The Stata Journal (2020)
20, Number 3, pp. 647–679 DOI: 10.1177/1536867X20953574

Smooth varying-coefficient models in Stata

Fernando Rios-Avila
Levy Economics Institute of Bard College
Annandale-on-Hudson, NY
friosavi@levy.org

Abstract. Nonparametric regressions are powerful statistical tools that can be used to model relationships between dependent and independent variables with minimal assumptions on the underlying functional forms. Despite their potential benefits, these models have two weaknesses: The added flexibility creates a curse of dimensionality, and procedures available for model selection, like cross-validation, have a high computational cost in samples with even moderate sizes. An alternative to fully nonparametric models is semiparametric models that combine the flexibility of nonparametric regressions with the structure of standard models. In this article, I describe the estimation of a particular type of semiparametric model known as the smooth varying-coefficient model (Hastie and Tibshirani, 1993, *Journal of the Royal Statistical Society, Series B* 55: 757–796), based on kernel regression methods, using a new set of commands within `vc_pack`. These commands aim to facilitate bandwidth selection and model estimation as well as create visualizations of the results.

Keywords: st0613, vc_pack, vc_bw, vc_bwalt, vc_reg, vc_bsreg, vc_preg, vc_predict, vc_test, vc_graph, smooth varying-coefficient models, kernel regression, cross-validation, semiparametric estimations

1 Introduction

Nonparametric regressions are powerful statistical tools that can be used to model relationships between dependent and independent variables with minimal assumptions on the underlying functional forms. This flexibility makes nonparametric regressions robust to functional form misspecification, which is one of the main advantages over standard regression analysis.

The added flexibility of nonparametric regressions comes at a cost. On the one hand, the added flexibility creates what is known as the curse of dimensionality. In essence, because nonparametric regressions imply the estimation of many parameters, accounting for interactions and nonlinearities, more data are required to obtain results with a similar level of precision to their parametric counterparts. On the other hand, while larger datasets can be used to reduce the curse of dimensionality, procedures used for model selection and estimations are often too computationally intensive, making the estimation of this type of model less practical in samples of moderate to large sizes. Perhaps because of these limitations, and until recent versions, Stata had a limited set of native commands for the estimation of nonparametric models. Even with the recent development of computing power, the estimation of full nonparametric models, using the currently available commands, remains a challenge when using large samples.[1]

One response to the main weakness of nonparametric methods has been the development of semiparametric methods. These methods combine the flexibility of nonparametric regressions with the structure of standard parametric models, reducing the curse of dimensionality and the computational cost of model selection and estimation.[2] In fact, many community-contributed commands have been proposed for the analysis of a large class of semiparametric models in Stata.[3]

A particular type of semiparametric model, the estimation of which has not been explored within Stata, is known as the smooth varying-coefficient model (SVCM) (Hastie and Tibshirani 1993). These models assume that the outcome Y is a function of two sets of characteristics, $\mathbf{X}$ and $\mathbf{Z}$, where the effect of $\mathbf{X}$ on Y follows some unspecified smooth function of $\mathbf{Z}$. As described by Henderson and Parmeter (2015), this method is particularly popular in applied settings because they are easy to estimate and interpret because the Y is a linear function of $\mathbf{X}$ conditional on $\mathbf{Z}$.

For example, as described in Hainmueller, Mummolo, and Xu (2019), SVCMs can be thought of as multiplicative interactive models where the variable $\mathbf{Z}$ behaves as a moderator for the treatment variables of interest, relaxing the linear assumption of the interaction. Alternatively, as described in Rios-Avila (2019), SVCMs can be used to extend standard Oaxaca–Blinder decomposition analysis to scenarios with continuous group variables, decomposing, for example, wage differences of individuals with different body mass indices. Furthermore, under assumptions of an exogenous treatment and unconfoundedness, SVCMs can be used to estimate heterogeneous dose treatment effects (see, for example, Hirano and Imbens [2004] for a discussion on continuous treatment effects).

In this article, I introduce a new set of commands that aims to facilitate the model selection, estimation, and visualization of SVCMs with a single smoothing variable $\mathbf{Z}$. `vc_bw` and `vc_bwalt` are commands used for model selection that implement a leave-one-out cross-validation (CV) procedure to select the optimal bandwidth. `vc_reg`, `vc_bsreg`, and `vc_preg` are commands used for the estimation of SVCMs across a selected set of points of interest, providing different alternatives for the estimation of standard errors. `vc_predict` and `vc_test` are commands that can be used to obtain model predictions and residuals, provide some statistics of the fit model, and provide some specification tests. `vc_graph` can be used to plot the smooth coefficients.

[1] Stata 15 introduced the command `npregress kernel`, which fits fully nonparametric models using kernel methods. More recently, Stata 16 introduced `npregress series`, which fits fully nonparametric models using series and spline methods.

[2] A brief review of the semiparametric method is provided in Cameron and Trivedi (2005, sec. 9.7). For a more in-depth revision of theory on semiparametric models, see Li and Racine (2007, chap. 7–11), whereas Henderson and Parmeter (2015) offer a more empirical discussion on these types of models.

[3] See Verardi (2013) for a brief review of commands for the estimation of semiparametric regressions in Stata.

The rest of the article is structured as follows. Section 2 reviews the estimation of SVCMs. Section 3 provides a detailed review of the implementation procedures and commands used for model selection, estimation, and postestimation. Section 4 illustrates the commands, and section 5 concludes.

2 Nonparametric regression and SVCM

2.1 Nonparametric regressions

Consider a model where Y is the dependent variable and $\mathbf{W}$ is a set of exogenous independent variables of dimension k. Without any assumption on the relationships between these variables, and assuming no omitted variable problem exists, the nonparametric regression model of Y given the k dimensional vector of variables $\mathbf{W}$ is given by

$$\begin{aligned} y_i &= g(\mathbf{W}_i) + e_i \\ E(e_i|\mathbf{W}_i) &= 0 \end{aligned} \tag{1}$$

Essentially, this model specification implies that $\mathbf{W}$ is related to Y following some unknown nonlinear functional form. The literature on nonparametric regressions suggests that these types of models can be fit in at least two ways. On the one hand, the function $g(\cdot)$ can be estimated by modeling the conditional mean function as a locally weighted average estimator

$$\widehat{g}(\mathbf{w}) = E(y_i|\mathbf{W}_i = \mathbf{w}) = \frac{\Sigma y_i K(\mathbf{W}_i, \mathbf{w}, \mathbf{h})}{\Sigma K(\mathbf{W}_i, \mathbf{w}, \mathbf{h})}$$

where $\mathbf{h} = \{h_{w1}, \dots, h_{wk}\}$ is a vector of bandwidths, $K(\cdot)$ is a joint kernel function

$$K(\mathbf{W}_i, \mathbf{w}, \mathbf{h}) = \prod_{j=1}^{k} K_j(W_{ij}, w_j, h_{wj})$$

and $K_j(W_{ij}, w_j, h_{wj})$ is a kernel function defined by the point of reference w_j and the bandwidth h_{wj}:[4]

$$K_j(W_{ij}, w_j, h_{wj}) = k_j\left(\frac{W_{ij} - w_j}{h_{wj}}\right) \tag{2}$$

This function $K(\cdot)$ gives more weight to observations that are close to the point $\mathbf{w}$ and uses the vector of bandwidths $\mathbf{h}$ to determine how much information is used for the estimation of the conditional mean. This procedure can be implemented in Stata using the command `npregress kernel`.

An alternative procedure is to estimate $g(\mathbf{w})$ using a set of predefined transformations and interactions of the original variables as explanatory variables. The most

[4] See appendix A for a list of kernel function definitions.

common practice is to use polynomial or splines basis of the original variables $S(\cdot)$ and their interactions $I(\cdot)$ and fit the following model:

$$y_i = S(\mathbf{W}_i)'\boldsymbol{\beta}_w + I\{S(\mathbf{W}_i)\}'\boldsymbol{\beta}_{I_W} + e_i \tag{3}$$

$\mathbf{B} = (\boldsymbol{\beta}_w, \boldsymbol{\beta}_I)$ are all the coefficients associated with each one of the terms of the transformations $S(\cdot)$ and interactions $I(\cdot)$. In this setting, the dimension of $\mathbf{B}$, or more specifically of $S(\cdot)$ and $I(\cdot)$, represent the tuning parameter that determines the roughness of $\widehat{g}()$. This procedure can be implemented in Stata using the command `npregress series`, using polynomials, splines, and B-splines basis.

As described in Li and Racine (2007) and Stinchcombe and Drukker (2013), in the case of kernel methods, the effective number of observations for the estimation of the conditional mean decreases rapidly as k increases and $\mathbf{h}$ goes to 0. In the case of transformations and interactions, the number of parameters that need estimation increases exponentially with the number of explanatory variables and the dimension of $S(\cdot)$ and $I(\cdot)$, rapidly reducing the degrees of freedom of the model.[5]

2.2 SVCM

SVCMs, as introduced by Hastie and Tibshirani (1993), assume that there is some structure in the model. Instead of estimating a function like (1), the authors suggest distinguishing two types of independent variables, $\mathbf{W} = (\mathbf{X}, \mathbf{Z})$. $\mathbf{X}$ are variables that have a linear effect of Y, but those effects are some unspecified nonlinear functions of $\mathbf{Z}$. This model is defined by

$$\begin{aligned} y_i &= \mathbf{X}_i'\beta(\mathbf{Z}_i) + e_i \\ E(e_i|\mathbf{X}_i, \mathbf{Z}_i) &= 0 \end{aligned} \tag{4}$$

This specification reduces the problem of the curse of dimensionality of the fit model compared with (1) by assuming $\mathbf{X}$ have a parametric effect on Y, conditional on $\mathbf{Z}$, allowing the coefficients $\beta(\mathbf{Z}_i)$ to be unknown smooth nonlinear functions of $\mathbf{Z}$. For simplicity, I will refer to $\mathbf{Z}$ as the set of smoothing variables. The existence of two types of variables raises the question of deciding which variables should be included in $\mathbf{X}$ or $\mathbf{Z}$. The empirical literature suggests that deciding which variables should be considered as part of the smoothing variables $\mathbf{Z}$ will depend on the research question of interest.

For example, Li et al. (2002) analyze the production function of the nonmetal mineral manufacturing industry in China by analyzing the marginal productivity of capital and labor ($\mathbf{X}$) and analyzing the heterogeneity based on expenditure on intermediate production and expenditure on management ($\mathbf{Z}$). Liu and Egan (2019) analyze recreation demand, focusing on the effect of travel cost and household income on households' willingness to pay ($\mathbf{X}$), allowing for heterogeneity across demographic characteristics, possession of a hunting/fishing license, and environmental organization membership

[5] In both cases, different strategies can be used to select the roughness or smoothness of the fit models. For a brief review of both strategies, see [R] **npregress intro**.

($\mathbf{Z}$). Centorrino and Racine (2017) revisit the role of experience, race, and geographical location ($\mathbf{X}$) as determinants of wages, analyzing the heterogeneity across education attainment ($\mathbf{Z}$). Polemis and Stengos (2015) analyze labor productivity as a function of labor share ratio, market size, capital, intermediate inputs, and energy cost ($\mathbf{X}$), analyzing the heterogeneity across a measure of market concentration ($\mathbf{Z}$).

Just like with nonparametric regressions, various methods have been proposed for the estimation of this type of model. Hastie and Tibshirani (1993) suggest estimating $\beta(\mathbf{Z}_i)$ using spline basis or penalized splines with respect to $\mathbf{Z}$. Hoover et al. (1998) and Li et al. (2002) suggest instead using local polynomial kernel regressions as a feasible strategy to estimate $\beta(\mathbf{Z}_i)$. More recently, Li and Racine (2010) expanded on the use of kernel methods for the estimation and inference of these types of models when $\mathbf{Z}$ is a mixture of continuous and discrete data.[6] In the next section, I describe the estimation of SVCMs using kernel methods when there is a single smoothing variable in $\mathbf{Z}$.

2.3 SVCM: Local kernel estimator

Consider a simplified version of the SVCM [(4)], as described in Li et al. (2002), where Y is the dependent variable, Z is a single continuous variable, and $\mathbf{X}$ is a set of variables including a constant. Because Z contains a single variable, the bandwidth h will be a single scalar, dropping the subscript j from (2).

Following Li and Racine (2007), the coefficients in (4) can be derived as follows. Starting from (4), premultiply both sides by $\mathbf{X}_i$, take expectations conditional on $Z_i = z$, and solve for $\beta(z)$, which yields

$$\begin{aligned} E(\mathbf{X}_i y_i | Z_i = z) &= E(\mathbf{X}_i \mathbf{X}_i' | Z_i = z)\beta(z) + E(\mathbf{X}_i e_i | Z_i = z) \\ \beta(z) &= E(\mathbf{X}_i \mathbf{X}_i' | Z_i = z)^{-1} E(\mathbf{X}_i y_i | Z_i = z) \end{aligned} \tag{5}$$

Using sample data, (5) can be an unfeasible estimator of $\beta(z)$ because there may be few or no observations for which $Z_i = z$, making $\beta(z)$ impossible to estimate.[7] As an alternative, a feasible estimation for (3) can be obtained using kernel methods for any point z,

$$\widehat{\beta}(z,h) = \left\{\sum \mathbf{X}_i \mathbf{X}_i' k\left(\frac{Z_i - z}{h}\right)\right\}^{-1} \left\{\sum \mathbf{X}_i y_i k\left(\frac{Z_i - z}{h}\right)\right\}$$

or the equivalent in matrix form,

$$\widehat{\beta}(z,h) = \{\mathbf{X}'\mathcal{K}_h(z)\mathbf{X}\}^{-1}\{\mathbf{X}'\mathcal{K}_h(z)\mathbf{Y}\} \tag{6}$$

[6] Most methodologies implementing SVCMs are based on the assumption that $\mathbf{X}$ and $\mathbf{Z}$ are exogenous. Discussion on the estimation of SVCMs when $\mathbf{X}$ is endogenous can be found in Cai et al. (2006), whereas the estimation of models when $\mathbf{Z}$ is endogenous has been discussed and proposed in Centorrino and Racine (2017), Delgado et al. (2020), and Rios-Avila (2019). This, however, is beyond the scope of this article.

[7] The estimator in (4) exists only if $E(\mathbf{X}_i \mathbf{X}_i' | Z_i = z)$ is full rank, but this may not be the case when using sample data.

where $k(\cdot)$ is the kernel function, as defined in (2), that gives more weight to observations where Z_i is closer to z, given the bandwidth h. $\mathcal{K}_h(z)$ is an $N \times N$ diagonal matrix with the ith element equal to $k\{(Z_i - z)/h_z\}$. Equation (6) constitutes the local constant (LC) estimator of SVCM.

A drawback of the LC estimator is that it is well known for its potentially large bias when estimating functions near boundaries of the support of Z. A simple solution to reduce this bias is to use a local linear (LL) estimator, based on a first-order approximation of the coefficients $\beta(Z_i)$. This implies that instead of estimating (4), one can instead fit the following model:

$$\begin{aligned}
\beta(Z_i) &\cong \beta(z) + (Z_i - z)\frac{\partial\beta(z)}{\partial z} \\
y_i &\cong \mathbf{X}_i' \left\{\beta(z) + (Z_i - z)\frac{\partial\beta(z)}{\partial z}\right\} + e_i \\
y_i &\cong \mathbf{X}_i'\beta(z) + \mathbf{X}_i'(Z_i - z)\frac{\partial\beta(z)}{\partial z} + e_i \qquad (7)
\end{aligned}$$

This implies that an approximation for y_i can be obtained using a linear expansion with respect to $\beta(z)$ and that the closer Z_i is to z, the more accurate the approximation will be.

Define $\mathcal{X}_i = \{\mathbf{X}_i \ (Z_i - z) \otimes \mathbf{X}_i\}$ to be the ith row of $\mathcal{X}$ and $\otimes$ to be the Kronecker product, such that $(Z_i - z) \otimes \mathbf{X}_i$ indicates that each variable in X_i is multiplied by the auxiliary variable $(Z_i - z)$. Based on (6), the coefficients $\beta(z)$ and $\partial\beta(z)/\partial z$ can be estimated as

$$\begin{bmatrix} \widehat{\beta}(z,h) \\ \frac{\partial\widehat{\beta}(z,h)}{\partial z} \end{bmatrix} = \{\mathcal{X}'\mathcal{K}_h(z)\mathcal{X}\}^{-1}\{\mathcal{X}'\mathcal{K}_h(z)\mathbf{Y}\} \qquad (8)$$

where $\widehat{\beta}(z,h)$ constitutes the LL estimator of $\beta(z)$ and $\partial\widehat{\beta}(z,h)/\partial z$ is the first derivative of that coefficient with respect to any point z, $\partial\beta(z)/\partial z$.

2.4 Example: SVCM and weighted least squares

While it may not seem evident, (4) and (7) show that the estimation of SVCMs using kernel methods can be easily obtained using weighted ordinary least squares (OLS), where weights are defined by the kernel functions. To show this, let's consider the dataset "Fictional data on monthly drunk driving citations" (`dui.dta`) and a simple model that assumes that citations are a linear function of college, taxes, city size, and fines. This model can be fit using the following command:

```
regress citations i.college i.taxes i.csize fines
```

Say that you are interested in analyzing how the effect of `college`, `taxes`, and `csize` varies as a function of `fines`.[8] Assume for simplicity that you are interested in

[8] One option could be to assume that the effects vary linearly with respect to fines. In such a case, the following model could be fit using `regress citations c.fines##i.(college taxes csize)`.

one point of the distribution: fines at the 10th percentile (= 9). In this example, there are enough observations with values exactly equal to 9, so it is possible to fit the model using this constraint. Because we are estimating regressions for specific values of fines, this variable is taken out of the specification:

```
regress citations i.college i.taxes i.csize if fines==9
```

In general, it may be more convenient to fit the model using kernel functions as weights. As discussed in the literature, the choice of kernel function is not as important as the choice of bandwidth. For simplicity, I will use a Gaussian kernel with a bandwidth $h = 0.5$. This is directly implemented using the `normalden()` function, with the smoothing variable `fines` as the first argument, the point of interest (9) as the second argument, and the bandwidth ($h = 0.5$) as the third argument:

```
regress citations i.college i.taxes i.csize [aw=normalden(fines,9,0.5)]
```

This example implements the LC estimators following (5). For the implementation of the LL estimator, an auxiliary variable needs to be constructed $(Z_i - z)$ `df=fines-9`. This variable is created and added to the model specification, creating interactions with all other explanatory variables. Using factor notation, we see this is straightforward:

```
regress citations i.(college taxes csize)##c.df [aw=normalden(fines,9,0.5)]
```

To show how these models compare with each other, figures 1a and 1b provide a simple plot of the coefficients associated with `college` and `taxes`, using the three specifications sketched above and using every distinct value of `fines`, comparing them with the standard regression estimations.

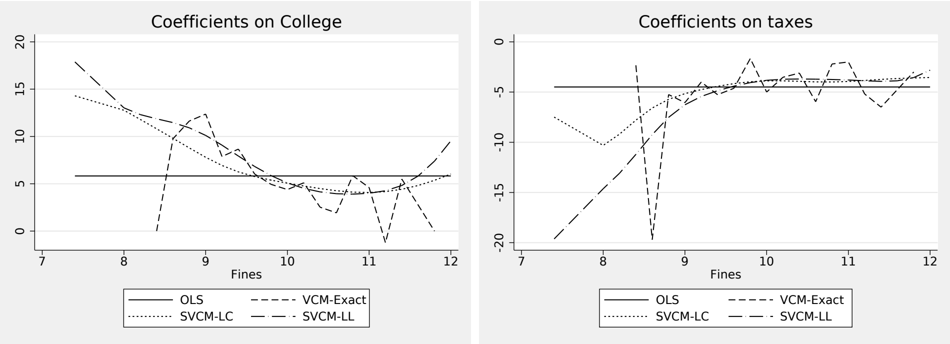

Figure 1. Varying-coefficient models (VCM) across fines: college and taxes

"VCM-Exact" corresponds to the models that constrain data to $Z_i = z$, whereas "SVCM-LC" and "SVCM-LL" indicate the estimates come from the LC and LL estimators of the SVCM model, respectively. Note that there are no estimations for the "VCM-Exact" model at the boundaries of the distribution of `fines` because there are simply not enough observations to obtain those estimates. Also, note that the "VCM-Exact" produces volatile coefficients. Both "SVCM-LC" and "SVCM-LL" produce smooth plots.

The LC estimators are somewhat flat at the boundaries of the distribution, which is to be expected. In contrast, the LL estimator seems to be less affected by the boundary bias, following more closely "VCM-Exact" coefficients. At this point, however, nothing can be said in terms of statistical inference regarding the merits of either model.

While this simple illustration shows the simplicity of fitting SVCMs, there are many details regarding model choice and statistical inference that require further examination. In the next section, I discuss some of the details regarding these problems, introducing the commands in `vc_pack` that can be used to fit SVCMs with a single smoothing variable.

3 SVCM: vc_pack

3.1 Model selection: vc_bw and vc_bwalt

Leave-one-out CV

The most important aspect of the estimation of SVCMs is the choice of the bandwidth parameter h. While larger bandwidths may help to reduce the variance of the estimations by allowing more information to be used on the local estimation process, it will increase the bias of the estimators by restricting the model flexibility. In contrast, smaller bandwidths can reduce bias, allowing for greater flexibility in the estimation but at the cost of higher variability.[9]

The illustration presented in the previous section is an example of this phenomenon. The standard OLS coefficients can be considered as an extreme scenario, where the bandwidth h is so large that all observations receive equal weight regardless of the point of interest. This is guaranteed to obtain the minimum variance for the estimated parameters but with a potentially large cost in terms of model bias. On the opposite side of the spectrum, the results where the regressions are estimated using samples restricted to observations with a specific value of `fines` (VCM-Exact) are based on a bandwidth h that is essentially 0. While this is the most flexible model possible given the data, figure 1 also shows that the results are highly volatile, and estimations were not feasible for some areas.

While there are many suggestions in the literature regarding bandwidth selection (see, for example, Zhang and Lee [2000]), the methodology used here is based on a leave-one-out CV procedure. Consider the model described in (4) and a sample of size N. The optimal bandwidth h^* is such that it minimizes the CV criteria defined as

$$\begin{aligned} h^* &= \min_h \mathrm{CV}(h) = \min_h \sum_{i=1}^{N} \omega(z_i) \left\{ y_i - \mathbf{X}_i' \widehat{\beta}_{-i}(z_i, h) \right\}^2 \\ &= \min_h \sum_{i=1}^{N} \omega(z_i) \left\{ y_i - \widehat{y}_{-i}(h) \right\}^2 \end{aligned}$$

[9] In the context of series, polynomials, and splines, the tradeoff between variance and bias is determined by the dimension of the series transformations $S(\cdot)$ and the interactions $I(\cdot)$.

where $\widehat{\beta}_{-i}(z_i,h)$ is the leave-one-out estimator of $\widehat{\beta}(z_i,h)$, conditional on a bandwidth h, that excludes the ith observation and $\widehat{y}_{-i}(h)$ is the leave-one-out prediction of the SVCM. $\omega(z_i)$ is a weighting function that is used to reduce the influence of areas where the distribution of Z is scant. While this seems a computationally intensive process that requires the estimation of n different sets of parameters, the actual estimation of the CV criteria requires the estimation of fewer equations based on characteristics of the data and properties of linear regressions.

On the one hand, even if Z is a continuous variable in nature, it is often recorded as partially discrete data. A person's age, for example, is a variable that is continuous in nature but is often measured in years. This implies that the number of distinct coefficients $\widehat{\beta}(z_i,h)$ is likely to be less than the number of observations in the sample N.

On the other hand, the estimation of the CV(h) does not require the explicit estimation of $\widehat{\beta}_{-i}(z_i,h)$, but the estimation of the leave-one-out error $\widetilde{e}_i(h)=y_i-\widehat{y}_{-i}(h)$. With linear regressions, one can obtain $\widetilde{e}_i(h)$ by rescaling the SVCM error $\widehat{e}(h)=y_i-\mathbf{X}_i'\widehat{\beta}(z_i,h)$ using the leverage statistic $[\text{lev}_i(z_i,h)]$[10]

$$\widetilde{e}_i(h)=y_i-\widehat{y}_{-i}(h)=\frac{y_i-\widehat{y}_i(h)}{1-\text{lev}_i(z_i,h)}=\frac{y_i-\mathbf{X}_i'\widehat{\beta}(z_i,h)}{1-\text{lev}_i(z_i,h)}=\frac{\widehat{e}_i(h)}{1-\text{lev}_i(z_i,h)} \tag{9}$$

where $\text{lev}_i(z_i,h)$ is the local leverage statistic, defined as the ith diagonal element of the local projection matrix $\mathbf{H}_{z_i}(h)=\mathcal{X}\{\mathcal{X}'\mathcal{K}_h(z_i)\mathcal{X}\}^{-1}\mathcal{X}'\mathcal{K}_h(z_i)$:

$$\text{lev}_i(z_i,h)=\mathcal{X}_i\left\{\mathcal{X}'\mathcal{K}_h(z_i)\mathcal{X}\right\}^{-1}\mathcal{X}_i'\times k(0) \tag{10}$$

Using this shortcut, we can rewrite CV(h) to reflect only the number of necessary regressions that need to be estimated. Consider the vector $\mathbf{z}_{pr}=(z_1,z_2,\ldots,z_J)$ of all unique values of Z, with $z_j<z_{j+1}$. Using this, we can rewrite the CV(h) as

$$\text{CV}(h)=\sum_{j=1}^{J}\sum_{i|Z_i=z_j}\omega(z_j)\left(\frac{y_i-\mathbf{X}_i'\widehat{\beta}(z_j,h)}{1-\text{lev}_i(z_j,h)}\right)^2=\sum_{j=1}^{J}\sum_{i|Z_i=z_j}\omega(z_j)\widetilde{e}_i(h)^2 \tag{11}$$

While (11) shows the number of estimated equations (J) is potentially smaller than the total number of observations in the sample (N), in some applications J may still be too large to allow for a fast estimation of CV$(\cdot)$. A feasible alternative in such cases is to use what Hoti and Holmström (2003) and Ichimura and Todd (2007) denominate as block or binned LL regressions to obtain an approximation of the CV criteria.

[10] Seber and Lee (2003, chap. 10) provide a simple demonstration of this identity for linear regression models. In addition, Hoover et al. (1998) suggest using a similar expression to speed up the calculation of the CV criteria.

Consider the vector $\mathbf{z}_{pr} = (z_1, z_2, \ldots, z_J)$ of all unique values of Z that are organized in P nonoverlapping bins Γ_p of width δ and a center equal to $\overline{z}_p$ such that

$$\delta = \frac{z_J - z_1}{p}; \quad \overline{z}_p = z_1 + \delta p - 0.5\delta \quad \forall p = 1 \ldots P \tag{12}$$
$$z_j \in \Gamma_p \quad \text{if} \quad \overline{z}_p - \frac{\delta}{2} \leq z_j < \overline{z}_p + \frac{\delta}{2} \quad \forall p = 1 \ldots P$$

Instead of estimating J sets of parameters, for each distinct value of Z, one estimates P sets of parameters using the points of reference $\overline{\mathbf{z}}_{pr} = (\overline{z}_1, \overline{z}_2 \ldots, \overline{z}_P)$. These parameters are used to obtain linear approximations around $\overline{z}_p$ for the predicted values $[\widehat{\widehat{y}}_i(h)]$, predicted errors $[\widehat{\widehat{e}}_i(h)]$, and leverage statistics $\widehat{\text{lev}}_i(\overline{z}_p, h)$ for all observations within their corresponding bins:

$$\widehat{y}_i(h) \cong \widehat{\widehat{y}}_i(h) = \mathbf{X}_i'\widehat{\beta}\left(\overline{z}_p, h\right) + \mathbf{X}_i'\left(Z_i - \overline{z}_p\right) \frac{\partial \widehat{\beta}\left(\overline{z}_p, h\right)}{\partial \overline{z}_p}$$
$$\text{if} \quad Z_i \in \Gamma_p \quad \forall p = 1 \ldots P \tag{13}$$
$$\widehat{e}_i(h) \cong \widehat{\widehat{e}}_i(h) = y_i - \widehat{\widehat{y}}_i(h) \tag{14}$$
$$\text{lev}_i(z_i, h) \cong \widehat{\text{lev}}_i(\overline{z}_p, h) = \mathcal{X}_i(\mathcal{X}'\mathcal{K}_h(\overline{z}_p)\mathcal{X})^{-1}\mathcal{X}_i' \times k\left(\frac{Z_i - \overline{z}_p}{h}\right)$$
$$\text{if} \quad Z_i \in \Gamma_p \quad \forall p = 1 \ldots P \tag{15}$$

Using these expressions, we can approximate the leave-one-out error $(\widetilde{\widetilde{e}}_i)$ for observation i with $Z_i = z_j$ and $z_j \in \Gamma_p$ as follows:

$$\widetilde{e}_i(h) \cong \widetilde{\widetilde{e}}_i(h) = \frac{y_i - \widehat{\widehat{y}}_i(h)}{1 - \widehat{\text{lev}}_i(\overline{z}_p, h)} = \frac{\widehat{\widehat{e}}_i(h)}{1 - \widehat{\text{lev}}_i(\overline{z}_p, h)} \tag{16}$$

This can be used to obtain an alternative expression for the CV criteria,

$$\text{CV}(h) \cong \widehat{\text{CV}}(h) = \sum_{p=1}^{P} \sum_{z_j \in \Gamma_p} \sum_{i|Z_i = z_j} \omega(z_j) \left(\frac{y_i - \widehat{\widehat{y}}_i(h)}{1 - \widehat{\text{lev}}_i(\overline{z}_p, h)}\right)^2$$
$$= \sum_{p=1}^{P} \sum_{z_j \in \Gamma_p} \sum_{i|Z_i = z_j} \omega(z_j) \widetilde{\widetilde{e}}_i(h)^2 \tag{17}$$

which reduces the number of estimated equations from J to P. We can see that as the number of groups P increases and the smaller the bin width δ is, the better the approximation of $\widehat{\text{CV}}(\cdot)$ to $\text{CV}(\cdot)$. As shown in Hoti and Holmström (2003), binned LL kernel regressions can provide good approximations for the overall model predictions as long as the ratio between the implicit bandwidth used for the construction of the bins and optimal bandwidth (δ/h^*) is relatively small.[11] In addition, even if one considers

[11] Simulations provided in Hoti and Holmström (2003) suggest that the binned estimator accuracy, measured by the relative integrated squared error, is similar to the unbinned estimator when $\delta/h^* < 0.3$ for Gaussian kernels and $\delta/h^* < 0.1$ for the Epanechnikov, triangle, and biweight kernels.

the bandwidth $\widehat{h}^*$ based on the approximation $\widehat{\text{CV}}(\cdot)$ to be a poor approximation of the full-information bandwidth h^*, it can still be used for exploratory analysis and as a starting point for the estimation of h^*, reducing the computational cost of bandwidth selection.

Automatic model selection

`vc_pack` offers two commands for the automatic model selection based on the modified CV procedure previously described, minimizing the objective function $\log\{\text{CV}(h)\}$. `vc_bw` implements a Newton–Raphson algorithm that works well when the objective CV function is smooth and differentiable, with a local minimum. This is an iterative algorithm that searches for optimal bandwidth h^* using

$$h_t = h_{t-1} - A \times \frac{\partial \log\{\text{CV}(h)\}}{\partial h} \, \text{abs}\left(\frac{\partial^2 \log\{\text{CV}(h)\}}{\partial h^2}\right)^{-1}\Bigg|_{h=h_{t-1}}$$

stopping when h_t and h_{t-1} are sufficiently close and selecting $h^* = h_t$. The first- and second-order derivatives are estimated using numerical methods with three points of reference. The scalar A is equal to 1 as long as there is an improvement in the maximization process [that is, $\text{CV}(h_t) < \text{CV}(h_{t-1})$]. Otherwise, A is reduced by half until an improvement is found.

`vc_bwalt` implements a bisection-type algorithm that works well in a larger set of scenarios, especially when $\text{CV}(\cdot)$ is not smooth or a differentiable function of h, but it may be slower in finding the optimal bandwidth. The algorithm starts with three points of reference: $h_0^0 < h_0^1 < h_0^2$. If the optimal bandwidth h^* is between h_0^0 and h_0^2 [that is, $\text{CV}(h_0^1) < \text{CV}(h_0^0)$ and $\text{CV}(h_0^1) < \text{CV}(h_0^2)$], the algorithm will evaluate the CV criteria using midpoints between h_0^1 and h_0^0 and h_0^1 and h_0^2 and update the reference points so $h_1^0 < h_1^1 < h_1^2$, with h_1^1 corresponding to the bandwidth with the lowest $\text{CV}(\cdot)$ and h_1^0 and h_1^2 corresponding to the two closest previously evaluated points of reference that are above and below h_1^1. If h^* is potentially smaller than h_1^0 [that is, $\text{CV}(h_1^0) < \text{CV}(h_1^1) < \text{CV}(h_1^2)$], a fourth point $h_1^x < h_1^0$ will be evaluated until finding a point such that $\text{CV}(h_1^x) > \text{CV}(h_1^0)$, which suggests h^* is between h_1^x and h_1^1. A similar process is implemented if h^* is potentially larger than h_1^2. The algorithm stops when h_t^0 and h_t^2 are sufficiently close, selecting $h^* = h_t^1$.

Both commands use the following syntax:

```
vc_bw[alt] depvar [indepvars] [if] [in], vcoeff(svar) [knots(#k) km(#km)
   bwi(#) trimsample(trimvar) kernel(kernel) plot]
```

depvar is the dependent variable Y, *indepvars* is the list of all independent variables $\mathbf{X}$ that we assume to have a conditional linear effect on the dependent variable Y, and *svar* is the smoothing variable Z.

`vcoeff(`*svar*`)` indicates the variable to be used for the estimation of smooth varying coefficients. `vcoeff()` is required.

`knots(`*#k*`)` and `km(`*#km*`)` are options that can be used to request the minimization of the approximate criteria $\widehat{\text{CV}}(h)$ as described in (17). Using `knots(`*#k*`)` with $\#k \geq 1$ requests the creation of a new variable that groups the smoothing variable *svar* into $\#k+1$ groups of equal width. Using `knots(0)` indicates to create $\#k+1$ groups, where *#k* is the nearest integer of $\min(N^{.5}, 10 \times \log_{10} N)$. When `knots(0)` is used, one can also use the option `km(`*#km*`)` so that *#k* is the nearest integer of $\min(N^{.5}, 10 \times \log_{10} N) \times \#km$.[12] Whenever `knots(`*#k*`)` is used, the command reports the number of knots used and the implicit bin width δ [see (12)].

The default is to use all distinct values in the smoothing variable, up to 500 distinct values. When more than 500 distinct values are detected, the command uses the options `knots(0) km(2)`. While there is nothing to indicate that this rule of thumb provides the most appropriate number of knots and implicit bin width (δ), simulations presented in Hoti and Holmström (2003) suggest that the approximate $\widehat{\text{CV}}$ criteria is reasonable if $(\delta/h^*) < 0.3$ when using Gaussian kernels, and $(\delta/h^*) < 0.1$ when using Epanechnikov, biweight, and triangular kernels.

Using the option `knots(-2)` requests the estimation of the CV criteria for all distinct values in the conditioning variable.

`bwi(`*#*`)` provides the command with an initial value h_0 for searching for the optimal bandwidth. The default uses the bandwidth from the command `lpoly` using the same kernel function declared in `kernel()`.

`trimsample(`*trimvar*`)` provides the name of a binary variable, *trimvar*, that indicates the subsample of the data that will be used for the estimation of the CV criteria. Observations with *trimvar* equal to zero are not used for the CV calculation. This plays the role of the weighting function $\omega(z_i)$.

`kernel(`*kernel*`)` indicates the kernel function [see (2)] that will be used to create the local weights and estimate the local regressions. The default is `kernel(gaussian)`, but other kernels are allowed.[13]

`plot` requests the command to plot all the bandwidths h and $\text{CV}(h)$ estimated internally. This can be used for visual inspection to verify the bandwidth is indeed minimizing the objective function.

After finishing the minimization process, the program stores the optimal bandwidth, the kernel function, and the smoothing variable name as globals: `$opbw_`, `$kernel_`, and `$vcoeff_`. This is done so other programs in the package can reuse this information.

[12] Stata uses this expression to define the number of bins used for a histogram as a default.

[13] See appendix A for the full list of kernels and functions available for the estimation.

3.2 Model estimation and inference: vc_reg, vc_preg, and vc_bsreg

Estimation of the variance–covariance matrix

As shown in section 2, once the bandwidth is selected, the estimation of the SVCM is a simple process that requires three steps:

1. Select the point or points of interest for which the model will be fit (typically a subset of all possible values of the smoothing variable z).
2. Construct the appropriate kernel weights based on the points of interest, selected kernel function, and the optimal bandwidth h^*.
3. Construct the auxiliary variable $(Z_i - z)$, which will be interacted with all independent variables in the model.

Once the auxiliary variables have been created, one can obtain the model coefficients and their gradients, conditional on all the selected points of interest, by estimating (7) using kernel-weighted least squares as in (8). The next step is the estimation of standard errors of the estimated parameters to obtain statistical inferences from the SVCM.

Following Li et al. (2002) and Li and Racine (2007, 2010), a feasible estimator for the asymptotic variance–covariance matrix of the SVCM, given a point of interest z and bandwidth h^*, can be obtained as follows:[14]

$$\begin{aligned}\widehat{\Sigma}_{\beta}^{as}(z,h^*) &= \widehat{\mathrm{Var}}_{as}\begin{bmatrix}\widehat{\beta}(z,h^*)\\ \frac{\partial\widehat{\beta}(z,h^*)}{\partial z}\end{bmatrix}\\ &= q_c\left\{\mathcal{X}'\mathcal{K}_h(z)\mathcal{X}'\right\}^{-1}\left\{\mathcal{X}'\mathcal{K}_h(z)\mathbf{D}\mathcal{K}_h(z)\mathcal{X}\right\}\left\{\mathcal{X}'\mathcal{K}_h(z)\mathcal{X}'\right\}^{-1} \qquad (18)\end{aligned}$$

$\mathbf{D}$ is a diagonal matrix where the ith element is equal to $\widehat{e}_i(h^*)^2$, and $\mathcal{K}_h(z)$ and $\mathcal{X}$ are defined as in (8). There is little guidance in the literature regarding q_c in the context of kernel regressions. Li et al. (2002) and Li and Racine (2007, 2010) assume $q_c = 1$, which is valid asymptotically. Note, however, that the expression given by (16) is the same as the robust standard errors for weighted least squares. Standard practice in those cases is to use $q_c = N/\{N - \dim(\mathcal{X})\}$, where $\dim(\mathcal{X})$ indicates the total number of coefficients that need to be fit in the model and N is the sample size. In semiparametric and nonparametric models, however, one must differentiate between sample size N and effective sample size (see section 3.3 on expected kernel observations).

Following the literature on the estimation of robust standard errors under heteroskedasticity (Long and Ervin 2000), one can estimate the variance–covariance matrix, substituting $\widehat{e}_i(h^*)^2$ with $\{\widehat{e}_i(h^*)^2\}/\{1-\mathrm{lev}_i(z,h^*)\}$ or $[\{\widehat{e}_i(h^*)\}/\{1-\mathrm{lev}_i(z,h^*)\}]^2$ in the diagonal matrix $\mathbf{D}$, where $\mathrm{lev}_i(z,h^*)$ is the leverage statistic as defined in (10). Here

[14] According to Li et al. (2002), the variance–covariance matrix for the SVCM can be consistently estimated by (18) if $E(e_i|x,z) = 0$, with conditional heteroskedasticity of unknown form $\sigma_e^2(x,z)$ and if $k(\cdot)$ is a standard second-order kernel function. Furthermore, it also assumes that as $N \to \infty$ and $h \to 0$, $Nh \to \infty$.

$q_c = 1$. This is equivalent to the estimation of heteroskedasticity-consistent 2 (HC2) and heteroskedasticity-consistent 3 (HC3) standard errors. According to Long and Ervin (2000), for the standard linear model, HC2 and HC3 outperform robust standard errors when the model is heteroskedastic and samples are relatively small ($N < 250$). While there is no formal study on the use of HC2 and HC3 standard errors when combined with SVCMs, I surmise that these standard errors may also be better than robust standard errors when the expected or effective sample size is small.

There is a debate regarding the use of analytical variance–covariance matrices in the framework of nonparametric kernel regressions. Cattaneo and Jansson (2018) advocate for the use of resampling methods, specifically paired bootstrap samples, to correctly estimate the variance–covariance matrix of the estimated coefficients when fitting kernel-based semiparametric models. In fact, they indicate percentile-based confidence intervals provide better coverage because paired bootstrap automatically corrects for nonnegligible estimation bias.[15] Broadly speaking, the paired bootstrap procedure, adapted to the estimation of SVCMs, is as follows:

1. Using the original sample $\mathbf{S} = (Y, \mathbf{X}, \mathbf{Z})$, estimate the coefficients $\widehat{\beta}(z, h^*)$ and $\partial\widehat{\beta}(z, h^*)/\partial z$, using the bandwidth h^* and all points of interest.

2. Obtain a paired bootstrap sample B with replacement from the original sample, and estimate $\widehat{\beta}_1^S(z, h^*)$ and $\partial\widehat{\beta}_1^S(z, h^*)/\partial z$ using the same points of interest as in 1 and bandwidth h^*.

3. Repeat 2 M times. The bootstrap standard errors for the coefficients are defined as

$$\widehat{\Sigma}_\beta^{\text{boot}}(z, h^*) = \widehat{\text{Var}}\begin{bmatrix}\widehat{\beta}(z, h^*) \\ \frac{\partial\widehat{\beta}(z, h^*)}{\partial z}\end{bmatrix} = \frac{1}{M-1}\begin{bmatrix}\widehat{\beta}^S(z, h^*) - E\{\widehat{\beta}^S(z, h^*)\} \\ \frac{\partial\widehat{\beta}^S(z, h^*)}{\partial z} - E\left(\frac{\partial\widehat{\beta}^S(z, h^*)}{\partial z}\right)\end{bmatrix}' \begin{bmatrix}\widehat{\beta}^S(z, h^*) - E\{\widehat{\beta}^S(z, h^*)\} \\ \frac{\partial\widehat{\beta}^S(z, h^*)}{\partial z} - E\left(\frac{\partial\widehat{\beta}^S(z, h^*)}{\partial z}\right)\end{bmatrix}$$

where $\widehat{\beta}^S(z, h^*)$ and $\partial\widehat{\beta}^S(z, h^*)/\partial z$ are vectors containing all the coefficients $\widehat{\beta}_m^S(z, h^*)$ and $\partial\widehat{\beta}_m^S(z, h^*)/\partial z$ that were estimated for each bootstrap sample $m = 1 \dots M$. The percentile confidence interval is defined as the lower $\alpha/2$ and upper $\alpha/2$ quantiles of the empirical distribution of $\widehat{\beta}_m^S(z, h^*)$ and $\partial\widehat{\beta}_m^S(z, h^*)/\partial z$, where α is the significance level.

[15] Note that Cattaneo and Jansson (2018) do not explicitly analyze the validity of their findings in the framework of SVCMs but rather provide general conclusions for what they call semiparametric kernel-based estimators. As a reference, `npregress kernel` reports the percentile confidence intervals by default using a paired bootstrap resampling procedure.

Implementation: vc_reg, vc_preg, and vc_bsreg

`vc_pack` offers three commands for the estimation of SVCMs, offering various alternatives for the estimation of the variance–covariance matrix ($\boldsymbol{\Sigma}_\beta$). `vc_reg` and `vc_preg` fit SVCMs using (18) for the estimation of $\boldsymbol{\Sigma}_\beta$, using different definitions for the model error $\widehat{e}_i(h^*)$. Bootstrap standard errors and percentile-based confidence intervals can be obtained using the command `vc_bsreg`.

`vc_preg` uses the SVCM error defined as $\widehat{e}_i(h^*) = y_i - \mathbf{X}_i' \widehat{\beta}(z_i, h^*)$ for the estimation of the asymptotic standard errors. `vc_reg` instead uses $\widehat{\widehat{e}}_i(h^*, z) = y_i - \mathbf{X}_i'\widehat{\beta}(z, h^*) - \mathbf{X}_i'(Z_i - z)\partial\widehat{\beta}(z, h^*)/\partial z$, which is the LL approximation of $\widehat{e}_i$ for the point of reference z.

While `vc_preg` produces the correct asymptotic standard errors, as suggested by Li and Racine (2007, 2010), it may be slow because the command fits the SVCM for all points of the smoothing variable Z to obtain the $\widehat{e}_i(h^*)$. `vc_reg` is faster by default because it uses only the LL approximation $\widehat{\widehat{e}}_i(h^*, z)$ and does not require additional steps for the estimation of the standard errors. These standard errors, however, contain approximation errors that increase the farther Z_i is from the point of reference z but can be used as a first quick approximation to analyze the data and draw statistical inferences. Empirically, `vc_reg` produces results that are comparable with the ones produced by `vc_preg` because observations where $\widehat{\widehat{e}}_i(h^*, z)$ and $\widehat{e}_i(h^*)$ differ greatly will have a small influence in the estimation of standard errors [(18)] because z_i is also likely to be far from the point of reference z.

The three commands share the same basic syntax:

`vc_`[`bs`|`p`]`reg` *depvar* [*indepvars*] [*if*] [*in*] [`, vcoeff(`*svar*`) bw(`*#*`)`
`kernel(`*kernel*`) cluster(`*varname*`) robust hc2 hc3 k(`*#*`) klist(`*numlist*`)`]

As with `vc_bw`[`alt`], *depvar* is the dependent variable Y, and *indepvars* are the set of explanatory variables ($\mathbf{X}$) that will have a linear effect on *depvar*, conditional on the smoothing variable *svar* (Z). `kernel()` and `bw()` are used to provide specific information regarding the model estimation. The default is to use information stored in `$vcoeff_`, `$kernel_`, and `$opbw_`.

Because the richness of SVCM comes from the estimation of the linear effects as a function of the smoothing variable Z, these commands offer two alternatives to select the points of interest over which the local regressions will be estimated. The option `k(`*#*`)`, which must be equal to or greater than 2, requests to estimate k regressions using equidistant points between the 1st and 99th percentiles of *svar*. `klist(`*numlist*`)` requests to estimate LL regression using each number of the list *numlist* as a reference point. When `klist()` contains a single number, the standard regression output is reported. Otherwise, when `k(`*#*`)` or `klist(`*numlist*`)` are used to fit two or more models, `vc_`[`bs`|`p`]`reg` produces no output but stores the betas and variance–covariance matrices for each regression as separate matrices in `e(b`*#*`)` and `e(V`*#*`)`. This information can be used to create plots of the coefficients across *svar*. Both vc_reg and vc_preg produce robust standard errors by default [(18)] but can also report HC2 and HC3 standard errors by using hc2 or hc3, respectively, as options. Clustered standard errors are also possible using the option cluster(cluster *varname*) but cannot be combined with the hc2 or hc3 options.

Because vc_preg requires full-information errors for the estimation of the variance–covariance matrix, by default, the command will obtain predictions for the errors $\widehat{e}_i(h)$ and leverage statistics $\text{lev}_i(z_i, h)$, using all distinct values of the smoothing variable Z (*svar*). Because this can be computationally costly, similar to our discussion on the calculation of the CV criteria, one can use the options knots() and km() to reduce the number of internally estimated regressions. This command uses the same default options as vc_bw[alt]. When binning options are used, the errors and leverage approximations defined in (13)–(15) are implemented. Alternatively, one can also provide the command with previously estimated sample errors $[\widehat{e}_i(h^*)]$ and leverage statistics $[\text{lev}_i(z, h^*)]$ using options err(*varname*) and lev(*varname*), respectively.

vc_bsreg estimates bootstrap standard errors using a paired bootstrap strategy. Following the syntax for the command bootstrap, one can specify information for strata() and cluster() as well as define a seed() for the reproducible generation of the random samples. The default number of bootstrap samples is 50, but that can be changed using the option reps(#). In addition to the bootstrap standard errors, vc_bsreg stores the 95% percentile confidence interval, but it can be changed to other levels with the option pci(#), using any number between 0 and 100.

When estimating a single equation, vc_reg, vc_preg, and vc_bsreg store two variables in the dataset: _delta_, containing $Z_i - z$, and _kwgt_, containing the standardized kernel weights (see the next section).

3.3 Model postestimation: vc_predict and vc_test

vc_pack provides two commands that can be used to obtain summary statistics of the model as well as report some tests for model specification against parametric alternatives. The first command, vc_predict, has a similar syntax to vc_bw[alt] and vc_[p|bs]reg:

```
vc_predict depvar [indepvars] [, vcoeff(svar) bw(#) kernel(kernel)
   yhat(newvar) res(newvar) looe(newvar) lvrg(newvar) stest knots(#)
   km(#) ]
```

In addition to the options previously described, vc_predict can be used to obtain predictions of the model $\widehat{y}_i(h^*)$ (yhat(*newvar*)), the model residual $\widehat{e}_i(h^*)$ (res(*newvar*)), the leverage statistic $\text{lev}_i(z_i, h)$ (lvrg(*newvar*)), or the leave-one-out error $\widetilde{e}(h^*)$ (looe(*newvar*)) [(9) and (10)]. Each one of these options requires one to specify a new variable name (*newvar*) to store the specified information. One can also use the options knots() and km() to speed up the computing process, in which case the approximations described in (13)–(15) are used.

Residuals and leverage from this command can be used, for example, for the estimation of SVCMs using vc_preg. This command also provides some basic information about the model and performs some specification tests when the option stest is used. The next section describes the methods and formulas reported by this command.

Log mean squared leave-one-out errors

Consider the SVCM described in (4). Given the smoothing variable Z (*svar*), kernel function (kernel()), and bandwidth (bw()), vc_predict reports the log of the mean squared leave-one-out error (MSLOOE),

$$\log \text{MSLOOE} = \log \left\{ \sum_{j=1}^{J} \sum_{i|Z_i=z_j} \left(\frac{y_i - \widehat{y}_i(h)}{1 - \text{lev}_i(z_j, h)} \right)^2 \right\}$$

when no binning option is used, or its approximation,

$$\log \text{MSLOOE} = \log \left\{ \sum_{p=1}^{P} \sum_{z_j \in \Gamma_p} \sum_{i|Z_i=z_j} \left(\frac{y_i - \widehat{\widehat{y}}_i(h)}{1 - \widehat{\text{lev}}_i(\overline{z}_p, h)} \right)^2 \right\}$$

when binning options (knots() km()) are used. This is the same statistic used for the model selection, except that it does not use the weighting factor $\omega(z_i)$ for its calculation.

Goodness-of-fit $\mathbf{R^2}$

vc_predict produces two measures of the goodness-of-fit statistic that are direct analogues to standard linear models. The first one is based on the standard decomposition of the sum of squares,

$$R_1^2 = 1 - \frac{\text{SSR}}{\text{SST}} = 1 - \frac{\Sigma\{y_i - \widehat{y}_i(h)\}^2}{\Sigma(y_i - \overline{y})^2} \tag{19}$$

which is the same one used by npregress kernel. Because this statistic is known to produce undesirable results, such as negative values for R_1^2, vc_predict also reports the goodness-of-fit statistic suggested in Henderson and Parmeter (2015):

$$R_2^2 = \frac{\left[\Sigma\left(y_i - \overline{y}\right)\left\{\widehat{y}_i(h) - \overline{y}\right\}\right]^2}{\Sigma\left(y_i - \overline{y}\right)^2 \Sigma\left\{\widehat{y}_i(h) - \overline{y}\right\}^2} \tag{20}$$

When binning options are used, $\widehat{y}_i(h)$ is substituted by $\widehat{\widehat{y}}_i(h)$ in (19) and (20).

Model and residual degrees of freedom

The effective number of degrees of freedom is a statistic that has proven useful in the literature of nonparametric econometrics for the comparison of models with different types of smoothers. Following the terminology from Hastie and Tibshirani (1990), consider any parametric and nonparametric model with a projection matrix $\mathbf{S}$ of dimension $N \times N$ such that $\widehat{\mathbf{Y}} = \mathbf{SY}$, where $\widehat{\mathbf{Y}}$ is an $N \times 1$ vector of the predicted values corresponding to any particular model. Hastie and Tibshirani (1990) emphasize two estimators for the estimation of the number of degrees of freedom:

$$
\begin{aligned}
\mathrm{df}_1 &= \mathrm{tr}(\mathbf{S}) \\
\mathrm{df}_2 &= \mathrm{tr}(2\mathbf{S} - \mathbf{SS}')
\end{aligned} \tag{21}
$$

In the context of linear regression models, where the projection matrix $\mathbf{S} = \mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}$, these definitions are equivalent to each other. However, in the case of kernel regressions and penalized smooth spline regressions, the matrix $\mathbf{S}$ is not symmetrical and the above definitions of degrees of freedom will differ from each other. df_1 is commonly used as an approximation of the number of degrees of freedom of the model, whereas $N - \mathrm{df}_2$ is used as the number of residual degrees of freedom.

For the specific case of SVCMs, the projection matrix $\mathbf{S}$ is defined as

$$
\mathbf{S} = \sum_{j=1}^{J} \boldsymbol{\gamma}_{z_j} \mathcal{X} \left\{ \mathcal{X}' \mathcal{K}_h(z_j) \mathcal{X} \right\}^{-1} \mathcal{X}' \mathcal{K}_h(z_j)
$$

where $\boldsymbol{\gamma}_{z_j}$ is an $N \times N$ matrix with the ith diagonal element equal to 1 if $Z_i = z_j$ and 0 elsewhere. This implies that the first measure of degrees of freedom is equivalent to

$$
\mathrm{df}_1 = \mathrm{tr}(\mathbf{S}) = \sum_{i=1}^{N} S_{ii} = \sum_{i=1}^{N} \mathrm{lev}_i(z_i, h) \tag{22}
$$

The second measure of degrees of freedom is computationally more difficult to estimate because it requires N^2 operations. As an alternative, Hastie and Tibshirani (1990) suggest using the following approximation:

$$
\mathrm{df}_2 = \mathrm{tr}(2\mathbf{S} - \mathbf{SS}') \cong 1.25\,\mathrm{tr}(\mathbf{S}) - 0.5 = 1.25\,\mathrm{df}_1 - 0.5 \tag{23}
$$

`vc_predict` reports df_1 and $n - \mathrm{df}_2$ as measures of model and residual degrees of freedom, respectively. When binning options are used, $\mathrm{lev}_i(z_i, h)$ is substituted by $\widehat{\mathrm{lev}}_i(\overline{z}_p, h)$ in (22).

Expected kernel observations

One of the drawbacks of nonparametric regression analysis is the more rapid the decline of the effective number of observations used for the estimation of the parameters of interest, the larger the number of explanatory variables used in the model (the curse of dimensionality) and the smaller the bandwidths. To provide the user with a statistic summarizing the amount of information used through the estimation process, researchers commonly report $n|h|$ as the expected number of kernel observations $E(\text{Kobs})$, where $|h|$ is the product of all bandwidths of the explanatory variables.[16] This statistic, however, can be misleading.

Consider the estimation of a model with a single independent variable for which an optimal bandwidth h^* is selected. If the scale of the independent variable doubles, the optimal bandwidth of the rescaled variable will double, but $E(\text{Kobs})$ should remain the same. The statistic $n|h|$, however, suggests that the $E(\text{Kobs})$ has doubled as well.[17]

As an alternative measure to $n|h|$, I propose a statistic based on what I denominate standardized kernel weights $K_w(Z_i, z, h)$, which are defined as[18]

$$K_w(Z_i, z, h) = k_w\left(\frac{Z_i - z}{h}\right) = \frac{1}{k(0)} k\left(\frac{Z_i - z}{h}\right)$$

These kernel weights are guaranteed to fall between 0 and 1. While this change in the scale of local weights has no impact on the estimation of the point estimates of the models, it provides a more intuitive understanding of the role of weights in the estimation process. Observations where Z_i is equal to z will receive a weight of 1, and one can consider that the information of that observation is fully used when estimating the LL regression. If an observation has a $k_w(\cdot)$ of, say, 0.5, one can consider that the information contributed by that observation to the local kernel regression is half of an observation where $Z_i = z$. Finally, observations with a $k_w(\cdot) = 0$ do not contribute to the local estimation at all. These kernel weights can be used to estimate the effective number of observations $\{\text{Kobs}(z)\}$ used for the estimation of the parameters of interest for a given point of reference z:[19]

$$\text{Kobs}(z) = \sum_{i=1}^{n} k_w\left(\frac{Z_i - z}{h}\right) \tag{24}$$

Because areas with higher density use more observations than areas where z is sparsely distributed, the expected number of kernel observations $E(\text{Kobs})$ can be defined as the simple weighted average of $\text{Kobs}(z_j)$ using all observations in the sample. This leads to

$$E(\text{Kobs}) = \frac{1}{N}\sum_{j=1}^{J} N_j \text{Kobs}(z_j) = \frac{1}{N}\sum_{j=1}^{J} N_j \sum_{i=1}^{n} k_w\left(\frac{Z_i - z_j}{h}\right) \tag{25}$$

[16] `npregress kernel` reports this statistic as “expected kernel observations”.

[17] To prevent unexpected results, `npregress kernel` sets the maximum value $E(\text{Kobs})$ as the sample size.

[18] See appendix A for a list of standardized kernel weight functions.

[19] This statistic can also be extended to multivariable kernel regression models by simply using the standardized kernels across all independent variables.

where N_j is the number of observations with $Z_i = z_j$. When binning options are used, the estimator is

$$\text{Kobs}(\overline{z}_p) = \sum_{i=1}^{n} k_w \left(\frac{Z_i - \overline{z}_p}{h} \right)$$

$$\widehat{E}(\text{Kobs}) = \frac{1}{N} \sum_{p=1}^{P} N_p \text{Kobs}(\overline{z}_p) = \frac{1}{N} \sum_{p=1}^{P} N_p \sum_{i=1}^{n} k_w \left(\frac{Z_i - \overline{z}_p}{h} \right)$$

where N_p is the number of observations that fall within the pth bin.

If Z is continuous, this statistic has two convenient properties with respect to the bandwidth h:

$$\lim_{h \to 0} E(\text{Kobs}) = 1 \quad \text{and} \quad \lim_{h \to \infty} E(\text{Kobs}) = N$$

This provides a more intuitive understanding of the effect of the bandwidth on the average amount of information used for the estimation of local regressions compared with the standard parametric model. At least, there will be one observation for the estimation of the local estimation, and at most, all the data will be used for each local estimation. This statistic is also reported after `vc_predict`.

Specification tests

In addition to reporting the basic summary statistics described above, `vc_predict` can produce basic specification tests when the option `stest` is specified. The specification tests follow Hastie and Tibshirani (1990) and provide what the authors call an approximate F test, comparing the SVCM with four parametric alternatives:

$$\text{Model 0:} \quad y_i = (\mathbf{X}_i'; Z_i)\boldsymbol{\beta}_0 + \epsilon_i \quad \text{with} \quad \text{df}_p = q + 1 \tag{26}$$

$$\text{Model 1:} \quad y_i = (\mathbf{X}_i'; Z_i \otimes \mathbf{X}_i')\boldsymbol{\beta}_1 + \epsilon_i \quad \text{with} \quad \text{df}_p = 2q \tag{27}$$

$$\text{Model 2:} \quad y_i = (\mathbf{X}_i'; Z_i \otimes \mathbf{X}_i'; Z_i^2 \otimes \mathbf{X}_i')\boldsymbol{\beta}_2 + \epsilon_i \quad \text{with} \quad \text{df}_p = 3q \tag{28}$$

$$\text{Model 3:} \quad y_i = (\mathbf{X}_i'; Z_i \otimes \mathbf{X}_i'; Z_i^2 \otimes \mathbf{X}_i'; Z_i^3 \otimes \mathbf{X}_i')\boldsymbol{\beta}_3 + \epsilon_i \quad \text{with} \quad \text{df}_p = 4q \tag{29}$$

q is the number of explanatory variables defined in $\mathbf{X}$ plus the constant. Define $n - \text{df}_2$ as the residual degrees of freedom of the SVCM [see (21)] and $\widehat{\epsilon}_i$ as the predicted residual for the parametric model 0, 1, 2, or 3. The approximate F statistic is defined as

$$aF = \frac{\Sigma\widehat{\epsilon}_i^2 - \Sigma\widehat{e}_i(h^*)^2}{\Sigma\widehat{e}_i(h^*)^2} \times \frac{n - \text{df}_2}{\text{df}_2 - \text{df}_p}$$

The null hypothesis (H_0) is that the parametric model 0, 1, 2, or 3 is correctly specified, whereas the alternative hypothesis is that the SVCM is correct. While the exact distribution of this statistic is not known, Hastie and Tibshirani (1990, 65) suggest using critical values for an F statistic with $n - \text{df}_2$ degrees of freedom in the numerator and $\text{df}_2 - \text{df}_p$ degrees of freedom in the denominator, a rough test for a quick inspection of the model specification. When binning options are used, $\Sigma\widehat{e}_i(h^*)^2$ is substituted by $\Sigma\widehat{\widehat{e}}_i(h^*)^2$ (13)–(15).

Cai, Fan, and Yao (2000) specification test: vc_test

Because the exact distribution of the approximate F statistic is not known, vc_pack also offers the implementation of the specification test proposed by Cai, Fan, and Yao (2000), based on a wild bootstrap approach, as described in Henderson and Parmeter (2015). The test statistic is constructed similarly to the approximate F statistic but without adjusting for the differences in degrees of freedom,

$$\widehat{\mathcal{J}} = \frac{\Sigma\widehat{\epsilon}_i^2 - \Sigma\widehat{e}_i(h^*)^2}{\Sigma\widehat{e}_i(h^*)^2} \tag{30}$$

where $\widehat{\epsilon}_i$ corresponds to the residuals for the parametric model—see (26)–(29)—and $\widehat{e}_i(h^*)$ corresponds to the residuals of the SVCM. The null hypothesis (H_0), which states that the parametric model is correctly specified, is rejected in favor of the SVCM if the statistic $\widehat{\mathcal{J}}$ is above some critical value.

Because the distribution of the statistic $\mathcal{J}$ is not known, a wild bootstrap procedure can be used to obtain its empirical distribution using the following procedure:

1. Define $\widehat{\epsilon}_i$ to be a predicted residual based on the parametric model [(26)–(29)].
2. Construct a new dependent variable y_w^*, using a two-point wild bootstrap error as follows:
$$y_{w,i}^* = (y_i - \widehat{\epsilon}_i) + \widehat{\epsilon}_i \times \left(\frac{1+\sqrt{5}}{2} - v_i\sqrt{5}\right)$$
v_i follows a Bernoulli distribution with $p = \{(1+\sqrt{5})/(2\sqrt{5})\}$.
3. Using the new dependent variable $y_{w,i}^*$, refit the parametric model and SVCM, using the optimal bandwidth h^*, and calculate the statistic $\widehat{\mathcal{J}}_w^*$.
4. Repeat 2 and 3 enough times to obtain the empirical distribution of the statistic $\mathcal{J}$.
If $\widehat{\mathcal{J}}$ is greater than the upper α-percentile of the empirical distribution obtained through the wild bootstrap procedure, one can reject the null hypothesis.

The command vc_test implements this specification test using the following syntax.

vc_test *depvar* [*indepvars*] [*if*] [*in*] [, vcoeff(*svar*) bw(*#*) kernel(*kernel*) knots(*#*) km(*#*) degree(*#d*) wbsrep(*#wb*)]

As with previous commands, one has to specify the dependent and independent variables in the model, but specifying vcoeff(*svar*), kernel, and bandwidth is optional. The program uses the information stored by vc_bw[alt] by default. Because the test requires the estimation of the whole model multiple times, one can specify the options knots() and km() to implement the binned version of the statistic and increase the speed of the computations. This substitutes $\widehat{e}_i(h^*)$ with $\widehat{\widehat{e}}_i(h^*)$ in (30).

`degree(`*#d*`)` is used to define the model under the null hypothesis. *#d* can take the values 0, 1, 2, or 3, which correspond to the models described in (26)–(29). The default is `degree(0)`.

`wbsrep(`*#wb*`)` is used to indicate the number of wild bootstrap repetitions used for the estimation of the empirical distribution of the statistic $\mathcal{J}$. The default number of repetitions is 50. The command reports the 90th, 95th, and 97.5th percentiles of the empirical distribution of $\mathcal{J}$ to be used as critical values.

3.4 Model visualization: vc_graph

One attractive feature of semiparametric models in general, and SVCMs in particular, is the potential to visualize effects across the range of the explanatory variables that enter the model nonparametrically. These plots can be used for a richer interpretation of marginal effects. As described in section 3.2, when `vc_`[`bs`|`p`]`reg` is used to fit models for more than one point of reference, the command produces no report but stores the coefficients, variances, and confidence intervals in `e()`.

`vc_graph` is a command that can be used as a postestimation tool to produce plots of coefficients of the independent variables or their gradients, using the information estimated via `vc_`[`bs`|`p`]`reg`. The command uses the following syntax:

vc_graph [*varlist*] [**, ci(***#***) constant delta xvar(***xvarname***) graph(***stub***)**
  rarea ci_off pci addgraph(*str***)**]

varlist can contain a subset of all independent variables used in the estimation of the SVCM. If factor variables and interactions are used, the same format must be used when using `vc_graph`.

`ci(`*#*`)` sets the level of the confidence intervals, using any number between 0–100. The default is `ci(95)`. Confidence intervals can be omitted from the plot using the option `ci_off`.

`constant` is used to plot the varying coefficients associated with the constant.

`delta` plots the gradients $\partial\beta(z)/\partial z$ of the variables listed in *varlist*. The default is to plot the coefficients $\beta(z)$. If the options `delta` and `constant` are used, `vc_graph` will plot the coefficient of the auxiliary variable $(Z - z)$.

`xvar(`*xvarname*`)` specifies to use a different variable to plot the smooth varying coefficients, as long as this variable *xvarname* is a monotonic transformation of the original variable *svar* used of the estimation. For example, say that the SVCM model was fit using the variable *svar* as the smoothing variable because it has fewer areas with scant distribution. The researcher, however, is interested in plotting coefficients across *svar1*, rather than *svar*. If *svar1* is a monotonic transformation of *svar*, using the option `xvar(`*svar1*`)` requests plotting coefficients using *svar1* on the horizontal axis. Internally, the mapping between the points of reference from *svar* to *svar1* is done using an LL approximation if the exact values are not available.[20]

`graph(`*stub*`)` provides a *stub* to be used as prefix for the created plots, which are saved in memory. The default is `graph(grph)`, which numbers the plots consecutively.

`rarea` requests using an “area” graph for the estimation of the confidence intervals; the default is to use `rcap`.

`ci_off` requests to plot only the point estimates but not the confidence intervals.

`pci`, when used with `vc_bsreg`, requests to plot the percentile-based confidence intervals rather than normal-based intervals. This option cannot be used with `ci()`.

`addgraph(`*str*`)` requests to add a plot to the generated graph. For example, `vc_graph x1, addplot(scatter g x1)` will create a `twoway` graph that includes the `scatter g x1`.

4 Illustration: Determinants of drunk driving citations: The role of fines

For this illustration, I use the fictitious `dui.dta`, presented in section 2.4, to analyze how the number of drunk driving citations is affected by whether a jurisdiction taxes alcohol, whether there is a college in the jurisdiction, and whether the jurisdiction is in a small, medium, or large city, conditional on fines imposed for drunk driving.

I start the analysis by using `vc_bw` to select the optimal bandwidth, using the leave-one-out CV strategy and the default options.

```
. webuse dui
(Fictional data on monthly drunk driving citations)
. vc_bw citations taxes college i.csize, vcoeff(fines)
Kernel: gaussian
Iteration: 0 BW:   0.5539764 CV:   3.129985
Iteration: 1 BW:   0.6870523 CV:   3.120199
Iteration: 2 BW:   0.7343729 CV:   3.119504
Iteration: 3 BW:   0.7397456 CV:   3.119497
Iteration: 4 BW:   0.7397999 CV:   3.119497
Bandwidth stored in global $opbw_
Kernel function stored in global $kernel_
VC variable name stored in global $vcoeff_
```

The command suggests a bandwidth of 0.7398, suggesting the bandwidth used in section 2.4 may have been undersmoothing the results.

[20] There is no robust theoretical discussion in regard to using transformations of independent variables for the selection and estimation of nonparametric and semiparametric models. However, I surmise that monotonic transformations can be used as an alternative to variable bandwidths by allowing more information to be used in areas with low density, reducing the variance of the estimator.

Next, I obtain simple summary statistics of the model using `vc_predict`. I also request to report an approximate F test for model specification against the models where `fines` is added as an interaction in the model.

```
. vc_predict citations taxes college i.csize, vcoeff(fines) stest
Smooth Varying coefficients model
Dep variable       : citations
Indep variables    : taxes college i.csize
Smoothing variable : fines
Kernel             : gaussian
Bandwidth          :    0.73980
Log MSLOOER        :    3.11950
Dof residual       :    477.146
Dof model          :     18.684
SSR                :  10323.152
SSE                :  37886.159
SST                :  47950.838
R2-1 1-SSR/SST     :    0.78471
R2-2               :    0.79010
E(Kernel obs)      :    277.835

Specification Test approximate F-statistic
H0: Parametric Model
H1: SVCM y=x*b(z)+e
Alternative parametric models:
Model 0 y=x*b0+g*z+e
F-Stat: 8.24705 with pval 0.00000
Model 1 y=x*b0+g*z+(z*x)b1+e
F-Stat: 5.80964 with pval 0.00000
Model 2 y=x*b0+g*z+(z*x)*b1+(z^2*x)*b2+e
F-Stat: 0.75977 with pval 0.65174
Model 3 y=x*b0+g*z+(z*x)*b1+(z^2*x)*b2+(z^3*x)*b3+e
F-Stat: -2.07399 with pval 1.00000
```

The report indicates that the model uses approximately 18.7 degrees of freedom [(22)], whereas the residuals have 477.15 degrees of freedom [(23)]. The model has an R^2 that is larger than the simple regression model ($R^2_{\mathrm{OLS}} = 0.718$) but is somewhat smaller than the R^2 obtained using the full nonparametric model ($R^2_{np} = 0.81$).[21] The second measure of R^2 [see (20)] is larger than the standard measure of goodness of fit. Finally, the expected number of kernel observations is 277.8 [(25)], suggesting that, on average, half of the whole sample is used for each local regression.

The approximate F test suggests rejecting models 0 and 1 in favor of the SVCM, but one cannot reject the null hypothesis that a model with a quadratic interaction with fines is correctly specified. The local fit of the model with a cubic interaction seems to be better than the SVCM, which explains why the F statistic is negative. I also use `vc_test` to implement the alternative specification test, comparing the same parametric models with the SVCM. For this example, I use 200 repetitions, using the option `wbsrep(200)`. Because model 0 was overwhelmingly rejected and model 3 seems to have a better fit than the SVCM, only the results comparing models 1 and 2 are shown:

[21] See the do-file that accompanies this article to see where this R^2 comes from.

```
. vc_test citations taxes college i.csize, degree(1) wbsrep(200) seed(1)
  (output omitted)
Specification test.
H0: y=x*b0+g*z+(z*x)*b1+e
H1: y=x*b(z)+e
J-Statistic      :0.16869
Critical Values
90th   Percentile:0.09382
95th   Percentile:0.10351
97.5th Percentile:0.10685
. vc_test citations taxes college i.csize, degree(2) wbsrep(200) seed(1)
  (output omitted)
Specification test.
H0: y=x*b0+g*z+(z*x)*b1+(z^2*x)*b2+e
H1: y=x*b(z)+e
J-Statistic      :0.01410
Critical Values
90th   Percentile:0.01177
95th   Percentile:0.01490
97.5th Percentile:0.01725
```

The results are consistent with the approximate F statistic. In the first case, the $\mathcal{J}$ statistic of the model is 0.16869, which is larger than the 97.5th percentile of the empirical distribution of the statistic, suggesting the rejection of the null of a linear interaction. The test comparing with the parametric quadratic interaction model is less conclusive. The $\mathcal{J}$ statistic is 0.0141, which suggests rejection of the null at 10% of confidence, but the null cannot be rejected at 5%. Despite these results, I will go ahead and fit the SVCM.

To provide an overview of the results that are similar to the output of standard regression models, I provide the conditional effects using the 10th, 50th, and 90th percentiles of fines as the points of reference, which correspond to 9, 10, and 11. This is done with the three commands available in vc_pack, using the basic syntax:

```
vc_reg citations taxes college i.csize, klist(9)
vc_preg citations taxes college i.csize, klist(9)
vc_bsreg citations taxes college i.csize, klist(9) seed(1)

vc_reg citations taxes college i.csize, klist(10)
vc_preg citations taxes college i.csize, klist(10)
vc_bsreg citations taxes college i.csize, klist(10) seed(1)

vc_reg citations taxes college i.csize, klist(11)
vc_preg citations taxes college i.csize, klist(11)
vc_bsreg citations taxes college i.csize, klist(11) seed(1)
```

The results from these regressions are shown in table 1, columns 2–4, showing the standard errors obtained with all three commands (robust, F robust, and bootstrap). Column 1 presents the results for the standard regression model. For the SVCM, two subcolumns are provided. The left subcolumn shows the conditional effects $\beta(z)$, whereas the right subcolumn shows the gradient of that effect $\partial\beta(z)/\partial z$.

Table 1. Determinants of number of monthly citations, conditional on fines. SVCM

Number of monthly	OLS	SVCM					
drunk driving citations	(1)	(2)		(3)		(4)	
		Fines = 9		Fines = 10		Fines = 11	
		$\beta(z)$	$\frac{\partial\beta(z)}{\partial z}$	$\beta(z)$	$\frac{\partial\beta(z)}{\partial z}$	$\beta(z)$	$\frac{\partial\beta(z)}{\partial z}$
1 if alcoholic beverages are taxed	−4.494	−6.377	3.008	−3.959	1.093	−3.843	−0.0505
Robust std err	(0.582)	(1.147)	(1.373)	(0.496)	(0.823)	(0.736)	(0.788)
F robust std err		(1.059)	(1.210)	(0.493)	(0.787)	(0.711)	(0.751)
Bootstrap std err	(0.638)	(1.322)	(1.525)	(0.498)	(0.967)	(0.812)	(0.833)
1 if college town; 0 otherwise	5.828	9.871	−4.578	5.305	−3.191	3.797	−1.024
Robust std err	(0.588)	(1.113)	(1.318)	(0.516)	(0.896)	(0.888)	(0.963)
F robust std err		(1.021)	(1.164)	(0.513)	(0.836)	(0.860)	(0.915)
Bootstrap std err	(0.634)	(1.201)	(1.381)	(0.470)	(0.972)	(0.884)	(0.926)
City size							
medium	5.492	6.734	−1.299	5.284	−2.332	3.051	−2.196
Robust std err	(0.532)	(0.973)	(1.125)	(0.535)	(0.785)	(0.782)	(0.843)
F robust std err		(0.936)	(1.069)	(0.538)	(0.786)	(0.772)	(0.831)
Bootstrap std err	(0.547)	(0.932)	(1.265)	(0.588)	(0.760)	(0.833)	(0.958)
large	11.24	14.99	−4.863	10.60	−3.779	7.784	−2.691
Robust std err	(0.571)	(1.146)	(1.373)	(0.510)	(0.852)	(0.750)	(0.764)
F robust std err		(1.071)	(1.233)	(0.509)	(0.822)	(0.741)	(0.751)
Bootstrap std err	(0.610)	(1.095)	(1.323)	(0.553)	(0.809)	(0.749)	(0.812)
Drunk driving fines (thousands of dollars)	−7.690		−8.256		−4.906		−3.673
Robust std err	(0.384)		(1.327)		(0.782)		(0.816)
F robust std err			(1.211)		(0.792)		(0.804)
Bootstrap std err	(0.405)		(1.473)		(0.787)		(0.810)
Constant	94.22	23.96		16.80		12.93	
Robust std err	(3.949)	(1.168)		(0.474)		(0.746)	
F robust std err		(1.099)		(0.478)		(0.737)	
Bootstrap std err	(4.117)	(1.255)		(0.501)		(0.819)	
N obs and *K* obs	500	243.19		341.64		203.36	

NOTES: Robust std err corresponds to the output with `vc_reg`, *F* robust standard errors were estimated with `vc_preg`, and bootstrap standard errors with `vc_bsreg`. *K* obs is defined as the sum of standardized kernel weights based on the point of reference [see (24)].

Overall, robust standard errors obtained with the LL approximation (`vc_reg`) seem to be a reasonably good approximation to the full-information robust standard errors (`vc_preg`), with the largest discrepancies observed in areas where the density of the distribution of fines is low (at the top and bottom). These estimates are also consistent with the bootstrap standard errors (`vc_bsreg`).

To complement the information in this table and before we provide an interpretation of the results, figure 2 provides a plot with 95% confidence intervals for the conditional effects of all variables in the model, using a predefined set of points of interest. I first use `vc_preg` to fit the SVCM:[22]

```
. vc_preg citations taxes college i.csize, klist(7.4(.2)12)
Estimating SVCM over 24 point(s) of reference
Smoothing variable:  fines
Kernel function   :  gaussian
Bandwidth         : 0.73980
vce               : robust
Estimating Full model
More than 1 point of reference specified
Results will not be saved in equation form but as matrices
```

The main difference with the previous examples is that the option `klist()` contains a list of numbers, or reference points, over which I am requesting the SVCM to be fit. It indicates that there are 24 points of reference, from 7.4 to 12. Once the estimation is finished, figure 2 can be reproduced with the following commands:

[22] Results using the bootstrap procedure and percentile confidence intervals are provided in the accompanying do-file or upon request.

```
. vc_graph taxes college i.csize
. graph combine grph1 grph2 grph3 grph4
```

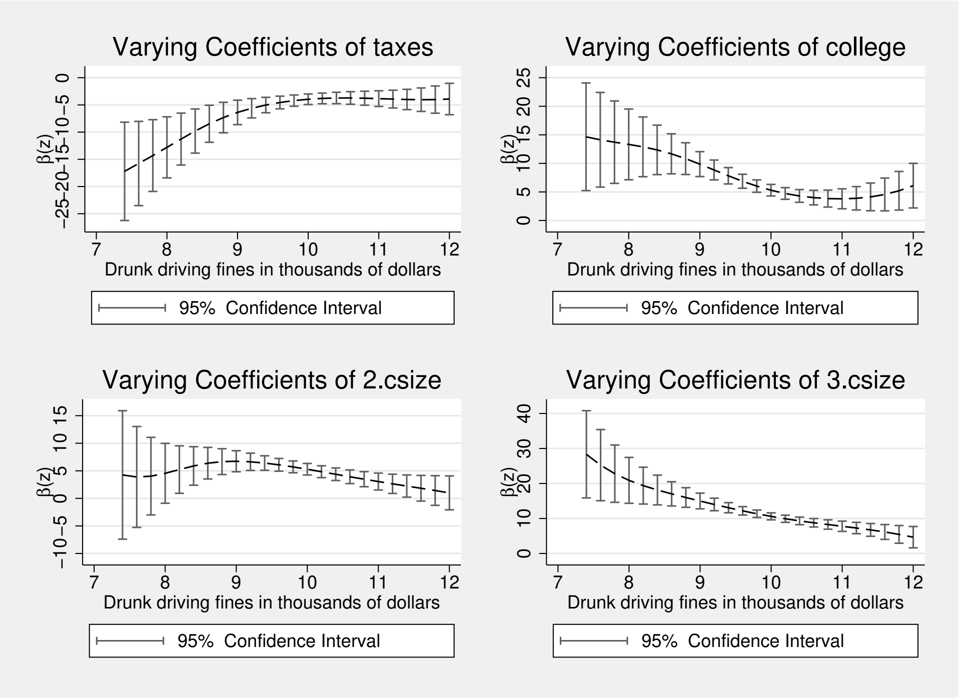

Figure 2. SVCM: Conditional effects across fines

If one is interested in the gradients $\partial\beta(z)/\partial z$, they can be plotted using the following commands:

```
. vc_graph taxes college i.csize, delta
. graph combine grph1 grph2 grph3 grph4
```

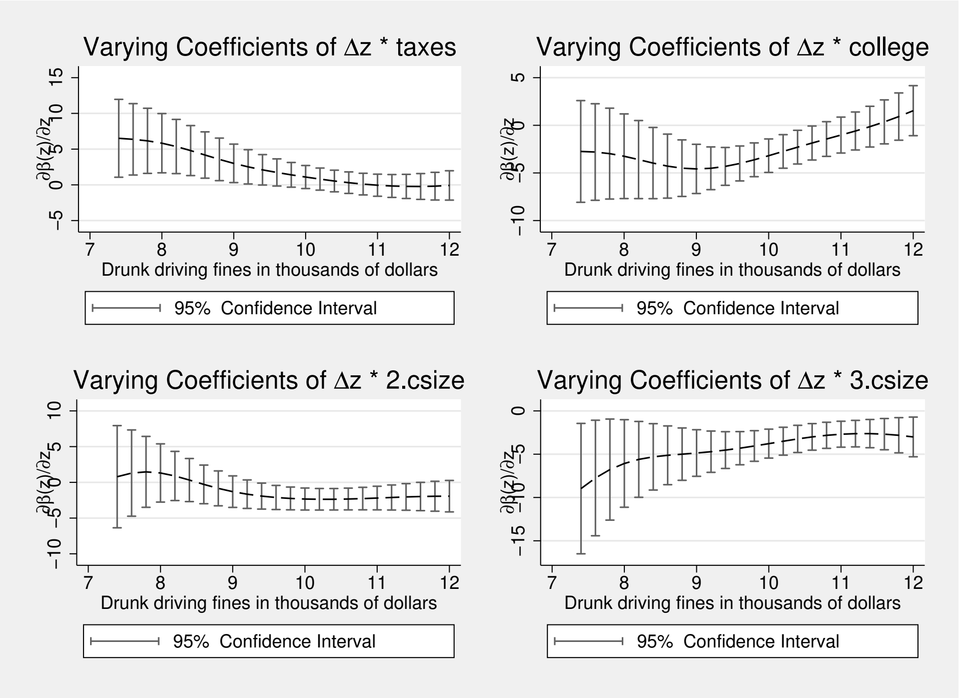

Figure 3. SVCM: Change of the conditional effects across fines $\partial\beta(z)/\partial z$

An interpretation of these results can be given as follows. Overall, when alcoholic beverages are taxed, the number of monthly citations per month decreases by 4.5 units (table 1 column 1). This effect is larger in jurisdictions with low fines, with a point estimate ranging from 15 to just below 4, and in jurisdictions with fines levels above 10. No differences can be observed in the conditional effect for fine levels above 10. This is mirrored by the fact that the estimate of $\partial\beta(z)/\partial z$ in figure 3 is statistically equal to zero.

If there is a college campus in town, the number of citations per month is about 5.8 higher. The conditional impact of college declines as fines increase by almost 10 points between the minimum and maximum levels of fines in the distribution. Based on the estimates from figure 3, when fines are above 11, the change in the effect of college on citations is no longer statistically significant. If the jurisdiction is located in a medium city, the impact on the number of citations is relatively small and statistically significant but shows no statistically significant change across fines. Finally, if the jurisdiction is located in a large city, the conditional impact is large, ranging from 10 to 30 additional citations per month, declining as fines increase. Note from these figures that most estimates for fines under 9 show large confidence intervals because less than 10% of the data fall below this threshold.

5 Conclusions

SVCMs are an alternative to full nonparametric models that can be used to analyze relationships between dependent and independent variables under the assumption that those relationships are linear conditional on a smaller set of explanatory variables. They are less affected by the curse of dimensionality problem because fewer variables enter the estimation nonparametrically. In this article, I reviewed the model selection, estimation, and testing for these types of models and introduced a set of commands, `vc_pack`, that aims to facilitate the estimation of such models when coefficients are assumed to vary with respect to a single smoothing variable. An empirical application illustrated the usefulness of the procedure.

6 Programs and supplemental materials

To install a snapshot of the corresponding software files as they existed at the time of publication of this article, type

```
. net sj 20-3
. net install st0613     (to install program files, if available)
. net get st0613         (to install ancillary files, if available)
```

7 References

Cai, Z., M. Das, H. Xiong, and X. Wu. 2006. Functional coefficient instrumental variables models. *Journal of Econometrics* 133: 207–241. https://doi.org/10.1016/j.jeconom.2005.03.014.

Cai, Z., J. Fan, and Q. Yao. 2000. Functional-coefficient regression models for nonlinear time series. *Journal of the American Statistical Association* 95: 941–956. https://doi.org/10.1080/01621459.2000.10474284.

Cameron, A. C., and P. K. Trivedi. 2005. *Microeconometrics: Methods and Applications*. New York: Cambridge University Press.

Cattaneo, M. D., and M. Jansson. 2018. Kernel-based semiparametric estimators: Small bandwidth asymptotics and bootstrap consistency. *Econometrica* 86: 955–995. https://doi.org/10.3982/ECTA12701.

Centorrino, S., and J. S. Racine. 2017. Semiparametric varying coefficient models with endogenous covariates. *Annals of Economics and Statistics* 128: 261–295. https://doi.org/10.15609/annaeconstat2009.128.0261.

Delgado, M. S., D. Ozabaci, Y. Sun, and S. C. Kumbhakar. 2020. Smooth coefficient models with endogenous environmental variables. *Econometric Reviews* 39: 158–180. https://doi.org/10.1080/07474938.2018.1552413.

Hainmueller, J., J. Mummolo, and Y. Xu. 2019. How much should we trust estimates from multiplicative interaction models? Simple tools to improve empirical practice. *Political Analysis* 27: 163–192. https://doi.org/10.1017/pan.2018.46.

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

———. 1993. Varying-coefficient models (with discussion). *Journal of the Royal Statistical Society, Series B* 55: 757–796.

Henderson, D. J., and C. F. Parmeter. 2015. *Applied Nonparametric Econometrics*. Cambridge: Cambridge University Press.

Hirano, K., and G. W. Imbens. 2004. The propensity score with continuous treatments. In *Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives*, ed. A. Gelman and X.-L. Meng, 73–84. Chichester, UK: Wiley. https://doi.org/10.1002/0470090456.ch7.

Hoover, D. R., J. A. Rice, C. O. Wu, and L.-P. Yang. 1998. Nonparametric smoothing estimates of time-varying coefficient models with longitudinal data. *Biometrika* 85: 809–822. https://doi.org/10.1093/biomet/85.4.809.

Hoti, F., and L. Holmström. 2003. On the estimation error in binned local linear regression. *Journal of Nonparametric Statistics* 15: 625–642. https://doi.org/10.1080/10485250310001605469.

Ichimura, H., and P. E. Todd. 2007. Implementing nonparametric and semiparametric estimators. In *Handbook of Econometrics, vol. 6B*, ed. J. J. Heckman and E. E. Leamer, 5369–5468. Amsterdam: Elsevier. https://doi.org/10.1016/S1573-4412(07)06074-6.

Li, Q., C. J. Huang, D. Li, and T.-T. Fu. 2002. Semiparametric smooth coefficient models. *Journal of Business & Economic Statistics* 20: 412–422. https://doi.org/10.1198/073500102288618531.

Li, Q., and J. S. Racine. 2007. *Nonparametric Econometrics: Theory and Practice*. Princeton, NJ: Princeton University Press.

———. 2010. Smooth varying-coefficient estimation and inference for qualitative and quantitative data. *Econometric Theory* 26: 1607–1637. https://doi.org/10.1017/S0266466609990739.

Liu, W., and K. J. Egan. 2019. A semiparametric smooth coefficient estimator for recreation demand. *Environmental and Resource Economics* 74: 1163–1187. https://doi.org/10.1007/s10640-019-00362-7.

Long, J. S., and L. H. Ervin. 2000. Using heteroscedasticity consistent standard errors in the linear regression model. *American Statistician* 54: 217–224. https://doi.org/10.2307/2685594.

Polemis, M. L., and T. Stengos. 2015. Does market structure affect labour productivity and wages? Evidence from a smooth coefficient semiparametric panel model. *Economics Letters* 137: 182–186. https://doi.org/10.1016/j.econlet.2015.11.004.

Rios-Avila, F. 2019. A semi-parametric approach to the Oaxaca–Blinder decomposition with continuous group variable and self-selection. *Econometrics* 7: 28. https://doi.org/10.3390/econometrics7020028.

Seber, G. A. F., and A. J. Lee. 2003. *Linear Regression Analysis*. 2nd ed. New York: Wiley.

Stinchcombe, M. B., and D. M. Drukker. 2013. Regression efficacy and the curse of dimensionality. In *Recent Advances and Future Directions in Causality, Prediction, and Specification Analysis: Essays in Honor of Halbert L. White Jr.*, ed. X. Chen and N. R. Swanson, 527–549. New York: Springer. https://doi.org/10.1007/978-1-4614-1653-1_20.

Verardi, V. 2013. Semiparametric regression in Stata. UK Stata Users Group meeting proceedings. https://www.stata.com/meeting/uk13/abstracts/materials/uk13_verardi.pdf.

Zhang, W., and S.-Y. Lee. 2000. Variable bandwidth selection in varying-coefficient models. *Journal of Multivariate Analysis* 74: 116–134. https://doi.org/10.1006/jmva.1999.1883.

About the author

Fernando Rios-Avila is a research scholar at Levy Economics Institute of Bard College under the Distribution of Income and Wealth program. His research interests include applied econometrics, labor economics, and poverty and inequality.

A Appendix: Kernel functions and standardized kernel weights

For the following definitions, $u = (Z_i - z)/h$, where Z_i is the evaluated point, z is the point of reference, and h is the bandwidth.

kernel	Kernel function	Standardized kernel weight
`gaussian`	$k(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}$	$k_w(u) = e^{-\frac{1}{2}u^2}$
`epan`	$k(u) = \frac{3}{4\sqrt{5}} \left(1 - \frac{u^2}{5}\right)$ if $\lvert u\rvert \le \sqrt{5}$	$k_w(u) = 1 - \frac{u^2}{5}$ if $\lvert u\rvert \le \sqrt{5}$
`epan2`	$k(u) = \frac{3}{4}(1 - u^2)$ if $\lvert u\rvert \le 1$	$k_w(u) = 1 - u^2$ if $\lvert u\rvert \le 1$
`biweight`	$k(u) = \frac{15}{16}(1 - u^2)^2$ if $\lvert u\rvert \le 1$	$k_w(u) = (1 - u^2)^2$ if $\lvert u\rvert \le 1$
`cosine`	$k(u) = \{1 + \cos(2\pi u)\}$ if $\lvert u\rvert \le 0.5$	$k_w(u) = \frac{1}{2}\{1 + \cos(2\pi u)\}$ if $\lvert u\rvert \le 0.5$
`parzen`	$k(u) = \begin{cases} \frac{4}{3} - 8 \times u^2 + 8 \times \lvert u\rvert \text{ if } \lvert u\rvert \le 0.5 \\ \frac{8}{3}(1 - \lvert u\rvert)^3 \text{ if } 0.5 \le \lvert u\rvert \le 1 \end{cases}$	$k_w(u) = \begin{cases} 1 - 6u^2 + 6 \times \lvert u\rvert^3 \text{ if } \lvert u\rvert \le 0.5 \\ 2(1 - \lvert u\rvert)^3 \text{ if } 0.5 \le \lvert u\rvert \le 1 \end{cases}$
`rectan`	$k(u) = \frac{1}{2}$ if $\lvert u\rvert \le 1$	$k_w(u) = 1$ if $\lvert u\rvert \le 1$
`trian`	$k(u) = 1 - \lvert u\rvert$ if $\lvert u\rvert \le 1$	$k_w(u) = 1 - \lvert u\rvert$ if $\lvert u\rvert \le 1$

NOTE: The option `kernel()` should be used as stated in this table when using all `vc_pack` commands.