# A Teachware Module to Instruct Statistics and Econometrics in XploRe 3.1

Isabel Proença[†]
Institut de Statistique
and
Center for Operations Research and Econometrics
Université Catholique de Louvain
Belgium

April 12, 1995

**Abstract**

The use of computer as a tool for teaching and learning is becoming widely accepted in all fields of science. In this paper an interactive system designed to instruct basic statistics and econometrics is presented. This system is written in the macro language of XploRe and is based in a highly interactive graphic interface. The main structure of the system and some details about the programing language are refered as an example to inspire teachers how to develop further macros.

[1] Institut de Statistique, Voie du Roman Pays 34, 1348 Louvain-la-Neuve, Belgium. This work was developed within a project financed by the FDS (Fonds de Developpement Scientifique) and supervised by Professors W. Härdle and L. Simar.

# 1 Introduction

The progress of computer science associated with the progress of computer hardware are making computers a yet more accessible and important tool in assisting individuals to process information and computation. This fact enhanced the use of computers for educational and teaching purposes. That is, the current sophisticated potential of computers can be used in order to provide a way to acquire concepts and to evolve reasoning and problem solving-skills. Computer aided learning (CAL) and computer aided instruction (CAI) became established fields in computer science. CAI and CAL mean the use of computer as a tool for learning and presenting instruction that is individualized, interactive and guided.

In this paper software (teachware or courseware) will be presented that uses the modern computer capabilities with the purpose of improving the quality of learning and instruction of statistics. In the conception of this teachware the following basic ideas were taken into consideration.

1. The learning process should be interactive, exploratory and directed by the student (and not by the program).

2. The system should be fully menu driven and easily usable by students without the need of supervision. In another words, it must be simple and very user-friendly designed to be used essentially by persons who may have a very few knowledge about programming or computer handling.

3. Because some of the statistic and econometric techniques are highly graphical in nature a highly interactive graphical interface routine is needed to give good graphical representations of the procedures performance.

Interactive CAL systems have appreciable benefits. They save time. They provide fast manipulation of statistical procedures and give immediate feedback. They reduce computational effort and therefore allow students to concentrate on concepts and understanding principles. They make learning more attractive and pleasant and therefore can better motivate the student attentiveness.

# 2 The general structure of the teachware system

This section aims to describe the main characteristics of this teachware implementation like the computer environment, the users interface and the teaching/learning setting.

## 2.1 The computer environment

Softwares can be broadly divided in two categories: closed softwares and open softwares. Closed softwares are systems where the user can not modify or manipulate procedures according to his will. He has to follow the options offered by the system in an automatic and direct way. In open softwares the user has to learn a macro-language and build himself the procedures necessary to achieve the aimed results. Usually teachwares are closed softwares. The system proposed in this report aim to be more flexible than the so called closed softwares and simultaneously keep their user–friendly properties. This means, it should allow the teacher to manipulate the procedures and tasks in order to suit particular teaching needs and it should be direct and very easy–to–use for students. For these reasons the system was written using a macro language. It was also required that this teachware should run in microcomputers IBM compatible.

The system is written in the macro–language of XploRe 3.1. The reasons of this choice lie mainly on the high interactive capabilities and useful graphic interface properties of XploRe. XploRe is a high performance computing environment special designed for statistical/econometrical problems. It is entirely written in C language and is a complete integrated system including graphics, programmable macros, an interpreter and many analytical commands forming libraries. These libraries are exhaustive and make some tasks very easy and fast to implement namely tasks related with smoothing techniques. It uses the matrix as basic data element which does not require dimensioning. This allows many numerical problems to be solved easier using XploRe than using a language like Pascal, Fortran or C instead. The results can be easy displayed as graphics with good-quality output to plotters and printers.

The teachware presented here is a module constituted by a set of XploRe macros. Each teaching subject is attached to a particular macro. However, to perform some specific tasks other macros have to be called within these main macros. Nevertheless, those will be sort of "hidden" to the user.

## 2.2 The users interface

The teachware system under analysis is highly interactive. The computer communicates with the user via a system of tree–structured menus. Users communicate by pressing a key that identifies their option. Menus describe the tasks and actions that can be pursued. Each action is identified by a specific key on the computer keyboard. The user has an active role on the process of performing a task and getting a result. This seems to be more attractive for the student and more effective on learning than simply having the solution (and the particular action that originates it) predetermined by the program. The aim is to allow the student to carry out iteratively a variety of actions and see the respective consequence of those until he reaches the "best" solution. Criteria are shown in order to evaluate the actual solution and to guide the student to achieve the "best" answer. The results of an action are shown graphically and numerically.

## 2.3 The teaching/learning setting

The target population of this teachware system are undergraduate students of statistic and/or econometric courses. These students do not need to know XploRe language nor to have special knowledge about computers to be able to use the teachware module. However they need a pre–instruction on the subjects which the module is concerned with. This prior instruction should be given by the teacher in the lectures or by reading the appropriate course textbooks.

The teachware module can provide a rich interaction between the teacher and the student. It can be used in the classroom by the teacher (using the appropriate technology for overall projection of the computer screen) being a more efficient substitute of the blackboard. More efficient in the sense that not only the computations are much more easy and faster but also it avoids the sometimes painful attempts of the teacher to draw graphics using the chalk. The teachware module can be used as well by the student alone whether at home (in order to get self-instruction) or in the exercises classes under the tutoring of the teacher.

# 3 Description of the main macros in the module

When the implementation of a teachware system for statistics and econometrics was planned it was decided to focused initially on basic subjects of this fields traditional taught to undergraduate students. The list of these subjects is given below.

1. Reasoning of the OLS rule.
2. Impact of influent observations on the OLS rule.
3. Principal component analysis.
4. Density estimation.
5. Nonparametric regression.

The reasons of this choice are identifiable in the following. The OLS rule is the most popular technique used in basic econometrics mainly because of its simplicity and widely use. OLS is the first rule learned by students how to fit a model to describe a data set. Outliers in data can be a serious problem interfering in the quality of a fit. They may arrive often when dealing with real data. It is important to know how utilitarian methods like the OLS react (concerning robustness) when outliers are present. Principal component analysis is a useful technique in a basic approach for dimension reduction with many applications. Density estimation is considerably requested in applied statistics. Here we are concerned essentially with the kernel method given its well known superior performance related to other methods like the histogram. Nowadays computer packages can give automatically a kernel density estimate having the user only to choose the bandwidth value (and sometimes also the kernel). The choice of bandwidth is the most interesting problem related with kernel estimation in the sense that the shape of the estimate is influenced by the bandwidth (and is practically the same for all kernels). It is important for the student to know the meaning and consequences of undersmoothing and oversmoothing the data. Nonparametric regression has recently become more usable due to the increasing speed and facilities provided by modern computers. It is a valuable tool for an exploratory analysis of data in order to guide (or even to validate) the choice of a parametric model.

The macros that constitute the teachware presented here are stored in the XploRe library named `TEACHWARE`. To get access to this tutoring system the user has to start XploRe. In XploRe command line he has to call the library `TEACHWARE` and the macro `TWARE` by typing:

```
library(TEACHWARE)
TWARE(0)
```

A menu will appear describing all the tasks that can be performed. Each task corresponds to a certain macro. To perform a task the user has only to put the cursor in the item with the task name and press return. Automatically the respective macro starts to run. In some situations another menu may appear showing several options related to this task. An option is selected in the same way as before. In general the screen is divided in several windows. One window shows a graphical display of the object(s) related to the task representing an initial situation. After analyzing this situation the user can take an action to produce a desired modification. The specific action is taken by pressing a matching key. In another window a help menu which presents all the possible actions that can be pursued and indicates the key necessary to press in order to do it is shown. In the last windows some information will be provided concerning numerical results or other graphical representations that describe the state of the actual situation or show criteria to evaluate it. When the user carries out an action by pressing the matching key automatically all windows will be updated showing the situation resulting from that action. The user can end the task (exit the macro) whenever he wants. In almost all the tasks the user can manipulate the input objects by manipulating the parameters of the simulated data or by introducing a data set of his choice. The module includes the macros named respectively `TWARE`, `TWLINREG`, `TWINFOBS`, `TWPCOMPO`, `TWDENES1`, `TWDENES2`, `TWREGSMO` and `TWREGEST` corresponding each one respectively to the subjects (tasks) mentioned before. TWARE gives the interactive initial menu and calls the macro corresponding to the task selected by the user. The other macros referred need a more detailed analysis. This analysis will be presented below.

## 3.1 Linear regression — `TWLINREG`

OLS gives a simple tool to summarize a given data set by a linear model. The students learn easily how to compute the coefficient estimates of the regression

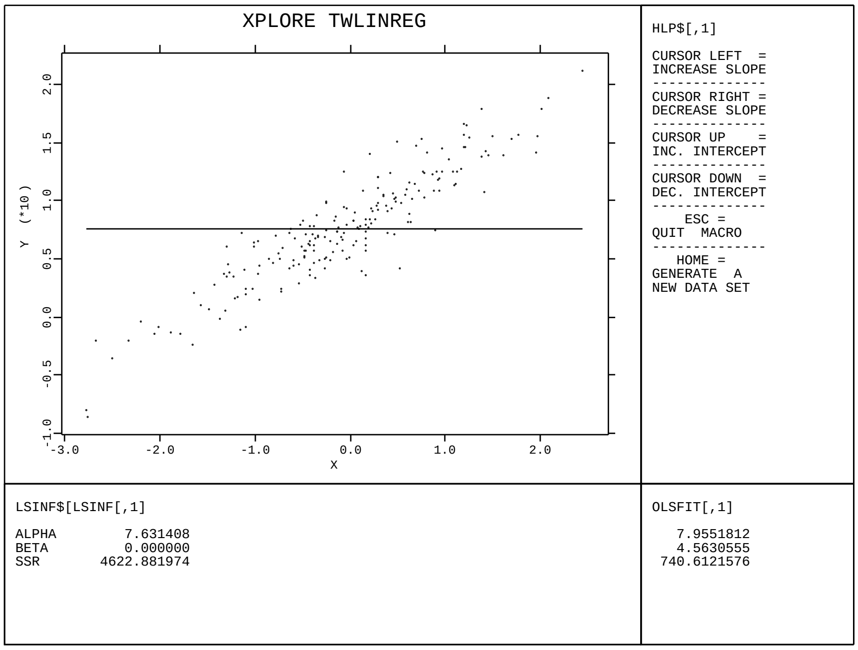


Figure 1: The initial plot of TWLINREG. In the upper left window a data set is displayed together with a line that gives a first fit of the data. In the window below the value of the parameters of this line are shown together with the SSR of the corresponding fit. The smaller window gives the OLS results while the upper right window shows an help menu.

line but they should also understand the performance in practice of this technique. That is, how the fitted line is determined for a given data set. The aim is to show that the data is projected into the direction that minimizes the $L_2$ distance from the observations, the sum of squared residuals (SSR). To illustrate this purpose students are confronted with a "good" data set (in the sense that it is adequate to perform on OLS) and a line (direction) that projects the data and represents a linear fit. The data is generated from the linear model with random error $Y = \alpha + \beta X + \sigma U$ verifying the classic conditions. Students can move the line in any direction and visually measure the distance between the line and the data. They can also see numerically the importance of this distance evaluated by the SSR. Moreover, they can perceive that any other linear fit besides the OLS originates bigger residuals or in other words is more apart from the bulk of the data. The student can manipulate the value of the parameters in the model in order to generate different data sets. This can be specially interesting to show how the accuracy of the OLS estimates depends on the variance of the error term.

When the macro starts to run a graphic with the simulated data set and an initial fit is shown together with the parameter values of this actual fit and the respective SSR, as it is shown in figure 1. The student should move the line (according to the actions shown in the help menu) and simultaneously check the changes in SSR. For instances, the student can move the line from the initial position to the position shown in figure 2 (by pressing successively cursor left) making an improving in the SSR. However OLS results (presented in the smaller window) are still better.

The actions that can be performed are:

- increase/decrease the slope of the line in the screen;
- increase/decrease the intercept of the line in the screen;
- enter new values for the parameters $\alpha, \beta, \sigma$;
- quit the macro and return to initial menu.

## 3.2 Influent observations — `TWINFOBS`

OLS is a useful rule but it does not perform equally good for all data sets. A common problem when analyzing real data is the presence of outliers (as it was already referred). Outliers are observations that assume clearly a distinct value compared to the rest of the data. Outliers can arrive in the dependent variable or in the explanatory variables with different consequences in estimation. Outliers in the explanatories can be quickly identified by a high influence index measured by the elements in the diagonal of the "hat" matrix, the matrix $X(X^TX)^{-1}X^T$. In most of the cases queer values in the observations are due to measurement errors. In this circumstance it is adequate to use an estimation method that will restrict the influention of this observations on the resulting estimate and reflect more the

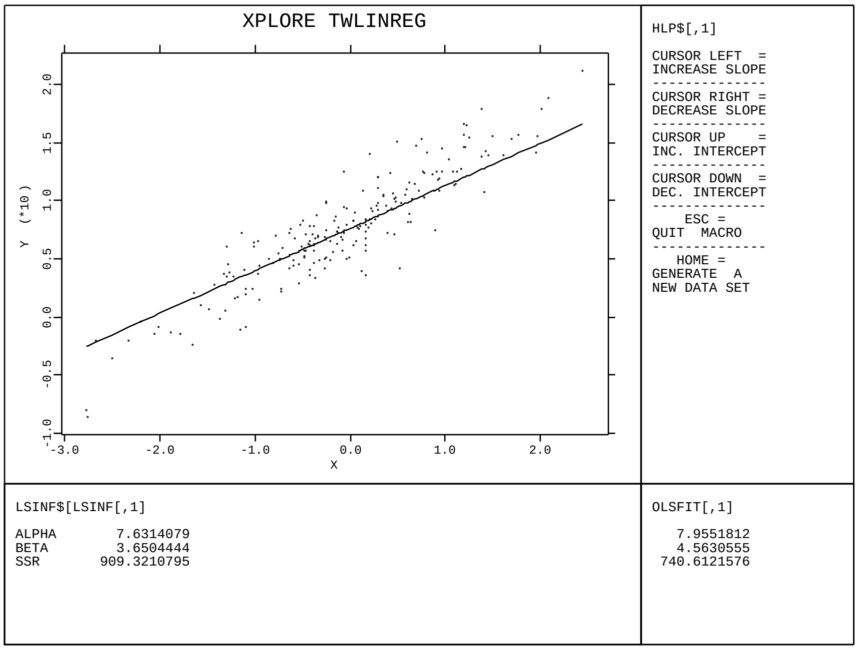


Figure 2: The result obtained after moving the line displayed before by increasing the slope. The SSR has diminished but OLS results are still better.

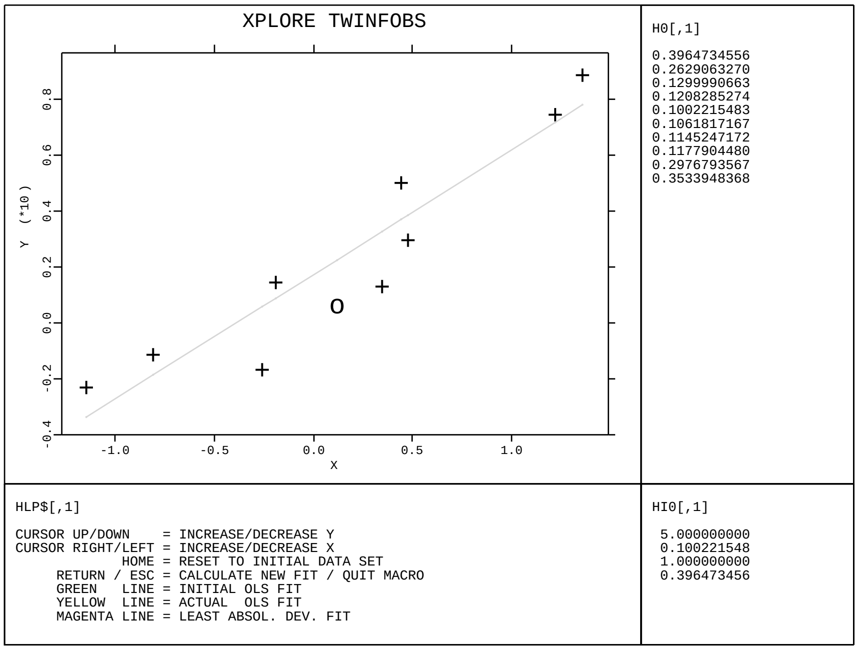


Figure 3: The initial plot of `TWINFOBS`. The upper left window shows a data set together with the OLS fit. The fifth observation is identified by a circle and can be manipulated in order to simulate an outlier. In the upper right window the diagonal of the "hat" matrix is displayed while in the smaller window the influence value of the fifth observation and the most influent observation are displayed together with the observation number. In the lower left a help menu is shown.

behavior of the bulk of the data, that is, a robust method. OLS is not a robust method to outliers and students should be aware of that.

The aim of TWINFOBS is to show how sensitive the OLS rule is to changes in the observations value, whether the change is in the explanatory or in the dependent variable. This set–up is useful to show how OLS behaves in presence of outliers in the data. The data is simulated from a classic linear model with random errors and is displayed together with the OLS fit as can be seen in figure 3 In the upper right window the elements of the diagonal of the "hat" matrix are also shown while the smaller window contains the maximal value of this elements and the value corresponding to fifth observation (the one represented by a circle).

The user has the possibility of modifying the position of one observation (the one represented by the circle in figure 3) by changing the value of $X$ or $Y$ within the graphic display (as far as he wants) and see how the OLS fit is changed. The

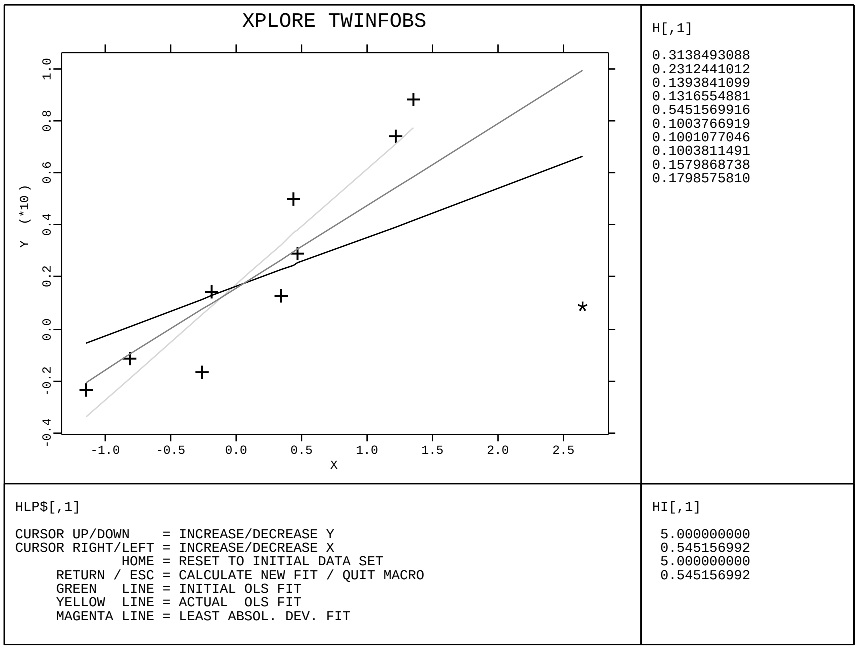


Figure 4: After moving the fifth observation (now represented by a star) a situation was reached where the OLS of the manipulated data (black line) has moved from the bulk of the data in direction to the outlier. The original OLS fit (dashed shadowed line) is also shown together with the L1-line (grey line). Here the L1–line is more robust than the OLS.

L1–line fit (resulting from minimizing the sum of the absolute value of residuals) is also shown as a more robust alternative to OLS. However, this fit is not completely robust to outliers in $X$. Figure 3 shows one situation where the L1–line is more robust than the OLS in presence of a outlier in the explanatory variable. Note that after been manipulated the fifth observation is now represented by a star.

The possible actions that can be undertaken are:

- increase/decrease the value of the observation in X;
- increase/decrease the value of the observation in Y;
- calculate the new fit for the modified data set;
- reset to initial data set;
- quit macro and return to initial menu.

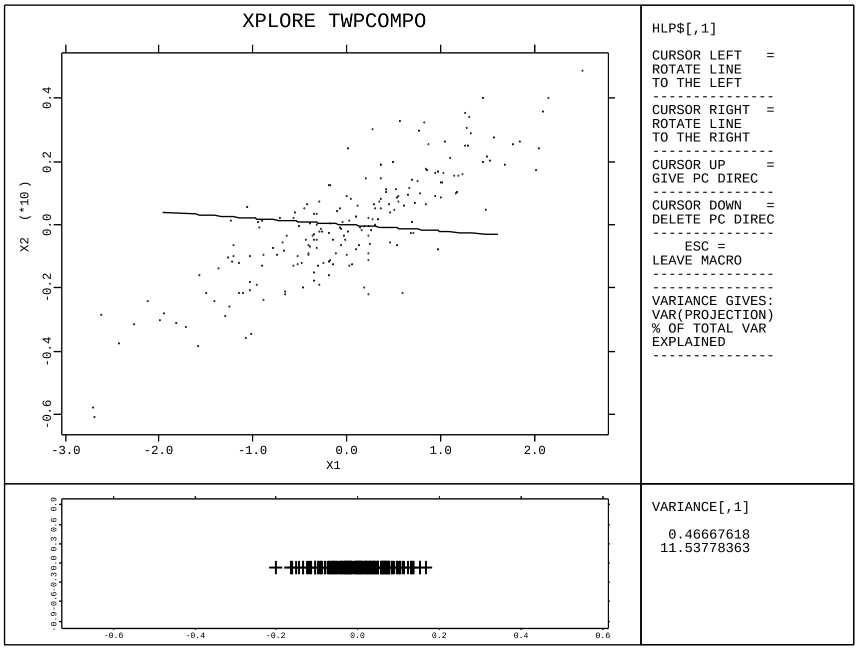


Figure 5: The initial plot of `TWPCOMPO`. In the upper left window a data set is displayed together with a line representing a direction to project the data. This projection is shown below. The smaller window displays the variance of the projection and the percentage of total variance explained while in the upper right a help menu is shown.

### 3.3 Principal component — `TWPCOMPO`

The aim of principal component analysis is to find appropriate linear combinations of the variables in the data (projections) that best summarize the general behavior of the data. The interest of this technique lies on the reduction of the dimension of the problem on substituting the original variables by a smaller number of projections, new variables, achieving mainly the same descriptive power of the original data. The principal components give the best projections in the sense that they incorporate better the variability of the original data. Consequently, these projections have maximal variance (among all linear combinations of the original variables where the square of the weights sum to one).

On showing the performance of principal component analysis graphically we are severely restricted on the number of variables to use. XploRe provides two-dimensional and three-dimensional plots. Three-dimensional plots are originated by projecting the third variable in a plan generated by the other two. This plan can be rotated interactively (generating new projections) allowing to see the data set from several angles. In the framework of this teachware module it was shown to be most convenient to use two-dimensional plots. Even if in practice the principal component is used for problems with higher dimension an example with two variables can, nevertheless, well illustrate graphically the performance of the method.

The data used in `TWPCOMPO` is simulated from a bivariate normal distribution. The task of this macro consists on projecting the data in one direction which is given by a linear combination of the two variables (where the sum of the weights sums to one). In figure 5 the data set and an initial direction are shown in the upper left window. In the window below the data projected in this direction is also displayed. The smaller window gives the variance of this projection and the percentage of the total variance explained.

The student can manipulate this direction in order to find the best projection of the data by rotating iteratively the line in the upper graphic and see the changes on the projected data on the window below. One situation that can be achieved is given in figure 6. Here, the principal component is also given corresponding to the shadowed line. The smaller window shows that the principal component explains 93.28% of the total variance. The other direction in the graph (the black line) gives a projection of the data with less variance as can be seen in the lower left window but better than the initial projection.

After some trials the student will see that there is no other direction beside the principal component that gives a projection with higher variance.

The possible actions are:

- rotate line to the right/left;
- give principal component results;
- delete principal component;
- quit macro and return to initial menu.

### 3.4 Density estimation — `TWDENES1` and `TWDENES2`

These macros perform density estimation of a data set. As it was already mentioned they are focused on the kernel method. The kernel density estimator of a univariate data set $X_i$, $i = 1, \ldots, n$ is defined by

$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right),$$

where $K(\bullet)$ is the kernel and $h$ is the bandwidth. The kernel has to be a function symmetric around zero with integral equal to one. Härdle (1991) gives a good survey of kernel density estimation. To obtain a kernel estimate the kernel function

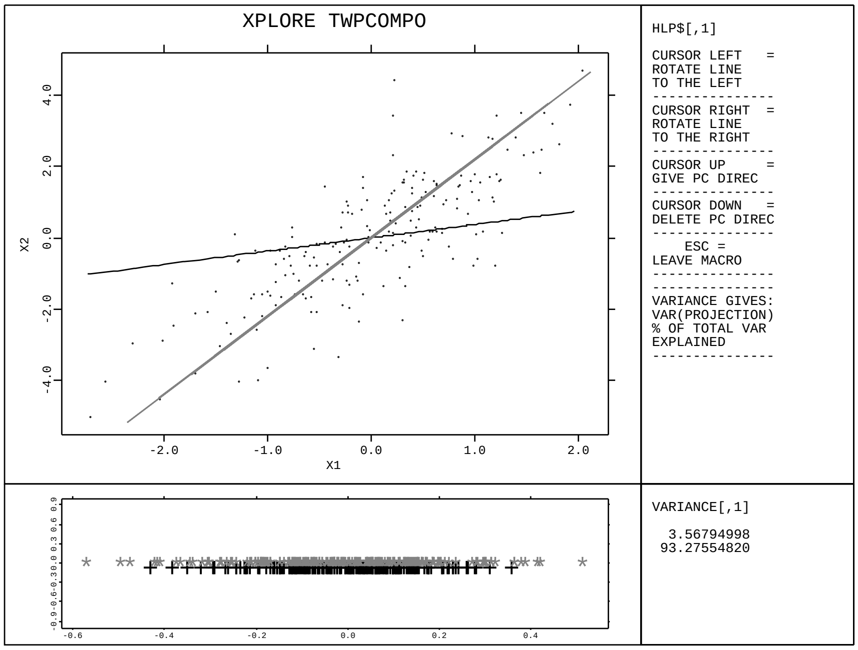


Figure 6: The upper left window shows a direction (black line) resulting from moving the initial line. Below the data projected in this direction is shown with crosses. Easily can be seen that this projection has more spread than the initial in figure 5. The principal component is also plotted in the upper left (shadowed line). The data projected in the principal component is shown with stars in the lower left. This projection has the biggest variance. The smaller window shows the results concerning the principal component.

and the bandwidth value have to be chosen. It is proved that the particular kernel used has not much effect on the shape of the density estimate (Härdle 1991, p. 78). So that, the choice of the kernel is not important. One of the most used kernels in practice is the standard normal. On the contrary, the bandwidth value does affect the resulting estimate in means of its bias and its variance. The bias of the kernel estimate increases with $h$ while its variance decreases with $h$. Bandwidth choice is a crucial problem on kernel density estimation which is not simpler given that it is necessary to well managed the trade-off between the bias and the variance.

A "good" bandwidth can be defined to be the one that minimizes the mean integrated square error (MISE) of the kernel estimate. However to calculate the MISE the true density of the data should be known. In real applications this is not the case and other criteria should be used. One popular criterion indicates to choose the bandwidth that minimizes the least squares cross-validation (LSCV) function. For details about this method see Härdle (1991). On the other hand, the rule of thumb gives a quick way of choosing the bandwidth. This rule consists on using

$$\hat{h} \cong 1.06\hat{\sigma}n^{-1/5}.$$

There is also a more robust variant of the rule of thumb known as better rule of thumb which determines $h$ according to

$$\hat{h} \cong 1.06 min(\hat{\sigma}n^{-1/5}, \frac{\hat{R}}{1.34}).$$

In the formulas above $\hat{\sigma}$ is the sample estimate of the standard deviation of $X$ and $\hat{R}$ is the interquartile range of the data. These rules were deduced assuming that the true distribution of $X$ is gaussian. So that, whenever the data comes from a distribution which is not too far from the gaussian the rule of thumb (or the better rule of thumb) should provide a good value for $h$.

The kernel density estimates calculated in the this teachware system were obtained using a gaussian kernel and the technique of binning (considering 250 bins) to speed up calculations. The binning technique is explained in detail in Härdle (1991). The criteria considered to guide in bandwidth choice were the minimum of MISE for simulated data and the minimum of LSCV for real data. The macros work as follows. `TWDENES1` reads by default the data which is stored in the XploRe data directory in the file "chron.dat". However the macro may read any other data set indicated by the user (provided that the file specification where this data is stored is correctly given to the program). `TWDENES2` simulates a data set from a density function. The user can choose this density from three different options.The options are: a symmetric density (which is a standard normal); a bimodal density (which is a mixture of normals) and an asymmetric density (also a mixture of normals). The sample size is also defined by the user.

After the data has been read or simulated (depending whether the macro is) a first kernel density estimate appears in the screen. This estimate is calculated for a bandwidth $h$ given by the empirical rule $h = 5d$ where $d$ is the bandwidth.

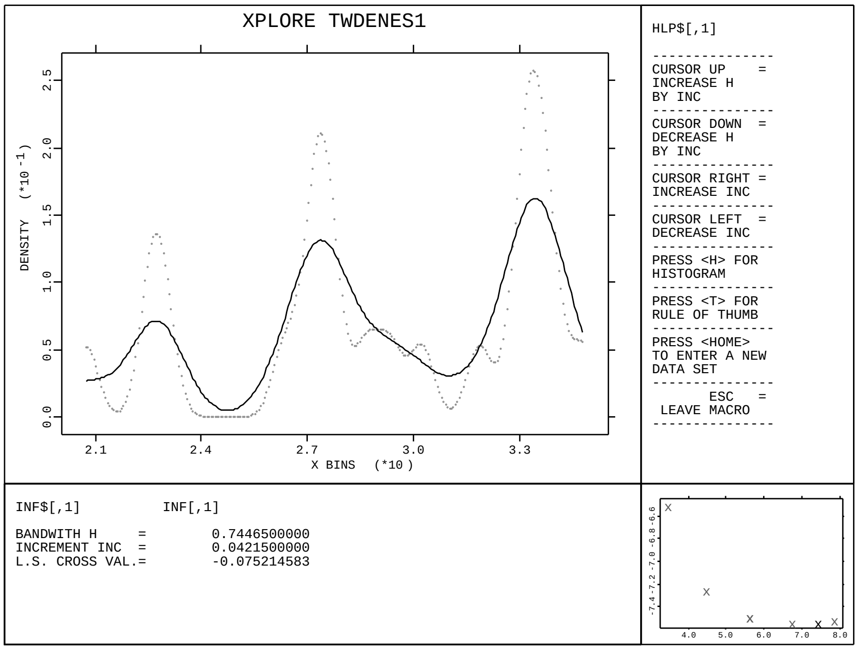


Figure 7: The upper left window shows a two kernel density estimates of the “chron” data for different bandwidths. The shadowed line with points is the initial estimate obtained for a small bandwidth while the black line gives the actual estimate which is smoother. Below, values for the bandwidth and LSCV concerning the actual estimate are shown together with the increment to change the bandwidth in order to replace the actual estimate. In the lower right the LSCV values for the different bandwidths tried before are shown. The actual estimate has the smallest value LSCV. The upper right shows an help menu.

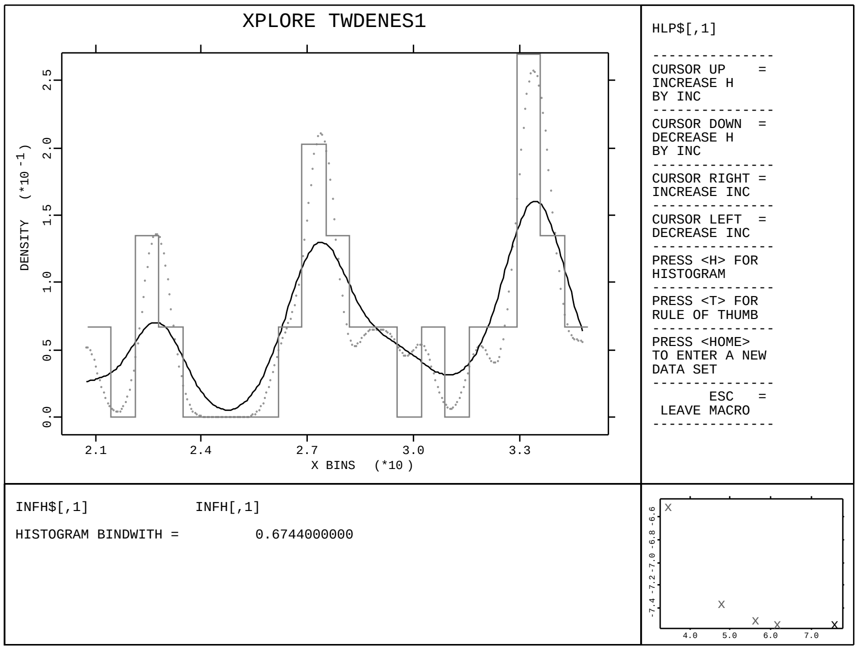


Figure 8: Here we have the same situation as in the previous figure with the addition of an histogram.

The user may change the bandwidth and see the effects of oversmoothing and undersmoothing the data and the behavior of the criterion function. In the lower left window a graphic of the criterion function for all bandwidths used until this moment is displayed. The point corresponding to the actual bandwidth is identified by a different color to easily help the user to locate the present criterion state. The actual bandwidth, the increment value to change it and the numerical value of the criterion function are also shown in another window. An histogram can also be obtained by pressing the key $h$. Figures 7 and 8 show a situation produced with `TWDENES1`. The points correspond to the initial density estimate of the data set and the black line to a smoother estimate which has the minimal MISE (for the bandwidths tried) as can be seen in the graphic in the smaller window. The second figure shows the same situation with the addition of an histogram.

The next figure was given by `TWDENES2`. The graphic shows the true simulated density which is asymmetric. The line in points is the density obtained with the better rule of thumb which is pretty close to the true density having a MISE very near to the minimum presented in the graphic in the lower right window. An oversmooth density estimate is given by the black line. The MISE of this estimate is represented by the point most in the right in the graphic in the in the smaller

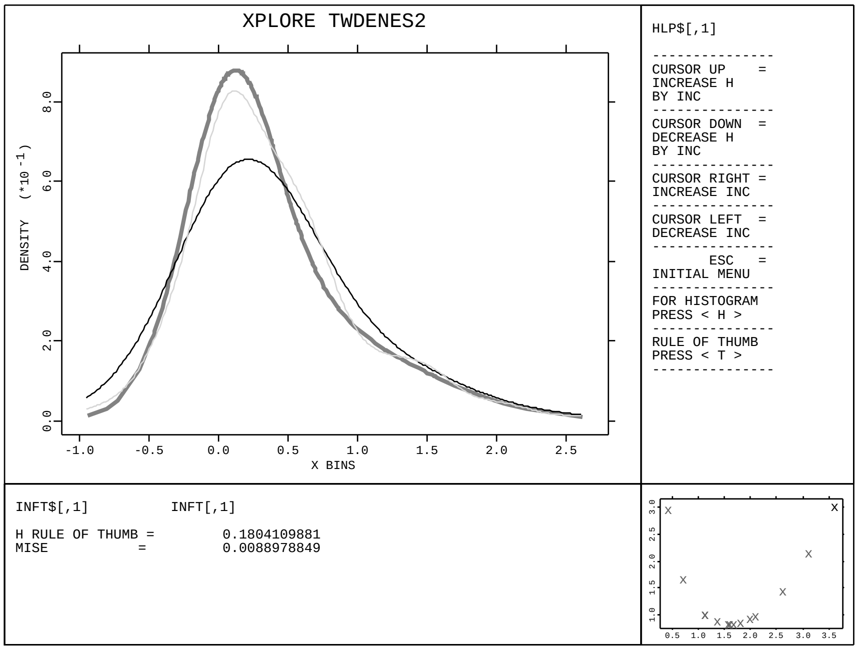


Figure 9: In the upper left window a simulated asymmetric density (shadowed thick line) is shown together with two kernel density estimates. One was obtained for the bandwidth given by the better rule of thumb (shadowed line in points) and the other for a bandwidth chosen by the user. The estimate for the better rule of thumb is pretty close to the true density having a very smaller MISE while the other estimate is oversmoothing the data. The plot of the MISE for the different bandwidths used before is shown in the lower right. The point most in the right gives the MISE for the actual estimate (the black line).

window.

The possible actions are:

- increase/decrease the bandwidth by an increment;
- increase/decrease the increment;
- calculate a histogram;
- calculate the kernel estimate with the *better rule of thumb*;
- read a new data set (for `TWDENES1`);

- return to last menu in order to choose the density from where to simulate the data (for `TWDENES2`);
- quit macro and return to initial menu.

### 3.5 Kernel regression — `TWREGEST`

Kernel estimation is a well accepted technique for nonparametric regression. Assume that a variable $Y$ is related to an exogenous variable $X$ by an unknown model. Suppose that the expectation of $Y$ conditional on $X$ verifies, $E[Y|X] = m(X)$ where $m(X)$ is an unknown function. An estimate of $m(X)$, the regression curve, can be determined nonparametrically using the Nadaraya-Watson estimator defined by

$$\hat{m}_h(x) = \frac{\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right) Y_i}{\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)},$$

where $K(\bullet)$ is the kernel. Details about kernel regression smoothing can be seen in Härdle (1990). Here, like in density estimation, the choice of the bandwidth is crucial (while the choice of the kernel plays no important role). Least squares cross–validation (LSCV) is a common method of bandwidth choice in kernel regression. The best bandwidth is the one that minimizes the LSCV function.

`TWREGEST` performs a kernel estimate of the regression of the one dimensional variable $X$ on $Y$. The main purpose is to show the flexibility of a kernel regression of a given data set and the effects of bandwidth choice on undersmoothing or oversmoothing. This macro evolves in the same way as `TWDENES1`. By default the data is read from the file "motcyc.dat" which is in the XploRe data directory. Any other data set can be read by the program provided that the user will input correctly the file specification where the data is stored. The estimates are calculated with binning using 250 bins. The initial bandwidth is calculated in the same way as in `TWDENES1`. For every bandwidth selected by the user the respective value in the LSCV function is given numerically and graphically. Figures 10 and 11 were produced by `TWREGEST`. They show the kernel regression estimate that minimizes the LSCV (for the bandwidths tried) given by the black line together with the data set. Figure 10 refers to the motcyc data and figure 11 to the geyser data. The shadowed line corresponds to a rougher kernel estimate.

The actions that can be pursued are:

- increase/decrease the bandwidth by an increment;
- increase/decrease the increment;
- read a new data set;
- quit macro and return to initial menu.

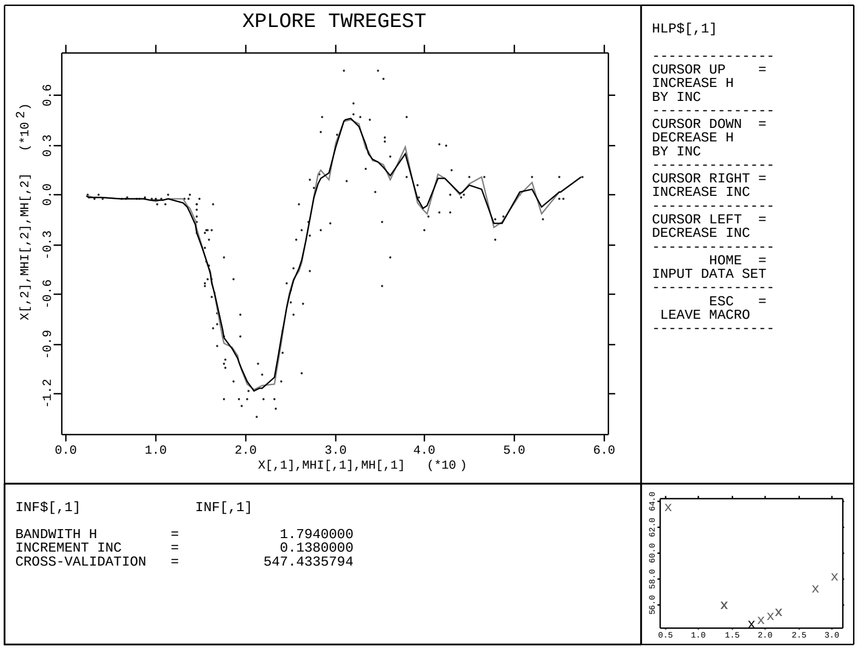


Figure 10: The plot in the upper left shows two kernel regression estimates for different bandwidths of the "motcyc" data. The initial estimate is the shadowed line with points while the smoother estimate (the black line) was obtained for a bandwidth chosen by the user. Below, values for the bandwidth and LSCV of the actual estimate are shown together with the actual increment to change the bandwidth in order to obtain another estimate. In the lower right a graphic with the LSCV for the bandwidths tried is displayed. The darker point with smaller LSCV value corresponds to the actual estimate. In the upper right an help menu is displayed

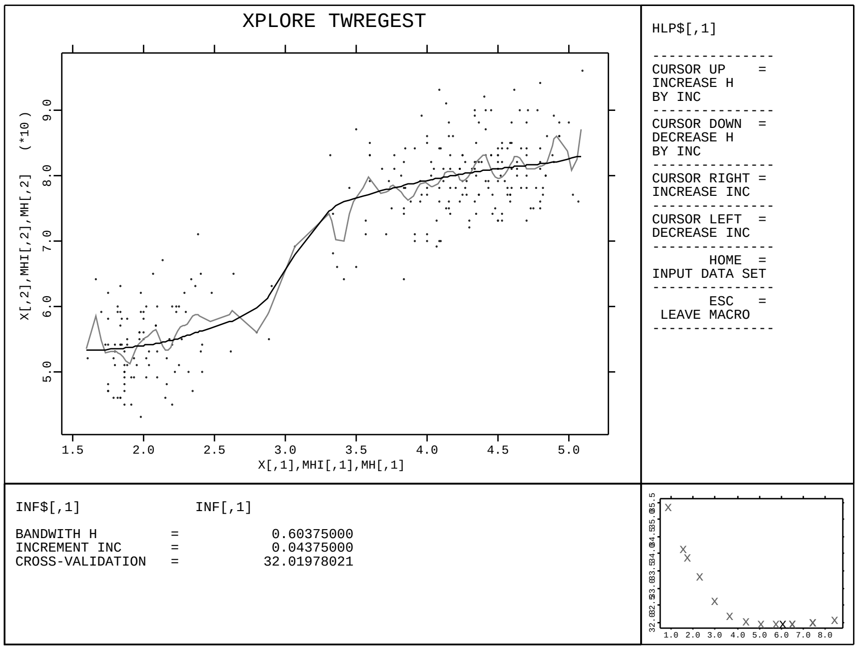


Figure 11: The plot in the upper left shows two kernel regression estimates for different bandwidths of the "geyser" data. The initial estimate is the shadowed line with points while the smoother estimate (the black line) was obtained for a bandwidth chosen by the user. Below, values for the bandwidth and LSCV of the actual estimate are shown together with the actual increment to change the bandwidth in order to obtain another estimate. In the lower right a graphic with the LSCV for the bandwidths tried is displayed. The darker point with smaller LSCV value corresponds to the actual estimate. In the upper right an help menu is displayed

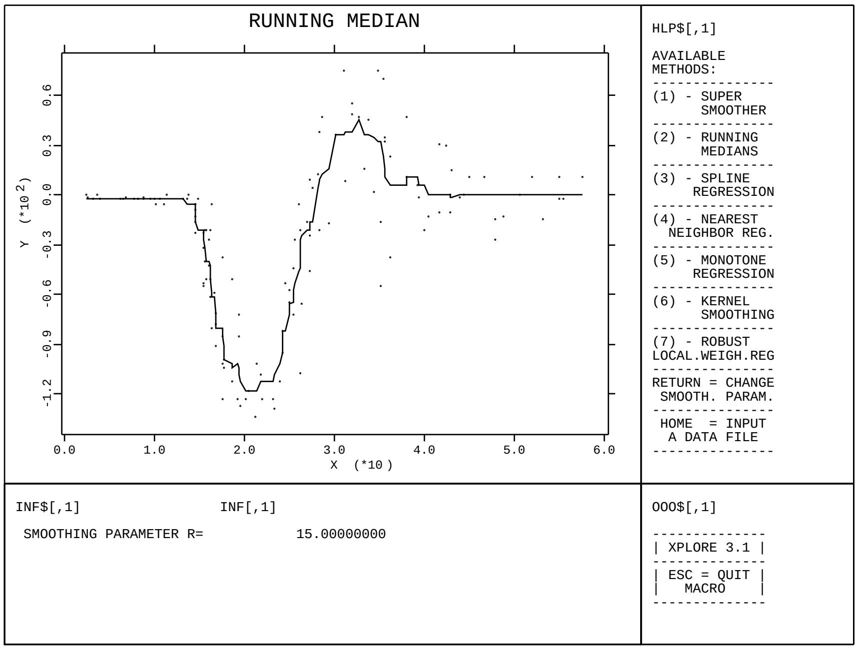


Figure 12: In the upper left window the "motcyc" data is plotted together with a running median fit. The smoothing parameter used in this fit is shown below. The upper right window shows an help menu.

### 3.6 Regression smoothers — `TWREGSMO`

There are much more methods of nonparametric regression beside the kernel regression by the Nadaraya-Watson estimator. The choice of one method depends mainly on the data in question and the purposes of the person who does the estimation. These methods may differ on robustness, smoothness, calculation complexity and speed, among others. Härdle (1990) makes a good survey of nonparametric regression methods.

The aim of this procedure is to show the performance of different regression smoothers for the same data set. The program reads the already mentioned "motcyc.dat" data and displays it on the screen. As before any other data set can be read. The user has to take an action indicating which method should produce the smoother fit. This fit will be then displayed on the same graphic where the data is. For the smoothing methods that depend on a bandwidth value the user has the possibility of changing this value. Figures 12 and 13 show respectively a running median and a spline fit.

The available actions are:

- calculate the supersmoother;

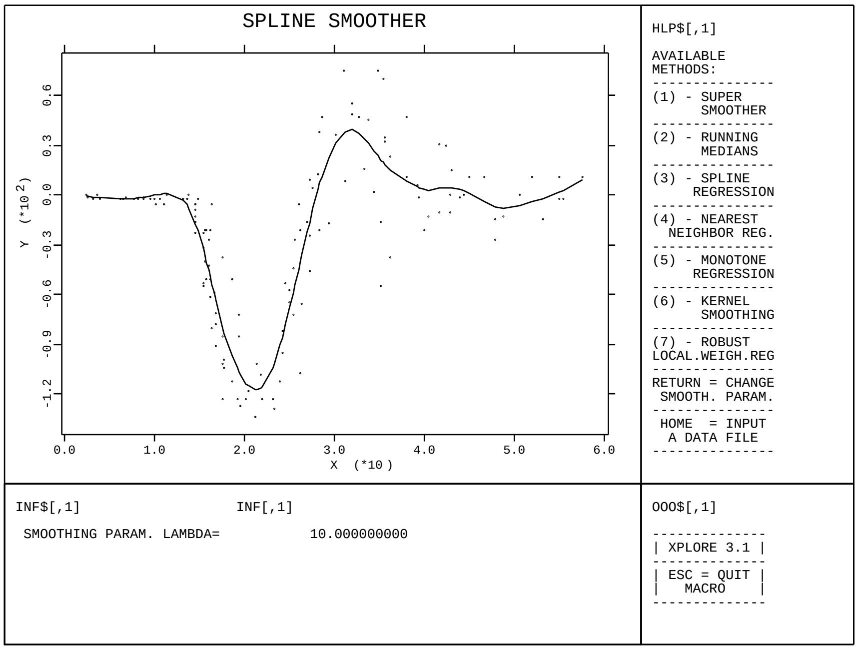


Figure 13: In the upper left window the "motcyc" data is plotted together with a spline fit. The smoothing parameter used in this fit is shown below. The upper right window shows an help menu.

- calculate a running medians smoother;
- calculate a spline regression smoother;
- calculate a nearest neighbor smoother;
- calculate the isotonic regression;
- calculate a kernel smoother;
- calculate a locally weighted regression smoother;
- input a new smoothing parameter;
- read a new data set;
- quit macro and return to initial menu.

### 3.7 Remarks

Before concluding this section a remark should be made. Procedures like `TWDENES1`, `TWREGSMO` and `TWREGEST` can also be used with purposes other than instructional. They can be also used as a quick and easy exploratory tool to analyze a given data set. For example, the user may use `TWREGSMO` to see how his data is fitted by different smoothers (for different bandwidths values) in order to know better his data set and to help to choose the method that better suits his purposes. With `TWDENES1` and `TWREGEST` he has a quick and easy way to see the effects of bandwidth selection in density or kernel regression evaluated by the LSCV rule.

## 4 Details on XploRe language

In this section some of the XploRe commands that have been used to write the teachware macros will be explained in some detail. The intention is to give some tools to those that do not know XploRe language to make them able either to built their own instruction macros or to modify those already existent in order to suit some designed purpose.

### 4.1 Creating windows and showing objects

One important feature of this teachware system is the need of showing simultaneously in the screen the display of more than one graphic together with numeric results and text strings like the help menu. In order to do it the computer screen has to be divided in several windows of different type.

The particular XploRe command for creating windows is **createdisplay**. To see a example how to work with createdisplay consider the following display given by the macro TWDENES2.

The four windows of the above display were generated with the command

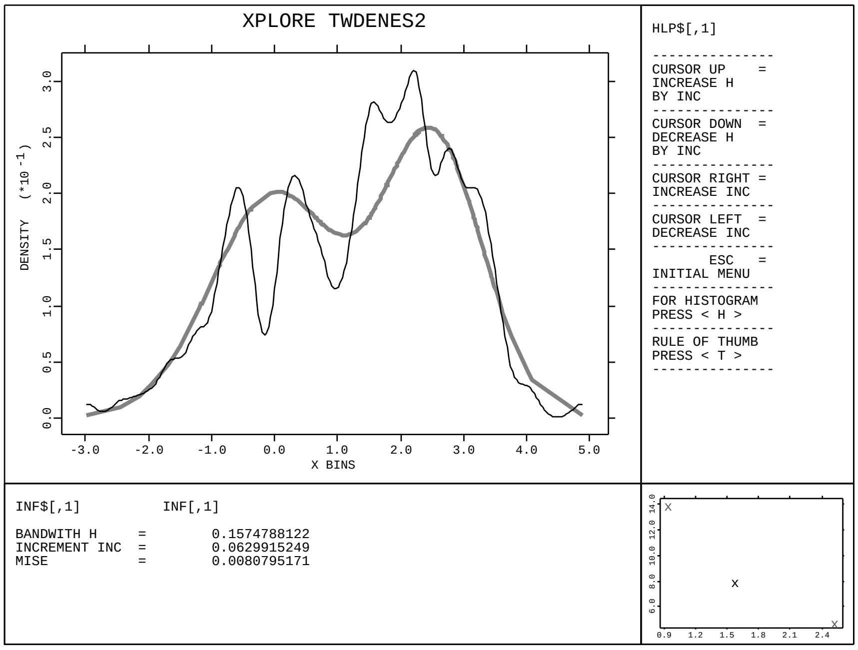


Figure 14: The initial display of `TWDENES2`. The screen was divided in four asymmetric windows using the XploRe command `createdisplay`.

```
createdisplay(displayname , (-2) (-2) , s2d text text s2d).
```

The parameter *displayname* gives the name of the display which identifies it as a XploRe object. The numbers in parenthesis refer to the number of respectively vertical and horizontal windows. Negative numbers create windows of asymmetric size. The last parameters indicate the type of the window. In this case the first and fourth windows show static two-dimensional graphics (s2d) while the second and third are designed to show matrices (float or string matrices).

In XploRe environment objects are displayed using the command `show`. As an example, to generate the display presented above from TWDENES2 the following *show* command was given (note that the *style* of the graphic was manipulated after the call of this command as it will be referred below).

```
show(tru fh s2d1, inf text1, hlp text2, misemsk s2d2)
```

The objects *tru* and *fh* are shown in the first window of type s2d. In the first window of type text the object *inf* (which is a matrix where the first column is a string vector and the second is a float vector) is displayed while in the second window of type text the string matrix *hlp* is exhibited. Finally the graphic in the lower right window was produced by showing the object *misemsk* in the second s2d type window.

## 4.2 Updating displays

A fundamental feature of the teachware system is the ability of replacing in a fast and easy way graphics and matrices that are displayed. That is, as soon as the user takes an action the display has to be fast updated showing the resulting situation. XploRe environment allows this procedure to be done easily using the command `update`.

### 4.2.1 Updating an object

Suppose that having the situation presented in the display shown above the user decides to increase the bandwidth in order to produce a smoother density estimate. After the action is communicated to the computer (by pressing the adequate keys) the display in figure 15 is given as a result.

The replacement of the old display by the new situation was produced with the commands

```
upd = update(fh 2 s2d1 solid line newproj)
upd = update(misemsk 1 s2d2 newproj)
upd = update(inf 1 text1).
```

The first `update` command replaces the new object *fh* (after been modified according to the action taken by the user) in the first s2d window. When the first `show` command was called XploRe retains the order with which each one of the objects were given. Returning to the example in figure 14 XploRe puts in the first

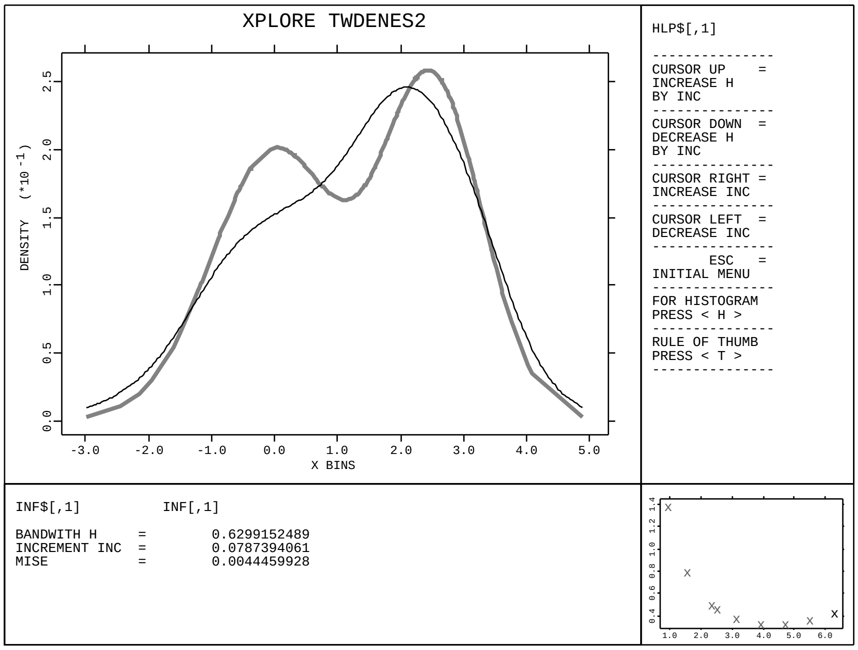


Figure 15: The black line in the upper left window was obtained by pressing cursor up. It has immediately replaced the line given in figure 14. This process is carried out with the XploRe command `update`.

position of the graphic the object *tru* and in the second position the object *fh*. This order is important for the update command because it is necessary to indicate which object is going to be replaced, that is, the first, second, etc... by given the order that the new one is going to take in the graphic. Eventually objects can also be added if this order is bigger than the number of objects already displayed previously. So that, in the prevailing example it is necessary to indicate that the new *fh* is going to take the position 2 (the same as before). Also, it is going to be drawn as a solid line. The parameter *newproj* is very useful because it allows the scale of the graphic to adapt for the new values assumed by the objects updated. If this parameter is not given the scale remains the same as before. With `update` it is also possible to replace or actualize matrices in text windows. In this example the matrix *inf* was also updated after being changed to incorporate the new bandwidth (and increment) values.

### 4.2.2 Updating the *style*

The command `update` is also useful to manipulate the *style* of the graphic, e.g., the title, the axes name, the drawstyle of the data (line, points, etc...) that usually is undertaken in XploRe interactively by the user. In order to obtain the display in figure 14 after the **show** command the following update commands have to be given

```
upd = update(tru 1 s2d1 title "XPLORE TWDENES2" xaxis "x bins")
upd = update(tru 1 s2d1 solid line thick yaxis "density")
upd = update(fh  2 s2d1 solid line).
```

## 4.3 Mask operations

Mask operations manipulate the color and outfit of a data point. XploRe has a special type of matrices — mask matrices — which allow to define a specific color and a specific style (point, star, etc...) for each data point. The mask matrix defined for a particular data set must have the same number of rows as the data matrix and has to be vertically concatenated with it. That is, having a data matrix (float type) $x$ and defining a mask, the mask type columns should be "glue" to the float columns resulting in a bigger matrix $x$. However when $x$ is displayed the mask columns are ignored. They are ignored as well in all mathematical operations applied to $x$. A mask matrix can be created using either the command **mask** or `vtocc`. The command `vtocc` can be more useful to handle when different configurations in color or style are assigned to the data points because the input matrix is numeric (with **mask** the color and style are given by text strings). That is, using **vtocc** the particular color combinated with a particular style required for a data point is identified by a specific number.

Consider the display in figure 16 produced by the macro **TWINFOBS**. Here a data set with ten points is displayed where one of the points (the one corresponding to the fifth observation) has a different color (yellow) and style (star). This is

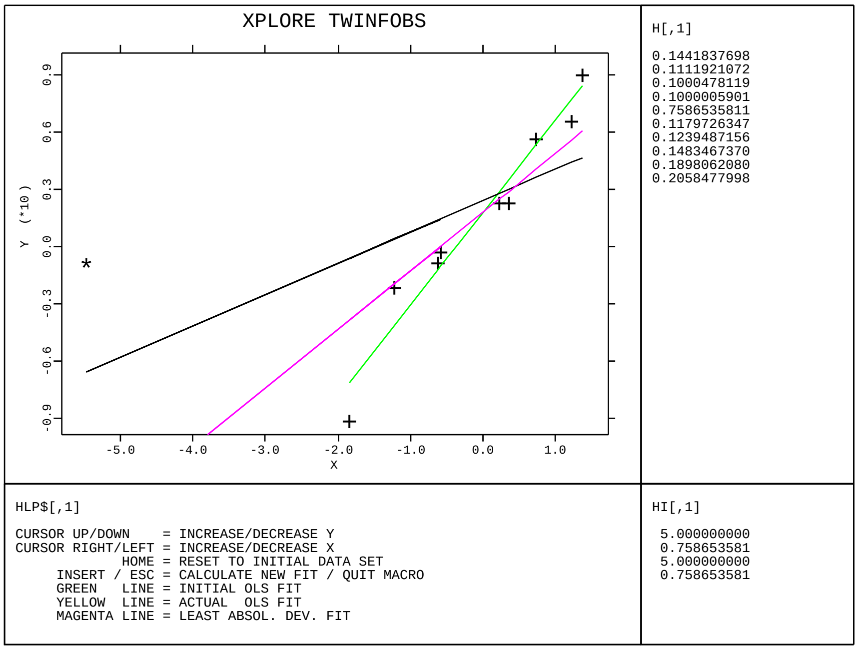


Figure 16: This display is an example of the XploRe features on masking. In color outputs the three lines have different colors. The black line (the OLS fit of the plotted data) should be yellow as also the point represented by the star (the outlier) while the other observations (in crosses) should be white. The line with the top more to the right should be magenta (the L1–line) while the last line (the OLS fit for the initial data) should be green. This colors can be assigned using the XploRe commands **mask** or `vtocc`.

the observation which value has been manipulated by the user. The line with the lower part more to the right is the OLS fit for the original data set and in color output is green, the black line is the OLS fit of the actual data set and for color output is yellow. Finally the last line should be magenta and is the L1–line also for the actual data set. Suppose that the data is stored in the float matrix *dat* and the OLS and L1–line fitted data are respectively in the float matrices *aols* and *al*1. The results of the OLS fit for the original data are in matrix *ools*. The mask operations to produce the outfit in the display above presented are obtained with the commands

```
ools = ools~mask(10 1 lgreen)
aols = aols~mask(10 1 lyellow)
al1  =  al1~mask(10 1 lmagenta)
stco = 31*matrix(10)
stco[5,.] = 95
dat  =  dat~vtocc(stco)
```

The first three lines define mask matrices with 10 rows, 1 column and with style by default *point* and color respectively light green, light yellow and light magenta. Then a vector of 10 rows with all elements equal to 31 was generated. The number 31 identifies *point* with the color white. Then the fifth observation is labeled with 95 which identifies *star* with color yellow. And finally the mask vector corresponding to *dat* is created with the `vtocc` command that interprets the numbers in *stco*. To obtain the display it is necessary to give a `show` or `update` command to those variables.

## 4.4 Producing interactiveness

Within the XploRe environment the computer understands the particular action undertaken by the user by means of the following commands:

- readcon
- readstr
- readval

### 4.4.1 Reading a key – `readcon`

XploRe assigns to each key in the keyboard a specific code number (for example the key corresponding to number 1 has the code number 49). The command `readcon` returns to the program the code number of the key that has been pressed. It has useful optional parameters that allow, for example, the program to pause until a key is pressed or until a key belonging only to a pre-specified subset of keys is pressed.

When the user chooses an action and transmits it by pressing a designed key the program can identify this key with the `readcon` command and then select the action from the set of all possible actions by a conditional branching structure. This procedure can be illustrated analyzing the XploRe code in the macro **TWLINREG**.

In **TWLINREG** the user may implement one of the following actions:

- increase/decrease the intercept *alp* by pressing *cursor UP/DOWN*;
- increase/decrease the slope *bet* by pressing *cursor LEFT/RIGHT*;
- insert new data
- generate a new data set by pressing *HOME*;
- quit the macro by pressing *ESC*

The conditional branching structure that allows the program to select and run one of these actions is presented below.

```
k = readcon(-1 usekeys) ; waits until a key which code number is in
                        ; the vector usekeys is pressed
exit((k.=27).|(k.=327)  ; exits the loop if ESC or Home are pressed
if     (k.=328)         ; cursor up
  alp = alp + dalp
elseif (k.=336)         ; cursor down
  alp = alp - dalp
elseif (k.=331)         ; cursor left
  bet = bet + dbet
elseif (k.=333)         ; cursor right
  bet = bet - dbet
endif
yfit   = x * (alp|bet)  ; fitted y in the actual direction
datfit = x[.,2]~yfit
ssr    = sum((y - yfit)^2)
tdat   = alp|bet|ssr1
upd    = update(zfit 2 s2d1 line solid)
lsinf  = nam~tdat
upd    = update(lsinf 1 text1)
```

### 4.4.2 Reading a string – readstr

Sometimes an action has to be transmitted as a text. This is the case when the user wants to analyze a particular data set and has to input the name specification of the data file. XploRe enables the program to read a text typed by the user via the command **readstr**. When **readstr** is called a box appears in the screen where the user can input the text. It is possible to send some message to the user inside this box. Then, a string matrix is created with the text typed by the user. In the macro **TWDENES1** this command is used in the following way

```
dat = readstr("input the name of the data file")
x = read(dat)     ; reads the file inputed by the user
```

If the data file is not stored in the XploRe data directory then the full path name has to be given.

### 4.4.3 Reading a value – `readval`

When the action has to be transmitted as a numeric value (usually a value that the user wishes to assign to some parameter) XploRe uses the command **`readval`** to fetch it. The command **`readval`** works very similarly to **`readstr`** but creates a float matrix with the user input instead of a string. **`TWREGSMO`** is one of the macros that uses this command. For certain smoothing procedures the user may change the smoothing parameter and input a value of his choice. This is the case for the running medians smoother where the smoother parameter is identified by $r$. The command is used as follows.

```
r = readval("input new r (r has to be odd)")
```

A box will appear in the screen with the message given in the command and the user has to insert a number on it. Automatically the smoothing parameter $r$ is set to this value.

### 4.4.4 Others – `message` and `writecon`

In some occasions the program needs to send a message to the user. This can be done using the command **message**. An example is provided by the macro **`TWDENES1`**. When the user asks for the histogram or the estimate given by the better rule of thumb the program should pause as long as the user wishes to analyze the resulting picture because when it continues the display will be set to the previous situation. With **`message`** this is possible to do by giving the directive

```
message("press any key"  WAIT)
```

After **`message`** is called the text between the quotation marks will appear on the computer screen (in the command line). The optional parameter *wait* obliges the program to hold in pause until a key is pressed.

Another useful command in creating interactiveness is **`writecon`**. This command inserts a XploRe code into the keybuffer. This has the same result as if the user would press the key corresponding to the code inserted. In the Teachware macros **`writecon`** is used very often to simulate the pressing of the key *ESC*. Within XploRe environment when the **`show`** command is invoked and a object is displayed in a window the program automatically will stay in the window in order to the user to manipulate the graphic outfit interactively. To leave the window and continue the program the user has to press *ESC*. In the teachware macro the graphic outfit is always controlled by the program and not by the user. So that, it is necessary that after a **`show`** call the program leaves automatically the window. This is done by typing

```
writecon(27)       ; 27 is the code of ESC key
show(tru fh s2d1)
```

Note that **`writecon`** has to be given before **`show`**.

### 4.5 Calling external procedures

Some of the teachware macros already analyzed need to call external procedures which are not included in any library. Procedures to generate the distributions in `TWDENES2` or a sample from those distributions, procedures to calculate the MISE function or the LSCV for density estimation or kernel regression are an example. They are sufficiently complex to be kept independent in order not to burden the code of the interactive macros. These procedures have to be stored in a file with name equal to the procedure name and they are all stored in the same directory as the teachware macros (usually is in the subdirectory *lib* of the *xplore3* directory). Before they are called their existence and localization has to be recognized by the program using the command `func`. However the whole path has to be given to `func` and this path depends on where XploRe is installed in each personal computer. Here the command **environ** can be very useful. This command returns a text string containing the environment variable *xplore3*, that is, the path where the main executable files of XploRe are stored. To illustrate how the external procedure *twmise* can be called from TWDENES2 consider the following code segment,

```
ftwnam = environ + ("\lib\twmise")
func(ftwnam)
mise   = twmise(n va mu w h)
```

In the first code line the text matrix *ftwnam* is created containing the whole path where the file *twmise.xpl* is stored. Then this file is loaded into the program by the command `func` and then finally the procedure may be invoked.

# 5 Conclusions

The potential of the computer as a tool to provide instruction is enormous and is being recognized as such in all fields of education. The actual accessibility of computers and their sophistication stimulate the development of software for educational purposes (teachware or courseware).

This paper presents a teachware module to support the instruction of basic statistics and econometrics. This teachware system is fully interactive and can be easily learned and used by persons that do not have a special knowledge about computers nor programming languages. It is written in a macro-language, XploRe 3.1, in order to allow teachers to manipulate the program without much effort by introducing changes or building additional procedures that suit their teaching needs.

The system presented is still in development. However the basic structure involving the programming language is achieved and can easily be used to inspire and help other persons in making their own extensions.

## 6 Acknowledgments

I am very grateful to the supervisors of this project, Prof. W. Härdle and Prof. L. Simar, for encouragement and valuable suggestions. I would like to thank M. De Wolf and the Unité d'Informatique of U.C.L. for their bibliographic help on computer science. I am also grateful to S. Klinke and B. Turlach for their useful comments. A special thank is due to S. Klinke by the support provided with XploRe 3.1.

## References

Härdle, W. (1990). *Applied Nonparametric Regression*, Econometric Society Monographs, 19, Cambridge University Press.

Härdle, W. (1991). *Smoothing Techniques with Implementation in S*, Springer–Verlag, New York.