Package ‘DBEST’

January 20, 2025

Type Package

Title Detecting Breakpoints and Estimating Segments in Trend

Version 1.8

Date 2017-10-23

Author Sadegh Jamali, Hristo Tomov

Maintainer Hristo Tomov <tomov.hristo@gmail.com>

Description A program for analyzing vegetation time series, with two algorithms: 1) change detection algorithm that detects trend changes, determines their type (abrupt or non-abrupt), and estimates their timing, magnitude, number, and direction; 2) generalization algorithm that simplifies the temporal trend into main features. The user can set the number of major breakpoints or magnitude of greatest changes of interest for detection, and can control the generalization process by setting an additional parameter of generalization-percentage.

License GPL-2

Imports graphics, stats, zoo

NeedsCompilation no

Repository CRAN

Date/Publication 2017-10-23 13:50:47 UTC

Contents

DBEST *Detecting Breakpoints and Estimating Segments in Trend (DBEST)*

Description

A program for analyzing vegetation time series, with two algorithms: 1) change detection algorithm that detects trend changes, determines their type (abrupt or non-abrupt), and estimates their timing, magnitude, number, and direction; 2) generalization algorithm that simplifies the temporal trend into main features. The user can set the number of major breakpoints or magnitude of greatest changes of interest for detection, and can control the generalization process by setting an additional parameter of generalization-percentage.

Usage

```
DBEST(data, data.type, seasonality = -1, algorithm, breakpoints.no = -1,
generalization.percent = -1, change.magnitude = -1, first.level.shift,
second.level.shift, duration, distance.threshold, alpha, plot = -1)
```

Arguments

`data` univariate time-series to be analysed. This should be an object of class 'ts'/'zoo' with a frequency greater than one without NA's or a vector without NA's. If the input data is a vector the algorithm will automatically assign a start year.

`data.type` the data type. There are two options: "cyclical" for time series with a seasonal cycle, or "non-cyclical" for time series without seasonal cycle (e.g. deseasonalized data)

`seasonality` the seasonality period as a number. If the input data type is non-cyclical this variable should be omitted or set to 'none'/'null'. This parameter will overwrite the frequency value of an input object of class 'ts'/'zoo'. However, if the input data is an object of class 'ts'/'zoo' and the seasonality period is omitted, then the algorithm will use as a seasonality period the frequency value from the input data.

`algorithm` the algorithm mode. There are two options: "change detection" and "generalization".

`breakpoints.no` the number of greatest changes to be detected (change detection algorithm); the number of major breakpoints to be included in the generalized trend (generalization algorithm). This parameter should be omitted if 'generalization.percent' or 'change.magnitude' is in use.

`change.magnitude` the lowest magnitude for the changes to be detected (change detection algorithm); the largest variation allowed within a generalized segment (generalization algorithm). This parameter should be omitted if 'breakpoints.no' or 'generalization.percent' is in use.

`generalization.percent`
: the highest percent (between 0 to 100) the trend should be generalized; (0 = the least-simplified trend; 100 = the most-simplified trend). This parameter should be omitted if 'breakpoints.no' or 'change.magnitude' is in use.

`first.level.shift`
: the first level-shift-threshold value. This parameter corresponds to the lowest absolute difference in the time-series data between the level-shift point (abrupt change) and the next data point.

`second.level.shift`
: the second level-shift-threshold value. This parameter corresponds to the lowest absolute difference in the means of the data calculated over the duration period before and after the level-shift point.

`duration`
: the duration threshold value. This parameter corresponds to the lowest time period (time steps) within which the shift in the mean of the data level, before and after the level-shift point, persists; and, the lowest spacing (time steps) between successive level-shift points.

`distance.threshold`
: the distance threshold value. This correspond to the the lowest perpendicular distance from farthest data point to the straight line passing through every pair of successive peak and valley points. The algorithm will estimate a distance threshold if this parameter is set to 'default'.

`alpha`
: the statistical significance level value used for testing the significance of detected changes.

`plot`
: display figures. This parameter could be omitted or set to: "on", "fig1", "fig2" or "off". The "fig1" option will display the input data and the estimated trend, plus the trend local change. The "fig2" option will display a graph with the decomposition of the time-series, including the actual data, the trend, the seasonal component and the remainder. The "on" option displays both 'figure 1' and 'figure 2'. The "off" option displays no figure.

Details

An object of the class "DBEST" is a list with elements depending on whether the generalization algorithm or change detection algorithm is used.

Value

`BreakpointNo`
: the number of breakpoints or changes detected.

`SegmentNo`
: the number of segments estimated by the algorithm.

`Start`
: a list with numbers representing the starting points of the changes as time-steps.

`Duration`
: a list with numbers representing the durations of the changes as time-steps.

`End`
: a list with numbers representing the ending points of the changes as time-steps.

`Change`
: a list with the values of the changes.

`ChangeType`
: a list with the types of the changes as numbers which could be 0 or 1. The numbers correspond to a non-abrupt change (0) or abrupt change (1).

`Significance`	a list with the statistical significances of the changes as numbers which could be 0 or 1. The numbers correspond to a statistically in-significant change (0) or significant change (1).
`RMSE`	the calculated Root Mean Squares Error of the fit.
`MAD`	the calculated Maximum Absolute Difference of the fit.

Note

1) DBEST detects changes requested by the user, and determines the type of the detected changes based on the definition of abrupt change (level-shift) made by user. The user can define what properties a data point must have, based on the studied application, to be considered as abrupt change or a level-shift. This is done using the three arguments of: first.level.shift, second.level.shift, and duration. For example, for the vegetation change application using monthly NDVI time-series studied in Jamali et al. 2015, an abrupt changes is a one time-step change >= 0.1 (NDVI units) that results in a shift >= 0.2 (NDVI units) in the mean level of NDVI, and the shift is valid for at-least two years (24 months).

2) Dashed vertical lines mark the STARTING point of detected changes.

3) Abrupt changes are in RED and non-abrupt changes are in ORANGE.

4) Here, DBEST uses a seasonal-trend decomposition method (STL) with slightly different setting parameters compared to that used in Jamali et al. 2015. This may lead to little change to the plots of the test data below compared to figures 4 and 5 published in Jamali et al. 2015.

Author(s)

Sadegh Jamali, Hristo Tomov

References

Jamali S, Jönsson P, Eklundh L, Ardö J, Seaquist J (2015). Detecting changes in vegetation trends using time series segmentation. Remote Sensing of Environment, 156, 182-195. http://dx.doi.org/10.1016/j.rse.2014.09.010

Tomov H (2016). Automated temporal NDVI analysis over the Middle East for the period 1982 – 2010. http://lup.lub.lu.se/student-papers/record/8871893

See Also

`plot.DBEST` for plotting of DBEST() results.

Examples

```
# NDVI data for Site 1 and Site 2 used in Fig. 5, Jamali et al. 2015
data(NDVI.Site1)
NDVI.Site1 <- ts(NDVI.Site1, start=1982, frequency=12)

data(NDVI.Site2)
NDVI.Site2 <- ts(NDVI.Site2, start=1982, frequency=12)

# Trend of NDVI data for Site 1 and Site 2 used in Fig. 4, Jamali et al. 2015)
data(TREND.Site1)
```

```
data(TREND.Site2)

# Examples for DBEST's change detection algorithm
# detecting three greatest changes in NDVI (Fig. 5a, b)
DBEST.Fig5a <- DBEST(data=NDVI.Site1, data.type="cyclical",
                 seasonality=12, algorithm="change detection",
                 breakpoints.no=3, first.level.shift=0.1,
                 second.level.shift=0.2, duration=24,
                 distance.threshold="default", alpha=0.05, plot="on")

DBEST.Fig5b <- DBEST(data=NDVI.Site2, data.type="cyclical",
                 seasonality=12, algorithm="change detection",
                 breakpoints.no=3, first.level.shift=0.1,
                 second.level.shift=0.2, duration=24,
                 distance.threshold="default", alpha=0.05, plot="on")

# detecting changes >= 0.2 NDVI units
DBEST.examp1 <- DBEST(data=NDVI.Site1, data.type="cyclical",
                  seasonality=12, algorithm="change detection",
                  change.magnitude=0.2, first.level.shift=0.1,
                  second.level.shift=0.2, duration=24,
                  distance.threshold="default", alpha=0.05, plot="fig1")

DBEST.examp2 <- DBEST(data=TREND.Site2, data.type="non-cyclical",
                  seasonality="none", algorithm="change detection",
                  change.magnitude=0.2, first.level.shift=0.1,
                  second.level.shift=0.2, duration=24,
                  distance.threshold="default", alpha=0.05, plot="fig1")

## Not run:
# Examples for DBEST's generalization algorithm
# the most-simplified trend
DBEST.Fig4a <- DBEST(data=TREND.Site1, data.type="non-cyclical",
                   seasonality="none", algorithm="generalization",
                   generalization.percent=100, first.level.shift=0.1,
                   second.level.shift=0.2, duration=24,
                   distance.threshold="default", alpha=0.05, plot="fig1")

DBEST.examp3 <- DBEST(data=NDVI.Site2, data.type="cyclical",
                  seasonality=12, algorithm="generalization",
                  generalization.percent=100, first.level.shift=0.1,
                  second.level.shift=0.2, duration=24,
                  distance.threshold="default", alpha=0.05, plot="fig1")

# one breakpoint included in the generalized trend
DBEST.Fig4b <- DBEST(data=TREND.Site1, data.type="non-cyclical",
                 seasonality="none", algorithm="generalization",
                 breakpoints.no=1, first.level.shift=0.1,
                 second.level.shift=0.2, duration=24,
                 distance.threshold="default", alpha=0.05, plot="fig1")

DBEST.examp4 <- DBEST(data=NDVI.Site2, data.type="cyclical",
                  seasonality=12, algorithm="generalization",
```

```
                  breakpoints.no=1, first.level.shift=0.1,
                  second.level.shift=0.2, duration=24,
                  distance.threshold="default", alpha=0.05, plot="fig1")

# the largest variation allowed within the generalized segments <= 0.1 NDVI units
DBEST.Fig4c <- DBEST(data=TREND.Site1, data.type="non-cyclical",
                 seasonality="none", algorithm="generalization",
                 change.magnitude=0.1, first.level.shift=0.1,
                 second.level.shift=0.2, duration=24,
                 distance.threshold="default", alpha=0.05, plot="fig1")

DBEST.examp5 <- DBEST(data=NDVI.Site2, data.type="cyclical",
                  seasonality=12, algorithm="generalization",
                  change.magnitude=0.2, first.level.shift=0.1,
                  second.level.shift=0.1, duration=24,
                  distance.threshold="default", alpha=0.05, plot="fig1")

# the least-simplified trend
DBEST.Fig4d <- DBEST(data=TREND.Site1, data.type="non-cyclical",
                 seasonality="none", algorithm="generalization",
                 generalization.percent=0, first.level.shift=0.1,
                 second.level.shift=0.2, duration=24,
                 distance.threshold="default", alpha=0.05, plot="fig1")

DBEST.examp6 <- DBEST(data=NDVI.Site2, data.type="cyclical",
                  seasonality=12, algorithm="generalization",
                  generalization.percent=0, first.level.shift=0.1,
                  second.level.shift=0.2, duration=24,
                  distance.threshold="default", alpha=0.05, plot="fig1")

## End(Not run)
```

NDVI.Site1 *Site 1*

Description

NDVI Test data (Site 1 in Jamali et al. 2015)

Usage

```
data("NDVI.Site1")
```

Format

The format is: num [1:300] 0.507 0.582 0.466 0.661 0.708 ...

Source

Jamali S, Jönsson P, Eklundh L, Ardö J, Seaquist J (2015). Detecting changes in vegetation trends using time series segmentation. Remote Sensing of Environment, 156, 182-195. http://dx.doi.org/10.1016/j.rse.2014.09.010

Examples

```
data(NDVI.Site1)
## maybe str(NDVI.Site1) ; plot(NDVI.Site1) ...
```

`NDVI.Site2` *Site 2*

Description

NDVI Test data (Site 2 in Jamali et al. 2015)

Usage

```
data("NDVI.Site2")
```

Format

The format is: num [1:300] 0.125 0.2 0.121 0.155 0.206 ...

Source

Jamali S, Jönsson P, Eklundh L, Ardö J, Seaquist J (2015). Detecting changes in vegetation trends using time series segmentation. Remote Sensing of Environment, 156, 182-195. http://dx.doi.org/10.1016/j.rse.2014.09.010

Examples

```
data(NDVI.Site2)
## maybe str(NDVI.Site2) ; plot(NDVI.Site2) ...
```

`plot.DBEST` *Methods for objects of class "DBEST".*

Description

Plot methods for objects of class "DBEST".

Usage

```
## S3 method for class 'DBEST'
plot(x, figure = 1, ...)
```

Arguments

x	DBEST object.
figure	If omitted, the plot function displays 'figure 1' and 'figure 2'. If this option is set to 1, the plot function will display the input data and the estimated trend, plus the trend local change (figure 1). If this option is set to 2, the plot function will display a graph with the decomposition of the time-series, including the actual data, the trend, the seasonal component and the remainder.
...	further arguments passed to the plot function.

Author(s)

Hristo Tomov

See Also

See also as DBEST

Examples

```
## Not run:
# Display figure 1 - the input data and the estimated trend, plus the trend local change.
plot(DBEST.obj, figure=1)

# Display figure 2 - the decomposition of the time-series, including the actual data,
# the trend, the seasonal component and the remainder.
plot(DBEST.obj, figure=2)


## End(Not run)
```

TREND.Site1 *Site 1*

Description

Trend of NDVI Test data (Site 1 in Jamali et al. 2015)

Usage

```
data("TREND.Site1")
```

Format

The format is: num [1:300] 0.507 0.582 0.466 0.661 0.708 ...

Source

Jamali S, Jönsson P, Eklundh L, Ardö J, Seaquist J (2015). Detecting changes in vegetation trends using time series segmentation. Remote Sensing of Environment, 156, 182-195. http://dx.doi.org/10.1016/j.rse.2014.09.010

Examples

```
data(TREND.Site1)
## maybe str(TREND.Site1) ; plot(TREND.Site1) ...
```

TREND.Site2 *Site 2*

Description

Trend of NDVI Test data (Site 2 in Jamali et al. 2015)

Usage

```
data("TREND.Site2")
```

Format

The format is: num [1:300] 0.125 0.2 0.121 0.155 0.206 ...

Source

Jamali S, Jönsson P, Eklundh L, Ardö J, Seaquist J (2015). Detecting changes in vegetation trends using time series segmentation. Remote Sensing of Environment, 156, 182-195. http://dx.doi.org/10.1016/j.rse.2014.09.010

Examples

```
data(TREND.Site2)
## maybe str(TREND.Site2) ; plot(TREND.Site2) ...
```

Index