Package ‘SiZer’

February 19, 2015

Version 0.1-4
Date 2011-3-21
Title SiZer: Significant Zero Crossings
Author Derek Sonderegger <derek.sonderegger@gmail.com>
Maintainer Derek Sonderegger <derek.sonderegger@gmail.com>
Depends R (>= 2.4.0), splines, boot

Description Calculates and plots the SiZer map for scatterplot data.
   A SiZer map is a way of examining when the p-th derivative of a
   scatterplot-smoother is significantly negative, possibly zero
   or significantly positive across a range of smoothing
   bandwidths.

License GPL (>= 2)
URL http://www.r-project.org
Repository CRAN
Date/Publication 2012-10-29 08:57:41
NeedsCompilation no

R topics documented:

   Arkansas ................................................................. 2
   bent.cable .............................................................. 2
   locally.weighted.polynomial ...................................... 4
   piecewise.linear ..................................................... 5
   plot.LocallyWeightedPolynomial .................................. 7
   plot.SiZer ............................................................. 8
   SiZer ................................................................. 10

Index 12
**Arkansas**  
*Time Series of Macroinvertebrates Abundance in the Arkansas River.*

**Description**

A time series of 16 years (5 replicates per year) of mayfly (Ephemeroptera:Heptageniidae) abundance in the fall at the monitoring station AR1.

**Usage**

```r
data(Arkansas)
```

**Format**

A data frame with 90 observations on the following 2 variables.

- `year` a numeric vector
- `sqrt.mayflies` a numeric vector of abundance values

**Source**


**Examples**

```r
data(Arkansas)
plot(Arkansas)
```

---

**bent.cable**  
*Fits a bent-cable model to the given data*

**Description**

Fits a bent-cable model to the given data by exhaustively searching the 2-dimensional parameter space to find the maximum likelihood estimators for $\alpha$ and $\gamma$.

**Usage**

```r
bent.cable(x, y, grid.size=100)
```

**Arguments**

- `x` The independent variable
- `y` The dependent variable
- `grid.size` How many $\alpha$ and $\gamma$ values to examine. The total number of parameter combinations examined is `grid.size` squared.
Details

Fit the model which is essentially a piecewise linear model with a quadratic curve of length $2\gamma$ connecting the two linear pieces.

The reason for searching the space exhaustively is because the bent-cable model often has a likelihood surface with a very flat ridge instead of definite peak. While the exhaustive search is slow, at least it is possible to examine the contour plot of the likelihood surface.

Value

A list of 7 elements:

- loglikelihood: A matrix of log-likelihood values.
- SSE: A matrix of sum-of-square-error values.
- alphas: A vector of alpha values examined.
- gammas: A vector of gamma values examined.
- alpha: The MLE estimate of alpha.
- gamma: The MLE estimate of gamma.
- model: The \texttt{lm} fit after \textit{alpha} and \textit{gamma} are known.

Author(s)

Derek Sonderegger

References


See Also

\texttt{piecewise.linear}

Examples

data(arkansas)
x <- Arkansas$year
y <- Arkansas$sqrt.mayflies

# For a more accurate estimate, increase grid.size
model <- bent.cable(x, y, grid.size=20)
plot(x, y)
x.grid <- seq(min(x), max(x), length=200)
lines(x.grid, predict(model, x.grid), col='red')
locally.weighted.polynomial

Locally-Weighted Polynomial Regression Smoother

Description

Smoothes the given bivariate data using kernel regression.

Usage

locally.weighted.polynomial(x, y = NA, x.grid = NA,
degree = 1, kernel.type = "Normal")

Arguments

x Vector of data for the independent variable
y Vector of data for the dependent variable
h The bandwidth for the kernel
x.grid What x-values should the value of the smoother be calculated at.
degree The degree of the polynomial to be fit at each x-value. The default is to fit a
linear regression, i.e., degree = 1.
kernelforme What kernel to use. Valid choices are 'Normal', 'Epanechnikov', 'biweight',
and 'triweight'

Details

The confidence intervals are created using the row-wise method of Hannig and Marron (2006).
Notice that the derivative to be estimated must be less than or equal to the degree of the polynomial
initially fit to the data.
If the bandwidth is not given, the Sheather-Jones bandwidth selection method is used.

Value

Returns a LocallyWeightedPolynomial object that has the following elements:

data A structure of the data used to generate the smoothing curve
h The bandwidth used to generate the smoothing curve.
x.grid The grid of x-values that we have estimated function value and derivative(s) for.
degrees.freedom The effective sample size at each grid point
beta A matrix of estimated beta values. The number of rows is degrees+1, while the
number of columns is the same as the length of x.grid. Notice that
\[ \hat{f}(x_i) = \beta[1, i] \]
Fit a piecewise linear model

Fit a degree 1 spline with 1 knot point where the location of the knot point is unknown.
Usage

```r
piecewise.linear(x, y, middle = 1, CI = FALSE,
bootstrap.samples = 1000, sig.level = 0.05)
```

Arguments

- `x`: Vector of data for the x-axis.
- `y`: Vector of data for the y-axis.
- `middle`: A scalar in $[0, 1]$. This represents the range that the change-point can occur in. 0 means the change-point must occur at the middle of the range of x-values. 1 means that the change-point can occur anywhere along the range of the x-values.
- `CI`: Whether or not a bootstrap confidence interval should be calculated.
- `bootstrap.samples`: The number of bootstrap samples to take.
- `sig.level`: What significance level to use for the confidence intervals.

Details

The bootstrap samples are taken by resampling the raw data points. Often a more appropriate bootstrap sample would be to calculate the residuals and then add a randomly selected residual to each y-value.

Value

A list of 5 elements is returned:

- `change.point`: The estimate of $\alpha$.
- `model`: The resulting `lm` object once $\alpha$ is known.
- `x`: The x-values used.
- `y`: The y-values used.
- `CI`: Whether or not the confidence interval was calculated.
- `intervals`: If the CIs were calculated, this is a matrix of the upper and lower intervals.

Author(s)

Derek Sonderegger

References


plot.LocallyWeightedPolynomial

See Also

~~objects to See Also as help, ~~~

Examples

data(Arkansas)
x <- Arkansas$year
y <- Arkansas$sqrt.mayflies

model <- piecewise.linear(x, y, CI=FALSE)
plot(model)
print(model)
predict(model, 2001)

plot.LocallyWeightedPolynomial

Plot a LocallyWeightedPolynomial object

Description

 Creates a plot of an object created by locally.weighted.polynomial.

Usage

## S3 method for class 'LocallyWeightedPolynomial'
plot(x, deriv = 0, CI.method = 2,
     alpha = 0.05, use.ess = TRUE, draw.points = TRUE, ...)

Arguments

x LocallyWeightedPolynomial object
derv Derivative to be plotted. Default is 0 - which plots the smoothed function.
CI.method What method should be used to calculate the confidence interval about the estimated line. The methods are from Hannig and Marron (2006), where 1 is the point-wise estimate, and 2 is the row-wise estimate.
alpha The CI has a 1-alpha/2 level of significance.
use.ess ESS stands for the estimated sample size. If at any point along the x-axis, the ESS is too small, then we will not plot unless use.ess=FALSE.
draw.points Should the data points be included in the graph?
... Additional arguments to be passed to the graphing functions.

Author(s)

Derek Sonderegger
References


See Also

locally.weighted.polynomial

Examples

data('Arkansas')
x <- Arkansas$year
y <- Arkansas$sqrt.mayflies

model <- locally.weighted.polynomial(x,y)
plot(model)

model <- locally.weighted.polynomial(x,y,degree=2)
plot(model, deriv=1)
plot(model, deriv=2)

_________

plot.SiZer

**Plot a SiZer map**

Description

Plot a SiZer object that was created using SiZer().

Usage

```r
## S3 method for class 'SiZer'
## S3 method for class 'SiZer'
plot(x, ylab = expression(log[10](h)),
colorlist = c("red", "purple", "blue", "grey"), ...)
```

Arguments

- `x` An object created using SiZer()
- `ylab` What the y-axis should be labeled.
- `colorlist` What colors should be used. This is a vector that corresponds to 'decreasing', 'possibly zero', 'increasing', 'insufficient data'
- `...` Any other parameters to be passed to the function `image`.
Details

The white lines in the SiZer map give a graphical representation of the bandwidth. The horizontal distance between the lines is $2h$.

Value

None

Author(s)

Derek Sonderegger

References


See Also

SiZer, locally.weighted.polynomial

Examples

# data('Arkansas')
# x <- Arkansas$year
# y <- Arkansas$sqrt.mayflies

# Calculate the SiZer map for the first derivative
# SiZer.1 <- SiZer(x, y, h=c(.5,10), degree=1, derv=1)
# plot(SiZer.1)

# Calculate the SiZer map for the second derivative
# SiZer.2 <- SiZer(x, y, h=c(.5,10), degree=2, derv=2);
# plot(SiZer.2)

# By setting the grid.length larger, we get a more detailed SiZer
# map but it takes longer to compute.
#
# SiZer.3 <- SiZer(x, y, h=c(.5,10), grid.length=100, degree=1, derv=1)
# plot(SiZer.3)
SiZer

**Calculate SiZer Map**

---

**Description**

Calculates the SiZer map from a given set of X and Y variables.

**Usage**

SiZer(x, y, h=NA, x.grid=NA, degree=NA, derv=1, grid.length=41)

**Arguments**

- **x**: data vector for the independent axis
- **y**: data vector for the dependent axis
- **h**: An integer representing how many bandwidths should be considered, or vector of length 2 representing the upper and lower limits h should take, or a vector of length greater than two indicating which bandwidths to examine.
- **x.grid**: An integer representing how many bins to use along the x-axis, or a vector of length 2 representing the upper and lower limits the x-axis should take, or a vector of length greater than two indicating which x-values the derivative should be evaluated at.
- **grid.length**: The default length of the h.grid or x.grid if the length of either is not given.
- **derv**: The order of derivative for which to make the SiZer map.
- **degree**: The degree of the local weighted polynomial used to smooth the data. This must be greater than or equal to derv.

**Details**

SiZer stands for the Significant Zero crossings of the derivative. There are two dominate approaches in smoothing bivariate data: locally weighted regression or penalized splines. Both approaches require the use of a ‘bandwidth’ parameter that controls how much smoothing should be done. Unfortunately there is no uniformly best bandwidth selection procedure. SiZer (Chaudhuri and Marron, 1999) is a procedure that looks across a range of bandwidths and classifies the p-th derivative of the smoother into one of three states: significantly increasing (blue), possibly zero (purple), or significantly negative (red).

**Value**

Returns an SiZer object which has the following components:

- **x.grid**: Vector of x-values at which the derivative was evaluated.
- **h.grid**: Vector of bandwidth values for which a smoothing function was calculated.
- **slopes**: Matrix of what category a particular x-value and bandwidth falls into (Increasing=1, Possibly Zero=0, Decreasing=-1, Not Enough Data=2).
SiZer

Author(s)
Derek Sonderegger

References

See Also
plot.Nsizer, locally.weighted.polynomial

Examples
data('Arkansas')
x <- Arkansas$year
y <- Arkansas$sqrt.mayflies
plot(x,y)

# Calculate the SiZer map for the first derivative
SiZer.1 <- SiZer(x, y, h=c(.5,10), degree=1, derv=1)
plot(SiZer.1)

# Calculate the SiZer map for the second derivative
SiZer.2 <- SiZer(x, y, h=c(.5,10), degree=2, derv=2);
plot(SiZer.2)

# By setting the grid.length larger, we get a more detailed SiZer
# map but it takes longer to compute.
# SiZer.3 <- SiZer(x, y, h=c(.5,10), grid.length=100, degree=1, derv=1)
# plot(SiZer.3)
Index

*Topic **datasets**
Arkansas, 2

*Topic **hplot**
plot.LocallyWeightedPolynomial, 7
plot.Sizer, 8
Sizer, 10

*Topic **regression**
bent.cable, 2
piecewise.linear, 5

*Topic **smooth**
locally.weighted.polynomial, 4
plot.Sizer, 8
Sizer, 10

Arkansas, 2
bent.cable, 2
calc.CI.LocallyWeightedPolynomial
(locally.weighted.polynomial), 4

find.state
(locally.weighted.polynomial), 4
find.states
(locally.weighted.polynomial), 4

h.grid.create (Sizer), 10
help, 7
interpSpline, 5
kernel.h (locally.weighted.polynomial), 4

locally.weighted.polynomial, 4, 8, 9, 11
loess, 5
logLik.bent.cable (bent.cable), 2

logLik.PiecewiseLinear
(piecewise.linear), 5

piecewise.linear, 3, 5
plot.LocallyWeightedPolynomial, 5, 7
plot.PiecewiseLinear
(piecewise.linear), 5
plot.Sizer, 8, 11
positive.part
(locally.weighted.polynomial), 4
predict.bent.cable (bent.cable), 2
predict.PiecewiseLinear
(piecewise.linear), 5
print.PiecewiseLinear
(piecewise.linear), 5

Sizer, 5, 9, 10
smooth.spline, 5

x.grid.create
(locally.weighted.polynomial), 4

12