Package: autoFRK (via r-universe)

October 29, 2026

Type Package

Title Automatic Fixed Rank Kriging

Description Automatic fixed rank kriging for (irregularly located) spatial data using a class of basis functions with multi-resolution features and ordered in terms of their resolutions. The model parameters are estimated by maximum likelihood (ML) and the number of basis functions is determined by Akaike's information criterion (AIC). For spatial data with either one realization or independent replicates, the ML estimates and AIC are efficiently computed using their closed-form expressions when no missing value occurs. Details regarding the basis function construction, parameter estimation, and AIC calculation can be found in Tzeng and Huang (2018) <doi:10.1080/00401706.2017.1345701>. For data with missing values, the ML estimates are obtained using the expectation- maximization algorithm. Apart from the number of basis functions, there are no other tuning parameters, making the method fully automatic. Users can also include a stationary structure in the spatial covariance, which utilizes 'LatticeKrig' package.

License GPL (>= 2)

Version 1.4.4

Date 2026-01-15

Depends R (>= 3.5.0), spam

Imports fields (>= 6.9.1), filehashSQLite, filehash, MASS, mgcv, LatticeKrig (>= 5.4), FNN, filematrix, Rcpp, methods

LinkingTo Rcpp, RSpectra, RcppEigen, RcppParallel

Author ShengLi Tzeng [aut, cre], Hsin-Cheng Huang [aut], Wen-Ting Wang [ctb], Douglas Nychka [ctb]

Maintainer ShengLi Tzeng <slt.cmu@gmail.com>

NeedsCompilation yes

RoxygenNote 7.1.1

Repository https://cran.r-universe.dev

Date/Publication 2026-01-17 09:35:37 UTC

RemoteUrl https://github.com/cran/autoFRK

RemoteRef HEAD

RemoteSha d7274954ee4408ca7975efaea54040407603750a

Contents

autoFRK 2
mrts 5
predict.FRK 6
predict.mrts 8

Index **9**

`autoFRK` *Automatic Fixed Rank Kriging*

Description

This function performs resolution adaptive fixed rank kriging based on spatial data observed at one or multiple time points via the following spatial random-effects model:

$$z[t] = \mu + G \cdot w[t] + \eta[t] + e[t], w[t] \sim N(0, M), e[t] \sim N(0, s \cdot D); t = 1, ..., T,$$

where $z[t]$ is an n-vector of (partially) observed data at n locations, μ is an n-vector of deterministic mean values, D is a given n by n matrix, G is a given n by K matrix, $\eta[t]$ is an n-vector of random variables corresponding to a spatial stationary process, and $w[t]$ is a K-vector of unobservable random weights. Parameters are estimated by maximum likelihood in a closed-form expression. The matrix G corresponding to basis functions is given by an ordered class of thin-plate spline functions, with the number of basis functions selected by Akaike's information criterion.

Usage

```
autoFRK(
  Data,
  loc,
  mu = 0,
  D = diag.spam(NROW(Data)),
  G = NULL,
  finescale = FALSE,
  maxit = 50,
  tolerance = 0.1^6,
  maxK = NULL,
  Kseq = NULL,
  method = c("fast", "EM"),
```

```
  n.neighbor = 3,
  maxknot = 5000
)
```

Arguments

Data	*n* by *T* data matrix (NA allowed) with $z[t]$ as the *t*-th column.
loc	*n* by *d* matrix of coordinates corresponding to *n* locations.
mu	*n*-vector or scalar for μ; Default is 0.
D	*n* by *n* matrix (preferably sparse) for the covariance matrix of the measurement errors up to a constant scale. Default is an identity matrix.
G	*n* by *K* matrix of basis function values with each column being a basis function taken values at loc. Default is NULL, which is automatic determined.
finescale	logical; if TRUE then a (approximate) stationary finer scale process $\eta[t]$ will be included based on LatticeKrig pacakge. In such a case, only the diagonals of D would be taken into account. Default is FALSE.
maxit	maximum number of iterations. Default is 50.
tolerance	precision tolerance for convergence check. Default is 0.1^6.
maxK	maximum number of basis functions considered. Default is $10 \cdot \sqrt{n}$ for *n>100* or *n* for *n<=100*.
Kseq	user-specified vector of numbers of basis functions considered. Default is NULL, which is determined from maxK.
method	"fast" or " EM"; if "fast" then the missing data are filled in using k-nearest-neighbor imputation; if "EM" then the missing data are taken care by the EM algorithm. Default is "fast".
n.neighbor	number of neighbors to be used in the "fast" imputation method. Default is 3.
maxknot	maximum number of knots to be used in generating basis functions. Default is 5000.

Details

The function computes the ML estimate of M using the closed-form expression in Tzeng and Huang (2018). If the user would like to specify a D other than an identity matrix for a large *n*, it is better to provided via spam function in spam package.

Value

an object of class FRK is returned, which is a list of the following components:

M	ML estimate of M.
s	estimate for the scale parameter of measurement errors.
negloglik	negative log-likelihood.
w	*K* by *T* matrix with $w[t]$ as the *t*-th column.
V	*K* by *K* matrix of the prediction error covariance matrix of $w[t]$.
G	user specified basis function matrix or an automatically generated mrts object.
LKobj	a list from calling LKrig.MLE in LatticeKrig package if useLK=TRUE; otherwise NULL. See that package for details.

Author(s)

ShengLi Tzeng and Hsin-Cheng Huang.

References

Tzeng, S., & Huang, H.-C. (2018). Resolution Adaptive Fixed Rank Kriging, Technometrics, https://doi.org/10.1080/00401706.2017.1345701.

See Also

predict.FRK

Examples

```
#### generating data from two eigenfunctions
originalPar <- par(no.readonly = TRUE)
set.seed(0)
n <- 150
s <- 5
grid1 <- grid2 <- seq(0, 1, l = 30)
grids <- expand.grid(grid1, grid2)
fn <- matrix(0, 900, 2)
fn[, 1] <- cos(sqrt((grids[, 1] - 0)^2 + (grids[, 2] - 1)^2) * pi)
fn[, 2] <- cos(sqrt((grids[, 1] - 0.75)^2 + (grids[, 2] - 0.25)^2) * 2 * pi)

#### single realization simulation example
w <- c(rnorm(1, sd = 5), rnorm(1, sd = 3))
y <- fn %*% w
obs <- sample(900, n)
z <- y[obs] + rnorm(n) * sqrt(s)
X <- grids[obs, ]

#### method1: automatic selection and prediction
one.imat <- autoFRK(Data = z, loc = X, maxK = 15)
yhat <- predict(one.imat, newloc = grids)

#### method2: user-specified basis functions
G <- mrts(X, 15)
Gpred <- predict(G, newx = grids)
one.usr <- autoFRK(Data = z, loc = X, G = G)
yhat2 <- predict(one.usr, newloc = grids, basis = Gpred)

require(fields)
par(mfrow = c(2, 2))
image.plot(matrix(y, 30, 30), main = "True")
points(X, cex = 0.5, col = "grey")
image.plot(matrix(yhat$pred.value, 30, 30), main = "Predicted")
points(X, cex = 0.5, col = "grey")
image.plot(matrix(yhat2$pred.value, 30, 30), main = "Predicted (method 2)")
points(X, cex = 0.5, col = "grey")
plot(yhat$pred.value, yhat2$pred.value, mgp = c(2, 0.5, 0))
par(originalPar)
```

```
#### end of single realization simulation example

#### independent multi-realization simulation example
set.seed(0)
wt <- matrix(0, 2, 20)
for (tt in 1:20) wt[, tt] <- c(rnorm(1, sd = 5), rnorm(1, sd = 3))
yt <- fn %*% wt
obs <- sample(900, n)
zt <- yt[obs, ] + matrix(rnorm(n * 20), n, 20) * sqrt(s)
X <- grids[obs, ]
multi.imat <- autoFRK(Data = zt, loc = X, maxK = 15)
Gpred <- predict(multi.imat$G, newx = grids)

G <- multi.imat$G
Mhat <- multi.imat$M
dec <- eigen(G %*% Mhat %*% t(G))
fhat <- Gpred %*% Mhat %*% t(G) %*% dec$vector[, 1:2]
par(mfrow = c(2, 2))
image.plot(matrix(fn[, 1], 30, 30), main = "True Eigenfn 1")
image.plot(matrix(fn[, 2], 30, 30), main = "True Eigenfn 2")
image.plot(matrix(fhat[, 1], 30, 30), main = "Estimated Eigenfn 1")
image.plot(matrix(fhat[, 2], 30, 30), main = "Estimated Eigenfn 2")
par(originalPar)
#### end of independent multi-realization simulation example
```

`mrts` *Multi-Resolution Thin-plate Spline Basis Functions*

Description

This function generates multi-resolution thin-plate spline basis functions. The basis functions are (descendingly) ordered in terms of their degrees of smoothness with a higher-order function corresponding to larger-scale features and a lower-order one corresponding to smaller-scale details. They are useful in the spatio-temporal random effects model.

Usage

```
mrts(knot, k, x = NULL, maxknot = 5000)
```

Arguments

`knot`	m by d matrix ($d<=3$) for m locations of d-dimensional knots as in ordinary splines. Missing values are not allowed.
`k`	the number ($<=m$) of basis functions.
`x`	n by d matrix of coordinates corresponding to n locations where the values of basis functions to be evaluated. Default is `NULL`, which uses the m by d matrix in `knot`.
`maxknot`	maximum number of knots to be used in generating basis functions. If `maxknot` $< m$, a deterministic subset selection of knots will be used. For using all knots, set `maxknot`$>=m$.

Value

An `mrts` object is generated. If `x=NULL` (default) it returns an m by k matrix of k basis function taken values at knots. With `x` given, it returns n by k matrix for basis functions taken values at `x`.

Author(s)

ShengLi Tzeng and Hsin-Cheng Huang.

References

Tzeng, S., & Huang, H. C. (2018). Resolution Adaptive Fixed Rank Kriging. Technometrics, https://doi.org/10.1080/00401706.2017.1345701. Tzeng, S., & Huang, H. C. (2015). Multi-Resolution Spatial Random-Effects Models for Irregularly Spaced Data. arXiv preprint arXiv:1504.05659.

See Also

`predict.mrts`

Examples

```
originalPar <- par(no.readonly = TRUE)
knot <- seq(0, 1, l = 30)
b <- mrts(knot, 30)
x0 <- seq(0, 1, l = 200)
bx <- predict(b, x0)
par(mfrow = c(5, 6), mar = c(0, 0, 0, 0))
for (i in 1:30) {
  plot(bx[, i], type = "l", axes = FALSE)
  box()
}
par(originalPar)
```

---

predict.FRK *Predict method for Fixed Rank Kriging*

Description

Predicted values and estimate of standard errors based on an "`autoFRK`" model object.

Usage

```
## S3 method for class 'FRK'
predict(
  object,
  obsData = NULL,
  obsloc = NULL,
  mu.obs = 0,
  newloc = obsloc,
```

```
  basis = NULL,
  mu.new = 0,
  se.report = FALSE,
  ...
)
```

Arguments

object	a model object obtained from "autoFRK".
obsData	a vector with observed data used for prediction. Default is NULL, which uses the Data input from object.
obsloc	a matrix with rows being coordinates of observation locations for obsData. Only object using mrts basis functions can have obsloc different from the loc input of object; not applicable for user-specified basis functions. Default is NULL, which uses the loc input of object.
mu.obs	a vector or scalar for the deterministic mean values at obsloc. Default is 0.
newloc	a matrix with rows being coordinates of new locations for prediction. Default is NULL, which gives prediction at the locations of the observed data.
basis	a matrix with each column being a basis function taken values at newloc. It can be omitted if object was fitted using default mrts basis functions.
mu.new	a vector or scalar for the deterministic mean values at newloc. Default is 0.
se.report	logical; if TRUE then the standard error of prediction is reported.
...	not used but needed for the S3 generic/method compatibility.

Value

A list with the components described below.

pred.value	a matrix with the *(i,t)* element being the predicted value at *i*-th location and time *t*.
se	a vector with the *i*-th element being the standard error of the predicted value at the *i*-th location.

Author(s)

ShengLi Tzeng and Hsin-Cheng Huang.

See Also

autoFRK

predict.mrts *Multi-Resolution Thin-plate Spline Basis Functions*

Description

Evaluate multi-resolution thin-plate spline basis functions at given locations. This function provides a generic prediction method for `mrts` objects, in a similar way as `predict.ns` and `predict.bs` in `splines` package.

Usage

```
## S3 method for class 'mrts'
predict(object, newx, ...)
```

Arguments

`object`	object produced from calling mrts.
`newx`	an *n* by *d* matrix of coordinates corresponding to *n* locations.
`...`	not used but needed for the S3 generic/method compatibility.

Value

an *n* by *k* matrix of the *k* basis function in `object` taken values at `newx`.

Author(s)

ShengLi Tzeng and Hsin-Cheng Huang.

See Also

`mrts`

Index

autoFRK, 2, *7*

mrts, 5, *8*

predict.FRK, *4*, 6
predict.mrts, *6*, 8