Journal of Statistical Software

August 2024, Volume 110, Issue 1. *doi: 10.18637/jss.v110.i01*

anomaly: Detection of Anomalous Structure in Time Series Data

Alex Fisch
Lancaster University

Daniel Grose
Lancaster University

Idris A. Eckley
Lancaster University

Paul Fearnhead
Lancaster University

Lawrence Bardwell
Lancaster University

Abstract

One of the contemporary challenges in anomaly detection is the ability to detect, and differentiate between, both point and collective anomalies within a data sequence or time series. The **anomaly** package has been developed to provide users with a choice of anomaly detection methods and, in particular, provides an implementation of the recently proposed collective and point anomaly family of anomaly detection algorithms. This article describes the methods implemented whilst also highlighting their application to simulated data as well as real data examples contained in the package.

Keywords: anomaly detection, point anomaly, collective anomaly, BARD, CAPA, PASS.

1. Introduction

Within this article, we focus on the challenge of detecting anomalies within data sequences. Anomaly detection has become an increasingly important area of research activity due to its wide ranging application: from fault detection (Theissler 2017; Zhao, Liu, Hu, and Yan 2018), to fraud prevention (Ahmed, Mahmood, and Islam 2016), and system monitoring (Goh, Adepu, Tan, and Lee 2017). In broad terms, anomalies are observations that do not conform with the general or local pattern of the data and are commonly considered to fall into one of three categories: global anomalies, contextual anomalies, or collective anomalies (Chandola, Banerjee, and Kumar 2009). Global anomalies and contextual anomalies are defined as single observations that are outliers with regards to the complete dataset and their local context respectively. Conversely, collective anomalies are defined as sequences of observations that are not anomalous when considered individually, but together form an anomalous pattern (Fisch, Eckley, and Fearnhead 2022).

In parallel with the methodological development of statistical anomaly detection for data sequences, a number of software implementations have been developed. For example, within R, the **anomalize** package (Dancho and Vaughan 2023) provides an implementation of two point anomaly approaches, based on the interquartile range and generalized extreme studentized deviate test respectively, following the removal of any seasonal and trend components. Similarly **otsad** (Iturria, Carrasco, Herrera, Charramendieta, and Intxausti 2019) also provides a suite of approaches for the detection of point anomalies, whilst **cbar** (Seonghyun 2017) seeks to identify contextual anomalies using a Bayesian framework. Conversely, **tsoutliers** (López-de-Lacalle 2024) seeks to detect innovative and additive outliers together within time series, whilst **oddstream** (Talagala, Hyndman, and Smith-Miles 2019) implements an algorithm for the detection of anomalous series within newly arrived collections of series and **stray** (Talagala 2020) implements the HDoutliers algorithm for various settings including the detection of anomalies in high-dimensional data. Whilst the aforementioned packages arguably represent the current state of the statistical art at the time of writing, a number of other contributions have been made by researchers in other disciplines: see for example Python packages including **anomatools** (Vercruyssen 2022), **adtk** (Arundo Analytics, Inc 2020) and **PySAD** Vilmaz (2023), and Julia contributions including **MultivariateAnomalies** (Flach *et al.* 2017) and **AnomalyDetection** (Škvára, Pevný, and Šmídl 2018).

This paper describes the **anomaly** package (Fisch, Grose, Bardwell, Eckley, and Fearnhead 2024), available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=anomaly`, that implements a number of recently proposed methods for anomaly detection. For univariate data there is the collective and point anomaly detection method of Fisch *et al.* (2022), that can detect both collective and point anomalies. For multivariate data there are three methods, a multivariate extension of CAPA (collective and point anomaly; Fisch, Eckley, and Fearnhead 2021), the proportion adaptive segment selection (PASS) method of Jeng, Cai, and Li (2012), and a Bayesian approach, Bayesian abnormal region detector (BARD; Bardwell and Fearnhead 2017).

The multivariate CAPA method and PASS are similar in that, for a given segment they use a likelihood-based approach to measure the evidence that it is anomalous for each component of the multivariate data stream, and then merge this evidence across components. They differ in how they merge this evidence, with PASS using higher criticism (Donoho and Jin 2004) and CAPA using a penalized likelihood approach. One disadvantage of the higher criticism approach for merging evidence is that it can lose power when only one or a very small number of components are anomalous. Furthermore, CAPA also allows for point anomalies in otherwise normal segments of data, and can be more robust to detecting collective anomalies when there are point anomalies in the data. CAPA can also allow for the anomalies segments to be slightly misaligned across different components.

The BARD method considers a similar model to that of CAPA or PASS, but is Bayesian and so its basic output are samples from the posterior distribution for where the collective anomalies are, and which components are anomalous. It does not allow for point anomalies. As with any Bayesian method, it requires the user to specify suitable priors, but the output is more flexible, and can more directly allow for quantifying uncertainty about the anomalies.

The article begins by providing a brief introduction to anomaly detection before proceeding to give a detailed treatment of each approach. In each case, the relevant methodology is introduced, describing the associated package functionality where appropriate. The methods are applied to a number of test datasets that are available with the package. These data sets comprise the machine temperature data introduced by Lavin and Ahmad (2015), and a microarray genomics dataset. The examples also include details of how the effects of autocorrelation can be accounted for through the adjustment of the method parameters or by applying transforms to preprocess the data prior to analysis.

2. Background

The suite of methods described in this article focuses on collective anomalies. Informally, collective anomalies are segments of data which are anomalous when compared against the general structure of the full data. The modelling paradigm is to assume that there is a common model for data outside the anomalous regions, for example that it is independent normally distributed with a fixed mean and variance, and that collective anomalies correspond to segments of the data that are inconsistent with this, for example due to having a different mean or variance. One approach to modelling this type of anomaly is via epidemic changepoints – a particular form of changepoints admitting one change away from the typical distribution of the data and one back to it at a later time (Fisch *et al.* 2022). Formally, in the univariate setting, data, $\{x_t\}$, are said to follow a parametric epidemic changepoint model if $\{x_t\}$ obey the parametric model $f(x_t, \boldsymbol{\theta}(t))$ at all times and the parameter $\boldsymbol{\theta}(t)$ satisfies

$$\theta(t) = \begin{cases} \theta_1 & s_1 < t \leq e_1, \\ & \vdots \\ \theta_K & s_K < t \leq e_K, \\ \theta_0 & \text{otherwise.} \end{cases} \tag{1}$$

Here $(s_1, e_1), \ldots, (s_K, e_K)$ denote the start and end points of K collective anomalies. The typical (baseline) behavior of the data sequence is defined by the parameter θ_0. Conditionally on the parameter $\theta(t)$, all observations x_t are assumed to be independent, with relaxations of this assumption being discussed in the following sections.

When extending to the multivariate setting, i.e., a p-dimensional multivariate time series, it is common to assume that the series are independent, but that their periods of anomalous behavior align. The copy number variations data set (Bleakley and Vert 2011) provides a good example of such behavior. In the absence of a copy number variation, data from different individuals can be assumed to be independent. However, when collective anomalies under the form of copy number variations occur, they typically affect a subset of the test subjects. Under such a model, it is well known that joint analysis can lead to significant improvements in detection power over analyzing each component individually (Donoho and Jin 2004). The subset multivariate epidemic changepoint model provides a natural model for this type of behavior. It assumes that

$$\boldsymbol{\theta}^{(i)}(t) = \begin{cases} \boldsymbol{\theta}_1^{(i)} & \text{if } s_1 < t \leq e_1 \text{ and } i \in \mathbf{J}_1, \\ \vdots & \\ \boldsymbol{\theta}_K^{(i)} & \text{if } s_K < t \leq e_K \text{ and } i \in \mathbf{J}_K, \\ \boldsymbol{\theta}_0^{(i)} & \text{otherwise,} \end{cases} \qquad 1 \leq i \leq p. \tag{2}$$

where, again, K is the number of collective anomalies with (s_k,e_k) denoting the start and end of the k-th collective anomaly. The k-th collective anomaly only affects the subset J_k

of time-series. If the i-th time-series is affected by the k-th collective anomaly, i.e., $i \in J_k$ then $\theta_k^{(i)}$ denotes its parameter value; with $\theta_0^{(i)}$ denoting the parameter governing the typical behavior of the i-th time-series.

3. The collective and point anomaly family

The collective and point anomaly (CAPA) family of algorithms (Fisch *et al.* 2022, 2021) differ from many other anomaly detection methods in that they seek to simultaneously detect and distinguish between both collective and point anomalies. CAPA assumes that the data $\{\mathbf{x}_t\}$ follow the model detailed in (1), when univariate or (2) when multivariate. Point anomalies are incorporated within the model as epidemic changes of length 1. When analyzing multivariate data, CAPA assumes that the collective anomalies don't overlap, i.e., that $e_1 \leq s_2, \dots, e_{K-1} \leq s_K$, whilst allowing for the alignment of collective anomalies to be imperfect, i.e., allowing the components to leave their typical state and return to it at slightly different times.

Whilst the CAPA procedure can allow for many different models for the data, the current implementation assumes that the data is independent and normally distributed, and that the data has been normalized so that the mean is 0 and variance is 1. Non-anomalous data points are drawn from a normal distribution with a specific mean and variance (that the CAPA algorithm will estimate). Collective anomalies correspond to regions where the mean or mean and variance of the data are different.

CAPA infers the number, K, and locations $(s_1, e_1), \dots, (s_K, e_K)$ of collective anomalies as well as the set O of point anomalies by maximizing the penalized saving function

$$\sum_{i=1}^{K} \left(S(s_i, e_i) - \beta(e_i - s_i)\right) + \sum_{t \in O} \left(S'(x_t) - \beta'\right), \tag{3}$$

with respect to $(s_1, e_1), \dots, (s_K, e_K)$ and K, subject to constraints on the maximum and minimum lengths of anomalies (see Fisch *et al.* 2022, for details). Here the saving statistic, $S(s, e)$, of a putative anomaly with start point, s, and end point, e, corresponds to the improvement in model fit obtained by modelling the data in segment (s, e) as a collective anomaly. Given this improvement will always be non-negative a penalty, $\beta(e_i - s_i + 1)$, potentially depending on the length of the putative anomaly is used to prevent false positives being flagged. The choice of the penalty is model dependent, and discussed in the following sections. Similarly, $S'(x_t)$ and β' denote the improvement in model fit by assuming observation, x_t, is a point anomaly.

CAPA makes some important independence assumptions, and also assumes that the mean and variance of the non-anomalous data is constant. As we see below, it can successfully be applied to situations where these assumptions do not hold. There are two approaches to do so. First we can transform the data so that the assumptions are more reasonable – this could be to remove the effect of common factors that induce dependence across components or applying a filter to remove autocorrelation from the noise. Alternatively we can inflate the default penalties so that we still have good properties if there are no collective anomalies. We give an example of this latter approach for the machine temperature data set below.

CAPA maximizes the penalized saving in (3) using an optimal partitioning algorithm (Jackson *et al.* 2005). By default, the runtime of CAPA family algorithms scales quadratically in the number of observations. In practice, the computational complexity can be reduced by applying a pruning technique developed by Killick, Fearnhead, and Eckley (2012) that is used in the **changepoint** package (Killick and Eckley 2014). It is particularly effective when a large number of anomalies is present – leading to a linear relationship between runtime and data size when the number of anomalies is proportional to the size of the data. Another way to reduce the runtime is to impose a maximum length, m, for anomalies, the runtime then scaling linearly in both the number of observations and m.

The **anomaly** package contain a single function, `capa`, for accessing both the univariate and multivariate methods. It has the following 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 β is missing and $p = 1$ then $\beta = 3\log(n)$ when the type is `"mean"` or `"robustmean"` and $\beta = 4\log(n)$ otherwise. If $p > 1$, `type = "meanvar"` or `type = "mean"` and `max_lag > 0` it defaults to the penalty regime 2' described in Fisch *et al.* (2022). 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 *et al.* (2022).

- `beta_tilde`: A numeric constant indicating the penalty for adding an additional point anomaly. It defaults to $3\log(np)$, 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 characterized by joint changes in mean and variance (the default), `"mean"` for collective anomalies characterized by changes in mean only, or `"robustmean"` (only allowed when $p = 1$) for collective anomalies characterized 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`. The computational cost of the CAPA algorithm can be reduced by decreasing the value of `max_seg_len`.

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

When the `x` argument to `capa` is one dimensional (i.e., a vector or $n \times 1$ array or matrix) the univariate method is used and the `max_lag` argument is ignored, otherwise, the multivariate method is employed. The `capa` function returns an S4 object of type '`capa.class`' for which the generic methods `plot` and `summary` have been provided.

3.1. Univariate CAPA

The **anomaly** package supports univariate CAPA via the `capa` function for detecting segments characterized by an anomalous mean or anomalous mean and variance. When investigating segments for an anomalous mean against a typical Gaussian background of mean 0 and variance 1. If we let $f(x; \mu, \sigma)$ denote the density function for a normal random variable with mean μ and variance σ^2 evaluated at x, then the savings for a collective anomaly are equal to the improvement in log-likelihood by fitting a segment as anomalous. When only the mean of an anomalous segment changes,

$$\max_{\mu} \sum_{t=s}^{e} \left\{ \log f(x_t; \mu, 1) - \log f(x_t; 0, 1) \right\}.$$

While the saving for a point anomaly is set to be the saving for a collective anomaly of length 1. This gives

$$S(s,e) = (e-s+1)\left(\bar{x}_{s:e}\right)^2 \qquad \text{and} \qquad S'(x) = x^2$$

where $\bar{x}_{s:e}$ denotes the mean of observations $x_s, \dots, x_e$. Conversely, when investigating segments for an anomalous mean and/or variance, savings is

$$\max_{\mu,\sigma} \sum_{t=s}^{e} \left\{ \log f(x_t; \mu, \sigma) - \log f(x_t; 0, 1) \right\},$$

with the saving for a point anomaly being that for a change in variance only in a segment of size 1. This gives

$$S(s,e) = \sum_{t=s}^{e} x_t^2 - (e-s+1)\left(1 + \log\left(\frac{\sum_{t=s}^{e}\left(x_t - \bar{x}_{s:e}\right)^2}{e-s+1}\right)\right) \quad , \quad S'(x) = x^2 - 1 - \log\left(e^{-\beta'} + x^2\right)$$

Note that the data, x, requires standardization using robust estimates for the typical mean (the median) and the typical variance (the median absolute deviation) obtained on the complete data series so that the above cost functions can be used. See (Fisch *et al.* 2021) for further details.

The argument `max_seg_len` sets the maximum length of a collective anomaly. It can be used to prevent the detection of weak but long anomalies which typically arise as a result of model misspecification and also to reduce the run time of the CAPA algorithm. It defaults to a value equal to the length of the data series. Care is needed, as if a value is set that is smaller than the size of the actual anomalous regions, then CAPA is likely to fit multiple collective anomalies to such a region.

By default, $\beta = 3\log(n)$ and $\beta = 4\log(n)$ are used for changes in mean and changes in mean and variance respectively, and $\beta' = 3\log n$ for all models, as they have been shown to control the number of false positives when all observations are independent and identically distributed (i.i.d.) Gaussian (Fisch *et al.* 2022, 2021). These default parameters have a tendency to return many false positives on structured, i.e., non independent, data. In this case, `beta` and `beta_tilde` should be inflated whilst keeping their ratio constant. When looking for changes in mean, using

$$\beta = \tilde{\beta} = 3\frac{1+\hat{\rho}}{1-\hat{\rho}}\log(n),$$

where $\hat{\rho}$ is a robust estimate for the AR(1) autocorrelation often yields good false positive control. For changes in mean and variance,

$$\beta = 4\frac{1+\hat{\rho}}{1-\hat{\rho}}\log(n), \quad \tilde{\beta} = 3\frac{1+\hat{\rho}}{1-\hat{\rho}}\log(n).$$

The specific factor is justified theoretically in Lavielle and Moulines (2000).

Alternatively, the data x_t can be directly transformed using

$$x'_t = \sqrt{\frac{1-\hat{\rho}}{1+\hat{\rho}}} \left(\frac{x_t - \hat{\mu}}{\hat{\sigma}} \right), \tag{4}$$

where $\hat{\mu}$ is the median, and $\hat{\sigma}$ is a robust estimator of the standard deviation of the data x, such as based on the inter-quartile range, or the median absolute deviation from the median. This transform should only be used when looking for mean anomalies.

Simulated data

To demonstrate univariate **capa** a data series of 5000 normally distributed observations with 3 collective anomalies and four point anomalies is analysed. The data can be reproduced using the code provided below, which also runs the analyses and summarizes the results.

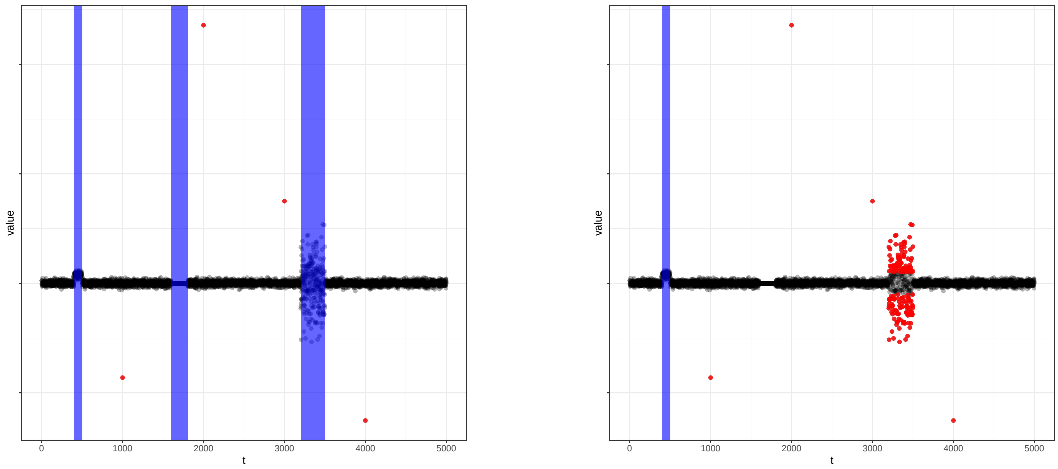
```
R> library("anomaly")
R> set.seed(0)
R> x <- rnorm(5000)
R> x[401:500] <- rnorm(100, 4, 1)
R> x[1601:1800] <- rnorm(200, 0, 0.01)
R> x[3201:3500] <- rnorm(300, 0, 10)
R> x[c(1000, 2000, 3000, 4000)] <- rnorm(4, 0, 100)
R> x <- (x - median(x)) / mad(x)
R> res <- capa(x)
R> summary(res)
```

```
Univariate CAPA detecting changes in mean and variance.
observations = 5000
minimum segment length = 10
maximum segment length = 5000

Point anomalies detected : 4
  location variate  strength
1     1000       1  43.07885
2     2000       1 117.84647
3     3000       1  37.49265
4     4000       1  62.67104

Collective anomalies detected : 3
  start  end variate start.lag end.lag  mean.change variance.change
1   401  500       1         0       0 14.597971638    4.990295e-04
2  1601 1800       1         0       0  0.001502774    9.869876e+01
3  3201 3500       1         0       0  0.036926415    7.764414e+00
```

```
R> plot(res)
```



(a) (b)

Figure 1: Visualization of data, collective and point anomalies detected by CAPA for the simulated univariate data. (a) detecting changes in mean and variance. (b) detecting changes in mean only. The blue regions correspond to the predicted anomalous segments and the red dots indicate point anomalies.

The `summary` method displays information regarding the analysis and details regarding the location and nature of the detected anomalies. The formatting demonstrates that `capa` correctly determines the presence of the anomalies in the simulated data. The `plot` function generates a '`ggplot`' object (Wickham 2016) which is shown in Figure 1a. The location of the collective anomalies are highlighted by vertical blue bands and the data point anomalies are shown in red. By default, CAPA detects both changes in mean and variance. The option `type = "mean"` can be used to detect changes in mean only.

```
R> res <- capa(x, type = "mean")
R> collective_anomalies(res)

    start end mean.change test.statistic
1    401 500    14.92774       1492.774

R> head(point_anomalies(res))

  location  strength
1     1000  43.07885
2     2000 117.84647
3     3000  37.49265
4     3201  11.44038
5     3202  16.52037
6     3203  10.58874
```

In this case, `capa` correctly identifies the collective change in mean and the point anomalies. However, as a consequence of CAPA now looking for changes in mean only, and assuming constant variance, the analysis results in changes in variance being classified as groups of point anomalies, see Figure 1b. The above example also demonstrates the `collective_anomalies` function, which is used to produce a data frame containing the location and change in mean for collective anomalies, and the `point_anomalies` function which provides the location and strength of the point anomalies.

As previously noted, the CAPA algorithm assumes that the data has been standardized. When this is not the case, false anomalous regions may be identified, as is the case in the following example.

```
R> res <- capa(1 + 2 * x, type = "mean")
R> nrow(collective_anomalies(res))
```

```
47
```

Real data: Machine temperature

To demonstrate the application of `capa` to real univariate data, a data stream from the Numenta Anomaly Benchmark corpus (Ahmad, Lavin, Purdy, and Agha 2017) consisting of temperature sensor data of an internal component of a large industrial machine is analysed. The dataset is included, with permission, in the **anomaly** package on the condition that derived work be kindly requested to acknowledge (Ahmad *et al.* 2017).

The machine temperature data consists of 22695 observations recorded at 5 minute intervals and contains three known anomalies as identified by an engineer working on the machine (Figure 2a). The first anomaly corresponds to a planned shutdown of the machine and the third anomaly to a catastrophic failure of the machine. The second anomaly, which can be difficult to detect, corresponds to the onset of a problem which led to the eventual system failure (Lavin and Ahmad 2015). Using `capa` with default parameters for the (normalized) data results in the detection of 97 collective anomalies.

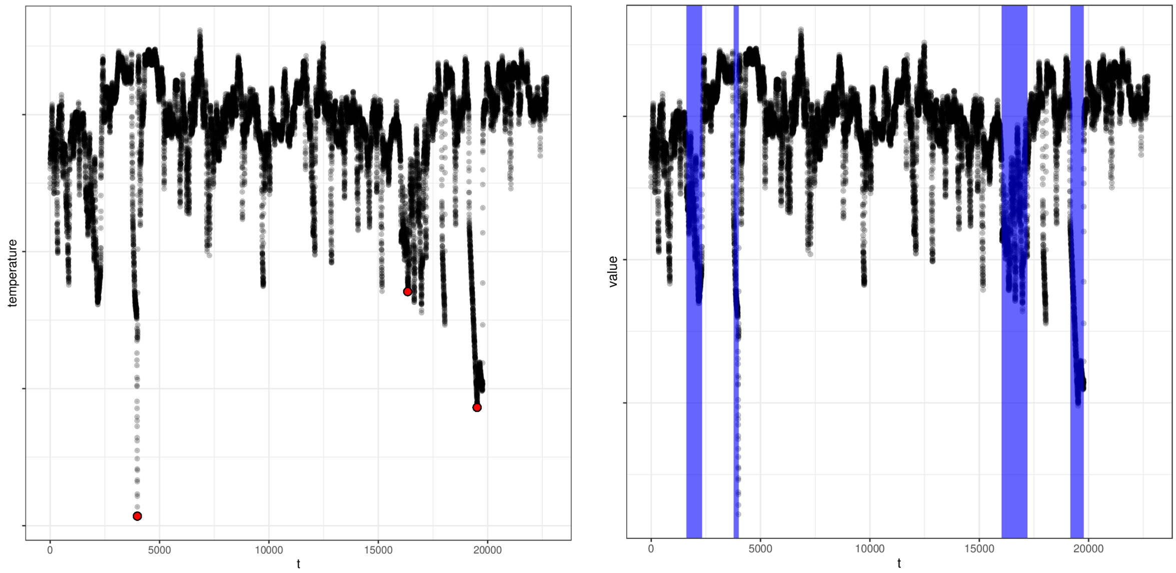
```
R> data("machinetemp", package = "anomaly")
R> attach(machinetemp)
R> x <- (temperature - median(temperature)) / mad(temperature)
R> res <- capa(x, type = "mean")
R> canoms <- collective_anomalies(res)
R> dim(canoms)[1]
```

```
[1] 97
```

One potential source of this over sensitivity is the presence of autocorrelation in the data. A robust estimate for the AR(1) autocorrelation $\hat{\rho}$ can be obtained using the `covMcd` method from the **robustbase** package.

```
R> library("robustbase")
R> n <- length(x)
R> x.lagged <- matrix(c(x[1:(n - 1)], x[2:n]), n - 1, 2)
R> rho_hat <- covMcd(x.lagged, cor = TRUE)$cor[1, 2]
```

which gives an estimate for $\hat{\rho}$ of 0.987.



(a) (b)

Figure 2: Time series of machine temperature data. (a) The highlighted data points show the locations of the anomalies identified by an engineer working on the machine. (b) The anomalies identified by **capa**. The second, third and fourth anomalies correspond well with the known anomalies. The first anomaly may be a false positive or an anomaly not recorded by the engineer.

As mentioned above, the default penalties have a tendency to return many false positives on structured, i.e., non independent, data. Instead we can inflate the penalties based on our estimates of the autocorrelation.

```
R> inflated_penalty <- 3 * (1 + rho_hat) / (1 - rho_hat) * log(n)
R> res <- capa(x, type = "mean", beta = inflated_penalty,
+    beta_tilde = inflated_penalty)
R> summary(res)
```

```
Univariate CAPA detecting changes in mean.
observations = 22695
minimum segment length = 10
maximum segment length = 22695

Point anomalies detected : 0

Collective anomalies detected : 4
  start   end variate start.lag end.lag mean.change test.statistic
1  1612  2327       1         0       0    9.148952       6550.650
2  3773  4002       1         0       0   25.648888       5899.244
3 16023 17204       1         0       0    8.191733       9682.628
4 19166 19775       1         0       0   39.426847      24050.377
```

```
R> plot(res)
```

The new predicted collective anomalies are shown in Figure 2b. The second, third and fourth collective anomalies detected using the modified penalty values correspond well with the known anomalies. The first anomaly on the other hand does not have a corresponding label which means that it is either a false positive or an anomaly corresponding to an event which has not been detected or recorded. Note that the test statistic for the first anomaly is stronger than for the second, but has smaller change in mean. This is inherent in the definition for the value of the test statistic used when inferring changes in mean, which is the change in mean multiplied by the length (duration) of the anomaly.

3.2. Multivariate CAPA

The `capa` function also has provision for analyzing multivariate data series using a multivariate version of the CAPA algorithm (Fisch *et al.* 2021). The algorithm assumes that the p components of the time series are independent of one another in all aspects except the locations of collective anomalies, which can affect any subset of the components. As with the univariate case, the current implementation of CAPA assumes non-anomalous data is independent normally distributed with a component specific mean and variance. Anomalous regions are then regions with a different mean, or a different mean and variance. The saving for a collective anomaly starting at s and ending at e involves aggregating the savings across components

$$S(s,e) = \max_k \sum_{i=1}^{k} \left(S_{(i)}(s,e) - \beta_i \right)$$

Here, $S_{(1)}(s,e) \geq \cdots \geq S_{(p)}(s,e)$ corresponds to the order statistics of the savings $S_1(s,e), \ldots, S_p(s,e)$, with $S_i(s,e)$ denoting the improvement in the individual components, as defined in Section 2. The β_i denote the typically decreasing marginal penalties or thresholds controlling false positives. Crucially, CAPA allows for the alignment of collective anomalies across components to be imperfect. In other words, certain components can lag by entering the anomalous state later and/or returning to their typical state earlier than others.

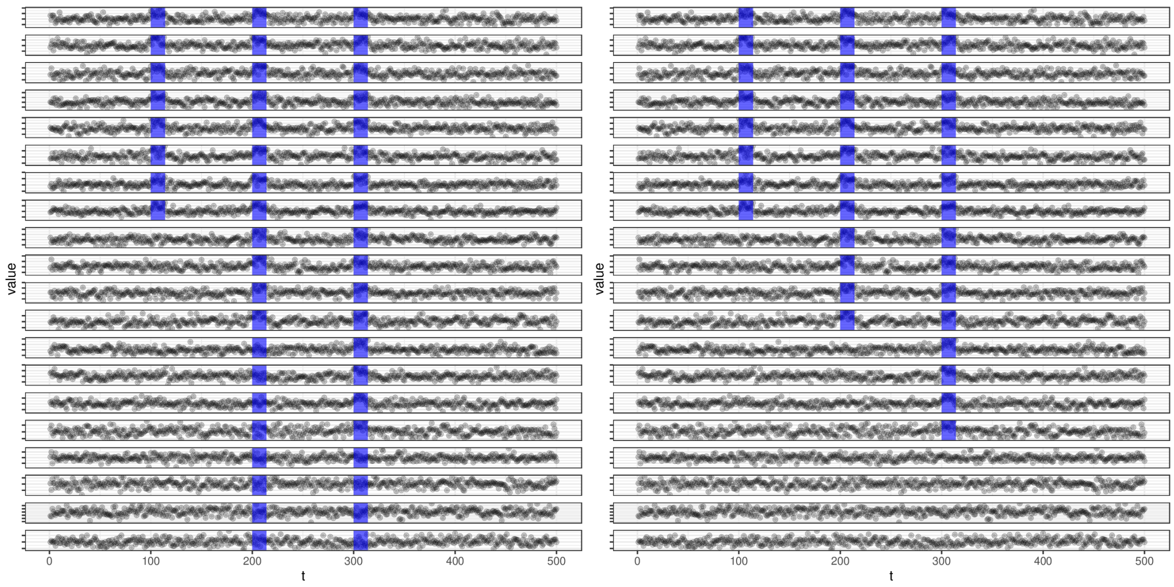
The (multivariate specific) `max_lag` argument in the `capa` function is used to set a limit on how much a collective anomaly in one variate can lag (or lead) a collective anomaly in another variate, whilst still being part of the same multivariate anomaly. The run time scales linearly with `max_lag`, though this dependence tends to be weak for small values of `max_lag`. The run time also scales linearly (up to logarithmic factors) with the number of components p. The default penalties are specific to i.i.d. data and tend to return many false positives when some of the p series contain, for example, auto-correlated structure. Extending the argument of Lavielle and Moulines (2000) to the multivariate setting, using

$$\tilde{\beta} = 2\frac{1+\hat{\rho}_1}{1-\hat{\rho}_1}\log(np) \qquad \beta_1 = 2\frac{1+\hat{\rho}_1}{1-\hat{\rho}_1}\log(np(w+1)) \qquad \beta_i = 2\frac{1+\hat{\rho}_i}{1-\hat{\rho}_i}\log(p(w+1)) \qquad 2 \leq i \leq p$$

can achieve good false positive control. Here, $\hat{\rho}_i$ is the i-th largest of the robust estimates for the AR(1) autocorrelation coefficients of the p series and w the value of the maximum lag.

Simulated data 1

To demonstrate multivariate CAPA, a simulated data set, `sim.data`, consisting of 500 observations on 200 variates which are $\mathcal{N}(0,1)$ distributed is used. The data is provided by the



(a)

(b)

Figure 3: CAPA results for Simulated data 1 in Section 3.2. The variates are displayed in order from the top down and the location and extent of the anomalous regions are indicated by the blue areas. (a) Using a penalty which maximizes power for the detection of collective anomalies but does not control false positives in the components. (b) Using a penalty which controls false positives in the components (at a loss of power).

anomaly package and contains three multivariate anomalies of length 15 located at $t = 100$, $t = 200$, and $t = 300$ for which the mean changes from 0 to 2. The anomalies affect variates 1 to 8, 1 to 12 and 1 to 16 respectively. Figure 3a shows a tile plot of the data and the anomaly locations as estimated by the following analysis.

```
R> data("simulated", package = "anomaly")
R> res <- capa(sim.data, type = "mean", min_seg_len = 2)
R> plot(res, subset = 1:20)
```

Clearly the overall positions of the anomalies have been located correctly however, many false positive anomalous segments have been fitted across most of the variates. This issue arises because the default penalty used by `capa` is tuned towards detection accuracy at the expense of false positive control in the number of components fitted as anomalous. False positive control can be recovered, at a loss of power against anomalies weakly affecting a lot of components, by using regime 2 from Fisch *et al.* (2021):

```
R> beta <- 2 * log(ncol(sim.data):1)
R> beta[1] <- beta[1] + 3 * log(nrow(sim.data))
R> res <- capa(sim.data, type = "mean", min_seg_len = 2, beta = beta)
R> plot(res, subset = 1:20)
```

As is apparent from Figure 3b, CAPA now controls false positives. Unfortunately, in general, optimal power and false positive control in the number of variates cannot both be achieved, as shown by Cai, Jeng, and Jin (2011).

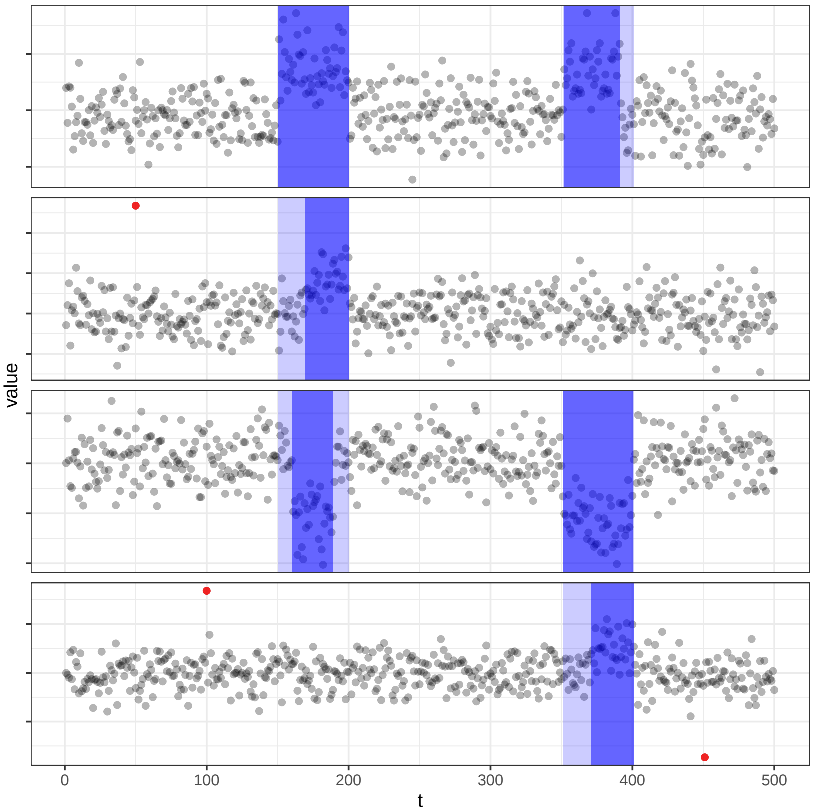


Figure 4: CAPA output on simulated data in Section 3.2. Collective anomalies are colored in blue, with lags displayed in light blue.

Simulated data 2

As mentioned previously, a maximum lag can be used when it is suspected that the collective anomalies do not perfectly align. This requires minor modifications to the argument structure

```
R> set.seed(0)
R> x1 <- rnorm(500)
R> x2 <- rnorm(500)
R> x3 <- rnorm(500)
R> x4 <- rnorm(500)
R> x1[151:200] <- x1[151:200] + 2
R> x2[171:200] <- x2[171:200] + 2
R> x3[161:190] <- x3[161:190] - 3
R> x1[351:390] <- x1[371:390] + 2
R> x3[351:400] <- x3[351:400] - 3
R> x4[371:400] <- x4[371:400] + 2
R> x4[451] <- x4[451] * max(1, abs(1 / x4[451])) * 6
R> x4[100] <- x4[100] * max(1, abs(1 / x4[100])) * 6
R> x2[050] <- x2[050] * max(1, abs(1 / x2[050])) * 6
R> x1 <- (x1 - median(x1)) / mad(x1)
R> x2 <- (x2 - median(x2)) / mad(x2)
```

```
R> x3 <- (x3 - median(x3)) / mad(x3)
R> x4 <- (x4 - median(x4)) / mad(x4)
R> x <- cbind(x1, x2, x3, x4)
R> res <- capa(x, max_lag = 20, type = "mean")
R> plot(res)
```

The output of this analysis can be found in Figure 4.

4. Proportion adaptive segment selection (PASS)

The **anomaly** package includes a scalable implementation of the proportion adaptive segment selection (PASS) algorithm by Jeng *et al.* (2012). PASS is an algorithm designed to efficiently scan long multi-variate sequences of data using a test statistic that detects multivariate collective anomalies in mean and/or variance. For each candidate collective anomaly, with a given start point and end point, s and e say, PASS tests each component individually for a mean anomaly thus obtaining p values $q_1, \dots, q_p$. These component specific p values are then ordered $q_{(1)} \leq \dots \leq q_{(p)}$ and combined into a test-statistic, q, for the segment (s, e) that combines information across the p values using higher criticism (Donoho and Jin 2004):

$$q = \max_{\alpha_0 \leq i \leq p} \left(\sqrt{p} \frac{\frac{i}{p} - q_{(i)}}{\sqrt{q_{(i)}(1 - q_{(i)})}} \right) \tag{5}$$

for an integer $\alpha_0 \geq 1$. To fit a multiple collective anomalies, an algorithm similar to circular binary segmentation (Olshen, Venkatraman, Lucito, and Wigler 2004), is used.

PASS inherits most of its hyper parameters and properties from higher criticism. In particular, it is often suggested to set $\alpha_0 > 1$, i.e., to disregard some of the lowest p values when using higher criticism to stabilize the procedure. However, this can lead to anomalies affecting fewer than α_0 components escaping detection. Furthermore, their approach requires selecting a suitable threshold value λ, which is typically increased with the data dimension n and p. Low values of α_0 can also make inflation of λ advisable; guidance on which is given in Jeng *et al.* (2012, Section 3.1). The method has been implemented in compiled code by following steps 1 to 8 in Jeng *et al.* (2012, Section 2.2) and has computational complexity $\mathcal{O}(\texttt{max\_seg\_len} \cdot np \log(p))$.

The **anomaly** package provides the function `pass` which accepts the following 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.

- `alpha`: An integer value greater then 0 corresponding to α_0 in Jeng *et al.* (2012). This value is used to stabilize the higher criticism based test statistic used by PASS leading to a better finite sample family-wise error rate. Anomalies affecting fewer than alpha components will be more likely to escape detection. The default value is 2.

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

- `max_seg_len`: A positive integer corresponding to the maximum segment length. This parameter corresponds to the maximum interval length, L, in Jeng *et al.* (2012). The default value is 10.

- `min_seg_len`: A positive integer (`max_seg_len` $\geq$ `min_seg_len` > 0) corresponding to the minimum segment length. The default value is 1.

4.1. PASS: Simulated example

The following code demonstrates how the `pass` method provided by **anomaly** is used. In this example the data, `sim.data`, is the same as that used in the `capa` example in Section 3.2.

```
R> library("anomaly")
R> data("simulated", package = "anomaly")
R> res <- pass(sim.data, max_seg_len = 20, alpha = 3)
R> collective_anomalies(res)
```

```
  start end      xstar
1   200 214 1519317784
2   100 114   42907782
3   300 315   22296743
```

The results show the start and end of each anomaly along with `xstar` denoting the value of the higher criticism test statistic for the segment, that is the value of q as defined by (5). Larger values indicate more evidence for a collective anomaly, and the segments are listed in decreasing order of `xstar`. The results are consistent with those provided by `capa` in that the three anomalies are all detected. However, unlike MVCAPA, PASS does not indicate which series are anomalous.

5. Bayesian abnormal region detector (BARD)

The Bayesian abnormal region detector (BARD, Bardwell and Fearnhead 2017) is a fully Bayesian method for estimating abnormal regions in multivariate data. It assumes that data has been normalized so that data for each variate in a normal region has mean 0 and variance 1, and that abnormal regions correspond to a change in mean. Specifically, the model is a special case of (2), where the parameter, $\theta^{(i)}(t)$ is the mean of variate i at time point t, and we model that, conditional on the parameters, the data are independent Gaussian.

As it is a Bayesian approach, BARD differs from CAPA and PASS in two aspects. First, the user has to specify prior distributions for aspects of the model such as the mean in abnormal segments, and the length of normal and abnormal segments. Second, the output of the algorithm will be draws from a posterior distribution, which can be used to produce a single estimate of the location of the abnormal segments or give some measure of uncertainty about where the abnormal segments are. Like PASS, BARD only gives information about where the abnormal segments are located and not which variates are abnormal within each segment.

The parametric form of the prior distributions assumed by BARD are as follows. Segment lengths are assumed to have a negative binomial distribution, with parameters (k_N, p_N) for normal segments and (k_A, p_A) for abnormal segments, where a negative binomial random variable with parameters (k, p) has probability mass function

$$\mathsf{P}(X = x) = \left(\begin{array}{c} x + k - 1 \\ x \end{array} \right) (1-p)^k p^x,$$

with $\mathsf{E}(X) = \frac{k(1-p)}{p}$ and $\mathsf{VAR}(X) = \frac{k(1-p)}{p^2}$.

For an abnormal segment we need to further define a prior for the segment mean, μ, and this is assumed to uniform on a range for $|\mu|$, with the sign of the mean being equally likely to be positive or negative. We also need to specify the average proportion of variates affected by an abnormal segment, and the probability that an abnormal segment is followed by a further abnormal segment.

The BARD algorithm proceeds in two stages. First it calculates an approximation to the joint posterior distribution for the number and location of the abnormal segments. The approximation comes first from using numerical integration to calculate marginal likelihoods, and second from using probabilistic pruning (also known as resampling) within a particle filter to ensure the algorithm's complexity is linear in the number of time points. These parts of the algorithm can be controlled by the user, but empirical evidence in Bardwell and Fearnhead (2017) suggest that the approximation error when using the default choices is small.

The second step of BARD is to draw a number of independent samples from the posterior. The individual draws can either be plotted to give a sense of the uncertainty around where the abnormal segments are, or can be summarized by a single point estimate of their location. The **anomaly** package provides functions to do both of these. The approach taken to summarize the posterior by a single point estimate is to consider marginally each time-point, t, and the proportion of draws which place t within an abnormal segment. Our point-estimate flags point t as within an abnormal segment if and only if this proportion of draws is above some user-chosen threshold. See Figure 5 for an example.

The **anomaly** package provides the function `bard` which accepts the following 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.
- `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 $\frac{5\texttt{p\_a}}{1-\texttt{p\_a}}$.
- `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 misspecified 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 value of `x`.
- `alpha`: Threshold used to control the resampling in the approximation of the posterior distribution at each time step. A sensible default is 1×10^{-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.

5.1. BARD: Simulated example

The following code demonstrates how the `bard` method provided by **anomaly** can be used. In this example the data, `sim.data`, is the same as that used in Section 4, with PASS.

```
R> library("anomaly")
R> data("simulated", package = "anomaly")
R> bard.res <- bard(sim.data)
```

The priors (`p_N`, `k_N`, `p_A` and `k_A`) for the two length of stay distributions for normal and abnormal segments were chosen to be quite vague but with abnormal segments being much smaller than their normal counterparts. The mean (standard deviation) for normal segments is 190 (62) whereas for abnormal segments it is 10 (4). With no particular knowledge of the process in question we took the probability that an abnormal segment (`pi_N`) is followed by a normal segment as 90%. This was relatively arbitrary and assigned a high prior probability to the classic epidemic changepoint model but still allows for two abnormal segments to follow each other (albeit in different variates). The proportion of variates assumed to be affected by an abnormal segment (`paffected`) was taken to be 5% of the total number of variates. This proportion is small enough to be able to locate rare anomalies. The prior for the mean $\mu \sim \mathcal{U}(\textbf{lower}, \textbf{upper})$ was taken to be in the range of 0.5 to 1.5. The lower limit of 0.5 gives the minimum change in mean we are interested in detecting. To calculate the marginal likelihood of abnormal segments numerical integration was used with a step size (`h`) set at 0.25. In the example, the default value for the threshold parameter α of 1×10^{-4} has been used.

The `bard` function returns an `S4` class that includes the posterior distribution of the abnormal segments given the observed data. To obtain samples from the posterior, and, from these, posterior estimates for the location collective anomalies the `sampler` function is used.

```
R> sampler.res <- sampler(bard.res, gamma = 1/3, num_draws = 1000)
R> show(sampler.res)
```

```
BARD sampler detecting changes in mean
observations = 500
variates = 200
Collective anomalies detected : 3
  start end LogMargLike
3   199 213    319.8889
4   299 313    311.7028
2    99 113    177.8095
```

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

A number of samples (`num_draws`) are taken from the posterior, and from these estimates of the location of the collective anomalies are obtained based on the asymmetric loss (Bardwell and Fearnhead 2017) using the parameter `gamma`. This loss will estimate a location t as part of a collective anomaly if the proportion of posterior samples that have t in a collective anomaly is greater than $1/(1+\gamma)$. The show function reports the resulting estimated collective anomalies, together with a measure of the evidence (`LogMargLike`) for the collective anomaly in terms of the log marginal likelihood for the region being a collective anomaly rather than a part of a normal region. Larger values imply a stronger anomaly, which is dependent on the length, change magnitude and number of affected variates of each collective anomaly. The example shows that the location and relative strength of the three abnormal segments are broadly consistent with those obtained using `capa` and `pass`.

As with `capa` and `pass`, the `plot` function can be used to visualize the data. However, the argument `marginals` can be used to display additional information as shown in Figure 5. The top plot shows different realizations from the posterior distribution. The values are either 0 or 1 indicating if the time point in the realization is part of a collective anomaly or not. The marginal probability of each time point being a collective anomaly is shown in the bottom plot. This is the fraction of sampled realizations that were found to be anomalous at each time. The dashed horizontal line is the threshold, for our choice of `gamma = 1/3`, for which a collective anomaly is inferred.

6. CAPA and PASS: Microarray data

This example examines microarray data for 20 individuals with a bladder tumor from the ACGH data set which is available in the **ecp** package (James and Matteson 2014).

```
R> library("ecp")
R> data("ACGH", package = "ecp")
R> acgh <- ACGH[[1]][, 1:20]
```

The data is highly autocorrelated so we transform each individual variate using (4) for both `capa` and `pass` to avoid false positives.

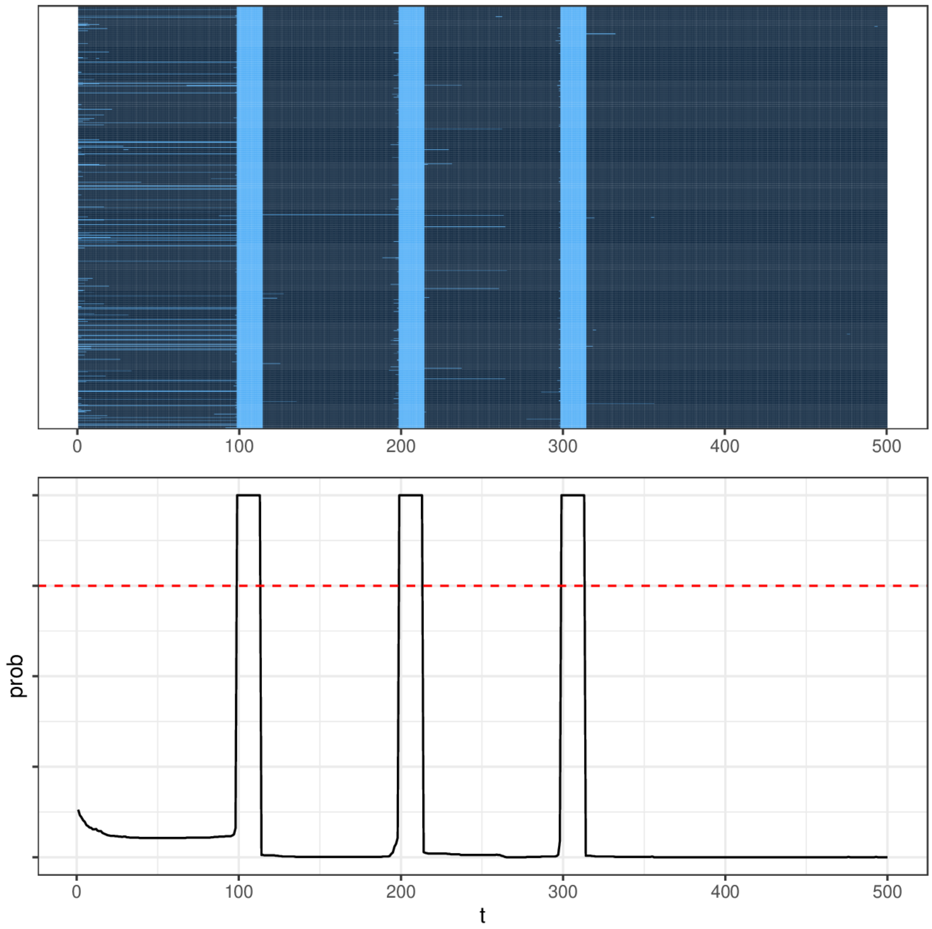


Figure 5: Plot of a sample of 1000 draws from the posterior distribution (top), each realization is shown in a separate row with light blue regions indicating the time points in that realization have been inferred as part of a collective anomaly. The marginal probability of a collective anomaly (bottom), with the user defined threshold (for $\gamma = 1/3$) for flagging a region as a collective anomaly (horizontal dashed line).

```
R> ac_corrected <- function(X) {
+    n <- length(X)
+    rcor <- covMcd(matrix(c(X[2:n], X[1:(n-1)]), ncol = 2), cor = TRUE)
+    psi <- rcor$cor[1, 2]
+    correction_factor <- sqrt((1 - psi) / (1 + psi))
+    return(correction_factor * (X - median(X)) / mad(X))
+ }
R> acgh <- acgh |> data.frame() |> sapply(ac_corrected)
```

The `acgh` data is analysed using `max_seg_len = 200` to ensure that detected anomalies can cover the whole length of a single chromosome and, for `capa`, the maximum lag was set to 50 to prevent adjacent anomalies overlapping.

```
R> res.capa <- capa(acgh, type = "mean", max_lag = 50, max_seg_len = 200)
```

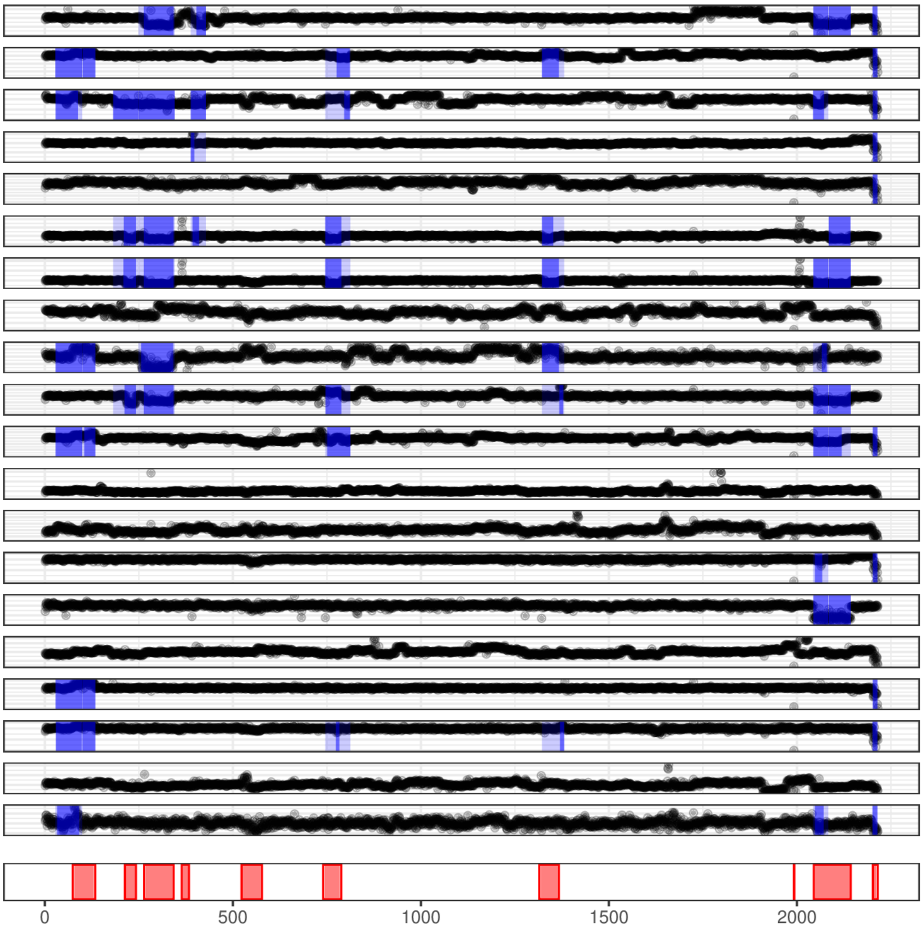


Figure 6: Locations of the 10 strongest anomalies for `capa` (blue) and `pass` (red).

```
R> res.pass <- pass(acgh, max_seg_len = 200, alpha = 3)
```

Both methods find that most of the region is anomalous (at least relative to the mean-0 model) so, for clarity of explanation, only the locations of the strongest 10 collective anomalies for each method are considered. These are shown in Figure 6 which demonstrates that the two sets of predicted anomalies are well alligned. Note however, only CAPA can distinguish between individuals that are affected by a specific anomaly or not.

7. Discussion

The detection of anomalous points and regions within data sequences and time series is becoming an increasingly important in many fields, from astrophysics to digital networking. The **anomaly** package implements a number of recently proposed, computationally efficient statistical approaches, accessible via a simple, easy to use R interface. Package **anomaly** provides a first implementation of the CAPA family of anomaly detection methods that can be used to detect both point anomalies in otherwise normal segments of data, as well as detecting collective anomalies. It also provides implementations of the BARD and PASS methods.

Distinctive plot classes have also been developed, allowing for the clear differentiation of anomaly types in both univariate and multivariate settings.

Each of the introduced methods are founded on independence assumptions though as described, in practice, they may be adapted to handle some (moderate) autocorrelation. Extension of these approaches to more general, time-dependent settings is the subject of current research. We hope to make such methods available within the package in due course, together with recently developed methods that allow for cross-dependence between series Tveten, Eckley, and Fearnhead (2022).

Each of the introduced methods also assumes that non-anomalous data is drawn from an underlying data generating process where the mean and variance are constant. In practice, trend and seasonality may be present within time series of interest. In such settings one might consider detrending/deseasonalising the data prior to running, e.g., CAPA. We advise care with such pre-processing, to ensure that anomalies of interest are not distorted by the process.

Acknowledgments

Fisch gratefully acknowledges funding provided by EPSRC and BT under the Industrial CASE scheme, and support from the EPSRC-funded STOR-i Centre for Doctoral Training. In addition, Eckley & Fearnhead gratefully acknowledge the financial support of EPSRC and BT through grants EP/N031938/1 and EP/R004935/1.

Section 3.1 uses data which is publicly available from the Numenta Anomaly Benchmark corpus (Ahmad *et al.* 2017) which can be accessed at `https://github.com/numenta/NAB`. It is included, with permission, in the anomaly package. Please remember to acknowledge Ahmad *et al.* (2017) when using the data.

References

Ahmad S, Lavin A, Purdy S, Agha Z (2017). "Unsupervised Real-Time Anomaly Detection for Streaming Data." *Neurocomputing*, **262**, 134–147. `doi:10.1016/j.neucom.2017.04.070`.

Ahmed M, Mahmood AN, Islam MR (2016). "A Survey of Anomaly Detection Techniques in Financial Domain." *Future Generation Computer Systems*, **55**, 278–288. `doi:10.1016/j.future.2015.01.001`.

Arundo Analytics, Inc (2020). ***adtk**: A Package for Unsepervised Time Series Anomaly Detection.* Python package version 0.6.2, URL `https://pypi.org/project/adtk/`.

Bardwell L, Fearnhead P (2017). "Bayesian Detection of Abnormal Segments in Multiple Time Series." *Bayesian Analysis*, **12**(1), 193–218. `doi:10.1214/16-ba998`.

Bleakley K, Vert JP (2011). "The Group Fused Lasso for Multiple Change-Point Detection." *arXiv 1106.4199*, arXiv.org E-Print Archive. `doi:10.48550/arXiv.1106.4199`.

Cai TT, Jeng JX, Jin J (2011). "Optimal Detection of Heterogeneous and Heteroscedastic Mixtures." *Journal of the Royal Statistical Society B*, **73**(5), 629–662. `doi:10.1111/j.1467-9868.2011.00778.x`.

Chandola V, Banerjee A, Kumar V (2009). "Anomaly Detection: A Survey." *ACM Computing Surveys*, **41**. doi:10.1145/1541880.1541882.

Dancho M, Vaughan D (2023). ***anomalize**: Tidy Anomaly Detection.* R package version 0.3.0, URL https://CRAN.R-project.org/package=anomalize.

Donoho D, Jin J (2004). "Higher Criticism for Detecting Sparse Heterogeneous Mixtures." *The Annals of Statistics*, **32**(3), 962–994. doi:10.1214/009053604000000265.

Fisch ATM, Eckley IA, Fearnhead P (2021). "Subset Multivariate Collective And Point Anomaly Detection." *Journal of Computational and Graphical Statistics*, **31**(2), 574–585. doi:10.1080/10618600.2021.1987257.

Fisch ATM, Eckley IA, Fearnhead P (2022). "A Linear Time Method for the Detection of Point and Collective Anomalies." *Statisical Analysis and Data Mining*, **15**, 494–508. doi:10.1002/sam.11586.

Fisch ATM, Grose DJ, Bardwell L, Eckley IA, Fearnhead P (2024). ***anomaly**: Detecting Anomalies in Data.* R package version 4.3.3, URL https://CRAN.R-project.org/package=anomaly.

Flach M, Gans F, Brenning A, Denzler J, Reichstein M, Rodner E, Bathiany S, Bodesheim P, Guanche Y, Sippel S, Mahecha MD (2017). "Multivariate Anomaly Detection for Earth Observations: A Comparison of Algorithms and Feature Extraction Techniques." *Earth System Dynamics*, **8**(3), 677–696. doi:10.5194/esd-8-677-2017.

Goh J, Adepu S, Tan M, Lee ZS (2017). "Anomaly Detection in Cyber Physical Systems Using Recurrent Neural Networks." In *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, pp. 140–145. IEEE. doi:10.1109/HASE.2017.36.

Iturria A, Carrasco J, Herrera F, Charramendieta S, Intxausti K (2019). ***otsad**: Online Time Series Anomaly Detectors.* R package version 0.2.0, URL https://CRAN.R-project.org/package=otsad.

Jackson B, Scargle JD, Barnes D, Arabhi S, Alt A, Gioumousis P, Gwin E, Sangtrakulcharoen P, Tan L, Tsai TT (2005). "An Algorithm for Optimal Partitioning of Data on an Interval." *IEEE Signal Processing Letters*, **12**(2), 105–108. doi:10.1109/lsp.2001.838216.

James NA, Matteson DS (2014). "**ecp**: An R Package for Nonparametric Multiple Change Point Analysis of Multivariate Data." *Journal of Statistical Software*, **62**(7), 1–25. doi:10.18637/jss.v062.i07.

Jeng XJ, Cai TT, Li H (2012). "Simultaneous Discovery of Rare and Common Segment Variants." *Biometrika*, **100**(1), 157–172. doi:10.1093/biomet/ass059.

Killick R, Eckley IA (2014). "**changepoint**: An R Package for Changepoint Analysis." *Journal of Statistical Software*, **58**(3), 1–19. doi:10.18637/jss.v058.i03.

Killick R, Fearnhead P, Eckley IA (2012). "Optimal Detection of Changepoints with a Linear Computational Cost." *Journal of the American Statistical Association*, **107**(500), 1590–1598. doi:10.1080/01621459.2012.737745.

López-de-Lacalle J (2024). ***tsoutliers**: Detection of Outliers in Time Series.* R package version 0.6-10, URL https://CRAN.R-project.org/package=tsoutliers.

Lavielle M, Moulines E (2000). "Least-Squares Estimation of an Unknown Number of Shifts in a Time Series." *Journal of Time Series Analysis*, **21**(1), 33–59. doi:10.1111/1467-9892.00172.

Lavin A, Ahmad S (2015). "Evaluating Real-Time Anomaly Detection Algorithms – The Numenta Anomaly Benchmark." In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pp. 38–44. doi:10.1109/ICMLA.2015.141.

Olshen AB, Venkatraman ES, Lucito R, Wigler M (2004). "Circular Binary Segmentation for the Analysis of Array-Based DNA Copy Number Data." *Biostatistics*, **5**(4), 557–572. doi:10.1093/biostatistics/kxh008.

Seonghyun K (2017). ***cbar**: Contextual Bayesian Anomaly Detection in R.* R package version 0.1.3, URL https://CRAN.R-project.org/package=cbar.

Škvára V, Pevný T, Šmídl V (2018). "Are Generative Deep Models for Novelty Detection Truly Better?" *arXiv 1807.05027*, arXiv.org E-Print Archive. doi:10.48550/arXiv.1807.05027.

Talagala PD (2020). ***stray**: Anomaly Detection in High Dimensional and Temporal Data.* R package version 0.1.1, URL https://CRAN.R-project.org/package=stray.

Talagala PD, Hyndman RJ, Smith-Miles K (2019). ***oddstream**: Outlier Detection in Data Streams.* R package version 0.5.0, URL https://CRAN.R-project.org/package=oddstream.

Theissler A (2017). "Detecting Known and Unknown Faults in Automotive Systems Using Ensemble-Based Anomaly Detection." *Knowledge-Based Systems*, **123**, 163–173. doi:10.1016/j.knosys.2017.02.023.

Tveten M, Eckley IA, Fearnhead P (2022). "Scalable Change-Point and Anomaly Detection in Cross-Correlated Data with an Application to Condition Monitoring." *The Annals of Applied Statistics*, **16**(2), 721–743. doi:10.1214/21-aoas1508.

Vercruyssen V (2022). ***anomatools**: A Compact Toolbox for Semi-Supervised Anomaly Detection.* Python package version 3.0.3, URL https://pypi.org/project/anomatools/.

Vilmaz SF (2023). ***pysad**: Python Streaming Anomaly Detection.* Python package version 0.2.0, URL https://pypi.org/project/pysad/.

Wickham H (2016). ***ggplot2**: Elegant Graphics for Data Analysis.* Springer-Verlag. doi:10.1007/978-0-387-98141-3.

Zhao H, Liu H, Hu W, Yan X (2018). "Anomaly Detection and Fault Analysis of Wind Turbine Components Based on Deep Learning Network." *Renewable Energy*, **127**, 825–834. doi:10.1016/j.renene.2018.05.024.

Affiliation:

Daniel Grose
Department of Mathematics and Statistics
Fylde College
Lancaster University
LA1 4YF, United Kingdom
E-mail: dan.grose@lancaster.ac.uk
URL: https://www.lancaster.ac.uk/sci-tech/about-us/people/daniel-grose

Journal of Statistical Software https://www.jstatsoft.org/
published by the Foundation for Open Access Statistics https://www.foastat.org/
August 2024, Volume 110, Issue 1 *Submitted:* 2020-10-15
doi:10.18637/jss.v110.i01 *Accepted:* 2023-12-21