The Stata Journal

The *Stata Journal*, electronic version (ISSN 1536-8734) is a publication of Stata Press, and Stata is a registered trademark of StataCorp LP.

The Stata Journal (2005)
5, Number 3, pp. 405–412

A multivariable scatterplot smoother

Patrick Royston
Cancer Division
MRC Clinical Trials Unit
222 Euston Road
London NW1 2DA
UK

Nicholas J. Cox
Durham University, UK
n.j.cox@durham.ac.uk

Abstract. We present an extension of Sasieni, Royston, and Cox's bivariate smoother `running` to the multivariable context. The software aims to provide a picture of the relation between a response variable and each of several continuous predictors simultaneously. This may be a valuable tool in exploratory data analysis, before constructing a more formal multiple regression model.

Keywords: gr0017, mrunning, running, scatterplot smoothing, multivariable regression analysis, running line

1 Introduction

The Stata program `running` performs scatterplot smoothing by running lines or running means (Sasieni 1995; Sasieni and Royston 1998; Sasieni, Royston, and Cox 2005). See Hastie and Tibshirani (1990, 15–16, 29–31) for a discussion of the theory behind bivariate nearest-neighbor smoothers, which are the building blocks used in the present work. In this article, we will extend `running` to the context of multivariable smoothing. Obtaining a picture of the relation between a response variable and each of several continuous predictors simultaneously may be a valuable tool in exploratory data analysis, particularly when the aim is to arrive at a parametric final model. The scatterplot smooth is "nonparametric" and as implemented here does not require the user to choose tuning parameters in order to get a reasonable representation of what may be a complex multivariate relationship.

The present implementation of multivariable scatterplot smoothing is in an ado-file called `mrunning`. Estimation of the smooth for each predictor is done by backfitting. Since each smooth is locally linear, the backfitting algorithm is guaranteed to converge (Breiman and Friedman 1985).

2 Syntax

`mrunning` is a regression-like command with the following syntax:

`mrunning` *yvar xvarlist* [*if*] [*in*] [*weight*] [`, adjust(`*varlist*`) ci`
`combine(`*combine_options*`) cycles(#) draw(`*numlist*`) generate(`*stub*`)`
`gense(`*sestub*`) nograph knn(`*varlist*`:#[ , `*varlist*`:# ...]) log mean`
`omit(`*numlist*`) predict(`*newvar*`) nopts repeat(`*varlist*`:#[ , `*varlist*`:# ...])`
`replace scatter(`*scatter_options*`) span(`*varlist*`:#[ , `*varlist*`:# ...])`
line_options]

Only `aweights` are allowed.

2.1 Options

`adjust(`*varlist*`)` adjusts linearly for *varlist*. In practice, this option should be used for binary predictors and continuous predictors for which a linear relationship is required.

`ci` produces a pointwise confidence interval for the smoothed values of *yvar*. The width is determined by the current value of the macro `$S_level`. `ci` is not available with `repeat()`.

`combine(`*combine_options*`)` specifies any of the options allowed by the `graph combine` command; see [G] **graph combine**. Useful examples are `combine(ycommon)` and `combine(saving(`*graphname*`))`.

`cycles(#)` sets the number of cycles. The default is `cycles(3)`.

`draw(`*numlist*`)` specifies that smooths for a subset of the variables in *xvarlist* be plotted. The elements of *numlist* are indexes determined by the order of the variables in *xvarlist*. For example, `mrunning y x1 x2 x3, draw(2 3)` would plot smooths only for variables `x2` and `x3`. By default, all variables in *xvarlist* are plotted. `draw()` takes precedence over `omit()` in the sense that variables included (by index) in *numlist* are plotted, even if they are excluded by `omit()`. See also `omit()`.

`generate(`*stub*`)` specifies that fitted values for each member of *xvarlist* be saved in new variables with names beginning with *stub*.

`gense(`*sestub*`)` specifies that standard errors of smooths for each member of *xvarlist* be saved to new variables whose names begin with *sestub*.

`nograph` suppresses the graph.

`knn(`*varlist*`:#[ , `*varlist*`:# ...])` controls the number k of nearest neighbors used on each side of the smoothed point. Different numbers of nearest neighbors may be specified for each *varlist*. The greater the value of k, the greater is the degree of smoothing.

`log` displays the squared correlation coefficient between *yvar* and the overall fitted values at each cycle for monitoring convergence. This option is provided mainly for pedagogic interest.

`mean` specifies running-mean least-squares smoothing; the default is running-line.

`omit(`*numlist*`)` specifies that smooths for a subset of the variables in *xvarlist* not be plotted. The elements of *numlist* are indexes determined by the order of the variables in *varlist*. For example, `mrunning y x1 x2 x3, omit(3)` would plot smooths only for variables `x1` and `x2`. By default, no variables in *xvarlist* are omitted. `draw()` takes precedence over `omit()`. See also `draw()`.

`predict(`*newvar*`)` specifies that the predicted values be saved in new variable *newvar*.

`nopts` suppresses the points in the plots. Only the lines representing the smooths (and where applicable, their confidence intervals) are drawn.

`repeat(`*varlist*`:`#[`,` *varlist*`:`# ...]`)` sets the number of smoothing passes for each member of each *varlist*. The default is 1. Standard errors of the smooth (see `ci` and `gense()` options) are not available for variables with `repeat()`> 1.

`replace` allows variables specified by any of the `generate()`, `gense()`, and `predict()` options to be replaced if they already exist.

`scatter(`*scatter_options*`)` specifies any of the options allowed by the `scatter` command; see [G] **graph twoway scatter**. These should be specified to control the rendering of the data points. The default includes `msymbol(oh)`, or `msymbol(p)` with over 299 observations.

`span(`*varlist*`:`#[`,` *varlist*`:`# ...]`)` sets the span for each member of each *varlist*. The span or proportion of the data is used to determine the symmetric nearest neighbors. If `span()` is specified and n is the number of observations, `knn()` is defined to be ($n \times$ `span()` $- 1)/2$. You cannot specify both `span()` and `knn()`.

line_options are any of the options allowed by the `line` command; see [G] **graph twoway line**. These should be specified to control the rendering of the smoothed lines or the overall graph.

3 Example

We will use as an example the diabetes data analyzed in some detail by Hastie and Tibshirani (1990). Figure 1 shows a scatterplot matrix of `cpep` (the log C-peptide concentration in log pmol/ml), `age` (the age of the patient), and `base` (minus the base deficit). There are 43 observations.

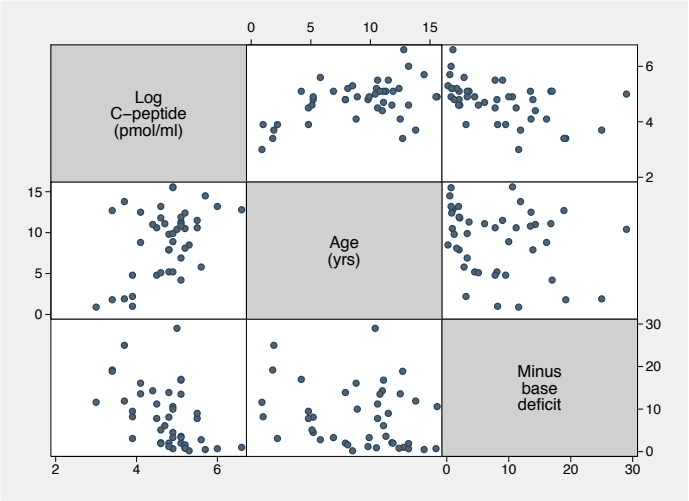

Figure 1: Scatterplot matrix for the diabetes data.

A bivariate relationship between `cpep` and each of the two predictors is apparent. To get an impression of the multivariable relationship, we use `mrunning`, requesting pointwise confidence intervals and the overall predictions of `cpep` to be stored in a new variable called `xb`:

```
. mrunning cpep age base, ci predict(xb)
43 observations, R-sq = 0.5160
```

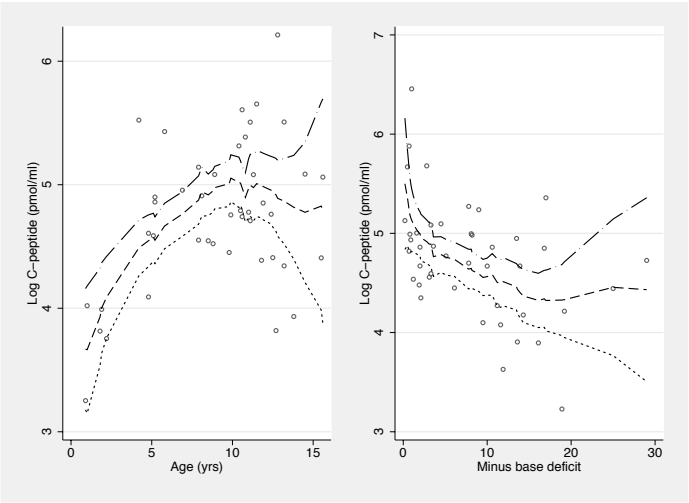

Figure 2: Multivariable running-line smooth for the diabetes data, with pointwise 95% confidence intervals.

We now see that there appears to be an increasing, nonlinear relationship between `cpep` and `age` (adjusting for `base`) and a decreasing, nonlinear relationship between `cpep` and `base` (adjusting for `age`). The multiple squared correlation coefficient between `cpep` and the overall predicted values, equal to the sum of the two smooths, is 0.516. The relationship between `cpep` and the overall predicted values `xb` is shown in figure 3. There is plenty of uncertainty but no obvious lack of fit.

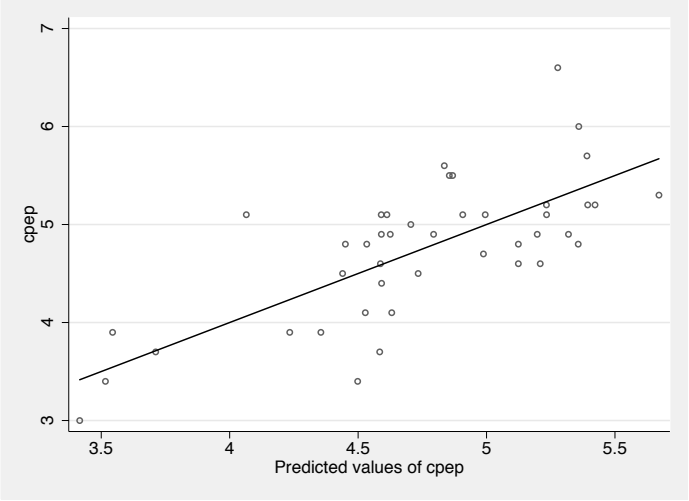

Figure 3: Relation between `cpep` and its predictor from smooths on `age` and `base`.

Convergence of the backfitting algorithm is rapid in this example. The values of R^2 for the first 8 cycles are 0.508067, 0.515340, 0.516031, 0.516138, 0.516159, 0.516162, 0.516163, and 0.516163. There is no further change in the sixth decimal place of R^2 with larger numbers of cycles.

Figure 4 shows the impact of increasing the smoothing by using the `repeat(2)` option. We cannot now obtain pointwise confidence intervals since these are not supported by `running` with `repeat(2)`.

(*Continued on next page*)

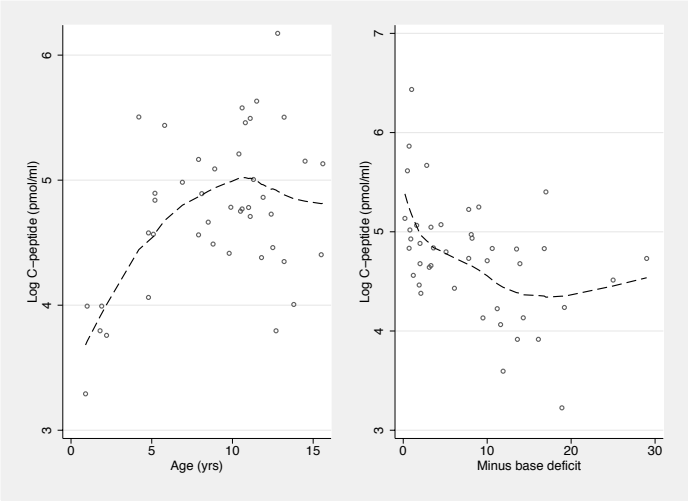

Figure 4: Multivariable running-line smooths with the `repeat(2)` option.

The lines are noticeably smoother, although the "message" is much the same. The R^2 has increased modestly from 0.516 to 0.544.

4 Technical notes

Suppose that there are $p \geq 1$ predictors $x_1, \ldots, x_p$. `mrunning` estimates the smooths $f_1(x_1), \ldots, f_p(x_p)$ by using a backfitting algorithm and a running-line smoother $S(y|x)$ for each predictor, as follows. Suppose that there are n observations

$(y_1, x_{11}, \ldots, x_{1p}), \ldots, (y_n, x_{n1}, \ldots, x_{np})$:

1. Initialize: $\alpha = \overline{y} = n^{-1} \sum_{i=1}^{n} y_i$; estimate $f_1(x_1), \ldots, f_p(x_p)$ by multiple linear regression; and center each smooth to have mean zero.

2. Cycle: $j = 1, \ldots, p, 1, \ldots, p, \ldots$ such that, for a typical observation,

$$f_j(x_j) = S\left\{ y - \alpha - \sum_{l=1,\ldots,p; l \neq j} f_l(x_l) \middle| x_j \right\}$$

3. Continue for `cycles()` rounds.

The running-line smoother $S(y|x)$ is provided by `running` (Sasieni, Royston, and Cox 2005). Details of the algorithm are given by Sasieni (1995). No convergence criterion is applied to the backfitting procedure. In practice, three cycles are usually more than sufficient to get results adequate for exploratory work. When the predictors are

highly correlated, it may be useful to increase the number of cycles; in case of doubt, the `log` option may be used together with `nograph` to monitor convergence of the explained variation statistic, R^2.

If adjustment is requested (the `adjust()` option), it is applied at the end of each backfitting round by multiple linear regression of the partial residuals

$$y_i - \alpha - \sum_{l=1}^{p} f_l(x_{il})$$

on the adjustment variables.

The smooths $f_j(x_j)$ are finally adjusted to have mean α, i.e.,

$$n^{-1} \sum_{i=1}^{n} f_j(x_{ij}) = \alpha$$

The overall predictor, as given by the `predict()` option, is defined as

$$\widehat{y}_i = \alpha + \sum_{l=1}^{p} \{ f_l(x_{il}) - \alpha \}$$

The points in the plots provided by `mrunning` for a given x_j are

$$y_i - \sum_{l \neq j} \{ f_l(x_{il}) - \alpha \}$$

that is, the partial residuals for the jth predictor plus α. These are plotted together with $f_j(x_{ij})$ against x_{ij}.

Note that `mrunning` estimates standard errors for each smooth of partial residuals on a given x_j. These SEs do not allow for correlation with other variables and are therefore underestimated; nevertheless, they are useful for exploratory work. If more accurate SEs are needed, we recommend the use of the bootstrap in conjunction with `mrunning` to compute them.

5 References

Breiman, L. and J. H. Friedman. 1985. Estimating optimal transformations for multiple regression and correlation. *Journal of the American Statistical Association* 80: 580–598.

Hastie, T. J. and R. J. Tibshirani. 1990. *Generalized Additive Models*. London: Chapman and Hall.

Sasieni, P. 1995. sed9: Symmetric nearest neighbor linear smoothers. *Stata Technical Bulletin* 24: 10–14. In *Stata Technical Bulletin Reprints*, vol. 4, 97–101. College Station, TX: Stata Press.

Sasieni, P. and P. Royston. 1998. sed9.1: Pointwise confidence intervals for running. *Stata Technical Bulletin* 41: 17–23. In *Stata Technical Bulletin Reprints*, vol. 7, 156–163. College Station, TX: Stata Press.

Sasieni, P., P. Royston, and N. J. Cox. 2005. sed9_2: Symmetric nearest neighbor linear smoothers. *Stata Journal* 5(2): 285.

About the Authors

Patrick Royston is a medical statistician of nearly 30 years of experience, with a strong interest in biostatistical methodology and in statistical computing and algorithms. At present, he works in clinical trials and related research issues in cancer. Currently, he is focusing on problems of model building and validation with survival data, including prognostic factors studies, on parametric modeling of survival data, and on novel trial designs.

Nicholas Cox is a statistically minded geographer at Durham University. He contributes talks, postings, FAQs, and programs to the Stata user community. He has also co-authored fifteen commands in official Stata. He was an author of several inserts in the *Stata Technical Bulletin* and is an Editor of the *Stata Journal*.