Package ‘anomaly’

October 12, 2022

Type Package

Title Detecting Anomalies in Data

Version 4.0.2

Date 2021-10-20

Description Implements Collective And Point Anomaly (CAPA) <arXiv:1806.01947>, Multi-Variate Collective And Point Anomaly (MVCAPA) <arXiv:1909.01691>, Proportion Adaptive Segment Selection (PASS) <doi:10.1093/biomet/ass059>, and Bayesian Abnormal Region Detector (BARD) <arXiv:1412.5565> methods for the detection of anomalies in time series data. Also includes sequential versions of CAPA and MVCAPA <arXiv:2009.06670>.

License GPL

Imports dplyr,rlang,methods,assertive,Rdpack,ggplot2,reshape2,Rcpp (>= 0.12.18),robustbase,cowplot,xts,zoo,robust,runner,magrittr,caTools

LinkingTo Rcpp,BH

Depends R (>= 3.5.0)

NeedsCompilation yes

RoxygenNote 7.1.1

RdMacros Rdpack

Author Alex Fisch [aut],
Daniel Grose [aut, cre],
Lawrence Bardwell [aut, ctb],
Idris Eckley [aut, ths],
Paul Fearnhead [aut, ths]

Maintainer Daniel Grose <dan.grose@lancaster.ac.uk>

Repository CRAN

Date/Publication 2021-10-20 19:40:02 UTC

R topics documented:

`ac_corrected` *Transforms the data X to account for autocorrelation.*

Description

Transforms the data X to account for autocorrelation by centring and scaling. It uses the transformation $X_i^{'} = \frac{X_i - \mu_i}{k_i \sigma_i}$, were μ_i and σ_i are robust estimates for the mean and standard deviation of each variate (column), X_i, of X. The estimates are calculated using the median and median absolute deviation. The scaling $k_i = \sqrt{\left(\frac{1+\phi_i}{1-\phi_i}\right)}$, with ϕ_i a robust estimate for the autocorrelation at lag 1, is used to account for AR(1) structure in the noise.

Usage

```
ac_corrected(X)
```

Arguments

`X` A numeric matrix containing the potentially multivariate data to be transformed. Each column corresponds to a component and each row to an observation. The time series data classes ts, xts, and zoo are also supported.

Value

A numeric matrix of the same dimension as X containing the transformed data.

Examples

```
library(anomaly)
# generate some multivariate data
set.seed(0)
X<-simulate(n=1000,p=4,mu=10,locations=c(200,400,600),
            duration=100,proportions=c(0.25,0.5,0.75))
# compare the medians of each variate and transformed variate
head(apply(X,2,median))
head(apply(ac_corrected(X),2,median))
# compare the variances of each variate and transformed variate
head(apply(X,2,var))
head(apply(ac_corrected(X),2,var))
```

bard *Detection of multivariate anomalous segments using BARD.*

Description

Implements the BARD (Bayesian Abnormal Region Detector) procedure of Bardwell and Fearnhead (2017). BARD is a fully Bayesian inference procedure which is able to give measures of uncertainty about the number and location of anomalous regions. It uses negative binomial prior distributions on the lengths of anomalous and non-anomalous regions as well as a uniform prior for the means of anomalous regions. Inference is conducted by solving a set of recursions. To reduce computational and storage costs a resampling step is included.

Usage

```
bard(
  x,
  p_N = 1/(nrow(x) + 1),
  p_A = 5/nrow(x),
  k_N = 1,
  k_A = (5 * p_A)/(1 - p_A),
  pi_N = 0.9,
  paffected = 0.05,
  lower = 2 * sqrt(log(nrow(x))/nrow(x)),
  upper = max(transform(x)),
  alpha = 1e-04,
  h = 0.25,
  transform = robustscale
)
```

Arguments

`x` An n x p real matrix representing n observations of p variates. The time series data classes ts, xts, and zoo are also supported.

`p_N` Hyper-parameter of the negative binomial distribution for the length of non-anomalous segments (probability of success). Defaults to

$$\frac{1}{n+1}.$$

`p_A` Hyper-parameter of the negative binomial distribution for the length of anomalous segments (probability of success). Defaults to

$$\frac{5}{n}.$$

`k_N` Hyper-parameter of the negative binomial distribution for the length of non-anomalous segments (size). Defaults to 1.

`k_A` Hyper-parameter of the negative binomial distribution for the length of anomalous segments (size). Defaults to

$$\frac{5p_A}{1-p_A}.$$

`pi_N` Probability that an anomalous segment is followed by a non-anomalous segment. Defaults to 0.9.

`paffected` Proportion of the variates believed to be affected by any given anomalous segment. Defaults to 5%. This parameter is relatively robust to being mis-specified and is studied empirically in Section 5.1 of Bardwell and Fearnhead (2017).

`lower` The lower limit of the the prior uniform distribution for the mean of an anomalous segment μ. Defaults to

$$2\sqrt{\frac{\log(n)}{n}}.$$

`upper` The upper limit of the prior uniform distribution for the mean of an anomalous segment μ. Defaults to the largest standardised value of x, i.e. `max(transform(x))`.

`alpha` Threshold used to control the resampling in the approximation of the posterior distribution at each time step. A sensible default is 1e-4. Decreasing alpha increases the accuracy of the posterior distribution but also increases the computational complexity of the algorithm.

`h` The step size in the numerical integration used to find the marginal likelihood. The quadrature points are located from `lower` to `upper` in steps of `h`. Defaults to 0.25. Decreasing this parameter increases the accuracy of the calculation for the marginal likelihood but increases computational complexity.

`transform` A function used to transform the data prior to analysis. The default value is to scale the data using the median and the median absolute deviation.

Value

An instance of the S4 object of type `.bard.class` containing the data `x`, procedure parameter values, and the results.

Notes on default hyper-parameters

This function gives certain default hyper-parameters for the two segment length distributions. We chose these to be quite flexible for a range of problems. For non-anomalous segments a geometric distribution was selected having an average segment length of n with the standard deviation being of the same order. For anomalous segments we chose parameters that gave an average length of 5 and a variance of n. These may not be suitable for all problems and the user is encouraged to tune these parameters.

References

Bardwell L, Fearnhead P (2017). "Bayesian Detection of Abnormal Segments in Multiple Time Series." *Bayesian Anal.*, **12**(1), 193–218.

See Also

sampler

Examples

```
library(anomaly)
set.seed(0)
sim.data<-simulate(n=500,p=50,mu=2,locations=c(100,200,300),
                   duration=6,proportions=c(0.04,0.06,0.08))
# run bard
bard.res<-bard(sim.data, alpha = 1e-3, h = 0.5)
sampler.res<-sampler(bard.res)
collective_anomalies(sampler.res)

plot(sampler.res,marginals=TRUE)
```

---

capa *A technique for detecting anomalous segments and points based on CAPA.*

Description

A technique for detecting anomalous segments and points based on CAPA (Collective And Point Anomalies) by Fisch et al. (2018). This is a generic method that can be used for both univariate and multivariate data. The specific method that is used for the analysis is deduced by capa from the dimensions of the data.

Usage

```
capa(
  x,
  beta = NULL,
  beta_tilde = NULL,
  type = "meanvar",
  min_seg_len = 10,
  max_seg_len = Inf,
  max_lag = 0,
  transform = robustscale
)
```

Arguments

`x` A numeric matrix with n rows and p columns containing the data which is to be inspected. The time series data classes ts, xts, and zoo are also supported.

`beta` A numeric vector of length p, giving the marginal penalties. If p > 1, type ="meanvar" or type = "mean" and max_lag > 0 it defaults to the penalty regime 2' described in Fisch, Eckley and Fearnhead (2019). If p > 1, type = "mean"/"meanvar" and max_lag = 0 it defaults to the pointwise minimum of the penalty regimes 1, 2, and 3 in Fisch, Eckley and Fearnhead (2019).

`beta_tilde` A numeric constant indicating the penalty for adding an additional point anomaly. It defaults to 3log(np), where n and p are the data dimensions.

`type` A string indicating which type of deviations from the baseline are considered. Can be "meanvar" for collective anomalies characterised by joint changes in mean and variance (the default), "mean" for collective anomalies characterised by changes in mean only, or "robustmean" for collective anomalies characterised by changes in mean only which can be polluted by outliers.

`min_seg_len` An integer indicating the minimum length of epidemic changes. It must be at least 2 and defaults to 10.

`max_seg_len` An integer indicating the maximum length of epidemic changes. It must be at least min_seg_len and defaults to Inf.

`max_lag` A non-negative integer indicating the maximum start or end lag. Only useful for multivariate data. Default value is 0.

`transform` A function used to centre the data prior to analysis by `capa`. This can, for example, be used to compensate for the effects of autocorrelation in the data. Importantly, the untransformed data remains available for post processing results obtained using `capa`. The package includes several methods that are commonly used for the transform, (see `robustscale` and `ac_corrected`), but a user defined function can be specified. The default values is `transform=robust_scale`.

Value

An instance of an S4 class of type capa.class.

References

Fisch A~T~M, Eckley I~A, Fearnhead P (2018). “A linear time method for the detection of point and collective anomalies.” *ArXiv e-prints*. https://arxiv.org/abs/1806.01947.

Examples

```
library(anomaly)
# generate some multivariate data
set.seed(0)
sim.data<-simulate(n=500,p=100,mu=2,locations=c(100,200,300),
                   duration=6,proportions=c(0.04,0.06,0.08))
res<-capa(sim.data,type="mean",min_seg_len=2,max_lag=5)
collective_anomalies(res)
plot(res)
```

---

capa.mv *Detection of multivariate anomalous segments and points using MV-CAPA.*

Description

This function implements MVCAPA (Multi-Variate Collective And Point Anomaly) from Fisch et al. (2019). It detects potentially lagged collective anomalies as well as point anomalies in multivariate time series data. The runtime of MVCAPA scales linearly (up to logarithmic factors) in `ncol(x)` and `maxlag`. If `max_seg_len` is not set, the runtime scales quadratically at worst and linearly at best in `nrow(x)`. If `max_seg_len` is set the runtime scales like `nrow(x)*max_seg_len`.

Usage

```
capa.mv(
  x,
  beta = NULL,
  beta_tilde = NULL,
  type = "meanvar",
  min_seg_len = 10,
  max_seg_len = Inf,
  max_lag = 0,
  transform = robustscale
)
```

Arguments

x	A numeric matrix with n rows and p columns containing the data which is to be inspected. The time series data classes ts, xts, and zoo are also supported.

`beta` A numeric vector of length p, giving the marginal penalties. If type ="meanvar" or if type = "mean"/"robustmean" and maxlag > 0 it defaults to the penalty regime 2' described in Fisch, Eckley, and Fearnhead (2019). If type = "mean"/"robustmean" and maxlag = 0 it defaults to the pointwise minimum of the penalty regimes 1, 2, and 3 in Fisch, Eckley, and Fearnhead (2019).

`beta_tilde` A numeric constant indicating the penalty for adding an additional point anomaly. It defaults to 3log(np), where n and p are the data dimensions.

`type` A string indicating which type of deviations from the baseline are considered. Can be "meanvar" for collective anomalies characterised by joint changes in mean and variance (the default), "mean" for collective anomalies characterised by changes in mean only, or "robustmean" for collective anomalies characterised by changes in mean only which can be polluted by outliers.

`min_seg_len` An integer indicating the minimum length of epidemic changes. It must be at least 2 and defaults to 10.

`max_seg_len` An integer indicating the maximum length of epidemic changes. It must be at least the min_seg_len and defaults to Inf.

`max_lag` A non-negative integer indicating the maximum start or end lag. Default value is 0.

`transform` A function used to transform the data prior to analysis by `capa.mv`. This can, for example, be used to compensate for the effects of autocorrelation in the data. Importantly, the untransformed data remains available for post processing results obtained using `capa.mv`. The package includes several methods that are commonly used for the transform, (see `robustscale` and `ac_corrected`), but a user defined function can be specified. The default value is `transform=robust_scale`.

Value

An instance of an S4 class of type capa.mv.class.

References

Fisch A~T~M, Eckley I~A, Fearnhead P (2019). “Subset multivariate collective and point anomaly detection.” *ArXiv e-prints*. https://arxiv.org/abs/1909.01691.

Examples

```
library(anomaly)

### generate some multivariate data

set.seed(0)
sim.data<-simulate(n=500,p=100,mu=2,locations=c(100,200,300),
                   duration=6,proportions=c(0.04,0.06,0.08))

### Apply MVCAPA

res<-capa.mv(sim.data,type="mean",min_seg_len=2)
plot(res)
```

```
### generate some multivariate data

set.seed(2018)
x1 = rnorm(500)
x2 = rnorm(500)
x3 = rnorm(500)
x4 = rnorm(500)

### Add two (lagged) collective anomalies

x1[151:200] = x1[151:200]+2
x2[171:200] = x2[171:200]+2
x3[161:190] = x3[161:190]-3

x1[351:390] = x1[371:390]+2
x3[351:400] = x3[351:400]-3
x4[371:400] = x4[371:400]+2

### Add point anomalies

x4[451] = x4[451]*max(1,abs(1/x4[451]))*5
x4[100] = x4[100]*max(1,abs(1/x4[100]))*5
x2[050] = x2[050]*max(1,abs(1/x2[050]))*5

my_x = cbind(x1,x2,x3,x4)

### Now apply MVCAPA

res<-capa.mv(my_x,max_lag=20,type="mean")

plot(res)
```

capa.uv *Detection of univariate anomalous segments and points using CAPA.*

Description

A technique for detecting anomalous segments and points in univariate time series data based on CAPA (Collective And Point Anomalies) by Fisch et al. (2018). CAPA assumes that the data has a certain mean and variance for most time points and detects segments in which the mean and/or variance deviates from the typical mean and variance as collective anomalies. It also detects point outliers and returns a measure of strength for the changes in mean and variance. If the number of anomalous windows scales linearly with the number of data points, CAPA scales linearly with the number of data points. At worst, if there are no anomalies at all and max_seg_len is unspecified, the computational cost of CAPA scales quadratically with the number of data points.

Usage

```
capa.uv(
  x,
  beta = NULL,
  beta_tilde = NULL,
  type = "meanvar",
  min_seg_len = 10,
  max_seg_len = Inf,
  transform = robustscale
)
```

Arguments

x	A numeric vector containing the data which is to be inspected. The time series data classes ts, xts, and zoo are also supported.
beta	A numeric vector of length 1 or `max_seg_len - min_seg_len + 1` indicating the penalty for adding additional collective anomalies of all possible lengths. If an argument of length 1 is provided the same penalty is used for all collective anomalies irrespective of their length. The default value is 4log(n), where n denotes the number of observations.
beta_tilde	A numeric constant indicating the penalty for adding an additional point anomaly. It defaults to 3log(n), where n denotes the number of observations.
type	A string indicating which type of deviations from the baseline are considered. Can be "meanvar" for collective anomalies characterised by joint changes in mean and variance (the default), "mean" for collective anomalies characterised by changes in mean only, or "robustmean" for collective anomalies characterised by changes in mean only which can be polluted by outliers.
min_seg_len	An integer indicating the minimum length of epidemic changes. It must be at least 2 and defaults to 10.
max_seg_len	An integer indicating the maximum length of epidemic changes. It must be at least the min_seg_len and defaults to Inf.
transform	A function used to transform the data prior to analysis by `capa.uv`. This can, for example, be used to compensate for the effects of autocorrelation in the data. Importantly, the untransformed data remains available for post processing results obtained using `capa.uv`. The package includes several methods that are commonly used for the transform, (see `robustscale` and `ac_corrected`), but a user defined function can be specified. The default values is `transform=robust_scale`.

Value

An instance of an S4 class of type capa.uv.class.

References

Fisch A~T~M, Eckley I~A, Fearnhead P (2018). “A linear time method for the detection of point and collective anomalies.” *ArXiv e-prints*. https://arxiv.org/abs/1806.01947.

Examples

```
library(anomaly)
data(machinetemp)
attach(machinetemp)
res<-capa.uv(temperature,type="mean")
canoms<-collective_anomalies(res)
dim(canoms)[1] # over fitted due to autocorrelation
psi<-0.98 # computed using covRob
inflated_penalty<-3*(1+psi)/(1-psi)*log(length(temperature))
res<-capa.uv(temperature,type="mean",beta=inflated_penalty,
             beta_tilde=inflated_penalty)
summary(res)
plot(res)

library(anomaly)
data(Lightcurves)
### Plot the data for Kepler 10965588: No transit apparent
plot(Lightcurves$Kepler10965588$Day,Lightcurves$Kepler10965588$Brightness,xlab = "Day",pch=".")
### Examine a period of 62.9 days for Kepler 10965588
binned_data = period_average(Lightcurves$Kepler10965588,62.9)
inferred_anomalies = capa.uv(binned_data)
plot(inferred_anomalies)
```

---

collective_anomalies *Collective anomaly location, lags, and mean/variance changes.*

Description

Creates a data frame containing collective anomaly locations, lags and changes in mean and variance as detected by capa.uv, capa.mv, scapa.uv, scapa.mv, capa, pass, and sampler.

For an object produced by capa.uv or scapa.uv, collective_anomalies returns a data frame with columns containing the start and end position of the anomaly, the change in mean due to the anomaly. When type="meanvar", the change in variance due to the anomaly is also returned in an additional column.

For an object produced by capa.mv or scapa.mv, collective_anomalies returns a data frame with columns containing the start and end position of the anomaly, the variates affected by the anomaly, as well as their the start and end lags. When type="mean"/"robustmean" only the change in mean is reported. When type="meanvar" both the change in mean and change in variance are included. If merged=FALSE (the default), then all the collective anomalies are processed individually even if they are common across multiple variates. If merged=TRUE, then the collective anomalies are grouped together across all variates that they appear in.

For an object produced by capa, collective_anomalies returns the same results as scapa.uv when the data is univariate, or the same results as scapa.mv when the data is multivariate.

For an object produced by pass or sampler returns a data frame containing the start, end and strength of the collective anomalies.

Usage

```
## S4 method for signature 'bard.sampler.class'
collective_anomalies(object)

## S4 method for signature 'capa.class'
collective_anomalies(object, epoch = nrow(object@data), merged = FALSE)

## S4 method for signature 'capa.mv.class'
collective_anomalies(object)

## S4 method for signature 'capa.uv.class'
collective_anomalies(object)

## S4 method for signature 'pass.class'
collective_anomalies(object)

## S4 method for signature 'scapa.mv.class'
collective_anomalies(object, epoch = nrow(object@data))

## S4 method for signature 'scapa.uv.class'
collective_anomalies(object, epoch = nrow(object@data))
```

Arguments

`object` An instance of an S4 class produced by `capa`, `capa.uv` and `capa.mv`.

`epoch` Positive integer. CAPA methods are sequential and as such, can generate results up to, and including, any epoch within the data series. This can be controlled by the value of `epoch` and is useful for examining how the inferred anomalies are modified as the data series grows. The default value for `epoch` is the length of the data series.

`merged` Boolean value. If `merged=TRUE` then collective anomalies that are common across multiple variates are merged together. This is useful when comparing the relative strength of multivariate collective anomalies. Default value is `merged=FALSE`. Note - `merged=TRUE` is currently only available when `type="mean"`.

Value

A data frame.

See Also

`capa`,`capa.uv`,`capa.mv`.

Lightcurves *Kepler Lightcurve data.*

Description

One of the most successful approaches for the detection of exoplanets is the so called transit method: A star's brightness is continuously measured over time by a powerful telescope. If one or multiple planets orbit this star the recorded luminosity of the star will exhibit periodically recurring dips due to the transits of the planet in front of the telescope's lens – an effect comparable to that of an eclipse. Given how small planets are compared to stars the transit signals are known to be very weak.

The stars included in this file all have known exoplanets with the following periods:

Kepler 1871056: 2 planets with orbital periods of 40.8 and 140.1 days
Kepler 2307415: 2 planets with orbital periods of 4.61 and 12.12 days
Kepler 3102384: 2 planets with orbital periods of 10.57 and 523.9 days
Kepler 3231341: 4 planets with orbital periods of 4.24, 8.15, 12.33, and 19.00 days
Kepler 3447722: 3 planets with orbital periods of 10.30, 16.09, and 35.68 days
Kepler 4139816: 4 planets with orbital periods of 3.34, 7.82, 20.06, and 46.18 days
Kepler 10965588: 1 planet with orbital period of 62.89 days

More information about the exoplanets above and more data can be found at
https://exoplanetarchive.ipac.caltech.edu/index.html

This research has made use of the NASA Exoplanet Archive, which is operated by the California Institute of Technology, under contract with the National Aeronautics and Space Administration under the Exoplanet Exploration Program. Please include the following acknowledgement when using this data : “This makes use of the NASA Exoplanet Archive, which is operated by the California Institute of Technology, under contract with the National Aeronautics and Space Administration under the Exoplanet Exploration Program".

Usage

```
data(Lightcurves)
```

Format

A list of seven dataframes named "Kepler1871056", "Kepler2307415", "Kepler3102384", "Kepler3231341", "Kepler3447722", "Kepler4139816", and "Kepler10965588". Each dataframe consists of two columns called "Brightness" and "Day", containing measurements of a star's brightness and the measurement's timestamp respectively.

`machinetemp` *Machine temperature data.*

Description

Temperature sensor data of an internal component of a large, industrial machine. The data contains three known anomalies. The first anomaly is a planned shutdown of the machine. The second anomaly is difficult to detect and directly led to the third anomaly, a catastrophic failure of the machine. The data consists of 22695 observations of machine temperature recorded at 5 minute intervals along with the date and time of the measurement. The data was obtained from the Numenta Anomaly Benchmark (Ahmad et al. 2017), which can be found at https://github.com/numenta/NAB.

Usage

```
data(machinetemp)
```

Format

A dataframe with 22695 rows and 2 columns. The first column contains the date and time of the temperature measurement. The second column contains the machine temperature.

References

Ahmad S, Lavin A, Purdy S, Agha Z (2017). "Unsupervised real-time anomaly detection for streaming data." *Neurocomputing*, **262**, 134 - 147. ISSN 0925-2312, doi: 10.1016/j.neucom.2017.04.070, Online Real-Time Learning Strategies for Data Streams, `https://www.sciencedirect.com/science/article/pii/S0925231217309864/`.

`moving_ac_corrected` *Transforms the data X to account for autocorrelation using a moving window and a burn-in.*

Description

Transforms the data X by centring and scaling using $X^{'}_{ij} = \frac{X_{ij}-\mu_{ij}}{k_i\sigma_{ij}}$ where μ_{ij} and σ_{ij} are robust estimates for location and scale based on the median and the median absolute deviation of each variate (column) X_i of X calculated on a moving window centred at j. The scaling $k_i = \sqrt{\left(\frac{1+\phi_i}{1-\phi_i}\right)}$ is a robust estimate for the autocorrelation at lag 1 calculated on an initial (burn-in) segment of the data where ϕ_i is calculated using a robust estimate for the autocorrelation of the burn-in segment.

Usage

```
moving_ac_corrected(X, burnin, window_size)
```

Arguments

`X` A numeric matrix containing the potentially multivariate data to be transformed. Each column corresponds to a component and each row to an observation. The time series data classes ts, xts, and zoo are also supported.

`burnin` A positive integer indicating the initial length of the data used to determine the value of ϕ_i.

`window_size` A positive integer indication the length of the moving window.

Value

A numeric matrix of the same dimension as X containing the transformed data.

`pass` *Detection of multivariate anomalous segments using PASS.*

Description

Implements the PASS (Proportion Adaptive Segment Selection) procedure of Jeng et al. (2012). PASS uses a higher criticism statistic to pool the information about the presence or absence of a collective anomaly across the components. It uses Circular Binary Segmentation to detect multiple collective anomalies.

Usage

```
pass(
  x,
  alpha = 2,
  lambda = NULL,
  max_seg_len = 10,
  min_seg_len = 1,
  transform = robustscale
)
```

Arguments

`x` An n x p real matrix representing n observations of p variates. The time series data classes ts, xts, and zoo are also supported.

`alpha` A positive integer > 0. This value is used to stabilise the higher criticism based test statistic used by PASS leading to a better finite sample familywise error rate. Anomalies affecting fewer than alpha components will however in all likelihood escape detection.

`lambda` A positive real value setting the threshold value for the familywise Type 1 error. The default value is $(1.1\log(n \times max\_seg\_len) + 2\log(\log(p)))/\sqrt{\log(\log(p))}$.

`max_seg_len` A positive integer (`max_seg_len` > 0) corresponding to the maximum segment length. This parameter corresponds to Lmax in Jeng et al. (2012). The default value is 10.

`min_seg_len` A positive integer (`max_seg_len` >= `min_seg_len` > 0) corresponding to the minimum segment length. This parameter corresponds to Lmin in Jeng et al. (2012). The default value is 1.

`transform` A function used to transform the data prior to analysis. The default value is to scale the data using the median and the median absolute deviation.

Value

An instance of an S4 object of type `.pass.class` containing the data `X`, procedure parameter values, and the results.

References

Jeng XJ, Cai TT, Li H (2012). "Simultaneous discovery of rare and common segment variants." *Biometrika*, **100**(1), 157-172. ISSN 0006-3444, doi: 10.1093/biomet/ass059, `https://academic.oup.com/biomet/article/100/1/157/193108`.

Examples

```
library(anomaly)
# generate some multivariate data
set.seed(0)
sim.data<-simulate(n=500,p=100,mu=2,locations=c(100,200,300),
                   duration=6,proportions=c(0.04,0.06,0.08))
res<-pass(sim.data)
summary(res)
plot(res,variate_names=TRUE)
```

`period_average` *A function to search the Kepler data for periodically recurring dips in luminosity.*

Description

The signal of transiting planets is very weak, especially if the planet is small. This function amplifies it by exploiting the periodicity of the signal. All observations times are taken modulo the period and then binned. An average is then taken within each bin, those averages then stored as a vector and returned. If the orbital period of an exoplanet (or an integer fraction thereof) is used as argument for "period", the signal to noise ratio of the transit is improved, which can allow for the planet's detection.

Usage

```
period_average(data, period)
```

Arguments

`data`	A dataframe with one column named "Day" and the other "Brightness", such as Kepler10965588 (included in the package).
`period`	A numeric which is larger than 0 representing the period (in days) which is to be examined.

Value

A vector of numerics.

Examples

```
library(anomaly)
data(Lightcurves)
### Plot the data for Kepler 10965588: No transit apparent
plot(Lightcurves$Kepler10965588$Day,Lightcurves$Kepler10965588$Brightness,xlab = "Day",pch=".")
### Examine a period of 62.9 days for Kepler 10965588
binned_data = period_average(Lightcurves$Kepler10965588,62.9)
inferred_anomalies = capa.uv(binned_data)
plot(inferred_anomalies)
```

`plot-bard.sampler.class` *Visualisation of data, collective and point anomalies.*

Description

Plot methods for S4 objects returned by `capa`, `capa.uv`, `capa.mv`, `scapa.uv`, `scapa.mv`, `pass`, and `sampler`.

The plot can either be a line plot or a tile plot, the type produced depending on the options provided to the `plot` function and/or the dimensions of the data associated with the S4 object.

Usage

```
## S4 method for signature 'bard.sampler.class,ANY'
plot(x, subset, variate_names, tile_plot, marginals = FALSE)

## S4 method for signature 'capa.class,ANY'
plot(x, subset, variate_names = FALSE, tile_plot, epoch = nrow(x@data))

## S4 method for signature 'capa.mv.class,ANY'
plot(x, subset, variate_names = FALSE, tile_plot)

## S4 method for signature 'capa.uv.class,ANY'
plot(x, variate_name = FALSE)
```

```
## S4 method for signature 'pass.class,ANY'
plot(x, subset, variate_names = FALSE, tile_plot)

## S4 method for signature 'scapa.mv.class,ANY'
plot(x, subset, variate_names = FALSE, tile_plot, epoch)

## S4 method for signature 'scapa.uv.class,ANY'
plot(x, epoch, variate_name = FALSE)
```

Arguments

`x`	An instance of an S4 class produced by `capa`, `capa.uv`, `capa.mv`, `pass`, or `sampler`.
`subset`	A numeric vector specifying a subset of the variates to be displayed. Default value is all of the variates present in the data.
`variate_names`	Logical value indicating if variate names should be displayed on the plot. This is useful when a large number of variates are being displayed as it makes the visualisation easier to interpret. Default value is FALSE.
`tile_plot`	Logical value. If TRUE then a tile plot of the data is produced. The data displayed in the tile plot is normalised to values in [0,1] for each variate. This type of plot is useful when the data contains are large number of variates. The default value is TRUE if the number of variates is greater than 20.
`marginals`	Logical value. If `marginals=TRUE` the plot will include visualisations of the marginal probablities of each time point being anomalous. The defualt is `marginals=FALSE`.
`epoch`	Positive integer. CAPA methods are sequential and as such, can generate results up to, and including, any epoch within the data series. This can be controlled by the value of `epoch` and is useful for examining how the inferred anomalies are modified as the data series grows. The default value for `epoch` is the length of the data series.
`variate_name`	Logical value indicating if the variate name should be displayed. Default value is `variate_name=FALSE`.

Value

A ggplot object.

See Also

`capa`,`capa.uv`,`capa.mv`,`pass`,`sampler`.

point_anomalies *Point anomaly location and strength.*

Description

Creates a data frame containing point anomaly locations and strengths as detected by capa, capa.uv, capa.mv, scapa.uv, and scapa.mv.

For an object produced by capa.uv or scapa.uv, the output is a data frame with columns containing the position and strength of the anomaly.

For an object produced by capa.mv or capa.mv, point_anomalies returns a data frame with columns containing the position, variate, and strength of the anomaly.

For an object produced by capa, point_anomalies returns the same results as scapa.uv when the data is univariate, and the same results as scapa.mv when the data is multivariate.

Usage

```
## S4 method for signature 'capa.class'
point_anomalies(object, epoch = nrow(object@data))

## S4 method for signature 'capa.mv.class'
point_anomalies(object)

## S4 method for signature 'capa.uv.class'
point_anomalies(object)

## S4 method for signature 'scapa.mv.class'
point_anomalies(object, epoch = nrow(object@data))

## S4 method for signature 'scapa.uv.class'
point_anomalies(object, epoch = nrow(object@data))
```

Arguments

object	An instance of an S4 class produced by capa, capa.uv, and capa.mv.
epoch	Positive integer. CAPA methods are sequential and as such, can generate results up to, and including, any epoch within the data series. This can be controlled by the value of epoch and is useful for examining how the inferred anomalies are modified as the data series grows. The default value for epoch is the length of the data series.

Value

A data frame.

See Also

capa, capa.uv, capa.mv.

robustscale *robustscale*

Description

Transforms the data X by centring and scaling using $X'_{ij} = \frac{X_i - \mu_i}{\sigma_i}$ where μ_i and σ_i are robust estimates for the mean and standard deviation of each variate (column), X_i, of the multivariate time series X. The estimates are calculated using the median and median absolute deviation. This method is the default value for the transform argument used by the capa function, since the capa method assumes that the typical distribution of the data is standard normal.

Usage

```
robustscale(X)
```

Arguments

X A numeric matrix containing the data to be transformed. Each column corresponds to a component and each row to an observation. The time series data classes ts, xts, and zoo are also supported.

Value

A numeric matrix containing the transformed data.

Examples

```
library(anomaly)
# generate some multivariate data
set.seed(0)
X<-simulate(n=1000,p=4,mu=10,locations=c(200,400,600),
            duration=100,proportions=c(0.25,0.5,0.75))
# compare the medians of each variate and transformed variate
head(apply(X,2,median))
head(apply(robustscale(X),2,median))
# compare the variances of each variate and transformed variate
head(apply(X,2,var))
head(apply(robustscale(X),2,var))
```

`sampler` *Post processing of BARD results.*

Description

Draw samples from the posterior distribution to give the locations of anomalous segments.

Usage

```
sampler(bard_result, gamma = 1/3, num_draws = 1000)
```

Arguments

`bard_result` An instance of the S4 class `.bard.class` containing a result returned by the `bard` function.

`gamma` Parameter of loss function giving the cost of a false negative i.e. incorrectly allocating an anomalous point as being non-anomalous. For more details see Section 3.5 of Bardwell and Fearnhead (2017).

`num_draws` Number of samples to draw from the posterior distribution.

Value

Returns an S4 class of type `bard.sampler.class`.

References

Bardwell L, Fearnhead P (2017). “Bayesian Detection of Abnormal Segments in Multiple Time Series.” *Bayesian Anal.*, **12**(1), 193–218.

See Also

`bard`

Examples

```
library(anomaly)
set.seed(0)
sim.data<-simulate(n=500,p=50,mu=2,locations=c(100,200,300),
duration=6,proportions=c(0.04,0.06,0.08))
# run bard
res<-bard(sim.data, alpha = 1e-3, h = 0.5)
# sample
sampler(res)
```

scapa.mv *Online detection of multivariate anomalous segments and points using SMVCAPA.*

Description

This function implements SMVCAPA from Fisch et al. (2019) in an as-if-online way. It detects potentially lagged collective anomalies as well as point anomalies in streaming data. The runtime scales linearly (up to logarithmic factors) in `ncol(x)`, `max_lag`, and `max_seg_len`. This version of `capa.uv` has a default value `transform=tierney` which uses sequential estimates for transforming the data prior to analysis. It also returns an S4 class which allows the results to be postprocessed as if the data had been analysed in an online fashion.

Usage

```
scapa.mv(
  x,
  beta = NULL,
  beta_tilde = NULL,
  type = "meanvar",
  min_seg_len = 10,
  max_seg_len = Inf,
  max_lag = 0,
  transform = tierney
)
```

Arguments

`x`	A numeric matrix with n rows and p columns containing the data which is to be inspected. The time series data classes ts, xts, and zoo are also supported.
`beta`	A numeric vector of length p, giving the marginal penalties. If type ="meanvar" or if type = "mean" and maxlag > 0 it defaults to the penalty regime 2' described in Fisch, Eckley and Fearnhead (2019). If type = "mean" and maxlag = 0 it defaults to the pointwise minimum of the penalty regimes 1, 2, and 3 in Fisch, Eckley and Fearnhead (2019).
`beta_tilde`	A numeric constant indicating the penalty for adding an additional point anomaly. It defaults to 3log(np), where n and p are the data dimensions.
`type`	A string indicating which type of deviations from the baseline are considered. Can be "meanvar" for collective anomalies characterised by joint changes in mean and variance (the default), "mean" for collective anomalies characterised by changes in mean only, or "robustmean" for collective anomalies characterised by changes in mean only which can be polluted by outliers.
`min_seg_len`	An integer indicating the minimum length of epidemic changes. It must be at least 2 and defaults to 10.
`max_seg_len`	An integer indicating the maximum length of epidemic changes. It must be at least the min_seg_len and defaults to Inf.

max_lag A non-negative integer indicating the maximum start or end lag. Default value is 0.

transform A function used to transform the data prior to analysis by scapa.mv. This can, for example, be used to compensate for the effects of autocorrelation in the data. Importantly, the untransformed data remains available for post processing results obtained using scapa.mv. The package includes a method which can be used for the transform, (see tierney, the default), but a user defined (ideally sequential) function can be specified.

Value

An S4 class of type scapa.mv.class.

References

Fisch A~T~M, Eckley I~A, Fearnhead P (2019). “Subset multivariate collective and point anomaly detection.” *ArXiv e-prints*. https://arxiv.org/abs/1909.01691.

Fisch ATM, Bardwell L, Eckley IA (2020). “Real Time Anomaly Detection And Categorisation.” 2009.06670.

Examples

```
library(anomaly)

### generate some multivariate data

set.seed(2018)
x1 = rnorm(500)
x2 = rnorm(500)
x3 = rnorm(500)
x4 = rnorm(500)

### Add two (lagged) collective anomalies

x1[151:200] = x1[151:200]+2
x2[171:200] = x2[171:200]+2
x3[161:190] = x3[161:190]-3

x1[351:390] = x1[371:390]+2
x3[351:400] = x3[351:400]-3
x4[371:400] = x4[371:400]+2

### Add point anomalies

x4[451] = x4[451]*max(1,abs(1/x4[451]))*5
x4[100] = x4[100]*max(1,abs(1/x4[100]))*5
x2[050] = x2[050]*max(1,abs(1/x2[050]))*5

my_x = cbind(x1,x2,x3,x4)

### Now apply MVCAPA
```

```
res<-scapa.mv(my_x,max_lag=20,type="mean")

### Examine the output at different times and see how the results are updated:

plot(res,epoch=155)
plot(res,epoch=170)
plot(res,epoch=210)
```

scapa.uv *Detection of univariate anomalous segments using SCAPA.*

Description

An offline as-if-online implementation of SCAPA (Sequential Collective And Point Anomalies) by Bardwell et al. (2019) for online collective and point anomaly detection. This version of `capa.uv` has a default value `transform=tierney` which uses sequential estimates for transforming the data prior to analysis. It also returns an S4 class which allows the results to be postprocessed at different time points as if the data had been analysed in an online fashion up to that point.

Usage

```
scapa.uv(
  x,
  beta = NULL,
  beta_tilde = NULL,
  type = "meanvar",
  min_seg_len = 10,
  max_seg_len = Inf,
  transform = tierney
)
```

Arguments

`x` A numeric vector containing the data which is to be inspected. The time series data classes ts, xts, and zoo are also supported.

`beta` A numeric vector of length 1 or `max_seg_len - min_seg_len + 1` indicating the penalty for adding additional collective anomalies of all possible lengths. If an argument of length 1 is provided the same penalty is used for all collective anomalies irrespective of their length. The default value is 4log(n), where n denotes the number of observations.

`beta_tilde` A numeric constant indicating the penalty for adding an additional point anomaly. It defaults to 3log(n), where n is the number of observations.

`type` A string indicating which type of deviations from the baseline are considered. Can be "meanvar" for collective anomalies characterised by joint changes in mean and variance (the default), "mean" for collective anomalies characterised by changes in mean only, or "robustmean" for collective anomalies characterised by changes in mean only which can be polluted by outliers.

`min_seg_len` An integer indicating the minimum length of epidemic changes. It must be at least 2 and defaults to 10.

`max_seg_len` An integer indicating the maximum length of epidemic changes. It must be at least the min_seg_len and defaults to Inf.

`transform` A function used to transform the data prior to analysis by `scapa.uv`. This can, for example, be used to compensate for the effects of autocorrelation in the data. Importantly, the untransformed data remains available for post processing results obtained using `scapa.uv`. The package includes a method which can be used for the transform, (see `tierney`, the default), but a user defined (ideally sequential) function can be specified.

Value

An S4 class of type scapa.uv.class.

References

Fisch A~T~M, Eckley I~A, Fearnhead P (2018). “A linear time method for the detection of point and collective anomalies.” *ArXiv e-prints*. https://arxiv.org/abs/1806.01947.

Fisch ATM, Bardwell L, Eckley IA (2020). “Real Time Anomaly Detection And Categorisation.” 2009.06670.

Examples

```
library(anomaly)

# Simulated data example
# Generate data typically following a normal distribution with mean 0 and variance 1.
# Then introduce 3 anomaly windows and 4 point outliers.

set.seed(2018)
x  = rnorm(5000)
x[1601:1700] = rnorm(100,0,0.01)
x[3201:3300] = rnorm(100,0,10)
x[4501:4550] = rnorm(50,10,1)
x[c(1000,2000,3000,4000)] = rnorm(4,0,100)
# use magrittr to pipe the data to the transform
library(magrittr)
trans<-.%>%tierney(1000)
res<-scapa.uv(x,transform=trans)

# Plot results at two different times and note that anomalies are re-evaluated:
plot(res,epoch=3201)
plot(res,epoch=3205)
```

show *Displays S4 objects produced by capa methods.*

Description

Displays S4 object produced by capa, capa.uv, capa.mv, pass, bard, and sampler. The output displayed depends on the type of S4 object passed to the method. For all types, the output indicates whether the data is univariate or multivariate, the number of observations in the data, and the type of change being detected.

Usage

```
## S4 method for signature 'bard.class'
show(object)

## S4 method for signature 'capa.class'
show(object)

## S4 method for signature 'capa.mv.class'
show(object)

## S4 method for signature 'capa.uv.class'
show(object)

## S4 method for signature 'pass.class'
show(object)
```

Arguments

object	An instance of an S4 class produced by capa, capa.uv, capa.mv, pass, bard, or sampler.

See Also

capa,capa.uv,capa.mv,pass,bard,sampler.

simulate *A function for generating simulated multivariate data*

Description

Generates multivariate simulated data having n observations and p variates. The data have a standard Gaussian distribution except at a specified number of locations where there is a change in mean in a proportion of the variates. The function is useful for generating data to demonstrate and assess multivariate anomaly detection methods such as capa.mv and pass.

Usage

```
simulate(
  n = 100,
  p = 10,
  mu = 1,
  locations = 40,
  durations = 20,
  proportions = 0.1
)
```

Arguments

`n`	The number of observations. The default is `n=100`.
`p`	The number of variates. The default is `p=10`.
`mu`	The change in mean. Default is `mu=1`.
`locations`	A vector of locations (or scalar for a single location) where the change in mean occurs. The default is `locations=20`.
`durations`	A scalar or vector (the same length as `locations`) of values indicating the duration for the change in mean. If the durations are all of the same length then a scalar value can be used. The default is `durations=20`.
`proportions`	A scalar or vector (the same length as `locations`) of values in the range (0,1] indicating the proportion of variates at each location that are affected by the change in mean. If the proportions are all same than a scalar value can be used. The default is `proportions=0.1`.

Value

A matrix with n rows and p columns

Examples

```
library(anomaly)
sim.data<-simulate(500,200,2,c(100,200,300),6,c(0.04,0.06,0.08))
```

`summary` *Summary of collective and point anomalies.*

Description

Summary methods for S4 objects returned by `capa`, `capa.uv`, `capa.mv`, `pass`, and `sampler`. The output displayed depends on the type of object passed to summary. For all types, the output indicates whether the data is univariate or multivariate, the number of observations in the data, and the type of change being detected.

For an object produced by `capa.uv` or `capa.mv`, `pass`, or `sampler`, `summary` displays a summary of the analysis.

For an object produced by `capa` is the same as for an object produced by `capa.uv` or `capa.mv`.

Usage

```
## S4 method for signature 'bard.sampler.class'
summary(object, ...)

## S4 method for signature 'capa.class'
summary(object, epoch = nrow(object@data))

## S4 method for signature 'capa.mv.class'
summary(object)

## S4 method for signature 'capa.uv.class'
summary(object)

## S4 method for signature 'pass.class'
summary(object, ...)
```

Arguments

`object` An instance of an S4 class produced by `capa`, `capa.uv`, `capa.mv`, or `pass`.

`...` Ignored.

`epoch` Positive integer. CAPA methods are sequential and as such, can generate results up to, and including, any epoch within the data series. This can be controlled by the value of `epoch` and is useful for examining how the inferred anomalies are modified as the data series grows. The default value for `epoch` is the length of the data series.

See Also

`capa`,`capa.uv`,`capa.mv`,`pass`,`sampler`.

`tierney` *tierney*

Description

Transforms the data X by centring and scaling using $X'_{ij} = \frac{X_{ij}-\mu_{ij}}{\sigma_{ij}}$ where μ_{ij} and σ_{ij} are robust quantile based sequential estimates for the mean and standard deviation of each variate (column) X_i of X calculated up to time j. The estimates μ_{ij} and σ_{ij} are calculated from sequential estimates for the median and inter-quartile range developed by Tierney et al (1983). This method is the default value for the transform argument used by the `scapa.uv` function.

Usage

```
tierney(X, burnin = 10)
```

Arguments

`X` A numeric matrix containing the data to be transformed. The time series data classes ts, xts, and zoo are also supported.

`burnin` Specifies the period used to stabalise the quantile estimates. The default value is 10.

Value

A numeric matrix containing the transformed data.

References

Schruben L, Singh H, Tierney L (1983). "Optimal Tests for Initialization Bias in Simulation Output." *Oper. Res.*, **31**(6), 1167–1178. ISSN 0030-364X, doi: 10.1287/opre.31.6.1167, `http://dx.doi.org/10.1287/opre.31.6.1167`.

Examples

```
library(anomaly)
data(machinetemp)
attach(machinetemp)
plot(temperature)
temperature<-tierney(temperature,burnin=4305)
plot(temperature)
```

Index