY.MLPKriging	<i>Get output centering value for an MLPKriging model</i>
--------------------	---

Description

Get output centering value for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
centerY(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

centerY.WarpKriging	<i>Get output centering value for a WarpKriging model</i>
---------------------	---

Description

Get output centering value for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'
centerY(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

classKriging	<i>Shortcut to provide functions to the S3 class "Kriging"</i>
--------------	--

Description

Shortcut to provide functions to the S3 class "Kriging"

Usage

```
classKriging(nk)
```

Arguments

nk	A pointer to a C++ object of class "Kriging"
----	--

Value

An object of class "Kriging" with methods to access and manipulate the data

classMLPKriging	<i>Shortcut to provide functions to the S3 class "MLPKriging"</i>
-----------------	---

Description

Shortcut to provide functions to the S3 class "MLPKriging"

Usage

```
classMLPKriging(obj)
```

Arguments

obj	A list with a ptr element pointing to a C++ MLPKriging object
-----	---

Value

An object of class "MLPKriging" with methods accessible via \$

classNestedKriging	<i>Shortcut to provide functions to the S3 class "NestedKriging"</i>
--------------------	--

Description

Shortcut to provide functions to the S3 class "NestedKriging"

Usage

```
classNestedKriging(nk)
```

Arguments

nk	A pointer to a C++ object of class "NestedKriging"
----	--

Value

An object of class "NestedKriging" with methods to access and manipulate the data

<code>classWarpKriging</code>	<i>Shortcut to provide functions to the S3 class "WarpKriging"</i>
-------------------------------	--

Description

Shortcut to provide functions to the S3 class "WarpKriging"

Usage

```
classWarpKriging(obj)
```

Arguments

<code>obj</code>	A list with a ptr element pointing to a C++ WarpKriging object
------------------	--

Value

An object of class "WarpKriging" with methods accessible via \$

<code>copy</code>	<i>Duplicate object.</i>
-------------------	--------------------------

Description

Duplicate a model given in object.

Usage

```
copy(object, ...)
```

Arguments

<code>object</code>	An object representing a fitted model.
<code>...</code>	Ignored.

Value

The copied object.

copy.Kriging	<i>Duplicate a Kriging Model</i>
--------------	----------------------------------

Description

Duplicate a Kriging Model

Usage

```
## S3 method for class 'Kriging'  
copy(object, ...)
```

Arguments

object	An S3 Kriging object.
...	Not used.

Value

The copy of object.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)  
set.seed(123)  
X <- as.matrix(runif(10))  
y <- f(X)  
  
k <- Kriging(y, X, kernel = "matern3_2", objective="LMP")  
print(k)  
  
print(copy(k))
```

copy.MLPKriging	<i>Deep copy of MLPKriging model</i>
-----------------	--------------------------------------

Description

Deep copy of MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'
copy(object, ...)
```

Arguments

object	MLPKriging object
...	ignored

Value

a new independent MLPKriging object

copy.WarpKriging	<i>Deep copy of WarpKriging model</i>
------------------	---------------------------------------

Description

Deep copy of WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'
copy(object, ...)
```

Arguments

object	WarpKriging object
...	ignored

Value

a new independent WarpKriging object

covMat	<i>covariance function</i>
--------	----------------------------

Description

Compute the covariance matrix of a model given in object, between given set of points.

Usage

```
covMat(object, ...)
```

Arguments

object	An object representing a fitted model.
...	Further arguments of function (eg. points, range).

Value

The covariance matrix.

covMat.Kriging	<i>Compute Covariance Matrix of Kriging Model</i>
----------------	---

Description

Compute Covariance Matrix of Kriging Model

Usage

```
## S3 method for class 'Kriging'  
covMat(object, x1, x2, ...)
```

Arguments

object	An S3 Kriging object.
x1	Numeric matrix of input points.
x2	Numeric matrix of input points.
...	Not used.

Value

A matrix of the covariance matrix of the Kriging model.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, kernel = "gauss")

x1 = runif(10)
x2 = runif(10)

covMat(k, x1, x2)
```

feature_dim	<i>Get feature dimensionality (d_out)</i>
-------------	---

Description

Get feature dimensionality (d_out)

Usage

```
feature_dim(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

feature_dim.MLPKriging	<i>Get feature dimensionality for an MLPKriging model</i>
------------------------	---

Description

Get feature dimensionality for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'
feature_dim(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

`feature_dim.WarpKriging`*Get feature dimensionality of warped space*

Description

Get feature dimensionality of warped space

Usage

```
## S3 method for class 'WarpKriging'  
feature_dim(object, ...)
```

Arguments

<code>object</code>	A Kriging/MLPKriging/WarpKriging model object.
<code>...</code>	Unused.

`fit`*Fit model on data.*

Description

Fit a model given in object.

Usage

```
fit(object, ...)
```

Arguments

<code>object</code>	An object representing a fitted model.
<code>...</code>	Further arguments of function

Value

No return value. Kriging object argument is modified.

fit.Kriging

*Fit Kriging object on given data.***Description**

The hyper-parameters (variance and vector of correlation ranges) are estimated thanks to the optimization of a criterion given by objective, using the method given in optim.

Usage

```
## S3 method for class 'Kriging'
fit(
  object,
  y,
  X,
  regmodel = c("constant", "linear", "interactive", "quadratic", "none"),
  normalize = FALSE,
  optim = c("BFGS", "none"),
  objective = c("LL", "LOO", "LMP"),
  parameters = NULL,
  noise = NULL,
  ...
)
```

Arguments

object	S3 Kriging object.
y	Numeric vector of response values.
X	Numeric matrix of input design.
regmodel	Universal Kriging linear trend: "constant", "linear", "interactive", "quadratic".
normalize	Logical. If TRUE both the input matrix X and the response y in normalized to take values in the interval [0, 1].
optim	Character giving the Optimization method used to fit hyper-parameters. Possible values are: "BFGS" and "none", the later simply keeping the values given in parameters. The method "BFGS" uses the gradient of the objective (note that "BFGS10" means 10 multi-start of BFGS).
objective	Character giving the objective function to optimize. Possible values are: "LL" for the Log-Likelihood, "LOO" for the Leave-One-Out sum of squares and "LMP" for the Log-Marginal Posterior.
parameters	Initial values for the hyper-parameters. When provided this must be named list with elements "sigma2" and "theta" containing the initial value(s) for the variance and for the range parameters. If theta is a matrix with more than one row, each row is used as a starting point for optimization.
noise	Either a numeric vector of per-observation noise variances, or "nugget" to estimate a homogeneous nugget, or NULL (default) for noise-free interpolation.
...	Ignored.

Value

No return value. Kriging object argument is modified.

Author(s)

Yann Richet <yann.richet@asnr.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
plot(f)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)
points(X, y, col = "blue", pch = 16)

k <- Kriging("matern3_2")
print(k)

fit(k,y,X)
print(k)
```

fit.MLPKriging

Fit an MLPKriging model to data

Description

(Re-)fit an already-constructed MLPKriging object on new data. The MLP architecture and kernel are kept from construction.

Usage

```
## S3 method for class 'MLPKriging'
fit(
  object,
  y,
  X,
  regmodel = "constant",
  normalize = FALSE,
  optim = "BFGS+Adam",
  objective = "LL",
  parameters = NULL,
  ...
)
```

Arguments

object	MLPKriging object
y	numeric vector of observations (n)
X	numeric matrix of inputs (n x d)
regmodel	trend: "constant", "linear", "quadratic"
normalize	logical; normalise inputs?
optim	optimiser
objective	"LL" (log-likelihood)
parameters	optional named list of tuning parameters
...	ignored

Value

No return value. MLPKriging object argument is modified.

fit.WarpKriging	<i>Fit a WarpKriging model to data</i>
-----------------	--

Description

(Re-)fit an already-constructed WarpKriging object on new data. The warping specification and kernel are kept from construction.

Usage

```
## S3 method for class 'WarpKriging'
fit(
  object,
  y,
  X,
  regmodel = "constant",
  normalize = FALSE,
  optim = "BFGS+Adam",
  objective = "LL",
  parameters = NULL,
  noise = NULL,
  ...
)
```

Arguments

object	WarpKriging object (created with the constructor or an empty-kernel call)
y	numeric vector of observations (n)
X	numeric matrix of inputs (n x d)
regmodel	trend: "constant", "linear", "quadratic"
normalize	logical; normalise continuous inputs?
optim	optimiser
objective	"LL" (log-likelihood)
parameters	optional named list of tuning parameters
noise	Either a numeric vector of per-observation noise variances, or "nugget" to estimate a homogeneous nugget, or NULL (default) for noise-free interpolation.
...	ignored

Value

No return value. WarpKriging object argument is modified.

F_	<i>Get trend matrix F</i>
----	---------------------------

Description

Get trend matrix F

Usage

F_(object, ...)

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

F_.MLPKriging	<i>Get trend matrix F for an MLPKriging model</i>
---------------	---

Description

Get trend matrix F for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'
F_(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

F_.WarpKriging	<i>Get trend matrix F for a WarpKriging model</i>
----------------	---

Description

Get trend matrix F for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'
F_(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

hidden_dims	<i>Get hidden layer sizes</i>
-------------	-------------------------------

Description

Get hidden layer sizes

Usage

hidden_dims(object, ...)

Arguments

- | | |
|--------|--|
| object | A Kriging/MLPKriging/WarpKriging model object. |
| ... | Unused. |

hidden_dims.MLPKriging	<i>Get hidden layer sizes for an MLPKriging model</i>
------------------------	---

Description

Get hidden layer sizes for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
hidden_dims(object, ...)
```

Arguments

- | | |
|--------|--|
| object | A Kriging/MLPKriging/WarpKriging model object. |
| ... | Unused. |

<code>is_fitted</code>	<i>Check if the model has been fitted</i>
------------------------	---

Description

Check if the model has been fitted

Usage

```
is_fitted(object, ...)
```

Arguments

<code>object</code>	A Kriging/MLPKriging/WarpKriging model object.
<code>...</code>	Unused.

<code>is_fitted.MLPKriging</code>	<i>Check whether an MLPKriging model is fitted</i>
-----------------------------------	--

Description

Check whether an MLPKriging model is fitted

Usage

```
## S3 method for class 'MLPKriging'
is_fitted(object, ...)
```

Arguments

<code>object</code>	A Kriging/MLPKriging/WarpKriging model object.
<code>...</code>	Unused.

is_fitted.WarpKriging *Check whether a WarpKriging model is fitted*

Description

Check whether a WarpKriging model is fitted

Usage

```
## S3 method for class 'WarpKriging'  
is_fitted(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

kernel	<i>Get kernel name</i>
--------	------------------------

Description

Get kernel name

Usage

```
kernel(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

Value

Character scalar naming the covariance kernel.

kernel.WarpKriging	<i>Get kernel name</i>
--------------------	------------------------

Description

Get kernel name

Usage

```
## S3 method for class 'WarpKriging'
kernel(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

KM	<i>Create an KM Object</i>
----	----------------------------

Description

Create an object of S4 class "KM" similar to a km object in the **DiceKriging** package.

Usage

```
KM(
  formula = ~1,
  design,
  response,
  covtype = c("matern5_2", "gauss", "matern3_2", "exp"),
  coef.trend = NULL,
  coef.cov = NULL,
  coef.var = NULL,
  nugget = NULL,
  nugget.estim = FALSE,
  noise.var = NULL,
  estim.method = c("MLE", "LOO"),
  penalty = NULL,
  optim.method = "BFGS",
  lower = NULL,
  upper = NULL,
  parinit = NULL,
  multistart = 1,
  control = NULL,
```

```

    gr = TRUE,
    iso = FALSE,
    scaling = FALSE,
    knots = NULL,
    kernel = NULL,
    ...
)

```

Arguments

formula	R formula object to setup the linear trend in Universal Kriging. Supports ~ 1 , $\sim \cdot$ and $\sim \cdot^2$.
design	Data frame. The design of experiments.
response	Vector of output values.
covtype	Covariance structure. For now all the kernels are tensor product kernels.
coef.trend	Optional value for a fixed vector of trend coefficients. If given, no optimization is done.
coef.cov	Optional value for a fixed correlation range value. If given, no optimization is done.
coef.var	Optional value for a fixed variance. If given, no optimization is done.
nugget	Ignored (kept for DiceKriging API compatibility).
nugget.estim	Logical. If TRUE, fit a nugget effect.
noise.var	Numeric vector of known per-point noise variances.
estim.method	Estimation criterion. "MLE" for Maximum-Likelihood or "LOO" for Leave-One-Out cross-validation.
penalty	Not implemented yet.
optim.method	Optimization algorithm used in the optimization of the objective given in estim.method. Supports "BFGS".
lower, upper	Not implemented yet.
parinit	Initial values for the correlation ranges which will be optimized using optim.method.
multistart, control, gr, iso	Not implemented yet.
scaling, knots, kernel	Not implemented yet.
...	Ignored.

Details

The class "KM" extends the "km" class of the **DiceKriging** package, hence has all slots of "km". It also has an extra slot "Kriging" slot which contains a copy of the original object.

Value

A KM object. See **Details**.

Author(s)

Yann Richet <yann.richet@asn.fr>

See Also

[km](#) in the **DiceKriging** package for more details on the slots.

Examples

```
# a 16-points factorial design, and the corresponding response
d <- 2; n <- 16
design.fact <- as.matrix(expand.grid(x1 = seq(0, 1, length = 4),
                                   x2 = seq(0, 1, length = 4)))
y <- apply(design.fact, 1, DiceKriging::branin)

# Using `km` from DiceKriging and a similar `KM` object
# kriging model 1 : matern5_2 covariance structure, no trend, no nugget effect
km1 <- DiceKriging::km(design = design.fact, response = y, covtype = "gauss",
                      parinit = c(.5, 1), control = list(trace = FALSE))
KM1 <- KM(design = design.fact, response = y, covtype = "gauss",
          parinit = c(.5, 1))
```

KM-class

S4 class for Kriging Models Extending the "km" Class

Description

This class is intended to be used either by using its own dedicated S4 methods or by using the S4 methods inherited from the "km" class of the **libKriging** package.

Slots

`d, n, X, y, p, F` Number of (numeric) inputs, number of observations, design matrix, response vector, number of trend variables, trend matrix.

`trend.formula, trend.coef` Formula used for the trend, vector $\hat{\beta}$ of estimated (or fixed) trend coefficients with length p .

`covariance` A S4 object with class "covTensorProduct" representing a covariance kernel.

`noise.flag, noise.var` Logical flag and numeric value for an optional noise term.

`known.param` A character code indicating what parameters are known.

`lower, upper` Bounds on the correlation range parameters.

`method, penalty, optim.method, control, gr, parinit` Objects defining the estimation criterion, the optimization.

`T, M, z` Auxiliary variables (matrices and vectors) that can be used in several computations.

`case` The possible concentration (a.k.a. profiling) of the likelihood.

`param.estim` Logical. Is an estimation used?

`Kriging` A copy of the Kriging object used to create the current KM object.

Author(s)

Yann Richet <yann.richet@asn.fr>

See Also

[km-class](#) in the **DiceKriging** package. The creator [KM](#).

Kriging	Create an object with S3 class "Kriging" using the libKriging library.
---------	---

Description

The hyper-parameters (variance and vector of correlation ranges) are estimated thanks to the optimization of a criterion given by objective, using the method given in optim.

Usage

```
Kriging(
  y = NULL,
  X = NULL,
  kernel = NULL,
  regmodel = c("constant", "linear", "interactive", "quadratic", "none"),
  normalize = FALSE,
  optim = c("BFGS", "none"),
  objective = c("LL", "LOO", "LMP"),
  parameters = NULL,
  noise = NULL
)
```

Arguments

y	Numeric vector of response values.
X	Numeric matrix of input design.
kernel	Character defining the covariance model: "exp", "gauss", "matern3_2", "matern5_2".
regmodel	Universal Kriging linear trend: "constant", "linear", "interactive", "quadratic".
normalize	Logical. If TRUE both the input matrix X and the response y in normalized to take values in the interval [0, 1].
optim	Character giving the Optimization method used to fit hyper-parameters. Possible values are: "BFGS" and "none", the later simply keeping the values given in parameters. The method "BFGS" uses the gradient of the objective (note that "BFGS10" means 10 multi-start of BFGS).

objective	Character giving the objective function to optimize. Possible values are: "LL" for the Log-Likelihood, "LOO" for the Leave-One-Out sum of squares, "LMP" for the Log-Marginal Posterior, and "VLL" or "VLL(m)" for the Vecchia approximated log-likelihood with m conditioning neighbors (default 30): each evaluation costs $O(n \cdot m^3)$ instead of $O(n^3)$, recommended for large designs in low dimension.
parameters	Initial values for the hyper-parameters. When provided this must be named list with elements "sigma2" and "theta" containing the initial value(s) for the variance and for the range parameters. If theta is a matrix with more than one row, each row is used as a starting point for optimization.
noise	Either a numeric vector of per-observation noise variances, or "nugget" to estimate a homogeneous nugget, or NULL (default) for noise-free interpolation.

Value

An object with S3 class "Kriging". Should be used with its predict, simulate, update methods.

Author(s)

Yann Richet <yann.richet@asnr.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)
## fit and print
k <- Kriging(y, X, kernel = "matern3_2")
print(k)

x <- as.matrix(seq(from = 0, to = 1, length.out = 101))
p <- predict(k, x = x, return_stdev = TRUE, return_cov = FALSE)

plot(f)
points(X, y)
lines(x, p$mean, col = "blue")
polygon(c(x, rev(x)), c(p$mean - 2 * p$stdev, rev(p$mean + 2 * p$stdev)),
border = NA, col = rgb(0, 0, 1, 0.2))

s <- simulate(k, nsim = 10, seed = 123, x = x)

matlines(x, s, col = rgb(0, 0, 1, 0.2), type = "l", lty = 1)
```

leaveOneOut	<i>Compute Leave-One-Out</i>
-------------	------------------------------

Description

Compute the leave-One-Out error of a model given in object.

Usage

```
leaveOneOut(object, ...)
```

Arguments

object	An object representing a fitted model.
...	Ignored.

Value

The Leave-One-Out sum of squares.

leaveOneOut.Kriging	<i>Get leaveOneOut of Kriging Model</i>
---------------------	---

Description

Get leaveOneOut of Kriging Model

Usage

```
## S3 method for class 'Kriging'
leaveOneOut(object, ...)
```

Arguments

object	An S3 Kriging object.
...	Not used.

Value

The leaveOneOut computed for fitted *theta*.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, kernel = "matern3_2", objective="LOO")
print(k)

leaveOneOut(k)
```

leaveOneOutFun	<i>Leave-One-Out function</i>
----------------	-------------------------------

Description

Compute the leave-One-Out error of a model given in object, at a different value of the parameters.

Usage

```
leaveOneOutFun(object, ...)
```

Arguments

object	An object representing a fitted model.
...	Further arguments of function (eg. range).

Value

The Leave-One-Out sum of squares.

leaveOneOutFun.Kriging	<i>Compute Leave-One-Out (LOO) error for an object with S3 class "Kriging" representing a kriging model.</i>
------------------------	--

Description

The returned value is the sum of squares $\sum_{i=1}^n [y_i - \hat{y}_{i,(-i)}]^2$ where $\hat{y}_{i,(-i)}$ is the prediction of y_i based on the the observations y_j with $j \neq i$.

Usage

```
## S3 method for class 'Kriging'
leaveOneOutFun(object, theta, return_grad = FALSE, bench = FALSE, ...)
```

Arguments

object	A Kriging object.
theta	A numeric vector of range parameters at which the LOO will be evaluated.
return_grad	Logical. Should the gradient (w.r.t. theta) be returned?
bench	Logical. Should the function display benchmarking output
...	Not used.

Value

The leave-One-Out value computed for the given vector θ of correlation ranges.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, kernel = "matern3_2", objective = "LOO", optim="BFGS")
print(k)

loo <- function(theta) leaveOneOutFun(k, theta)$leaveOneOut
t <- seq(from = 0.001, to = 2, length.out = 101)
plot(t, loo(t), type = "l")
abline(v = k$theta(), col = "blue")
```

leaveOneOutVec	<i>Leave-One-Out vector</i>
----------------	-----------------------------

Description

Compute the leave-One-Out vector error of a model given in object, at a different value of the parameters.

Usage

```
leaveOneOutVec(object, ...)
```

Arguments

object	An object representing a fitted model.
...	Further arguments of function (eg. range).

Value

The Leave-One-Out errors (mean and stdev) for each conditional point.

```
leaveOneOutVec.Kriging
```

Compute Leave-One-Out (LOO) vector error for an object with S3 class "Kriging" representing a kriging model.

Description

The returned value is the mean and stdev of $\hat{y}_{i,(-i)}$, the prediction of y_i based on the the observations y_j with $j \neq i$.

Usage

```
## S3 method for class 'Kriging'
leaveOneOutVec(object, theta, ...)
```

Arguments

object	A Kriging object.
theta	A numeric vector of range parameters at which the LOO will be evaluated.
...	Not used.

Value

The leave-One-Out vector computed for the given vector θ of correlation ranges.

Author(s)

Yann Richet <yann.richet@asnr.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(c(0.0, 0.25, 0.5, 0.75, 1.0))
y <- f(X)

k <- Kriging(y, X, kernel = "matern3_2")
print(k)

x <- as.matrix(seq(0, 1, , 101))
p <- predict(k, x, TRUE, FALSE)

plot(f)
points(X, y)
lines(x, p$mean, col = 'blue')
```

```

polygon(c(x, rev(x)), c(p$mean - 2 * p$stdev, rev(p$mean + 2 * p$stdev)),
        border = NA, col = rgb(0, 0, 1, 0.2))

# Compute leave-one-out (no range re-estimate) on 2nd point
X_no2 = X[-2,,drop=FALSE]
y_no2 = f(X_no2)
k_no2 = Kriging(y_no2, X_no2, "matern3_2", optim = "none", parameters = list(theta = k$theta()))
print(k_no2)

p_no2 <- predict(k_no2, x, TRUE, FALSE)
lines(x, p_no2$mean, col = 'red')
polygon(c(x, rev(x)), c(p_no2$mean - 2 * p_no2$stdev, rev(p_no2$mean + 2 * p_no2$stdev)),
        border = NA, col = rgb(1, 0, 0, 0.2))

# Use leaveOneOutVec to get the same
loov = k$leaveOneOutVec(matrix(k$theta()))
points(X[2], loov$mean[2], col='red')
lines(rep(X[2], 2), loov$mean[2]+2*c(-loov$stdev[2], loov$stdev[2]), col='red')

```

load	<i>Load any Kriging Model from a file storage. Back to base::load if not a Kriging object.</i>
------	--

Description

Load any Kriging Model from a file storage. Back to base::load if not a Kriging object.

Usage

```
load(filename, ...)
```

Arguments

filename	A file holding any Kriging object.
...	Arguments used by base::load.

Value

The loaded "Kriging object, or nothing if base::load is used (update parent environment).

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, kernel = "matern3_2", objective="LMP")
print(k)

outfile = tempfile("k.json")
save(k,outfile)

print(load(outfile))
```

load.Kriging

Load a Kriging Model from a file storage

Description

Load a Kriging Model from a file storage

Usage

```
load.Kriging(filename, ...)
```

Arguments

filename	File name to load from.
...	Not used.

Value

The loaded Kriging object.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, kernel = "matern3_2", objective="LMP")
print(k)

outfile = tempfile("k.json")
```

```

save(k,outfile)

print(load.Kriging(outfile))
unlink(outfile)

```

load.MLPKriging	<i>Load an MLPKriging model from file</i>
-----------------	---

Description

Load an MLPKriging model from file

Usage

```

## S3 method for class 'MLPKriging'
load(filename, ...)

```

Arguments

filename	path to saved file
...	ignored

Value

MLPKriging object

load.WarpKriging	<i>Load a WarpKriging model from file</i>
------------------	---

Description

Load a WarpKriging model from file

Usage

```

## S3 method for class 'WarpKriging'
load(filename, ...)

```

Arguments

filename	path to saved file
...	ignored

Value

WarpKriging object

logLikelihood	<i>Compute Log-Likelihood</i>
---------------	-------------------------------

Description

Compute the log-Likelihood of a model given in object.

Usage

```
logLikelihood(object, ...)
```

Arguments

object	An object representing a fitted model.
...	Ignored.

Value

The log-likelihood.

logLikelihood.Kriging	<i>Get Log-Likelihood of Kriging Model</i>
-----------------------	--

Description

Get Log-Likelihood of Kriging Model

Usage

```
## S3 method for class 'Kriging'  
logLikelihood(object, ...)
```

Arguments

object	An S3 Kriging object.
...	Not used.

Value

The log-Likelihood computed for fitted *theta*.

Author(s)

Yann Richet <yann.richet@asnr.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, kernel = "matern3_2", objective="LL")
print(k)

logLikelihood(k)
```

logLikelihood.WarpKriging

Log-likelihood of the fitted model

Description

Log-likelihood of the fitted model

Usage

```
## S3 method for class 'WarpKriging'
logLikelihood(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

logLikelihoodFun *Log-Likelihood function*

Description

Compute the log-Likelihood of a model given in object, at a different value of the parameters.

Usage

```
logLikelihoodFun(object, ...)
```

Arguments

object	An object representing a fitted model.
...	Further arguments of function (eg. range).

Value

The log-likelihood.

logLikelihoodFun.Kriging

Compute Log-Likelihood of Kriging Model

Description

Compute Log-Likelihood of Kriging Model

Usage

```
## S3 method for class 'Kriging'
logLikelihoodFun(
  object,
  theta,
  return_grad = FALSE,
  return_hess = FALSE,
  bench = FALSE,
  ...
)
```

Arguments

object	An S3 Kriging object.
theta	A numeric vector of (positive) range parameters at which the log-likelihood will be evaluated.
return_grad	Logical. Should the function return the gradient?
return_hess	Logical. Should the function return Hessian?
bench	Logical. Should the function display benchmarking output?
...	Not used.

Value

The log-Likelihood computed for given *theta*.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, kernel = "matern3_2")
```

```

print(k)

ll <- function(theta) logLikelihoodFun(k, theta)$logLikelihood

t <- seq(from = 0.001, to = 2, length.out = 101)
plot(t, ll(t), type = 'l')
abline(v = k$theta(), col = "blue")

```

logLikelihoodFun.MLPKriging

Evaluate log-likelihood at given GP theta

Description

Evaluate log-likelihood at given GP theta

Usage

```

## S3 method for class 'MLPKriging'
logLikelihoodFun(object, theta, return_grad = FALSE, return_hess = FALSE, ...)

```

Arguments

object	MLPKriging object
theta	range parameter vector
return_grad	return gradient?
return_hess	return hessian?
...	ignored

logLikelihoodFun.WarpKriging

Evaluate log-likelihood at given theta

Description

Evaluate log-likelihood at given theta

Usage

```

## S3 method for class 'WarpKriging'
logLikelihoodFun(object, theta, return_grad = FALSE, return_hess = FALSE, ...)

```

Arguments

object	WarpKriging object
theta	range parameter vector
return_grad	return gradient?
return_hess	return hessian?
...	ignored

Value

list with logLikelihood and optionally gradient, hessian

logMargPost	<i>Compute log-Marginal Posterior</i>
-------------	---------------------------------------

Description

Compute the log-Marginal Posterior of a model given in object.

Usage

```
logMargPost(object, ...)
```

Arguments

object	An object representing a fitted model.
...	Ignored.

Value

The log-marginal posterior.

logMargPost.Kriging	<i>Get logMargPost of Kriging Model</i>
---------------------	---

Description

Get logMargPost of Kriging Model

Usage

```
## S3 method for class 'Kriging'  
logMargPost(object, ...)
```

Arguments

object	An S3 Kriging object.
...	Not used.

Value

The logMargPost computed for fitted *theta*.

Author(s)

Yann Richet <yann.richet@asnr.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, kernel = "matern3_2", objective="LMP")
print(k)

logMargPost(k)
```

logMargPostFun	<i>log-Marginal Posterior function</i>
----------------	--

Description

Compute the log-Marginal Posterior of a model given in object, at a different value of the parameters.

Usage

```
logMargPostFun(object, ...)
```

Arguments

object	An object representing a fitted model.
...	Further arguments of function (eg. range).

Value

The log-marginal posterior.

logMargPostFun.Kriging

Compute the log-marginal posterior of a kriging model, using the prior XXXY.

Description

Compute the log-marginal posterior of a kriging model, using the prior XXXY.

Usage

```
## S3 method for class 'Kriging'
logMargPostFun(object, theta, return_grad = FALSE, bench = FALSE, ...)
```

Arguments

object	S3 Kriging object.
theta	Numeric vector of correlation range parameters at which the function is to be evaluated.
return_grad	Logical. Should the function return the gradient (w.r.t theta)?
bench	Logical. Should the function display benchmarking output?
...	Not used.

Value

The value of the log-marginal posterior computed for the given vector theta.

Author(s)

Yann Richet <yann.richet@asn.fr>

References

XXXY A reference describing the model (prior, ...)

See Also

[rgasp](#) in the RobustGaSP package.

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, "matern3_2", objective="LMP")
```

```

print(k)

lmp <- function(theta) logMargPostFun(k, theta)$logMargPost

t <- seq(from = 0.01, to = 2, length.out = 101)
plot(t, lmp(t), type = "l")
abline(v = k$theta(), col = "blue")

```

M	<i>Get whitened trend matrix M</i>
---	------------------------------------

Description

Get whitened trend matrix M

Usage

```
M(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

M.MLPKriging	<i>Get whitened trend matrix M for an MLPKriging model</i>
--------------	--

Description

Get whitened trend matrix M for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'
M(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

M.WarpKriging

Get whitened trend matrix M for a WarpKriging model

Description

Get whitened trend matrix M for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'
M(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

MLPKriging

Create an MLPKriging model (Deep Kernel Learning)

Description

Kriging with a joint multi-layer perceptron applied to all inputs before the GP kernel is evaluated. The MLP weights, GP range parameters, variance and trend are jointly fitted by maximising the concentrated log-likelihood.

Usage

```
MLPKriging(
  y,
  X,
  hidden_dims,
  d_out = 2,
  activation = "selu",
  kernel = "gauss",
  regmodel = "constant",
  normalize = FALSE,
  optim = "BFGS+Adam",
  objective = "LL",
  parameters = NULL
)
```


Arguments

y	numeric vector of observations (n)
X	numeric matrix of inputs (n x d)
hidden_dims	integer vector of hidden layer sizes, e.g. c(32, 16)
d_out	output feature dimensionality (default 2)
activation	activation function: "relu", "selu", "tanh", "sigmoid", "elu"
kernel	covariance kernel: "gauss", "matern3_2", "matern5_2", "exp"
regmodel	trend: "constant", "linear", "quadratic"
normalize	logical; normalise inputs?
optim	optimiser (default "BFGS+Adam")
objective	"LL" (log-likelihood)
parameters	optional named list of tuning parameters, e.g. list(max_iter_adam = "300", adam_lr = "0.001", max_iter_bfgs = "50")

Value

An S3 object of class "MLPKriging".

Examples

```
X <- as.matrix(seq(0.01, 0.99, length.out = 10))
f <- function(x) 1 - 1/2 * (sin(12*x)/(1+x) + 2*cos(7*x)*x^5 + 0.7)
y <- f(X)
k <- MLPKriging(y, X, hidden_dims = c(16, 8), d_out = 2,
  activation = "selu", kernel = "gauss")
print(k)
```

NestedKriging	Create an object with S3 class "NestedKriging" using the libKriging library.
---------------	---

Description

Divide-and-conquer Kriging for large designs: the data are partitioned in nb_groups groups, one **Kriging** submodel is fitted per group (then all share a common prior), and predictions are aggregated.

Usage

```

NestedKriging(
  y = NULL,
  X = NULL,
  kernel = NULL,
  nb_groups = NULL,
  aggregation = "NK",
  partition = "kmeans",
  seed = 123,
  regmodel = "constant",
  optim = "BFGS",
  objective = "LL",
  parameters = NULL,
  warping = NULL
)

```

Arguments

y	Numeric vector of response values.
X	Numeric matrix of input design.
kernel	Character defining the covariance model: "exp", "gauss", "matern3_2", "matern5_2".
nb_groups	Number of submodels (each of size $\sim \text{nrow}(X)/\text{nb_groups}$).
aggregation	Character: "NK" (optimal nested-kriging aggregation of Ruliere et al. 2018, default), "PoE", "gPoE", "BCM" or "rBCM".
partition	Character: "kmeans" (default) or "random".
seed	Integer seed for the partition.
regmodel	Linear trend; "NK" aggregation requires "constant".
optim	Character, hyper-parameter optimization method of the submodels: "BFGS" or "none".
objective	Character: "LL" (default), "L00", "LMP", or "VLL(m)" — the common prior is then estimated by one global Vecchia fit ($O(n \, m^3)$, cross-group information).
parameters	Initial or fixed values for the hyper-parameters (named list with "sigma2", "theta", "beta").
warping	Optional character vector of per-dimension warp specs (see WarpKriging); submodels are then WarpKriging.

Value

An object with S3 class "NestedKriging", to be used with its predict method.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(X) apply(X, 1, function(x) sin(3 * x[1]) + cos(5 * x[2]))
set.seed(123)
X <- matrix(runif(2 * 400), ncol = 2)
y <- f(X)
k <- NestedKriging(y, X, kernel = "matern5_2", nb_groups = 8)
x <- matrix(runif(2 * 100), ncol = 2)
p <- predict(k, x)
```

NoiseKM

*Create a KM object with heteroscedastic noise (deprecated)***Description**

This is a thin compatibility wrapper. Use [KM](#)(noise.var = ...) directly instead.

Usage

```
NoiseKM(..., noise.var)
```

Arguments

... Arguments passed to [KM](#).

noise.var Numeric vector of known per-point noise variances.

Value

A KM object. See [KM](#).

normalize

*Get normalize flag***Description**

Get normalize flag

Usage

```
normalize(object, ...)
```

Arguments

object A Kriging/MLPKriging/WarpKriging model object.

... Unused.

normalize.MLPKriging	<i>Get normalize flag for an MLPKriging model</i>
----------------------	---

Description

Get normalize flag for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
normalize(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

normalize.WarpKriging	<i>Get normalize flag for a WarpKriging model</i>
-----------------------	---

Description

Get normalize flag for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
normalize(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

NuggetKM

Create a KM object with nugget effect (deprecated)

Description

This is a thin compatibility wrapper. Use `KM(nugget.estim = TRUE)` directly instead.

Usage

```
NuggetKM(...)
```

Arguments

... Arguments passed to `KM`.

Value

A KM object. See `KM`.

predict,KM-method

Prediction Method for a KM Object

Description

Compute predictions for the response at new given input points. These conditional mean, the conditional standard deviation and confidence limits at the 95% level. Optionnally the conditional covariance can be returned as well.

Usage

```
## S4 method for signature 'KM'
predict(
  object,
  newdata,
  type = "UK",
  se.compute = TRUE,
  cov.compute = FALSE,
  light.return = TRUE,
  bias.correct = FALSE,
  checkNames = FALSE,
  ...
)
```

Arguments

<code>object</code>	KM object.
<code>newdata</code>	Matrix of "new" input points where to perform prediction.
<code>type</code>	character giving the kriging type. For now only "UK" is possible.
<code>se.compute</code>	Logical. Should the standard error be computed?
<code>cov.compute</code>	Logical. Should the covariance matrix between newdata points be computed?
<code>light.return</code>	Logical. If TRUE, no auxiliary results will be returned (such as the Cholesky root of the correlation matrix).
<code>bias.correct</code>	Logical. If TRUE the UK variance and covariance are .
<code>checkNames</code>	Logical to check the consistency of the column names between the design stored in <code>object@X</code> and the new one given <code>newdata</code> .
<code>...</code>	Ignored.

Details

Without a dedicated `predict` method for the class "KM", this method would have been inherited from the "km" class. The dedicated method is expected to run faster. A comparison can be made by coercing a KM object to a km object with `as.km` before calling `predict`.

Value

A named list. The elements are the conditional mean and standard deviation (mean and sd), the predicted trend (trend) and the confidence limits (lower95 and upper95). Optionnally, the conditional covariance matrix is returned in `cov`.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
## a 16-points factorial design, and the corresponding response
d <- 2; n <- 16
design.fact <- expand.grid(x1 = seq(0, 1, length = 4), x2 = seq(0, 1, length = 4))
y <- apply(design.fact, 1, DiceKriging::branin)

## library(DiceKriging)
## kriging model 1 : matern5_2 covariance structure, no trend, no nugget
## m1 <- km(design = design.fact, response = y, covtype = "gauss",
##         parinit = c(.5, 1), control = list(trace = FALSE))
KM1 <- KM(design = design.fact, response = y, covtype = "gauss",
         parinit = c(.5, 1))
Pred <- predict(KM1, newdata = matrix(.5, ncol = 2), type = "UK",
              checkNames = FALSE, light.return = TRUE)
```

predict.Kriging	<i>Predict from a Kriging object.</i>
-----------------	---------------------------------------

Description

Given "new" input points, the method compute the expectation, variance and (optionnally) the covariance of the corresponding stochastic process, conditional on the values at the input points used when fitting the model.

Usage

```
## S3 method for class 'Kriging'
predict(
  object,
  x,
  return_stdev = TRUE,
  return_cov = FALSE,
  return_deriv = FALSE,
  ...
)
```

Arguments

object	S3 Kriging object.
x	Input points where the prediction must be computed.
return_stdev	Logical. If TRUE the standard deviation is returned.
return_cov	Logical. If TRUE the covariance matrix of the predictions is returned.
return_deriv	Logical. If TRUE the derivatives of mean and sd of the predictions are returned.
...	Ignored.

Value

A list containing the element mean and possibly stdev and cov.

Note

The names of the formal arguments differ from those of the predict methods for the S4 classes "km" and "KM". The formal x corresponds to newdata, stdev corresponds to se.compute and cov to cov.compute. These names are chosen **Python** and **Octave** interfaces to **libKriging**.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
plot(f)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)
points(X, y, col = "blue", pch = 16)

k <- Kriging(y, X, "matern3_2")

x <- seq(from = 0, to = 1, length.out = 101)
p <- predict(k, x)

lines(x, p$mean, col = "blue")
polygon(c(x, rev(x)), c(p$mean - 2 * p$stdev, rev(p$mean + 2 * p$stdev)),
  border = NA, col = rgb(0, 0, 1, 0.2))
```

predict.MLPKriging	<i>Predict with an MLPKriging model</i>
--------------------	---

Description

Predict with an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'
predict(
  object,
  x,
  return_stdev = TRUE,
  return_cov = FALSE,
  return_deriv = FALSE,
  ...
)
```

Arguments

object	MLPKriging object
x	prediction matrix (m x d)
return_stdev	return standard deviations?
return_cov	return full covariance?
return_deriv	return derivatives of mean and stdev wrt x?
...	ignored

Value

list with mean, optionally stdev, cov, mean_deriv, stdev_deriv

predict.NestedKriging *Predict from a NestedKriging object.*

Description

Predict from a NestedKriging object.

Usage

```
## S3 method for class 'NestedKriging'  
predict(object, x, return_stdev = TRUE, ...)
```

Arguments

object	S3 NestedKriging object.
x	Input points (matrix) where to predict.
return_stdev	Logical, return standard deviation (default TRUE).
...	Ignored.

Value

A list with elements mean and (optionally) stdev.

predict.WarpKriging *Predict with a WarpKriging model*

Description

Predict with a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
predict(  
  object,  
  x,  
  return_stdev = TRUE,  
  return_cov = FALSE,  
  return_deriv = FALSE,  
  ...  
)
```

Arguments

object	WarpKriging object
x	prediction matrix (m x d)
return_stdev	return standard deviations?
return_cov	return full covariance?
return_deriv	return derivatives of mean and stdev wrt x?
...	ignored

Value

list with mean, optionally stdev, cov, mean_deriv, stdev_deriv

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
X <- as.matrix(seq(0.05, 0.95, length.out = 10))
y <- f(X)

wk <- WarpKriging(y, X, warping = "kumaraswamy", kernel = "gauss", optim = "BFGS")
x <- as.matrix(seq(0, 1, length.out = 101))
p <- wk$predict(x, return_stdev = TRUE)

plot(f)
points(X, y, col = "blue", pch = 16)
lines(x, p$mean, col = "blue")
polygon(c(x, rev(x)), c(p$mean - 2 * p$stdev, rev(p$mean + 2 * p$stdev)),
        border = NA, col = rgb(0, 0, 1, 0.2))
```

<code>print.Kriging</code>	<i>Print the content of a Kriging object.</i>
----------------------------	---

Description

Print the content of a Kriging object.

Usage

```
## S3 method for class 'Kriging'
print(x, ...)
```

Arguments

x	A (S3) Kriging Object.
...	Ignored.

Value

String of printed object.

Author(s)

Yann Richet <yann.richet@asnr.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)

k <- Kriging(y, X, "matern3_2")

print(k)
## same thing
k
```

```
print.NestedKriging
```

Print a NestedKriging object.

Description

Print a NestedKriging object.

Usage

```
## S3 method for class 'NestedKriging'
print(x, ...)
```

Arguments

<code>x</code>	S3 NestedKriging object.
<code>...</code>	Ignored.

regmodel	<i>Get regression model type</i>
----------	----------------------------------

Description

Get regression model type

Usage

```
regmodel(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

regmodel.MLPKriging	<i>Get regression model type for an MLPKriging model</i>
---------------------	--

Description

Get regression model type for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
regmodel(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

regmodel.WarpKriging	<i>Get regression model type for a WarpKriging model</i>
----------------------	--

Description

Get regression model type for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
regmodel(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

save	<i>Save a Kriging Model inside a file. Back to base::save if argument is not a Kriging object.</i>
------	--

Description

Save a Kriging Model inside a file. Back to base::save if argument is not a Kriging object.

Usage

```
save(object = NULL, filename = NULL, ...)
```

Arguments

object	An object representing a model.
filename	A file to save the object.
...	Arguments used by base::save.

Author(s)

Yann Richet <yann.richet@asn.fr>

save.Kriging	<i>Save a Kriging Model to a file storage</i>
--------------	---

Description

Save a Kriging Model to a file storage

Usage

```
## S3 method for class 'Kriging'  
save(object, filename, ...)
```

Arguments

object	An S3 Kriging object.
filename	File name to save in.
...	Not used.

Value

The loaded Kriging object.

Author(s)

Yann Richet <yann.richet@asnr.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)  
set.seed(123)  
X <- as.matrix(runif(10))  
y <- f(X)  
  
k <- Kriging(y, X, kernel = "matern3_2", objective="LMP")  
print(k)  
  
outfile = tempfile("k.json")  
save(k,outfile)  
unlink(outfile)
```

save.MLPKriging	<i>Save an MLPKriging model to file</i>
-----------------	---

Description

Save an MLPKriging model to file

Usage

```
## S3 method for class 'MLPKriging'  
save(object, filename, ...)
```

Arguments

object	MLPKriging object
filename	path to save file
...	ignored

save.WarpKriging	<i>Save a WarpKriging model to file</i>
------------------	---

Description

Save a WarpKriging model to file

Usage

```
## S3 method for class 'WarpKriging'  
save(object, filename, ...)
```

Arguments

object	WarpKriging object
filename	path to save file
...	ignored

scaleX	<i>Get input scaling vector</i>
--------	---------------------------------

Description

Get input scaling vector

Usage

scaleX(object, ...)

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

scaleX.MLPKriging	<i>Get input scaling vector for an MLPKriging model</i>
-------------------	---

Description

Get input scaling vector for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
scaleX(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

scaleX.WarpKriging	<i>Get input scaling vector for a WarpKriging model</i>
--------------------	---

Description

Get input scaling vector for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
scaleX(object, ...)
```

Arguments

- object A Kriging/MLPKriging/WarpKriging model object.
- ... Unused.

scaleY	<i>Get output scaling value</i>
--------	---------------------------------

Description

Get output scaling value

Usage

```
scaleY(object, ...)
```

Arguments

- object A Kriging/MLPKriging/WarpKriging model object.
- ... Unused.

scaleY.MLPKriging	<i>Get output scaling value for an MLPKriging model</i>
-------------------	---

Description

Get output scaling value for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
scaleY(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

scaleY.WarpKriging	<i>Get output scaling value for a WarpKriging model</i>
--------------------	---

Description

Get output scaling value for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
scaleY(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

sigma2	<i>Get process variance</i>
--------	-----------------------------

Description

Get process variance

Usage

```
sigma2(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

Value

Numeric process variance estimate.

sigma2.WarpKriging	<i>Get process variance (concentrated MLE)</i>
--------------------	--

Description

Get process variance (concentrated MLE)

Usage

```
## S3 method for class 'WarpKriging'  
sigma2(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

simulate, KM-method	<i>Simulation from a KM Object</i>
---------------------	------------------------------------

Description

The simulate method is used to simulate paths from the kriging model described in object.

Usage

```
## S4 method for signature 'KM'
simulate(
  object,
  nsim = 1,
  seed = NULL,
  newdata,
  cond = TRUE,
  nugget.sim = 0,
  checkNames = FALSE,
  ...
)
```

Arguments

object	A KM object.
nsim	Integer: number of response vectors to simulate.
seed	Random seed.
newdata	Numeric matrix with it rows giving the points where the simulation is to be performed.
cond	Logical telling wether the simulation is conditional or not. Only TRUE is accepted for now.
nugget.sim	Numeric. A postive nugget effect used to avoid numerical instability.
checkNames	Check consistency between the design data X within object and newdata. The default is FALSE. XXXY Not used!!!
...	Ignored.

Details

Without a dedicated simulate method for the class "KM", this method would have been inherited from the "km" class. The dedicated method is expected to run faster. A comparison can be made by coercing a KM object to a km object with [as.km](#) before calling simulate.

Value

A numeric matrix with `nrow(newdata)` rows and `nsim` columns containing as its columns the simulated paths at the input points given in newdata.

XXX method simulate KM

Author(s)

Yann Richet <yann.richet@asnr.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
plot(f)
set.seed(123)
X <- as.matrix(runif(5))
y <- f(X)
points(X, y, col = 'blue')
k <- KM(design = X, response = y, covtype = "gauss")
x <- seq(from = 0, to = 1, length.out = 101)
s_x <- simulate(k, nsim = 3, newdata = x)
lines(x, s_x[, 1], col = 'blue')
lines(x, s_x[, 2], col = 'blue')
lines(x, s_x[, 3], col = 'blue')
```

simulate.Kriging

Simulation from a Kriging model object.

Description

This method draws paths of the stochastic process at new input points conditional on the values at the input points used in the fit.

Usage

```
## S3 method for class 'Kriging'
simulate(
  object,
  nsim = 1,
  seed = 123,
  x,
  with_noise = NULL,
  will_update = FALSE,
  ...
)
```

Arguments

object	S3 Kriging object.
nsim	Number of simulations to perform.
seed	Random seed used.
x	Points in model input space where to simulate.

with_noise	Logical or numeric specification controlling whether observation noise is included in the simulated paths. Use TRUE to include fitted noise when available, FALSE to exclude nugget noise, or NULL for the default behaviour.
will_update	Set to TRUE if you plan to call update_simulate() later.
...	Ignored.

Value

a matrix with `nrow(x)` rows and `nsim` columns containing the simulated paths at the inputs points given in `x`.

Note

The names of the formal arguments differ from those of the `simulate` methods for the S4 classes "km" and "KM". The formal `x` corresponds to `newdata`. These names are chosen **Python** and **Octave** interfaces to **libKriging**.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
plot(f)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)
points(X, y, col = "blue")

k <- Kriging(y, X, kernel = "matern3_2")

x <- seq(from = 0, to = 1, length.out = 101)
s <- simulate(k, nsim = 3, x = x)

lines(x, s[, 1], col = "blue")
lines(x, s[, 2], col = "blue")
lines(x, s[, 3], col = "blue")
```

simulate.MLPKriging	<i>Simulate from an MLPKriging model</i>
---------------------	--

Description

Simulate from an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
simulate(object, nsim = 1, seed = 123, x, will_update = FALSE, ...)
```

Arguments

object	MLPKriging object
nsim	number of simulations
seed	random seed
x	simulation matrix (m x d)
will_update	logical; if TRUE keep the internal state for a subsequent update_simulate() call.
...	ignored

Value

matrix (m x nsim)

simulate.WarpKriging *Simulate from a WarpKriging model*

Description

Simulate from a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
simulate(object, nsim = 1, seed = 123, x, will_update = FALSE, ...)
```

Arguments

object	WarpKriging object
nsim	number of simulations
seed	random seed
x	simulation matrix (m x d)
will_update	logical; if TRUE, cache data for update_simulate
...	ignored

Value

matrix (m x nsim)

theta	<i>Get GP range parameters</i>
-------	--------------------------------

Description

Get GP range parameters

Usage

theta(object, ...)

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

Value

Numeric vector of range parameters.

theta.WarpKriging	<i>Get GP range parameters</i>
-------------------	--------------------------------

Description

Get GP range parameters

Usage

```
## S3 method for class 'WarpKriging'  
theta(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

T_	<i>Get Cholesky factor T</i>
----	------------------------------

Description

Get Cholesky factor T

Usage

T_(object, ...)

Arguments

- object A Kriging/MLPKriging/WarpKriging model object.
- ... Unused.

T_.MLPKriging	<i>Get Cholesky factor T for an MLPKriging model</i>
---------------	--

Description

Get Cholesky factor T for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
T_(object, ...)
```

Arguments

- object A Kriging/MLPKriging/WarpKriging model object.
- ... Unused.

T_.WarpKriging	<i>Get Cholesky factor T for a WarpKriging model</i>
----------------	--

Description

Get Cholesky factor T for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'
T_(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

update, KM-method	<i>Update a KM Object with New Points</i>
-------------------	---

Description

The update method is used when new observations are added to a fitted kriging model. Rather than fitting the model from scratch with the updated observations added, the results of the fit as stored in object are used to achieve some savings.

Usage

```
## S4 method for signature 'KM'
update(
  object,
  newX,
  newy,
  newX.alreadyExist = FALSE,
  cov.reestim = TRUE,
  trend.reestim = cov.reestim,
  nugget.reestim = FALSE,
  newnoise.var = NULL,
  kmcontrol = NULL,
  newF = NULL,
  ...
)
```

Arguments

<code>object</code>	A KM object.
<code>newX</code>	A numeric matrix containing the new design points. It must have <code>object@</code> columns in correspondence with those of the design matrix used to fit the model which is stored as <code>object@X</code> .
<code>newy</code>	A numeric vector of new response values, in correspondence with the rows of <code>newX</code> .
<code>newX.alreadyExist</code>	Logical. If TRUE, <code>newX</code> can contain some input points that are already in <code>object@X</code> .
<code>cov.reestim</code>	Logical. If TRUE, the vector <code>theta</code> of correlation ranges will be re-estimated using the new observations as well as the observations already used when fitting <code>object</code> . Only TRUE can be used for now.
<code>trend.reestim</code>	Logical. If TRUE the vector <code>beta</code> of trend coefficients will be re-estimated using all the observations. Only TRUE can be used for now.
<code>nugget.reestim</code>	Logical. If TRUE the nugget effect will be re-estimated using all the observations. Only FALSE can be used for now.
<code>newnoise.var</code>	Optional variance of an additional noise on the new response.
<code>kmcontrol</code>	A list of options to tune the fit. Not available yet.
<code>newF</code>	New trend matrix. XXXY?
<code>...</code>	Ignored.

Details

Without a dedicated update method for the class "KM", this would have been inherited from the class "km". The dedicated method is expected to run faster. A comparison can be made by coercing a KM object to a km object with [as.km](#) before calling `update`.

Value

The updated KM object.

Author(s)

Yann Richet <yann.richet@asn.fr>

See Also

[as.km](#) to coerce a KM object to the class "km".

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
plot(f)
set.seed(123)
X <- as.matrix(runif(5))
y <- f(X)
```

```

points(X, y, col = "blue")
KMobj <- KM(design = X, response = y, covtype = "gauss")
x <- seq(from = 0, to = 1, length.out = 101)
p_x <- predict(KMobj, x)
lines(x, p_x$mean, col = "blue")
lines(x, p_x$lower95, col = "blue")
lines(x, p_x$upper95, col = "blue")
newX <- as.matrix(runif(3))
newy <- f(newX)
points(newX, newy, col = "red")

## replace the object by its updated version
KMobj <- update(KMobj, newX=newX, newy=newy)

x <- seq(from = 0, to = 1, length.out = 101)
p2_x <- predict(KMobj, x)
lines(x, p2_x$mean, col = "red")
lines(x, p2_x$lower95, col = "red")
lines(x, p2_x$upper95, col = "red")

```

update.Kriging

Update a Kriging model object with new points

Description

Update a Kriging model object with new points

Usage

```

## S3 method for class 'Kriging'
update(object, y_u, ..., X_u = NULL, noise_u = NULL, refit = TRUE)

```

Arguments

object	S3 Kriging object.
y_u	Numeric vector of new responses (output).
...	Ignored.
X_u	Numeric matrix of new input points.
noise_u	Optional numeric vector of observation noise variances attached to y_u.
refit	Logical. If TRUE the model is refitted (default is TRUE).

Value

No return value. Kriging object argument is modified.

Caution

The method *does not return the updated object*, but instead changes the content of object. This behaviour is quite unusual in R and differs from the behaviour of `update.km` in **DiceKriging** and the update method for class KM.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1- 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x)*x^5 + 0.7)
plot(f)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)
points(X, y, col = "blue")

k <- Kriging(y, X, "matern3_2")

x <- seq(from = 0, to = 1, length.out = 101)
p <- predict(k, x)
lines(x, p$mean, col = "blue")
polygon(c(x, rev(x)), c(p$mean - 2 * p$stdev, rev(p$mean + 2 * p$stdev)),
  border = NA, col = rgb(0, 0, 1, 0.2))

X_u <- as.matrix(runif(3))
y_u <- f(X_u)
points(X_u, y_u, col = "red")

## change the content of the object 'k'
update(k, y_u, X_u)

## include design points to see interpolation
x <- sort(c(X,X_u,seq(from = 0, to = 1, length.out = 101)))
p2 <- predict(k, x)
lines(x, p2$mean, col = "red")
polygon(c(x, rev(x)), c(p2$mean - 2 * p2$stdev, rev(p2$mean + 2 * p2$stdev)),
  border = NA, col = rgb(1, 0, 0, 0.2))
```

update.MLPKriging

Update an MLPKriging model with new observations

Description

Update an MLPKriging model with new observations

Usage

```
## S3 method for class 'MLPKriging'  
update(object, y_u, X_u, refit = TRUE, ...)
```

Arguments

object	MLPKriging object
y_u	new observations
X_u	new input matrix
refit	Logical. If TRUE the model is refitted (default is TRUE).
...	ignored

update.WarpKriging	<i>Update a WarpKriging model with new observations</i>
--------------------	---

Description

Update a WarpKriging model with new observations

Usage

```
## S3 method for class 'WarpKriging'  
update(object, y_u, X_u, refit = TRUE, ...)
```

Arguments

object	WarpKriging object
y_u	new observations
X_u	new input matrix
refit	logical; if TRUE (default), re-optimize hyperparameters
...	ignored

update_simulate	<i>Update simulation of model on data.</i>
-----------------	--

Description

Update previous simulate of a model given in object.

Usage

```
update_simulate(object, ...)
```

Arguments

object	An object representing a fitted model.
...	Further arguments of function

Value

Updated simulation of model output.

update_simulate.Kriging	<i>Update previous simulation of a Kriging model object.</i>
-------------------------	--

Description

This method draws paths of the stochastic process conditional on the values at the input points used in the fit, plus the new input points and their values given as argument (known as 'update' points).

Usage

```
## S3 method for class 'Kriging'
update_simulate(object, y_u, ..., X_u = NULL, noise_u = NULL)
```

Arguments

object	S3 Kriging object.
y_u	Numeric vector of new responses (output).
...	Ignored.
X_u	Numeric matrix of new input points.
noise_u	Optional numeric vector of observation noise variances attached to y_u.

Value

a matrix with `nrow(x)` rows and `nsim` columns containing the simulated paths at the inputs points given in `x`.

Author(s)

Yann Richet <yann.richet@asn.fr>

Examples

```
f <- function(x) 1 - 1 / 2 * (sin(12 * x) / (1 + x) + 2 * cos(7 * x) * x^5 + 0.7)
plot(f)
set.seed(123)
X <- as.matrix(runif(10))
y <- f(X)
points(X, y, col = "blue")

k <- Kriging(y, X, kernel = "matern3_2")

x <- seq(from = 0, to = 1, length.out = 101)
s <- k$simulate(nsim = 3, x = x, will_update = TRUE)

lines(x, s[, 1], col = "blue")
lines(x, s[, 2], col = "blue")
lines(x, s[, 3], col = "blue")

X_u <- as.matrix(runif(3))
y_u <- f(X_u)
points(X_u, y_u, col = "red")

su <- k$update_simulate(y_u, X_u)

lines(x, su[, 1], col = "blue", lty=2)
lines(x, su[, 2], col = "blue", lty=2)
lines(x, su[, 3], col = "blue", lty=2)
```

update_simulate.MLPKriging

Update simulated paths with new observations (FOXY algorithm)

Description

Update simulated paths with new observations (FOXY algorithm)

Usage

```
## S3 method for class 'MLPKriging'
update_simulate(object, y_u, X_u, ...)
```


Arguments

object	MLPKriging object (must have called simulate with will_update=TRUE)
y_u	new observations
X_u	new input matrix
...	ignored

Value

matrix (m x nsim) of updated simulated paths

update_simulate.WarpKriging

Update simulated paths with new observations (FOXY algorithm)

Description

Update simulated paths with new observations (FOXY algorithm)

Usage

```
## S3 method for class 'WarpKriging'  
update_simulate(object, y_u, X_u, ...)
```

Arguments

object	WarpKriging object (must have called simulate with will_update=TRUE)
y_u	new observations
X_u	new input matrix
...	ignored

Value

matrix (m x nsim) of updated simulated paths

warping	<i>Get warping specifications as strings</i>
---------	--

Description

Get warping specifications as strings

Usage

```
warping(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

warping.WarpKriging	<i>Get warping specification for a WarpKriging model</i>
---------------------	--

Description

Get warping specification for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
warping(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

WarpKriging

*Create a WarpKriging model***Description**

Kriging with per-variable input warping. Each input dimension is independently transformed before the GP kernel is evaluated. Supports continuous, categorical, ordinal variables and joint deep kernel learning.

Usage

```
WarpKriging(
  y,
  X,
  warping,
  kernel = "gauss",
  regmodel = "constant",
  normalize = FALSE,
  optim = "BFGS+Adam",
  objective = "LL",
  parameters = NULL,
  noise = NULL
)
```

Arguments

y	numeric vector of observations (n)
X	numeric matrix of inputs (n x d)
warping	character vector of warp specifications, one per column of X. Use warp_*() helpers or plain strings: "none", "affine", "boxcox", "kumaraswamy", "neural_mono(8)", "mlp(16:8,2,selu)", "knots(3)", "knots(0.25:0.5:0.75)", "categorical(5,2)", "ordinal(4)".
kernel	covariance kernel: "gauss", "matern3_2", "matern5_2", "exp"
regmodel	trend: "constant", "linear", "quadratic"
normalize	logical; normalise continuous inputs?
optim	optimiser (currently only one bi-level strategy)
objective	"LL" (log-likelihood)
parameters	optional named list of tuning parameters, e.g. list(max_iter_adam = "300", adam_lr = "0.001", max_iter_bfgs = "50")
noise	Either a numeric vector of per-observation noise variances, or "nugget" to estimate a homogeneous nugget, or NULL (default) for noise-free interpolation.

Value

An S3 object of class "WarpKriging".

Examples

```
# Continuous with Kumaraswamy warping
X <- as.matrix(c(0.0, 0.25, 0.5, 0.75, 1.0))
f <- function(x) 1 - 1/2 * (sin(12*x)/(1+x) + 2*cos(7*x)*x^5 + 0.7)
y <- f(X)
k <- WarpKriging(y, X, warping = "kumaraswamy", kernel = "gauss")
print(k)

# Mixed: 1 continuous + 1 categorical (3 levels)
n <- 15
X_mix <- cbind(runif(n), rep(0:2, length.out = n))
y_mix <- sin(2 * pi * X_mix[, 1]) * c(1, 2, 0.5)[X_mix[, 2] + 1]
k2 <- WarpKriging(y_mix, X_mix,
  warping = c("mlp(16:8,2,selu)", "categorical(3,2)"),
  kernel = "matern5_2")
```

warp_affine	<i>Affine warping: $w(x) = a \cdot x + b$</i>
-------------	--

Description

Affine warping: $w(x) = a \cdot x + b$

Usage

```
warp_affine()
```

Value

string "affine"

warp_boxcox	<i>Box-Cox warping</i>
-------------	------------------------

Description

Box-Cox warping

Usage

```
warp_boxcox()
```

Value

string "boxcox"

warp_categorical	<i>Categorical embedding</i>
------------------	------------------------------

Description

Categorical embedding

Usage

```
warp_categorical(n_levels, embed_dim = 2)
```

Arguments

n_levels	number of levels (integer-coded 0..n_levels-1)
embed_dim	embedding dimensionality (default 2)

Value

string e.g. "categorical(5,2)"

warp_knots	<i>Piecewise-linear monotone warping with knots (Xiong et al. 2007)</i>
------------	---

Description

Implements the non-stationary input warping of Xiong, Chen, Apley & Ding (2007, *Int. J. Numer. Meth. Engng*). The domain $[0, 1]$ is split into $K + 1$ intervals by K interior knots, each carrying a learnable positive slope $s_k = \exp(r_k)$. The warp is:

$$w(x) = \sum_{j < k} s_j(t_{j+1} - t_j) + s_k(x - t_k)$$

for $x \in [t_k, t_{k+1})$, giving a continuous, monotone piecewise-linear function with $K + 1$ free parameters.

This is the same construction as the knots argument in **DiceKriging**.

Usage

```
warp_knots(n_knots = 3, knot_positions = NULL)
```

Arguments

n_knots	number of interior knots $K \geq 1$ (default 3)
knot_positions	optional numeric vector of K knot positions strictly inside $(0, 1)$, in increasing order. When NULL (default), knots are placed uniformly at $1/(K + 1), 2/(K + 1), \dots, K/(K + 1)$.

Value

warp specification string, e.g. "knots(3)" or "knots(0.25:0.5:0.75)"

References

Xiong, Y., Chen, W., Apley, D. & Ding, X. (2007). A non-stationary covariance-based Kriging method for metamodeling in engineering design. *International Journal for Numerical Methods in Engineering*, 71(6), 733–756.

See Also

[WarpKriging](#)

warp_kumaraswamy	<i>Kumaraswamy CDF warping on [0,1]</i>
------------------	---

Description

Kumaraswamy CDF warping on [0,1]

Usage

warp_kumaraswamy()

Value

string "kumaraswamy"

warp_mlp	<i>Per-variable MLP warping (unconstrained, multi-dim output)</i>
----------	---

Description

Per-variable MLP warping (unconstrained, multi-dim output)

Usage

warp_mlp(hidden_dims, d_out = 2, activation = "selu")

Arguments

hidden_dims	integer vector of hidden layer sizes
d_out	output dimensionality (default 2)
activation	activation: "relu","selu","tanh","sigmoid","elu"

Value

string e.g. "mlp(16:8,3,selu)"

warp_neural_mono	<i>Monotone neural network warping</i>
------------------	--

Description

Monotone neural network warping

Usage

warp_neural_mono(n_hidden = 8)

Arguments

n_hidden number of hidden units (default 8)

Value

string e.g. "neural_mono(16)"

warp_none	<i>No warping (identity)</i>
-----------	------------------------------

Description

No warping (identity)

Usage

warp_none()

Value

string "none"

warp_ordinal	<i>Ordinal warping (learned ordered positions)</i>
--------------	--

Description

Ordinal warping (learned ordered positions)

Usage

warp_ordinal(n_levels)

Arguments

n_levels number of ordered levels (0..n_levels-1)

Value

string e.g. "ordinal(4)"

X	<i>Get training input matrix</i>
---	----------------------------------

Description

Get training input matrix

Usage

X(object, ...)

Arguments

object A fitted model object
... ignored

Value

matrix of training inputs

X.MLPKriging	<i>Get training input matrix</i>
--------------	----------------------------------

Description

Get training input matrix

Usage

```
## S3 method for class 'MLPKriging'  
X(object, ...)
```

Arguments

object	MLPKriging object
...	ignored

Value

matrix of training inputs

y	<i>Get training output vector</i>
---	-----------------------------------

Description

Get training output vector

Usage

```
y(object, ...)
```

Arguments

object	A fitted model object
...	ignored

Value

vector of training outputs

y.MLPKriging	<i>Get training output vector</i>
--------------	-----------------------------------

Description

Get training output vector

Usage

```
## S3 method for class 'MLPKriging'
y(object, ...)
```

Arguments

object	MLPKriging object
...	ignored

Value

vector of training outputs

z	<i>Get whitened residuals z</i>
---	---------------------------------

Description

Get whitened residuals z

Usage

```
z(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

z.MLPKriging	<i>Get whitened residuals z for an MLPKriging model</i>
--------------	---

Description

Get whitened residuals z for an MLPKriging model

Usage

```
## S3 method for class 'MLPKriging'  
z(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

z.WarpKriging	<i>Get whitened residuals z for a WarpKriging model</i>
---------------	---

Description

Get whitened residuals z for a WarpKriging model

Usage

```
## S3 method for class 'WarpKriging'  
z(object, ...)
```

Arguments

object	A Kriging/MLPKriging/WarpKriging model object.
...	Unused.

Index

activation, 5
activation.MLPKriging, 5
as.km, 6, 54, 68, 75
as.km, Kriging, Kriging-method
 (as.km.Kriging), 6
as.km.Kriging, 6
as.list, Kriging, Kriging-method
 (as.list.Kriging), 7
as.list.Kriging, 7

beta, 8
beta.MLPKriging, 8
beta.WarpKriging, 9

centerX, 9
centerX.MLPKriging, 10
centerX.WarpKriging, 10
centerY, 11
centerY.MLPKriging, 11
centerY.WarpKriging, 12
classKriging, 12
classMLPKriging, 13
classNestedKriging, 13
classWarpKriging, 14
copy, 14
copy, Kriging, Kriging-method
 (copy.Kriging), 15
copy.Kriging, 15
copy.MLPKriging, 15
copy.WarpKriging, 16
covMat, 17
covMat, Kriging, Kriging-method
 (covMat.Kriging), 17
covMat.Kriging, 17

F_, 23
F_.MLPKriging, 24
F_.WarpKriging, 24
feature_dim, 18
feature_dim.MLPKriging, 18

feature_dim.WarpKriging, 19
fit, 19
fit.Kriging, 20
fit.MLPKriging, 21
fit.WarpKriging, 22

hidden_dims, 25
hidden_dims.MLPKriging, 25

is_fitted, 26
is_fitted.MLPKriging, 26
is_fitted.WarpKriging, 27

kernel, 27
kernel.WarpKriging, 28
KM, 28, 31, 51, 53
km, 30
KM-class, 30
Kriging, 31, 49

leaveOneOut, 33
leaveOneOut, Kriging, Kriging-method
 (leaveOneOut.Kriging), 33
leaveOneOut.Kriging, 33
leaveOneOutFun, 34
leaveOneOutFun, Kriging, Kriging-method
 (leaveOneOutFun.Kriging), 34
leaveOneOutFun.Kriging, 34
leaveOneOutVec, 35
leaveOneOutVec, Kriging, Kriging-method
 (leaveOneOutVec.Kriging), 36
leaveOneOutVec.Kriging, 36
load, 37
load.Kriging, 38
load.MLPKriging, 39
load.WarpKriging, 39
logLikelihood, 40
logLikelihood, Kriging, Kriging-method
 (logLikelihood.Kriging), 40
logLikelihood.Kriging, 40

- logLikelihood.WarpKriging, 41
- logLikelihoodFun, 41
- logLikelihoodFun.Kriging, Kriging-method
(logLikelihoodFun.Kriging), 42
- logLikelihoodFun.Kriging, 42
- logLikelihoodFun.MLPKriging, 43
- logLikelihoodFun.WarpKriging, 43
- logMargPost, 44
- logMargPost.Kriging, Kriging-method
(logMargPost.Kriging), 44
- logMargPost.Kriging, 44
- logMargPostFun, 45
- logMargPostFun.Kriging, Kriging-method
(logMargPostFun.Kriging), 46
- logMargPostFun.Kriging, 46
- M, 47
- M.MLPKriging, 47
- M.WarpKriging, 48
- MLPKriging, 48
- NestedKriging, 49
- NoiseKM, 51
- normalize, 51
- normalize.MLPKriging, 52
- normalize.WarpKriging, 52
- NuggetKM, 53
- predict, KM-method, 53
- predict.Kriging, 55
- predict.MLPKriging, 56
- predict.NestedKriging, 57
- predict.WarpKriging, 57
- print.Kriging, 58
- print.NestedKriging, 59
- regmodel, 60
- regmodel.MLPKriging, 60
- regmodel.WarpKriging, 61
- rgasp, 46
- save, 61
- save.Kriging, Kriging-method
(save.Kriging), 62
- save.Kriging, 62
- save.MLPKriging, 63
- save.WarpKriging, 63
- scaleX, 64
- scaleX.MLPKriging, 64
- scaleX.WarpKriging, 65
- scaleY, 65
- scaleY.MLPKriging, 66
- scaleY.WarpKriging, 66
- sigma2, 67
- sigma2.WarpKriging, 67
- simulate, KM-method, 68
- simulate.Kriging, 69
- simulate.MLPKriging, 70
- simulate.WarpKriging, 71
- T_, 73
- T_.MLPKriging, 73
- T_.WarpKriging, 74
- theta, 72
- theta.WarpKriging, 72
- update, KM-method, 74
- update.km, 77
- update.Kriging, 76
- update.MLPKriging, 77
- update.WarpKriging, 78
- update_simulate, 79
- update_simulate.Kriging, 79
- update_simulate.MLPKriging, 80
- update_simulate.WarpKriging, 81
- warp_affine, 84
- warp_boxcox, 84
- warp_categorical, 85
- warp_knots, 85
- warp_kumaraswamy, 86
- warp_mlp, 86
- warp_neural_mono, 87
- warp_none, 87
- warp_ordinal, 88
- warping, 82
- warping.WarpKriging, 82
- WarpKriging, 83, 86
- X, 88
- X.MLPKriging, 89
- y, 89
- y.MLPKriging, 90
- z, 90
- z.MLPKriging, 91
- z.WarpKriging, 91