

Package ‘lpridge’

February 20, 2015

Version 1.0-7

Date 2013-04-17

Title Local Polynomial (Ridge) Regression

Author Burkhardt Seifert <seifert@ifspm.unizh.ch> (S original);
Packaged for R by Martin Maechler

Maintainer Martin Maechler <maechler@stat.math.ethz.ch>

Description Local Polynomial Regression with Ridging.

Depends stats

License GPL (>= 2)

NeedsCompilation yes

Repository CRAN

Date/Publication 2013-04-17 14:06:37

R topics documented:

| | |
|-------------------|---|
| lpepa | 1 |
| lpridge | 3 |

| | |
|--------------|----------|
| Index | 6 |
|--------------|----------|

| | |
|-------|--|
| lpepa | <i>Local polynomial regression fitting with Epanechnikov weights</i> |
|-------|--|

Description

Fast and stable algorithm for nonparametric estimation of regression functions and their derivatives via **local** polynomials with **Epanechnikov** weight function.

Usage

```
lpepa(x, y, bandwidth, deriv = 0, n.out = 200, x.out = NULL,  
      order = deriv+1, mnew = 100, var = FALSE)
```

Arguments

| | |
|-----------|--|
| x | vector of design points, not necessarily ordered. |
| y | vector of observations of the same length as x. |
| bandwidth | bandwidth(s) for nonparametric estimation. Either a number or a vector of the same length as x.out. |
| deriv | order of derivative of the regression function to be estimated; defaults to <code>deriv = 0</code> . |
| n.out | number of output design points where the function has to be estimated. The default is <code>n.out=200</code> . |
| x.out | vector of output design points where the function has to be estimated. The default value is an equidistant grid of n.out points from <code>min(x)</code> to <code>max(x)</code> . |
| order | integer, order of the polynomial used for local polynomials. Must be ≤ 10 and defaults to <code>order = deriv+1</code> . |
| mnew | integer forcing to restart the algorithm after mnew updating steps. The default is <code>mnew = 100</code> . For <code>mnew = 1</code> you get a numerically “super-stable” algorithm (see reference SBE&G below). |
| var | logical flag: if TRUE, the variance of the estimator proportional to the residual variance is computed (see details). |

Details

More details are described in the first reference SBE&G (1994) below. In S&G, a bad finite sample behaviour of local polynomials for random designs was found. For practical use, we therefore propose local polynomial regression fitting with ridging, as implemented in the function `lpridge`. In `lpepa`, several parameters described in SBE&G are fixed either in the fortran routine or in the R-function. There, you find comments how to change them.

For `var=TRUE`, the variance of the estimator proportional to the residual variance is computed, i.e., the exact finite sample variance of the regression estimator is $\text{var}(\text{est}) = \text{est.var} * \sigma^2$.

Value

a list including used parameters and estimator.

| | |
|-----------|---|
| x | vector of ordered design points. |
| y | vector of observations ordered according to x. |
| bandwidth | vector of bandwidths actually used for nonparametric estimation. |
| deriv | order of derivative of the regression function estimated. |
| x.out | vector of ordered output design points. |
| order | order of the polynomial used for local polynomials. |
| mnew | force to restart the algorithm after mnew updating steps. |
| var | logical flag: whether the variance of the estimator was computed. |
| est | estimator of the derivative of order <code>deriv</code> of the regression function. |
| est.var | estimator of the variance of <code>est</code> (proportional to residual variance). |

References

See also <http://www.unizh.ch/biostat/> under ‘Manuscripts’ etc.

- Numerical stability and computational speed:

B. Seifert, M. Brockmann, J. Engel and T. Gasser (1994) Fast algorithms for nonparametric curve estimation. *J. Computational and Graphical Statistics* **3**, 192–213.

- Statistical properties:

Seifert, B. and Gasser, T. (1996) Finite sample variance of local polynomials: Analysis and solutions. *J. American Statistical Association* **91**(433), 267–275.

Seifert, B. and Gasser, T. (2000) Data adaptive ridging in local polynomial regression. *J. Computational and Graphical Statistics* **9**, 338–360.

Seifert, B. and Gasser, T. (1998) Ridging Methods in Local Polynomial Regression. in: S. Weisberg (ed), *Dimension Reduction, Computational Complexity, and Information*, Vol. **30** of Computing Science & Statistics, Interface Foundation of North America, 467–476.

Seifert, B. and Gasser, T. (1998) Local polynomial smoothing. in: *Encyclopedia of Statistical Sciences*, Update Vol. **2**, Wiley, 367–372.

Seifert, B., and Gasser, T. (1996) Variance properties of local polynomials and ensuing modifications. in: *Statistical Theory and Computational Aspects of Smoothing*, W. Härdle, M. G. Schimek (eds), Physica, 50–127.

See Also

[lpridge](#), and also [lowess](#) and [loess](#) which do local linear and quadratic regression quite a bit differently.

Examples

```
data(cars)
attach(cars)

epa.sd <- lpepa(speed,dist, bandw=5) # local polynomials

plot(speed, dist, main = "data(cars) & lp epanechnikov regression")
lines(epa.sd$x.out, epa.sd$est, col="red")
lines(lowess(speed,dist, f= .5), col="orange")
detach()
```

lpridge

Local polynomial regression fitting with ridging

Description

Fast and stable algorithm for nonparametric estimation of regression functions and their derivatives via local polynomials and local polynomial ridge regression with polynomial weight functions.

Usage

```
lpridge(x, y, bandwidth, deriv=0, n.out=200, x.out=NULL,
order = NULL, ridge = NULL, weight = "epa", mnew = 100,
var = FALSE)
```

Arguments

| | |
|-----------|--|
| x | vector of design points, not necessarily ordered. |
| y | vector of observations of the same length as x. |
| bandwidth | bandwidth for nonparametric estimation. Either a number or a vector of the same length as x.out. |
| deriv | order of derivative of the regression function to be estimated; default is 0. |
| n.out | number of output design points at which to evaluate the estimator; defaults to 200. |
| x.out | vector of output design points at which to evaluate the estimator; By default, an equidistant grid of n.out points from $\min(x)$ to $\max(x)$. |
| order | order of the polynomial used for local polynomials. The default value is $\text{deriv} + 1$. |
| ridge | ridging parameter. The default value performs a slight ridging (see "Details"). $\text{ridge} = 0$ leads to the local polynomial estimator without ridging. |
| weight | kernel weight function. The default value is $\text{weight} = \text{"epa"}$ for Epanechnikov weights. Other weights are "bi" for biweights (square of "epa") and "tri" for triweights (cube of "epa"). If weight is a vector, it is interpreted as vector of coefficients of the polynomial weight function. Thus, $\text{weight} = \text{"epa"}$ is equivalent to $\text{weight} = c(1,0,-1)$. |
| mnew | force to restart the algorithm after mnew updating steps. The default value is $\text{mnew} = 100$. For $\text{mnew} = 1$ you get a numerically "super-stable" algorithm (see reference SBE&G below). |
| var | logical flag: if TRUE, the variance of the estimator proportional to the residual variance is computed (see "Details" below). |

Details

described in the reference SBE&G below. Several parameters described there are fixed either in the fortran routine or in the R-function. There, you find comments how to change them.

In S&G, a bad finite sample behavior of local polynomials for random design was found, and ridging of the estimator was proposed. In `lpridge()`, we use a ridging matrix corresponding to the smoothness assumption “*The squared difference of the derivative of order deriv of the regression function at the point of estimation and the weighted mean of design points is bounded by the residual variance divided by the ridge parameter.*”

Thus, without any smoothness assumption, $\text{ridge} = 0$ would be appropriate, and for a nearly constant derivative of order deriv , a ridge parameter going to infinity behaves well. For equidistant design, ridging influences the estimator only at the boundary. Asymptotically, the influence of any non-increasing ridge parameter vanishes.

So far, our experience with the choice of a ridging parameter is limited. Therefore we have chosen a default value which performs a slight modification of the local polynomial estimator (with denotations h = bandwidth, d = deriv, and where $n_0 = \text{length}(x) * \text{mean}(\text{bandwidth}) / \text{diff}(\text{range}(x))$ is a mean number of observations in a smoothing interval):

$$\text{ridge} = 5\sqrt{n_0}h^{2d} / ((2d + 3)(2d + 5))$$

For var=TRUE, the variance of the estimator proportional to the residual variance is computed, i.e., the exact finite sample variance of the regression estimator is $\text{var}(\text{est}) = \text{est.var} * \text{sigma}^2$.

Value

a list including used parameters and estimator.

| | |
|-----------|--|
| x | vector of ordered design points. |
| y | vector of observations ordered according to x. |
| bandwidth | vector of bandwidths actually used for nonparametric estimation. |
| deriv | order of derivative of the regression function estimated. |
| x.out | vector of ordered output design points. |
| order | order of the polynomial used for local polynomials. |
| ridge | ridging parameter used. |
| weight | vector of coefficients of the kernel weight function. |
| mnew | force to restart the algorithm after mnew updating steps. |
| var | logical flag: whether the variance of the estimator was computed. |
| est | estimator of the derivative of order deriv of the regression function. |
| est.var | estimator of the variance of est (proportional to residual variance). |

References

The same as for [lpepa](#).

Examples

```
data(cars)
attach(cars)
plot(speed, dist, main = "data(cars) & lpRIDGE Regression")

myfit <- lpridge(speed,dist,bandw = 5, ridge=0) # local polynomials
lines(myfit$x.out,myfit$est,col=2)

myridge <- lpridge(speed,dist,bandw = 5) # local pol. ridge
lines(myridge$x.out,myridge$est,col=3)
mtext("bandw = 5")
legend(5,120, c("ridge = 0", "default ridging"), col = 2:3, lty = 1)
detach()
```

Index

*Topic **smooth**

lpepa, 1

lpridge, 3

loess, 3

lowess, 3

lpepa, 1, 5

lpridge, 2, 3, 3