lpoly — Kernel-weighted local polynomial smoothing

Description

`lpoly` performs a kernel-weighted local polynomial regression of *yvar* on *xvar* and displays a graph of the smoothed values with (optional) confidence bands.

Quick start

Kernel-weighted local polynomial regression of `y` on `x`

```
lpoly y x
```

Same as above, but specify a bandwidth of 2

```
lpoly y x, bwidth(2)
```

Same as above, but specify a degree of 1

```
lpoly y x, bwidth(2) degree(1)
```

Same as above, but use the alternative Epanechnikov kernel

```
lpoly y x, bwidth(2) degree(1) kernel(epan2)
```

Same as above, but create a new variable for the smoothing grid `g` and smoothed values `s`

```
lpoly y x, bwidth(2) degree(1) kernel(epan2) generate(g s)
```

With 95% confidence bands

```
lpoly y x, ci
```

Use `twoway` to graph multiple local polynomial fits

```
twoway scatter y x ||                    ///
    lpoly y x, degree(1) kernel(epan2) ||      ///
    lpoly y x, degree(1) kernel(epan2) bwidth(1) || ///
    lpoly y x, degree(1) kernel(epan2) bwidth(7) ||
```

Menu

Statistics > Nonparametric analysis > Local polynomial smoothing

Syntax

```
lpoly yvar xvar [if] [in] [weight] [, options]
```

options	Description
Main	
`kernel(`*kernel*`)`	specify kernel function; default is `kernel(epanechnikov)`
`bwidth(`#|*varname*`)`	specify kernel bandwidth
`degree(`#`)`	specify degree of the polynomial smooth; default is `degree(0)`
`generate(`[*newvar*$_x$] *newvar*$_s$`)`	store smoothing grid in *newvar*$_x$ and smoothed points in *newvar*$_s$
`n(`#`)`	obtain the smooth at # points; default is min(N, 50)
`at(`*varname*`)`	obtain the smooth at the values specified by *varname*
`nograph`	suppress graph
`noscatter`	suppress scatterplot only
SE/CI	
`ci`	plot confidence bands
`level(`#`)`	set confidence level; default is `level(95)`
`se(`*newvar*`)`	store standard errors in *newvar*
`pwidth(`#`)`	specify pilot bandwidth for standard error calculation
`var(`#|*varname*`)`	specify estimates of residual variance
Scatterplot	
marker_options	change look of markers (color, size, etc.)
marker_label_options	add marker labels; change look or position
Smoothed line	
`lineopts(`*cline_options*`)`	affect rendition of the smoothed line
CI plot	
`ciopts(`*cline_options*`)`	affect rendition of the confidence bands
Add plots	
`addplot(`*plot*`)`	add other plots to the generated graph
Y axis, X axis, Titles, Legend, Overall	
twoway_options	any options other than `by()` documented in [G-3] ***twoway_options***

kernel	Description
epanechnikov	Epanechnikov kernel function; the default
epan2	alternative Epanechnikov kernel function
biweight	biweight kernel function
cosine	cosine trace kernel function
gaussian	Gaussian kernel function
parzen	Parzen kernel function
rectangle	rectangle kernel function
triangle	triangle kernel function

collect is allowed; see [U] 11.1.10 Prefix commands.

fweights and aweights are allowed; see [U] 11.1.6 weight.

Options

Main

kernel(*kernel*) specifies the kernel function for use in calculating the weighted local polynomial estimate. The default is kernel(epanechnikov).

bwidth(# | *varname*) specifies the half-width of the kernel—the width of the smoothing window around each point. If bwidth() is not specified, a rule-of-thumb (ROT) bandwidth estimator is calculated and used. A local variable bandwidth may be specified in *varname*, in conjunction with an explicit smoothing grid using the at() option.

degree(#) specifies the degree of the polynomial to be used in the smoothing. The default is degree(0), meaning local-mean smoothing.

generate([*newvar*$_x$] *newvar*$_s$) stores the smoothing grid in *newvar*$_x$ and the smoothed values in *newvar*$_s$. If at() is not specified, then both *newvar*$_x$ and *newvar*$_s$ must be specified. Otherwise, only *newvar*$_s$ is to be specified.

n(#) specifies the number of points at which the smooth is to be calculated. The default is $\min(N, 50)$, where N is the number of observations.

at(*varname*) specifies a variable that contains the values at which the smooth should be calculated. By default, the smoothing is done on an equally spaced grid, but you can use at() to instead perform the smoothing at the observed x's, for example. This option also allows you to more easily obtain smooths for different variables or different subsamples of a variable and then overlay the estimates for comparison.

nograph suppresses drawing the graph of the estimated smooth. This option is often used with the generate() option.

noscatter suppresses superimposing a scatterplot of the observed data over the smooth. This option is useful when the number of resulting points would be so large as to clutter the graph.

SE/CI

`ci` plots confidence bands, using the confidence level specified in `level()`.

`level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is `level(95)` or as set by `set level`; see **[U] 20.8 Specifying the width of confidence intervals**.

`se(`*newvar*`)` stores the estimates of the standard errors in *newvar*. This option requires specifying `generate()` or `at()`.

`pwidth(#)` specifies the pilot bandwidth to be used for standard error computations. The default is chosen to be 1.5 times the value of the ROT bandwidth selector. If you specify `pwidth()` without specifying `se()` or `ci`, then the `ci` option is assumed.

`var(#|`*varname*`)` specifies an estimate of a constant residual variance or a variable containing estimates of the residual variances at each grid point required for standard error computation. By default, the residual variance at each smoothing point is estimated by the normalized weighted residual sum of squares obtained from locally fitting a polynomial of order $p + 2$, where p is the degree specified in `degree()`. `var(`*varname*`)` is allowed only if `at()` is specified. If you specify `var()` without specifying `se()` or `ci`, then the `ci` option is assumed.

Scatterplot

marker_options affect the rendition of markers drawn at the plotted points, including their shape, size, color, and outline; see [G-3] ***marker_options***.

marker_label_options specify if and how the markers are to be labeled; see [G-3] ***marker_label_options***.

Smoothed line

`lineopts(`*cline_options*`)` affects the rendition of the smoothed line; see [G-3] ***cline_options***.

CI plot

`ciopts(`*cline_options*`)` affects the rendition of the confidence bands; see [G-3] ***cline_options***.

Add plots

`addplot(`*plot*`)` provides a way to add other plots to the generated graph; see [G-3] ***addplot_option***.

Y axis, X axis, Titles, Legend, Overall

twoway_options are any of the options documented in [G-3] ***twoway_options***, excluding `by()`. These include options for titling the graph (see [G-3] ***title_options***) and for saving the graph to disk (see [G-3] ***saving_option***).

Remarks and examples

Remarks are presented under the following headings:

Introduction

The last 25 years or so has seen a significant outgrowth in the literature on scatterplot smoothing, otherwise known as univariate nonparametric regression. Of most appeal is the idea of making no assumptions about the functional form for the expected value of a response given a regressor, but instead allowing the data to "speak for themselves". Various methods and estimators fall into the category of nonparametric regression, including local mean smoothing as described independently by Nadaraya (1964) and Watson (1964), the Gasser and Müller (1979) estimator, locally weighted scatterplot smoothing (LOWESS) as described by Cleveland (1979), wavelets (for example, Donoho [1995]), and splines (Eubank 1999), to name a few. Much of the vast literature focuses on automating the amount of smoothing to be performed and dealing with the bias/variance tradeoff inherent to this type of estimation. For example, for Nadaraya–Watson the amount of smoothing is controlled by choosing a *bandwidth*.

Smoothing via local polynomials is by no means a new idea but instead one that has been rediscovered in recent years in articles such as Fan (1992). A natural extension of the local mean smoothing of Nadaraya–Watson, local polynomial regression involves fitting the response to a polynomial form of the regressor via locally weighted least squares. Higher-order polynomials have better bias properties than the zero-degree local polynomials of the Nadaraya–Watson estimator; in general, higher-order polynomials do not require bias adjustment at the boundary of the regression space. For a definitive reference on local polynomial smoothing, see Fan and Gijbels (1996).

Local polynomial smoothing

Consider a set of scatterplot data $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ from the model

$$y_i = m(x_i) + \sigma(x_i)\epsilon_i \tag{1}$$

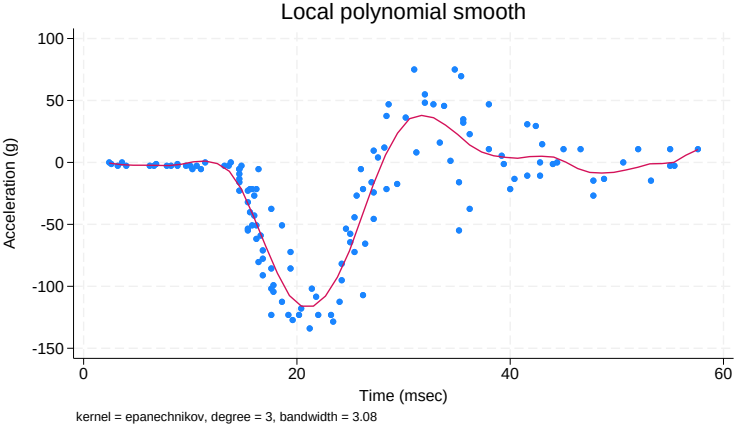
for some unknown mean and variance functions $m(\cdot)$ and $\sigma^2(\cdot)$, and symmetric errors ϵ_i with $E(\epsilon_i) = 0$ and $\text{Var}(\epsilon_i) = 1$. The goal is to estimate $m(x_0) = E[Y|X = x_0]$, making no assumption about the functional form of $m(\cdot)$.

`lpoly` estimates $m(x_0)$ as the constant term (intercept) of a regression, weighted by the kernel function specified in `kernel()`, of *yvar* on the polynomial terms $(xvar - x_0), (xvar - x_0)^2, \ldots, (xvar - x_0)^p$ for each smoothing point x_0. The degree of the polynomial, p, is specified in `degree()`, the amount of smoothing is controlled by the bandwidth specified in `bwidth()`, and the chosen kernel function is specified in `kernel()`.

▷ Example 1

Consider the motorcycle data as examined (among other places) in Fan and Gijbels (1996). The data consist of 133 observations and measure the acceleration (`accel` measured in grams [g]) of a dummy's head during impact over time (`time` measured in milliseconds). For these data, we use `lpoly` to fit a local cubic polynomial with the default bandwidth (obtained using the ROT method) and the default Epanechnikov kernel.

```
. use https://www.stata-press.com/data/r19/motorcycle
(Motorcycle data from Fan & Gijbels (1996))
. lpoly accel time, degree(3)
```



◁

❑ Technical note

`lpoly` allows specifying in `degree()` both odd and even orders of the polynomial to be used for the smoothing. However, the odd-order, $2k+1$, polynomial approximations are preferable. They have an extra parameter compared with the even-order, $2k$, approximations, which leads to a significant bias reduction and there is no increase of variability associated with adding this extra parameter. Using an odd order when estimating the regression function is therefore usually sufficient. For a more thorough discussion, see Fan and Gijbels (1996).

❑

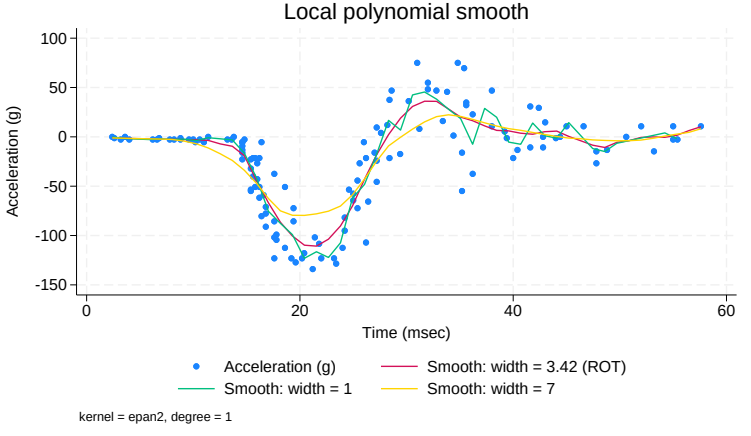
Choice of a bandwidth

The choice of a bandwidth is crucial for many smoothing techniques, including local polynomial smoothing. In general, using a large bandwidth gives smooths with a large bias, whereas a small bandwidth may result in highly variable smoothed values. Various techniques exist for optimal bandwidth selection. By default, `lpoly` uses the ROT method to estimate the bandwidth used for the smoothing; see *Methods and formulas* for details.

▷ Example 2

Using the `motorcycle` data, we demonstrate how a local linear polynomial fit changes using different bandwidths.

```
. lpoly accel time, degree(1) kernel(epan2) bwidth(1)
>  generate(at smooth1) nograph
. lpoly accel time, degree(1) kernel(epan2) bwidth(7) at(at)
> generate(smooth2) nograph
. label variable smooth1 "Smooth: width = 1"
. label variable smooth2 "Smooth: width = 7"
. lpoly accel time, degree(1) kernel(epan2) at(at) addplot(line smooth* at)
> legend(label(2 "Smooth: width = 3.42 (ROT)") cols(2) pos(6))
> note("kernel = epan2, degree = 1")
```

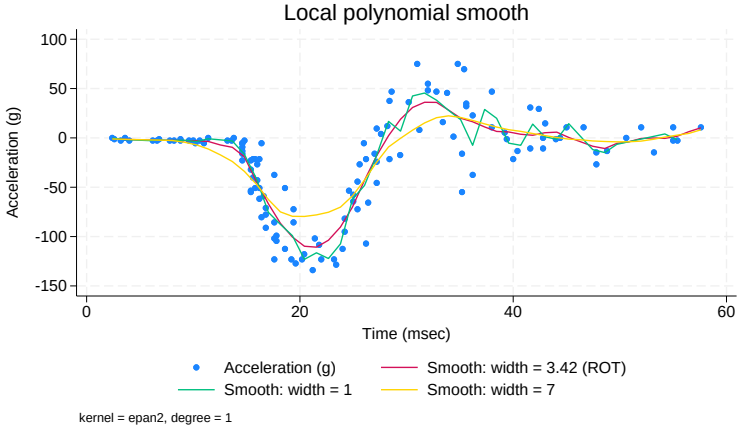


From this graph, we can see that the local linear polynomial fit with larger bandwidth (`width` = 7) corresponds to a smoother line but fails to fit the curvature of the scatterplot data. The smooth obtained using the width equal to one seems to fit most data points, but the corresponding line has several spikes indicating larger variability. The smooth obtained using the ROT bandwidth estimator seems to have a good tradeoff between the fit and variability in this example.

In the above, we also demonstrated how the `generate()` and `addplot()` options may be used to produce overlaid plots obtained from `lpoly` with different options. The `nograph` option saves time when you need to save only results with `generate()`.

However, to avoid generating variables manually, one can use `twoway lpoly` instead; see [G-2] **graph twoway lpoly** for more details.

```
. twoway scatter accel time  ||
>        lpoly accel time, degree(1) kernel(epan2) lpattern(solid) ||
>        lpoly accel time, degree(1) kernel(epan2) bwidth(1)       ||
>        lpoly accel time, degree(1) kernel(epan2) bwidth(7)       ||
>      , legend(label(2 "Smooth: width = 3.42 (ROT)")
>               label(3 "Smooth: width = 1")
>               label(4 "Smooth: width = 7") cols(2) pos(6))
>        title("Local polynomial smooth") note("kernel = epan2, degree = 1")
>        xtitle("Time (msec)") ytitle("Acceleration (g)")
```



The ROT estimate is commonly used as an initial guess for the amount of smoothing; this approach may be sufficient when the choice of a bandwidth is less important. In other cases, you can pick your own bandwidth.

When the shape of the regression function has a combination of peaked and flat regions, a variable bandwidth may be preferable over the constant bandwidth to allow for different degrees of smoothness in different regions. The `bwidth()` option allows you to specify the values of the local variable bandwidths as those stored in a variable in your data.

Similar issues with bias and variability arise when choosing a pilot bandwidth (the `pwidth()` option) used to compute standard errors of the local polynomial smoother. The default value is chosen to be $1.5 \times$ ROT. For a review of methods for pilot bandwidth selection, see Fan and Gijbels (1996).

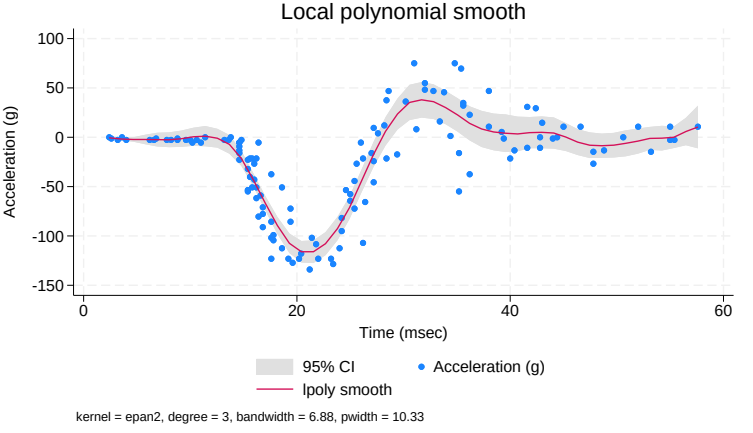
Confidence bands

The established asymptotic normality of the local polynomial estimators under certain conditions allows the construction of approximate confidence bands. `lpoly` offers the `ci` option to plot these bands.

▷ Example 3

Let us plot the confidence bands for the local polynomial fit from example 1.

```
. lpoly accel time, degree(3) kernel(epan2) ci legend(cols(2) pos(6))
```



You can obtain graphs with overlaid confidence bands by using `twoway lpolyci`; see [G-2] **graph twoway lpolyci** for examples.

◁

Constructing the confidence intervals involves computing standard errors obtained by taking a square root of the estimate of the conditional variance of the local polynomial estimator at each grid point x_0. Estimating the conditional variance requires fitting a polynomial of a higher order locally by using a different bandwidth, the pilot bandwidth. The value of the pilot bandwidth may be supplied by using `pwidth()`. By default, the value of $1.5 \times \text{ROT}$ is used. Also, estimates of the residual variance $\sigma^2(x_0)$ at each grid point, x_0, are required to obtain the estimates of the conditional variances. These estimates may be supplied by using the `var()` option. By default, they are computed using the normalized weighted residual sum of squares from a local polynomial fit of a higher order. See *Methods and formulas* for details. The standard errors may be saved by using `se()`.

Stored results

`lpoly` stores the following in `r()`:

Scalars

`r(degree)`	smoothing polynomial degree	`r(bwidth)`	bandwidth of the smooth
`r(ngrid)`	number of successful regressions	`r(pwidth)`	pilot bandwidth
`r(N)`	sample size		

Macros

`r(kernel)`	name of kernel

Methods and formulas

Consider model (1), written in matrix notation,

$$\mathbf{y} = m(\mathbf{x}) + \boldsymbol{\epsilon}$$

where $\mathbf{y}$ and $\mathbf{x}$ are the $n \times 1$ vectors of scatterplot values, $\boldsymbol{\epsilon}$ is the $n \times 1$ vector of errors with zero mean and covariance matrix $\boldsymbol{\Sigma} = \text{diag}\{\sigma(x_i)\}\mathbf{I}_n$, and $m()$ and $\sigma()$ are some unknown functions. Define $m(x_0) = E[Y|X = x_0]$ and $\sigma^2(x_0) = \text{Var}[Y|X = x_0]$ to be the conditional mean and conditional variance of random variable Y (residual variance), respectively, for some realization x_0 of random variable X.

The method of local polynomial smoothing is based on the approximation of $m(x)$ locally by a pth order polynomial in $(x - x_0)$ for some x in the neighborhood of x_0. For the scatterplot data $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, the pth-order local polynomial smooth $\widehat{m}(x_0)$ is equal to $\hat{\beta}_0$, an estimate of the intercept of the weighted linear regression,

$$\hat{\beta} = (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{y} \tag{2}$$

where $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p)^T$ is the vector of estimated regression coefficients (with $\{\hat{\beta}_j = (j!)^{-1}\widehat{m}^{(j)}(x)|_{x=x_0},\ j = 0, \ldots, p\}$ also representing estimated coefficients from a corresponding Taylor expansion); $\mathbf{X} = \{(x_i - x_0)^j\}_{i,j=1,0}^{n,p}$ is a design matrix; and $\mathbf{W} = \text{diag}\{K_h(x_i - x_0)\}_{n\times n}$ is a weight matrix with weights $K_h(\cdot)$ defined as $K_h(x) = h^{-1}K(x/h)$, with $K(\cdot)$ being a kernel function and h defining a bandwidth. The kernels are defined in *Methods and formulas* of [R] **kdensity**.

The default bandwidth is obtained using the ROT method of bandwidth selection. The ROT bandwidth is the plugin estimator of the asymptotically optimal constant bandwidth. This is the bandwidth that minimizes the conditional weighted mean integrated squared error. The ROT plugin bandwidth selector for the smoothing bandwidth h is defined as follows; assuming constant residual variance $\sigma^2(x_0) = \sigma^2$ and odd degree p:

$$\hat{h} = C_{0,p}(K)\left[\frac{\hat{\sigma}^2 \int w_0(x)dx}{n\int\{\widehat{m}^{(p+1)}(x)\}^2 w_0(x)f(x)dx}\right]^{1/(2p+3)} \tag{3}$$

where $C_{0,p}(K)$ is a constant, as defined in Fan and Gijbels (1996), that depends on the kernel function $K(\cdot)$, and the degree of a polynomial p and w_0 is chosen to be an indicator function on the interval $[\min_{\mathbf{x}} + 0.05 \times \text{range}_{\mathbf{x}}, \max_{\mathbf{x}} - 0.05 \times \text{range}_{\mathbf{x}}]$ with $\min_{\mathbf{x}}$, $\max_{\mathbf{x}}$, and $\text{range}_{\mathbf{x}}$ being, respectively, the minimum, maximum, and the range of $\mathbf{x}$. To obtain the estimates of a constant residual variance, $\hat{\sigma}^2$, and $(p+1)$th order derivative of $m(x)$, denoted as $\widehat{m}^{(p+1)}(x)$, a polynomial in $\mathbf{x}$ of order $(p+3)$ is fit globally to $\mathbf{y}$. $\hat{\sigma}^2$ is estimated as a standardized residual sum of squares from this fit.

The expression for the asymptotically optimal constant bandwidth used in constructing the ROT bandwidth estimator is derived for the odd-order polynomial approximations. For even-order polynomial fits the expression would depend not only on $m^{(p+1)}(x)$ but also on $m^{(p+2)}(x)$ and the design density and its derivative, $f(x)$ and $f'(x)$. Therefore, the ROT bandwidth selector would require estimation of these additional quantities. Instead, for an even-degree p of the local polynomial, `lpoly` uses the value of the ROT estimator (3) computed using degree $p+1$. As such, for even degrees this is not a plugin estimator of the asymptotically optimal constant bandwidth.

The estimates of the conditional variance of local polynomial estimators are obtained using

$$\widehat{\text{Var}}\{\widehat{m}(x_0)|X = x_0\} = \hat{\sigma}^2_m(x_0) = (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{W}^2\mathbf{X})(\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\hat{\sigma}^2(x_0) \tag{4}$$

where $\hat{\sigma}^2(x_0)$ is estimated by the normalized weighted residual sum of squares from the $(p+2)$th order polynomial fit using pilot bandwidth $h^\star$.

When the bias is negligible the normal-approximation method yields a $(1-\alpha)\times 100\%$ confidence interval for $m(x_0)$,

$$\{\widehat{m}(x_0) - z_{(1-\alpha/2)}\hat{\sigma}_m(x_0),\ \widehat{m}(x_0) + z_{(1-\alpha/2)}\hat{\sigma}_m(x_0)\}$$

where $z_{(1-\alpha/2)}$ is the $(1-\alpha/2)$th quantile of the standard Gaussian distribution, and $\widehat{m}(x_0)$ and $\hat{\sigma}_m(x_0)$ are as defined in (2) and (4), respectively.

References

Chetverikov, D., D. Kim, and D. Wilhelm. 2018. Nonparametric instrumental-variable estimation. *Stata Journal* 18: 937–950.

Cleveland, W. S. 1979. Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association* 74: 829–836. https://doi.org/10.2307/2286407.

Cox, N. J. 2005. Speaking Stata: Smoothing in various directions. *Stata Journal* 5: 574–593.

Donoho, D. L. 1995. Nonlinear solution of linear inverse problems by wavelet-vaguelette decomposition. *Applied and Computational Harmonic Analysis* 2: 101–126. https://doi.org/10.1006/acha.1995.1008.

Eubank, R. L. 1999. *Nonparametric Regression and Spline Smoothing*. 2nd ed. New York: Dekker. https://doi.org/10.1201/9781482273144.

Fan, J. 1992. Design-adaptive nonparametric regression. *Journal of the American Statistical Association* 87: 998–1004. https://doi.org/10.2307/2290637.

Fan, J., and I. Gijbels. 1996. *Local Polynomial Modelling and Its Applications*. London: Chapman and Hall. https://doi.org/10.1201/9780203748725.

Gasser, T., and H.-G. Müller. 1979. "Kernel estimation of regression functions". In *Smoothing Techniques for Curve Estimation*, edited by T. Gasser and M. Rosenblatt, vol. 757: 23–68. New York: Springer. https://doi.org/10.1007/BFb0098489.

Gutierrez, R. G., J. M. Linhart, and J. S. Pitblado. 2003. From the help desk: Local polynomial regression and Stata plugins. *Stata Journal* 3: 412–419.

Nadaraya, E. A. 1964. On estimating regression. *Theory of Probability and Its Applications* 9: 141–142. https://doi.org/10.1137/1109020.

Sheather, S. J., and M. C. Jones. 1991. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society*, B ser., 53: 683–690. https://doi.org/10.1111/j.2517-6161.1991.tb01857.x.

Verardi, V., and N. Debarsy. 2012. Robinson's square root of N consistent semiparametric regression estimator in Stata. *Stata Journal* 12: 726–735.

Watson, G. S. 1964. Smooth regression analysis. *Sankhyā*, A ser., 26: 359–372.

Also see

[R] **kdensity** — Univariate kernel density estimation

[R] **lowess** — Lowess smoothing

[R] **makespline** — Spline generation

[R] **npregress intro** — Introduction to nonparametric regression

[R] **smooth** — Robust nonlinear smoother

[G-2] **graph twoway lpoly** — Local polynomial smooth plots

[G-2] **graph twoway lpolyci** — Local polynomial smooth plots with CIs