Package ‘RobKF’

July 15, 2021

Type Package
Title Innovative and/or Additive Outlier Robust Kalman Filtering
Version 1.0.2
Date 2021-07-15
License GPL
Imports Rcpp (>= 1.0.2), Rdpack, ggplot2, reshape2, Matrix
RdMacros Rdpack
LinkingTo Rcpp, RcppEigen
RoxygenNote 7.1.1
NeedsCompilation yes
Author Alex TM Fisch [aut],
     Daniel Grose [aut, cre],
     Idris A Eckley [aut, ths],
     Paul Fearnhead [aut, ths],
     Lawrence Bardwell [aut, ctb]
Maintainer Daniel Grose <dan.grose@lancaster.ac.uk>
Repository CRAN
Date/Publication 2021-07-15 09:40:02 UTC

R topics documented:

AORKF_huber .............................................................. 2
AORKF_t ................................................................. 3
Generate_Data ............................................................ 5
IOAORKF ................................................................. 6
IORKF ................................................................. 8
A huberisation based additive outlier robust Kalman filter

Description
An additive outlier robust Kalman filter, based on the work by Ruckdeschel et al. (2014). This function assumes that the additions are potentially polluted by a heavy tailed process. The update equations are made robust to these via huberisation.

Usage
AORKF_huber(
  Y,
  mu_0,
  Sigma_0 = NULL,
  A,
  C,
  Sigma_Add,
  Sigma_Inn,
  h = 2,
  epsilon = 1e-06
)

Arguments
Y A list of matrices containing the observations to be filtered.
mu_0 A matrix indicating the mean of the prior for the hidden states.
Sigma_0 A matrix indicating the variance of the prior for the hidden states. It defaults to the limit of the variance of the Kalman filter.
A A matrix giving the updates for the hidden states.
C A matrix mapping the hidden states to the observed states.
Sigma_Add A positive definite matrix giving the additive noise covariance.
Sigma_Inn A positive definite matrix giving the innovative noise covariance.
h A numeric giving the huber threshold. It defaults to 2.
epsilon A positive numeric giving the precision to which the limit of the covariance is to be computed. It defaults to 0.000001.

Value
An rkf S3 class.
**AORKF_t**

**A t-distribution based additive outlier robust Kalman filter**

**Description**

An additive outlier robust Kalman filter, based on the work by Agamennoni et al. (2018). This function assumes that the additions are potentially polluted by a heavy tailed process, which is approximated by a t-distribution. Variational inference is used to approximate the posterior.

**Usage**

```r
AORKF_t(
  Y,
  mu_0,
  Sigma_0 = NULL,
  A,
  C,
  Sigma_Add,
  Sigma_Inn,
  s = 2,
  epsilon = 1e-06
)
```
Arguments

Y 
A list of matrices containing the observations to be filtered.

mu_0 
A matrix indicating the mean of the prior for the hidden states.

Sigma_0 
A matrix indicating the variance of the prior for the hidden states. It defaults to the limit of the variance of the Kalman filter.

A 
A matrix giving the updates for the hidden states.

C 
A matrix mapping the hidden states to the observed states.

Sigma_Add 
A positive definite matrix giving the additive noise covariance.

Sigma_Inn 
A positive definite matrix giving the innovative noise covariance.

s 
A numeric giving the shape of the t-distribution to be considered. It defaults to 2.

epsilon 
A positive numeric giving the precision to which the limit of the covariance, and the variational inferences is to be computed. It defaults to 0.000001.

Value

An rkf S3 class.

References


Examples

library(RobKF)

set.seed(2019)

A = matrix(c(1), nrow = 1, ncol = 1)
C = matrix(c(1), nrow = 1, ncol = 1)

Sigma_Inn = diag(1,1)*0.01
Sigma_Add = diag(1,1)

mu_0 = matrix(0,nrow=1,ncol=1)

Y_list = Generate_Data(1000,A,C,Sigma_Add,Sigma_Inn,mu_0,anomaly_loc = c(100,400,700), anomaly_type = c("Add","Add","Add"),anomaly_comp = c(1,1,1), anomaly_strength = c(10,10,10))

Output = AORKF_t(Y_list,mu_0,Sigma_0=NULL,A,C,Sigma_Add,Sigma_Inn)

plot(Output,conf_level = 0.9999)
Generate_Data

Simulate data from a Kalman model

Description

This function simulates data obeying a Kalman model whilst allowing the user to add innovative and additive anomalies.

Usage

Generate_Data(
  n,
  A,
  C,
  Sigma_Add,
  Sigma_Inn,
  mu_0 = NULL,
  anomaly_loc = integer(0),
  anomaly_type = character(0),
  anomaly_comp = integer(0),
  anomaly_strength = NULL
)

Arguments

n     A positive integer giving the number of observations desired
A     A matrix giving the updates for the hidden states.
C     A matrix mapping the hidden states to the observed states.
Sigma_Add  A positive definite diagonal matrix giving the additive noise covariance.
Sigma_Inn  A positive definite diagonal matrix giving the innovative noise covariance.
mu_0     A matrix indicating the mean of the prior for the hidden states. It defaults to a zero-vector.
anomaly_loc  A vector of integers giving the locations of anomalies.
anomaly_type  A vector of strings, either "Add" or "Inn" indicating whether the anomaly is additive or innovative.
anomaly_comp  A vector of integers giving the component affected by the anomalies.
anomaly_strength     A vector of numerics giving the strength of the anomalies (in sigmas).

Value

A list of matrices, each corresponding to an observation.
Examples

```
library(RobKF)
library(ggplot2)

set.seed(2018)

A = diag(2)*0.99
A[1,2] = -0.05
C = matrix(c(10,0.1),nrow=1)
mu = matrix(c(0,0),nrow=2)
Sigma_Inn = diag(c(1,0.01)*0.00001,nrow=2)
Sigma_Add = diag(c(1)*0.1,nrow=1)

Y_list = Generate_Data(100,A,C,Sigma_Add,Sigma_Inn, mu_0 = mu, anomaly_loc = c(10,30,50),
anomaly_type = c("Inn","Add","Inn"),
anomaly_comp = c(1,1,2), anomaly_strength = c(400,-10,3000))

qplot(1:100,unlist(Y_list),xlab="time",ylab="observation")+theme_minimal()
```

---

**IOAORKF**

An innovative and additive outlier robust Kalman filter

**Description**

An implementation of Computationally Efficient Bayesian Anomaly detection by Sequential Sampling (CE-BASS) by Fisch et al. (2020). This function assumes that both the innovations and additions are potentially polluted by a heavy tailed process, which is approximated by a t-distribution. To approximate the posterior, particles for the precision (inverse variance) are sampled using a robust approximation to the posterior. Conditionally on those samples, the classical Kalman updates are used.

**Usage**

```
IOAORKF(  
  Y,  
  mu_0,  
  Sigma_0 = NULL,  
  A,  
  C,  
  Sigma_Add,  
  Sigma_Inn,  
  Particles,  
  Descendants = 1,  
  s = 2,  
  anom_add_prob = NULL,
```
anom_inn_prob = NULL,
epsilon = 1e-06,
horizon_matrix = NULL
)

Arguments

Y
A list of matrices containing the observations to be filtered.

mu_0
A matrix indicating the mean of the prior for the hidden states.

Sigma_0
A matrix indicating the variance of the prior for the hidden states. It defaults to the limit of the variance of the Kalman filter.

A
A matrix giving the updates for the hidden states.

C
A matrix mapping the hidden states to the observed states.

Sigma_Add
A positive definite diagonal matrix giving the additive noise covariance.

Sigma_Inn
A positive definite diagonal matrix giving the innovative noise covariance.

Particles
An integer giving the number of particles to be maintained at each step. More particles lead to more accuracy, but also require more memory and CPU time. The parameter should be at least p + q + 1, where p s the dimension of the observations and q the dimension of the hidden states.

Descendants
An integer giving the number of descendants to be sampled for each of the possible anomalies. Increasing Descendants leads to higher accuracy but also higher memory and CPU requirements. The default value is 1.

s
A numeric giving the shape of the t-distribution to be considered. It defaults to 2.

anom_add_prob
A vector of probabilities with length equal to the dimension of the observations giving the probabilities of additive outliers in each of the components. It defaults to 1/10000.

anom_inn_prob
A vector of probabilities with length equal to the dimension of the hidden state giving the probabilities of innovative outliers in each of the components. It defaults to 1/10000.

epsilon
A positive numeric giving the precision to which the limit of the covariance is to be computed. It defaults to 0.000001.

horizon_matrix
A matrix of 0s and 1s giving the horizon’s at which innovative particles are to be resampled. It defaults to a k by q matrix, where k is the number of observations required for observability of the system and q is the dimension of the hidden states.

Value
An ioaorkf S3 class.

References

Examples

```r
library(RobKF)

set.seed(2018)

A = diag(2)*0.99
A[1,2] = -0.05
C = matrix(c(10,0.1),nrow=1)
mu = matrix(c(0,0),nrow=2)
Sigma_Inn = diag(c(1,0.01)*0.00001,nrow=2)
Sigma_Add = diag(c(1)*0.1,nrow=1)

Y_list = Generate_Data(100,A,C,Sigma_Add,Sigma_Inn, mu_0 = mu, anomaly_loc = c(10,30,50),
anomaly_type = c("Inn","Add","Inn"),
anomaly_comp = c(1,1,2), anomaly_strength = c(400,-10,3000))

horizon_matrix = matrix(1,nrow = 3 ,ncol = 2)

Particle_List = IOAORKF(Y_list,mu,Sigma_0=NULL,A,C,Sigma_Add,Sigma_Inn,Particles=20,
horizon_matrix=horizon_matrix)

plot(Particle_List)
summary(Particle_List)
```

IORKF_huber

A huberisation based innovative outlier robust Kalman filter

Description

An innovative outlier robust Kalman filter, based on the work by Ruckdeschel et al. (2014). This function assumes that the innovations are potentially polluted by a heavy tailed process. The update equations are made robust to these via huberisation.

Usage

```r
IORKF_huber(
  Y,  
  mu_0,  
  Sigma_0 = NULL,  
  A,  
  C,  
  Sigma_Add,  
  Sigma_Inn,  
  h = 2,  
  epsilon = 1e-06
)
```
Arguments

- **Y**: A list of matrices containing the observations to be filtered.
- **mu_0**: A matrix indicating the mean of the prior for the hidden states.
- **Sigma_0**: A matrix indicating the variance of the prior for the hidden states. It defaults to the limit of the variance of the Kalman filter.
- **A**: A matrix giving the updates for the hidden states.
- **C**: A matrix mapping the hidden states to the observed states.
- **Sigma_Add**: A positive definite matrix giving the additive noise covariance.
- **Sigma_Inn**: A positive definite matrix giving the innovative noise covariance.
- **h**: A numeric giving the huber threshold. It defaults to 2.
- **epsilon**: A positive numeric giving the precision to which the limit of the covariance is to be computed. It defaults to 0.000001.

Value

An rkf S3 class.

References


Examples

```r
library(RobKF)

set.seed(2019)

A = matrix(c(1), nrow = 1, ncol = 1)
C = matrix(c(1), nrow = 1, ncol = 1)

Sigma_Inn = diag(1,1)*0.01
Sigma_Add = diag(1,1)

mu_0 = matrix(0,nrow=1,ncol=1)

Y_list = Generate_Data(1000,A,C,Sigma_Add,Sigma_Inn,mu_0,anomaly_loc = c(100,400,700),
anomaly_type = c("Inn","Inn","Inn"),anomaly_comp = c(1,1,1),
anomaly_strength = c(50,80,-100))

Output = IORKF_huber(Y_list,mu_0,Sigma_0=NULL,A,C,Sigma_Add,Sigma_Inn,h=2)

plot(Output,conf_level = 0.9999)
```
The classical Kalman filter

**Usage**

```r
KF(Y, mu_0, Sigma_0 = NULL, A, C, Sigma_Add, Sigma_Inn, epsilon = 1e-06)
```

**Arguments**

- **Y**: A list of matrices containing the observations to be filtered.
- **mu_0**: A matrix indicating the mean of the prior for the hidden states.
- **Sigma_0**: A matrix indicating the variance of the prior for the hidden states. It defaults to the limit of the variance of the Kalman filter.
- **A**: A matrix giving the updates for the hidden states.
- **C**: A matrix mapping the hidden states to the observed states.
- **Sigma_Add**: A positive definite matrix giving the additive noise covariance.
- **Sigma_Inn**: A positive definite matrix giving the innovative noise covariance.
- **epsilon**: A positive numeric giving the precision to which the limit of the covariance is to be computed. It defaults to 0.000001.

**Value**

An rkf S3 class.

**References**


**Examples**

```r
library(RobKF)

set.seed(2019)

A = matrix(c(1), nrow = 1, ncol = 1)
C = matrix(c(1), nrow = 1, ncol = 1)

Sigma_Inn = diag(1,1)*0.01
Sigma_Add = diag(1,1)

mu_0 = matrix(0,nrow=1,ncol=1)
```
Y_list = Generate_Data(1000,A,C,Sigma_Add,Sigma_Inn,mu_0)
Output = KF(Y_list,mu_0,Sigma_0=NULL,A,C,Sigma_Add,Sigma_Inn)
plot(Output)

Description

A function to plot the output produced by `AORKF_t`, `AORKF_huber`, `IORKF_huber` or `IOAORKF`. One can specify a time during the run for which the output should be displayed.

Usage

```r
## S3 method for class 'ioaorkf'
plot(x, time = NULL, horizon = NULL, subset = NULL, ...)

## S3 method for class 'rkf'
plot(x, time = NULL, subset = NULL, conf_level = 0.95, ...)
```

Arguments

- **x**: An instance of an `ioaorkf` or `rkf` S3 class.
- **time**: A positive integer giving the time at which the output is to be displayed. It defaults to the number of observations.
- **horizon**: A positive integer giving the smoothing horizon that is to be used. It must be at least equal to the number of rows of the horizon matrix used to obtain the `ioaorkf` object.
- **subset**: A list of integers indicating the components of observations which are to be plotted.
- **conf_level**: A probability between 0 and 1 giving the confidence level at which the series are to be tested against anomalies. It defaults to 0.95.
- **...**: Ignored.

Value

A ggplot object.
### print

#### Description

A function to print the output produced by `AORKF_t`, `AORKF_huber`, `IORKF_huber` or `IOAORKF`. One can specify a time during the run for which the output should be displayed.

#### Usage

```r
## S3 method for class 'ioaorkf'
print(x, time = NULL, horizon = NULL, ...)

## S3 method for class 'rkf'
print(x, time = NULL, conf_level = 0.95, ...)
```

#### Arguments

- `x`: An instance of an `ioaorkf` or `rkf` S3 class.
- `time`: A positive integer giving the time at which the output is to be displayed. It defaults to the number of observations.
- `horizon`: A positive integer giving the smoothing horizon that is to be used. It must be at least equal to the number of rows of the horizon matrix used to obtain the `ioaorkf` object.
- `...`: Ignored.
- `conf_level`: A probability between 0 and 1 giving the confidence level at which the series are to be tested against anomalies. It defaults to 0.95.

### summary

#### Description

A function to summarise the output produced by `AORKF_t`, `AORKF_huber`, `IORKF_huber`, or `IOAORKF`. One can specify a time during the run for which the output should be displayed.

#### Usage

```r
## S3 method for class 'ioaorkf'
summary(object, time = NULL, horizon = NULL, ...)

## S3 method for class 'rkf'
summary(object, time = NULL, conf_level = 0.95, ...)
```
Arguments

- **object**
  An instance of an `ioaorkf` or `rkf` S3 class.

- **time**
  A positive integer giving the time at which the output is to be displayed. It defaults to the number of observations.

- **horizon**
  A positive integer giving the smoothing horizon that is to be used. It must be at least equal to the number of rows of the `horizonmatrix` used to obtain the `ioaorkf` object.

- **...**
  Ignored

- **conf_level**
  A probability between 0 and 1 giving the confidence level at which the series are to be tested against anomalies. It defaults to 0.95.
Index

AORKF_huber, 2, 11, 12
AORKF_t, 3, 11, 12

Generate_Data, 5

IOAORKF, 6, 11, 12
IORKF_huber, 8, 11, 12

KF, 10

plot, 11
print, 12

summary, 12