|  |  |
| --- | --- |
| loess {stats} | R Documentation |

**Local Polynomial Regression Fitting**

**Description**

Fit a polynomial surface determined by one or more numerical predictors, using local fitting.

**Usage**

loess(formula, data, weights, subset, na.action, model = FALSE,

span = 0.75, enp.target, degree = 2,

parametric = FALSE, drop.square = FALSE, normalize = TRUE,

family = c("gaussian", "symmetric"),

method = c("loess", "model.frame"),

control = loess.control(...), ...)

**Arguments**

|  |  |
| --- | --- |
| formula | a [formula](http://127.0.0.1:14695/library/stats/help/formula) specifying the numeric response and one to four numeric predictors (best specified via an interaction, but can also be specified additively). Will be coerced to a formula if necessary. |
| data | an optional data frame, list or environment (or object coercible by [as.data.frame](http://127.0.0.1:14695/library/stats/help/as.data.frame) to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which loess is called. |
| weights | optional weights for each case. |
| subset | an optional specification of a subset of the data to be used. |
| na.action | the action to be taken with missing values in the response or predictors. The default is given by getOption("na.action"). |
| model | should the model frame be returned? |
| span | the parameter *α* which controls the degree of smoothing. |
| enp.target | an alternative way to specify span, as the approximate equivalent number of parameters to be used. |
| degree | the degree of the polynomials to be used, normally 1 or 2. (Degree 0 is also allowed, but see the ‘Note’.) |
| parametric | should any terms be fitted globally rather than locally? Terms can be specified by name, number or as a logical vector of the same length as the number of predictors. |
| drop.square | for fits with more than one predictor and degree = 2, should the quadratic term be dropped for particular predictors? Terms are specified in the same way as for parametric. |
| normalize | should the predictors be normalized to a common scale if there is more than one? The normalization used is to set the 10% trimmed standard deviation to one. Set to false for spatial coordinate predictors and others known to be on a common scale. |
| family | if "gaussian" fitting is by least-squares, and if "symmetric" a re-descending M estimator is used with Tukey's biweight function. Can be abbreviated. |
| method | fit the model or just extract the model frame. Can be abbreviated. |
| control | control parameters: see [loess.control](http://127.0.0.1:14695/library/stats/help/loess.control). |
| ... | control parameters can also be supplied directly (*if* control is not specified). |

**Details**

Fitting is done locally. That is, for the fit at point *x*, the fit is made using points in a neighbourhood of *x*, weighted by their distance from *x* (with differences in ‘parametric’ variables being ignored when computing the distance). The size of the neighbourhood is controlled by *α* (set by span or enp.target). For *α < 1*, the neighbourhood includes proportion *α* of the points, and these have tricubic weighting (proportional to *(1 - (dist/maxdist)^3)^3*). For *α > 1*, all points are used, with the ‘maximum distance’ assumed to be *α^(1/p)* times the actual maximum distance for *p* explanatory variables.

For the default family, fitting is by (weighted) least squares. For family="symmetric" a few iterations of an M-estimation procedure with Tukey's biweight are used. Be aware that as the initial value is the least-squares fit, this need not be a very resistant fit.

It can be important to tune the control list to achieve acceptable speed. See [loess.control](http://127.0.0.1:14695/library/stats/help/loess.control) for details.

**Value**

An object of class "loess".

**Note**

As this is based on cloess, it is similar to but not identical to the loess function of S. In particular, conditioning is not implemented.

The memory usage of this implementation of loess is roughly quadratic in the number of points, with 1000 points taking about 10Mb.

degree = 0, local constant fitting, is allowed in this implementation but not documented in the reference. It seems very little tested, so use with caution.

**Author(s)**

B. D. Ripley, based on the cloess package of Cleveland, Grosse and Shyu.

**Source**

The 1998 version of cloess package of Cleveland, Grosse and Shyu. A later version is available as dloess at <http://www.netlib.org/a>.

**References**

W. S. Cleveland, E. Grosse and W. M. Shyu (1992) Local regression models. Chapter 8 of *Statistical Models in S* eds J.M. Chambers and T.J. Hastie, Wadsworth & Brooks/Cole.

**See Also**

[loess.control](http://127.0.0.1:14695/library/stats/help/loess.control), [predict.loess](http://127.0.0.1:14695/library/stats/help/predict.loess).

[lowess](http://127.0.0.1:14695/library/stats/help/lowess), the ancestor of loess (with different defaults!).

**Examples**

cars.lo <- loess(dist ~ speed, cars)

predict(cars.lo, data.frame(speed = seq(5, 30, 1)), se = TRUE)

# to allow extrapolation

cars.lo2 <- loess(dist ~ speed, cars,

control = loess.control(surface = "direct"))

predict(cars.lo2, data.frame(speed = seq(5, 30, 1)), se = TRUE)