Package ‘kfino’

November 3, 2022

Title Kalman Filter for Impulse Noised Outliers

Version 1.0.0

Author Bertrand Cloez [aut],
Isabelle Sanchez [aut, cre],
Benedicte Fontez [ctr]

Maintainer Isabelle Sanchez <isabelle.sanchez@inrae.fr>

Description A method for detecting outliers with a Kalman filter on impulsed noised outliers and prediction on cleaned data. 'kfino' is a robust sequential algorithm allowing to filter data with a large number of outliers. This algorithm is based on simple latent linear Gaussian processes as in the Kalman Filter method and is devoted to detect impulse-noised outliers. These are data points that differ significantly from other observations. 'ML' (Maximization Likelihood) and 'EM' (Expectation-Maximization algorithm) algorithms were implemented in 'kfino'. The method is described in full details in the following arXiv e-Print: <arXiv:2208.00961>.

License GPL-3

Depends R (>= 4.1.0)

Encoding UTF-8

LazyData TRUE

URL https://forgemia.inra.fr/isabelle.sanchez/kfino

BugReports https://forgemia.inra.fr/isabelle.sanchez/kfino/-/issues

Imports ggplot2, dplyr,

Suggests rmarkdown, knitr, testthat (>= 3.0.0), covr, foreach,
doParallel, parallel

VignetteBuilder knitr

RoxygenNote 7.2.1

Config/testthat/edition 3

NeedsCompilation no

Repository CRAN

Date/Publication 2022-11-03 08:26:44 UTC

R topics documented:

---

doutlier — *doutlier defines an outlier distribution (Surface of a trapezium) and uses input parameters given in the main function kfino_fit()*

Description

doutlier defines an outlier distribution (Surface of a trapezium) and uses input parameters given in the main function kfino_fit()

Usage

```
doutlier(y, K, expertMin, expertMax)
```

Arguments

`y`	numeric, point
`K`	numeric, constant value
`expertMin`	numeric, the minimal weight expected by the user
`expertMax`	numeric, the maximal weight expected by the user

Details

this function is used to calculate an outlier distribution following a trapezium shape. $y \mapsto \mathrm{doutlier}(y, K, \mathrm{expertMin}, \mathrm{expertMa}$
is the probability density function on $[\mathrm{expertMin}, \mathrm{expertMax}]$ which is linear and verifies $\mathrm{doutlier}(\mathrm{expertMax}, K, \mathrm{expertMin}, \mathrm{e}$
$K * \mathrm{doutlier}(\mathrm{expertMin}, K, \mathrm{expertMin}, \mathrm{expertMax})$. In particular, when $K=1$ this corresponds to the uniform distribution.

Value

a numeric value

Examples

```
doutlier(2,5,10,45)
```

kfino *kfino: Kalman Filtering*

Description

A method for detecting outliers with a Kalman filter on impulsed noised outliers and prediction on cleaned data. 'kfino' is a robust sequential algorithm allowing to filter data with a large number of outliers. This algorithm is based on simple latent linear Gaussian processes as in the Kalman Filter method and is devoted to detect impulse-noised outliers. These are data points that differ significantly from other observations. 'ML' (Maximization Likelihood) and 'EM' (Expectation-Maximization algorithm) algorithms were implemented in 'kfino'. The method is described in full details in the following arXiv e-Print: arXiv:2208.00961.

Details

xxxxxxxx xxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxx.

Author(s)

Maintainer: Isabelle Sanchez `<isabelle.sanchez@inrae.fr>`

Authors:

- Bertrand Cloez `<bertrand.cloez@inrae.fr>`

Other contributors:

- Benedicte Fontez `<benedicte.fontez@supagro.fr>` [contractor]

See Also

Useful links:

- `https://forgemia.inra.fr/isabelle.sanchez/kfino`
- Report bugs at `https://forgemia.inra.fr/isabelle.sanchez/kfino/-/issues`

kfino_fit *kfino_fit a function to detect outlier with a Kalman Filtering approach*

Description

kfino_fit a function to detect outlier with a Kalman Filtering approach

Usage

```
kfino_fit(
  datain,
  Tvar,
  Yvar,
  param = NULL,
  doOptim = TRUE,
  method = "ML",
  threshold = 0.5,
  kappa = 10,
  kappaOpt = 7,
  verbose = FALSE
)
```

Arguments

`datain` an input data.frame of one time course to study (unique IDE)

`Tvar` char, time column name in the data.frame datain, a numeric vector Tvar should be expressed as a proportion of day in seconds

`Yvar` char, name of the variable to predict in the data.frame datain

`param` list, a list of initialization parameters

`doOptim` logical, if TRUE optimization of the initial parameters, default TRUE

`method` character, the method used to optimize the initial parameters: Expectation-Maximization algorithm ‘"EM"‘ (faster) or Maximization Likelihood ‘"ML"‘ (more robust), default ‘"ML"‘

`threshold` numeric, threshold to qualify an observation as outlier according to the label_pred, default 0.5

`kappa` numeric, truncation setting for likelihood optimization over initial parameters, default 10

`kappaOpt` numeric, truncation setting for the filtering and outlier detection step with optimized parameters, default 7

`verbose` write details if TRUE (optional), default FALSE.

Details

The initialization parameter list ‘param‘ contains:

mm (optional) numeric, target weight, NULL if the user wants to optimize it

pp (optional) numeric, probability to be correctly weighed, NULL if the user wants to optimize it

m0 (optional) numeric, initial weight, NULL if the user wants to optimize it

aa numeric, rate of weight change, default 0.001

expertMin numeric, the minimal weight expected by the user

expertMax numeric, the maximal weight expected by the user

sigma2_m0 numeric, variance of m0, default 1

sigma2_mm numeric, variance of mm, related to the unit of Tvar, default 0.05

sigma2_pp numeric, variance of pp, related to the unit of Yvar, default 5

K numeric, a constant value in the outlier function (trapezium), by default K=5

seqp numeric vector, sequence of pp probability to be correctly weighted. default seq(0.5,0.7,0.1)

It should be given by the user based on their knowledge of the animal or the data set. All parameters are compulsory except m0, mm and pp that can be optimized by the algorithm. In the optimization step, those three parameters are initialized according to the input data (between the expert range) using quantile of the Y distribution (varying between 0.2 and 0.8 for m0 and 0.5 for mm). pp is a sequence varying between 0.5 and 0.7. A sub-sampling is performed to speed the algorithm if the number of possible observations studied is greater than 500. Optimization is performed using ‘"EM"‘ or ‘"ML"‘ method.

Value

a S3 list with two data frames and a list of vectors of kfino results

detectOutlier: The whole input data set with the detected outliers flagged and the prediction of the analyzed variable. the following columns are joined to the columns present in the input data set:

prediction the parameter of interest - Yvar - predicted

label_pred the probability of the value being well predicted

lwr lower bound of the confidence interval of the predicted value

upr upper bound of the confidence interval of the predicted value

flag flag of the value (OK value, KO value (outlier), OOR value (out of range values defined by the user in ‘kfino_fit‘ with ‘expertMin‘, ‘expertMax‘ input parameters). If flag == OOR the 4 previous columns are set to NA.

PredictionOK: A subset of ‘detectOutlier‘ data set with the predictions of the analyzed variable on possible values (OK and KO values)

kfino.results: kfino results (a list of vectors containing the prediction of the analyzed variable, the probability to be an outlier, the likelihood, the confidence interval of the prediction and the flag of the data) on input parameters that were optimized if the user chose this option

Examples

```
data(spring1)
library(dplyr)

# --- With Optimization on initial parameters - ML method
t0 <- Sys.time()
param1<-list(m0=NULL,
             mm=NULL,
             pp=NULL,
             aa=0.001,
             expertMin=30,
             expertMax=75,
             sigma2_m0=1,
             sigma2_mm=0.05,
             sigma2_pp=5,
             K=2,
             seqp=seq(0.5,0.7,0.1))

resu1<-kfino_fit(datain=spring1,
              Tvar="dateNum",Yvar="Poids",
              doOptim=TRUE,method="ML",param=param1,
              verbose=TRUE)
Sys.time() - t0

# --- Without Optimization on initial parameters
t0 <- Sys.time()
param2<-list(m0=41,
             mm=45,
             pp=0.5,
             aa=0.001,
             expertMin=30,
             expertMax=75,
             sigma2_m0=1,
             sigma2_mm=0.05,
             sigma2_pp=5,
             K=2,
             seqp=seq(0.5,0.7,0.1))
resu2<-kfino_fit(datain=spring1,
              Tvar="dateNum",Yvar="Poids",
              param=param2,
              doOptim=FALSE,
              verbose=FALSE)
Sys.time() - t0
```

kfino_plot *kfino_plot a graphical function for the result of a kfino run*

Description

kfino_plot a graphical function for the result of a kfino run

Usage

```
kfino_plot(
  resuin,
  typeG,
  Tvar,
  Yvar,
  Ident,
  title = NULL,
  labelX = NULL,
  labelY = NULL
)
```

Arguments

`resuin` a list resulting of the kfino algorithm

`typeG` char, type of graphic, either detection of outliers (with qualitative or quantitative display) or prediction. must be "quanti" or "quali" or "prediction"

`Tvar` char, time variable in the data.frame datain

`Yvar` char, variable which was analysed in the data.frame datain

`Ident` char, column name of the individual id to be analyzed

`title` char, a graph title

`labelX` char, a label for x-axis

`labelY` char, a label for y-axis

Details

The produced graphic can be, according to typeG:

quali This plot shows the detection of outliers with a qualitative rule: OK values (black), KO values (outliers, purple) and OOR values (out of range values defined by the user in ‘kfino_fit‘, red)

quanti This plot shows the detection of outliers with a quantitative display using the calculated probability of the kfino algorithm

prediction This plot shows the prediction of the analyzed variable plus the OK values. Prediction corresponds to E[X_t | Y_1...t] for each time point t. Between 2 time points, we used a simple linear interpolation.

Value

a ggplot2 graphic

Examples

```
data(spring1)
library(dplyr)

print(colnames(spring1))

# --- Without Optimisation on initial parameters
param2<-list(m0=41,
             mm=45,
             pp=0.5,
             aa=0.001,
             expertMin=30,
             expertMax=75,
             sigma2_m0=1,
             sigma2_mm=0.05,
             sigma2_pp=5,
             K=2,
             seqp=seq(0.5,0.7,0.1))
resu2<-kfino_fit(datain=spring1,
              Tvar="dateNum",Yvar="Poids",
              param=param2,
              doOptim=FALSE)

# flags are qualitative
kfino_plot(resuin=resu2,typeG="quali",
            Tvar="Day",Yvar="Poids",Ident="IDE",
            title="kfino spring1",
            labelX="Time (day)",labelY="Weight (kg)")

# flags are quantitative
kfino_plot(resuin=resu2,typeG="quanti",
            Tvar="Day",Yvar="Poids",Ident="IDE")

# predictions on OK values
kfino_plot(resuin=resu2,typeG="prediction",
            Tvar="Day",Yvar="Poids",Ident="IDE")
```

`lambs` *a dataset containing the WoW weighing for 4 animals of 1296 observations, https://doi.org/10.1016/j.compag.2018.08.022*

Description

A dataset for kfino algorithm

Usage

```
lambs
```

Format

a data.frame

Poids weight (in kg)

Date Date of weighing yyyy-mm-dd

IDE id of the animal

Day Date of weighing with day and time yyyy-mm-dd hh:mm:ss

dateNum a rescaled date - fraction of the whole observational time for one individual. $dateNum = (Heure - min(Heure))/86400 + (Date - min(Date))/86400$

`merinos1` *a dataset containing the WoW weighing for one animal (merinos lamb) of 397 observations. https://doi.org/10.1016/j.compag.2018.08.022*

Description

A dataset for kfino algorithm

Usage

```
merinos1
```

Format

a data.frame

Poids weight (in kg)

Date Date of weighing yyyy-mm-dd

IDE id of the animal

Day Date of weighing with day and time yyyy-mm-dd hh:mm:ss

dateNum a rescaled date - fraction of the whole observational time for one individual. $dateNum = (Heure - min(Heure))/86400 + (Date - min(Date))/86400$

`merinos2` *a dataset containing the WoW weighing for one animal (merinos lamb) of 345 observations, difficult to model. https://doi.org/10.1016/j.compag.2018.08.022*

Description

A dataset for kfino algorithm

Usage

```
merinos2
```

Format

a data.frame

Poids weight (in kg)

Date Date of weighing yyyy-mm-dd

IDE id of the animal

Day Date of weighing with day and time yyyy-mm-dd hh:mm:ss

dateNum a rescaled date - fraction of the whole observational time for one individual. $dateNum = (Heure - min(Heure))/86400 + (Date - min(Date))/86400$

`spring1` *a dataset containing the WoW weighing for one animal of 203 observations. https://doi.org/10.1016/j.compag.2018.08.022*

Description

A dataset for kfino algorithm

Usage

```
spring1
```

Format

a data.frame

Poids weight (in kg)

Date Date of weighing yyyy-mm-dd

IDE id of the animal

Day Date of weighing with day and time yyyy-mm-dd hh:mm:ss

dateNum a rescaled date - fraction of the whole observational time for one individual. $dateNum = (Heure - min(Heure))/86400 + (Date - min(Date))/86400$

`utils_EM` *utils_EM a function to estimate the parameters ‘m_0‘ , ‘mm‘, ‘pp‘ through an Expectation-Maximization (EM) method*

Description

utils_EM a function to estimate the parameters ‘m_0‘ , ‘mm‘, ‘pp‘ through an Expectation-Maximization (EM) method

Usage

```
utils_EM(param, kappaOpt, Y, Tps, N, scalingC)
```

Arguments

`param`	list, see initial parameter list in `kfino_fit`
`kappaOpt`	numeric, truncation setting for initial parameters’ optimization, default 7
`Y`	character, name of the numeric variable to predict in the data.frame datain
`Tps`	character, time column name in the data.frame datain, a numeric vector. Tvar can be expressed as a proportion of day in seconds
`N`	numeric, length of the numeric vector of Y values
`scalingC`	numeric, scaling constant. To be changed if the function is not able to calculate the likelihood because the number of data is large

Details

utils_EM is a tool function used in the main `kfino_fit` function. It uses the same input parameter list than the main function.

Value

a list:

m0 numeric, optimized m0

mm numeric, optimized mm

pp numeric, optimized pp

likelihood numeric, the calculated likelihood

Examples

```
set.seed(1234)
Y<-rnorm(n=10,mean=50,4)
Tps<-seq(1,10)
N=10
param2<-list(m0=41,
             mm=45,
             pp=0.5,
             aa=0.001,
             expertMin=30,
             expertMax=75,
             sigma2_m0=1,
             sigma2_mm=0.05,
             sigma2_pp=5,
             K=2,
             seqp=seq(0.5,0.7,0.1))
print(Y)
utils_EM(param=param2,kappaOpt=7,Y=Y,Tps=Tps,N=N,scalingC=6)
```

---

utils_fit — *utils_fit a fonction running the kfino algorithm to filter data and detect outliers under the knowledge of all parameters*

Description

utils_fit a fonction running the kfino algorithm to filter data and detect outliers under the knowledge of all parameters

Usage

```
utils_fit(param, threshold, kappa = 10, Y, Tps, N)
```

Arguments

`param` list, see initial parameter list in `kfino_fit`

`threshold` numeric, threshold for confidence interval, default 0.5

`kappa` numeric, truncation setting for likelihood optimization, default 10

`Y` character, name of the numeric variable to predict in the data.frame datain

`Tps` character, time column name in the data.frame datain, a numeric vector. Tvar can be expressed as a proportion of day in seconds

`N` numeric, length of the numeric vector of Y values

Details

utils_fit is a tool function used in the main `kfino_fit` function. It uses the same input parameter list than the main function.

Value

a list

prediction vector, the prediction of weights

label vector, probability to be an outlier

likelihood numeric, the calculated likelihood

lwr vector of lower bound confidence interval of the prediction

upr vector of upper bound confidence interval of the prediction

flag char, is an outlier or not

Examples

```
set.seed(1234)
Y<-rnorm(n=10,mean=50,4)
Tps<-seq(1,10)
N=10
param2<-list(m0=41,
             mm=45,
             pp=0.5,
             aa=0.001,
             expertMin=30,
             expertMax=75,
             sigma2_m0=1,
             sigma2_mm=0.05,
             sigma2_pp=5,
             K=2,
            seqp=seq(0.5,0.7,0.1))
print(Y)
utils_fit(param=param2,threshold=0.5,kappa=10,Y=Y,Tps=Tps,N=N)
```

Index