QuantifQuantile : an R package for performing quantile regression through optimal quantization

Isabelle Charlier, Davy Paindaveine, Jérôme Saracco

QuantifQuantile : an R package for performing quantile regression through optimal quantization

Isabelle Charlier[(1,2,3)*] Davy Paindaveine[(1,2)†] Jérôme Saracco[(3)]

November 6, 2014

[(1)] *Université Libre de Bruxelles, Département de Mathématique, Boulevard du Triomphe, Campus Plaine, CP210, B-1050, Bruxelles, Belgium.*

ischarli@ulb.ac.be, dpaindav@ulb.ac.be

[(2)] *ECARES, 50 Avenue F.D. Roosevelt, CP114/04, B-1050, Bruxelles, Belgium.*

[(3)] *Université de Bordeaux, Institut de Mathématiques de Bordeaux, UMR CNRS 5251 and INRIA Bordeaux Sud-Ouest, équipe CQFD, 351 Cours de la Libération, 33405 Talence.*

Jerome.Saracco@math.u-bordeaux1.fr

Abstract

Quantile regression allows to assess the impact of some covariate X on a response Y. An important application is the construction of reference curves and conditional prediction intervals for Y. Recently, Charlier et al. (2014a) developed a new nonparametric quantile regression method based on the concept of *optimal quantization*. In this paper, we describe an R package, called **QuantifQuantile**, that allows to perform quantization-based quantile regression. We describe the various functions of the package and provide examples.

Keywords : Nonparametric estimation, optimal quantization, quantile regression, R package

1 Introduction

In numerous applications, quantile regression is used to evaluate the impact of a d-dimensional covariate X on a (scalar) response variable Y. Quantile regression is an interesting alternative

*Research supported by a Bourse F.R.I.A. of the Fonds National de la Recherche Scientifique, Communauté française de Belgique.

†Research is supported by an A.R.C. contract from the Communauté Française de Belgique and by the IAP research network grant P7/06 of the Belgian government (Belgian Science Policy).

to standard regression whenever the conditional mean does not provide a satisfactory picture of the conditional distribution. Denoting by $F(\cdot|x)$ the conditional distribution of Y given $X = x$, the conditional quantile functions

$$x \mapsto q_\alpha(x) = \inf\{y \in \mathbb{R} : F(y|x) \geq \alpha\}, \qquad \alpha \in (0,1), \tag{1.1}$$

indeed always yield a complete description of the conditional distribution. For our purposes, it is useful to recall that the conditional quantiles in (1.1) can be equivalently defined as

$$q_\alpha(x) = \arg\min_{a \in \mathbb{R}} \mathrm{E}[\rho_\alpha(Y - a)|X = x], \tag{1.2}$$

where $\rho_\alpha(z) = \alpha z \mathbb{I}_{[z \geq 0]} - (1-\alpha) z \mathbb{I}_{[z<0]}$ is the so-called *check function*.

For fixed α, the quantile functions $x \mapsto q_\alpha(x)$ provide reference curves (when $d = 1$), one for each value of α. For fixed x, they provide conditional prediction intervals of the form $I_\alpha = [q_\alpha(x), q_{1-\alpha}(x)]$ ($\alpha < 1/2$). Such reference curves and prediction intervals are widely used, e.g., in economics, ecology, or lifetime analysis. In medicine, they are used to provide reference growth curves for children's height and weight given their age.

Many approaches have been developed to estimate conditional quantiles. After the seminal paper Koenker and Bassett (1978) that introduced linear quantile regression, much effort has been made to consider nonparametric quantile regression. The most classical estimators in this vein are the nearest-neighbor estimators from Bhattacharya and Gangopadhyay (1990) and the (kernel) local linear estimators from Yu and Jones (1998). For related work, we also refer to, e.g., Fan et al. (1994), Gannoun et al. (2002), and Yu et al. (2003).

Recently, Charlier et al. (2014a) developed a new nonparametric quantile regression method based on the concept of *optimal quantization*. Optimal quantization replaces the (typically continuous) covariate X with a discretized version $\widetilde{X}^N$ obtained by projecting X on a collection of N points (these N points, that form the *quantization grid*, are chosen to minimize the L_p-norm of $X - \widetilde{X}^N$; see Section 2.1). As shown in Charlier et al. (2014b), the resulting conditional quantile estimators compete very well with their classical nearest-neighbor or kernel competitors.

The goal of this paper is to describe an R package, called **QuantifQuantile**, that allows to perform quantization-based quantile regression. This includes the data-driven selection of the grid size N, the construction of the corresponding quantization grid, the computation of the resulting sample conditional quantiles, as well as (for $d = 1$ and $d = 2$) their graphical representation.

The paper is organized as follows. Section 2 first briefly recalls the construction of quantization-based quantile regression introduced in Charlier et al. (2014a,b) and then explains the various steps needed to obtain the resulting estimators. Section 3 lists the functions of **QuantifQuan-**

tile, and describes their inputs and outputs. Finally, Section 4 provides several examples that illustrate the use of the various functions.

2 Quantile regression through quantization

As mentioned above, the R package we are proposing in this paper implements the Charlier et al. (2014a,b) quantization-based methodology to perform nonparametric quantile regression. This section describes this methodology.

2.1 Approximating population conditional quantiles through quantization

Let $\gamma^N \in (\mathbb{R}^d)^N$ be a grid of size N, that is, a collection of N points in $\mathbb{R}^d$. For any $x \in \mathbb{R}^d$, we will denote by $\tilde{x}^{\gamma^N} = \text{Proj}_{\gamma^N}(x)$ the projection of x onto this grid, that is, the point of γ^N that is closest to x (absolute continuity assumption makes ties unimportant in the sequel). This allows to approximate the d-dimensional covariate X by its quantized version $\widetilde{X}^{\gamma^N}$.

Obviously, the choice of the grid has a significant impact on the quality of this approximation. Under the assumption that $\|X\|_p := \text{E}[|X|^p]^{1/p} < \infty$ (throughout, $|\cdot|$ denotes the Euclidean norm), optimal quantization selects the grid γ^N that minimizes the L_p-quantization error $\|X - \widetilde{X}^{\gamma^N}\|_p$. Such an optimal grid exists under the assumption that the distribution of X does not charge any hyperplane; see, e.g., Pagès (1998). In practice, an optimal grid is constructed using a *stochastic gradient algorithm* (see Section 2.2). For more details on quantization, the reader may refer to Pagès (1998) and Graf and Luschgy (2000).

Based on optimal quantization of X, we can approximate the conditional quantile $q_\alpha(x)$ in (1.2) by

$$\widetilde{q}_\alpha^N(x) := \arg\min_{a\in\mathbb{R}} \text{E}\big[\rho_\alpha(Y-a)|\widetilde{X}^N = \tilde{x}^N\big], \tag{2.1}$$

where $\widetilde{X}^N$ (resp., $\tilde{x}^N$) denotes the projection of X (resp., x) onto an optimal grid. It is shown in Charlier et al. (2014a) that, under mild assumptions, $\widetilde{q}_\alpha^N(x)$ converges to $q_\alpha(x)$ as $N \to \infty$, uniformly in x.

2.2 Obtaining an optimal N-grid

As we will see below, whenever independent copies $(X_1', Y_1)', \ldots, (X_n', Y_n)'$ of $(X', Y)'$ are available, the first step to obtain a sample version of (2.1) is to compute an optimal N-grid (we assume here that N is fixed). As already mentioned, this can be done through a stochastic gradient algorithm. This algorithm, called *Competitive Learning Vector Quantization (CLVQ)* when $p = 2$, is an iterative procedure that operates as follows :

- First, an initial grid — $\hat{\gamma}^{N,0}$, say — is chosen by sampling randomly without replacement among the X_i's.
- Second, n iterations are performed (one for each observation available). The grid $\hat{\gamma}^{N,t} = (\hat{\gamma}_1^{N,t}, \ldots, \hat{\gamma}_N^{N,t})$ at step t is obtained through

$$\hat{\gamma}_i^{N,t} = \begin{cases} \hat{\gamma}_i^{N,t-1} - \delta_t |\hat{\gamma}_i^{N,t-1} - X_t|^{p-1} \frac{\hat{\gamma}_i^{N,t-1} - X_t}{|\hat{\gamma}_i^{N,t-1} - X_t|} & \text{if } \mathrm{Proj}_{\hat{\gamma}^{N,t-1}}(X_t) = \hat{\gamma}_i^{N,t-1} \\ \hat{\gamma}_i^{N,t-1} & \text{otherwise,} \end{cases}$$

  where $(\delta_t), t \in \mathbb{N}_0$, is a deterministic sequence in $(0,1)$ such that $\sum_t \delta_t = \infty$ and $\sum_t \delta_t^2 < \infty$. At the t^{th} iteration, only one point of the grid moves, namely the one that is closest to X_t.

The optimal grid provided by this algorithm is then $\hat{\gamma}^{N,n}$.

2.3 Estimating conditional quantiles

Assume now that a sample $(X_1', Y_1)', \ldots, (X_n', Y_n)'$ of $(X', Y)'$ is indeed available. The sample analog of (2.1) is then defined as follows :

(S1) First, we compute the optimal grid $\hat{\gamma}^{N,n}$ through the stochastic gradient algorithm, and we write $\widehat{X}_i^N = \mathrm{Proj}_{\hat{\gamma}^{N,n}}(X_i)$, $i = 1, \ldots, n$.

(S2) Then, the conditional quantiles are estimated by

$$\widehat{q}_\alpha^{N,n}(x) = \arg\min_{a \in \mathbb{R}} \sum_{i=1}^{n} \rho_\alpha(Y_i - a) \mathbb{I}_{[\widehat{X}_i^N = \hat{x}^N]}, \tag{2.2}$$

  where $\hat{x}^N = \mathrm{Proj}_{\hat{\gamma}^{N,n}}(x)$. In practice, $\widehat{q}_\alpha^{N,n}(x)$ is simply evaluated as the sample α-quantile of the Y_i's for which $\widehat{X}_i^N = \mathrm{Proj}_{\hat{\gamma}^{N,n}}(X_i) = \hat{x}^N$.

It is shown in Charlier et al. (2014a) that, for N, x fixed and under mild assumptions, $\widehat{q}_\alpha^{N,n}(x)$ converges in probability to $\widetilde{q}_\alpha^N(x)$ as $n \to \infty$, provided that quantization is based on $p = 2$.

Clearly, the originality of our method is how it decides in (S2) which X_i's will be taken into account to estimate conditional quantiles given $X = x$. For the sake of comparison, the local linear and local constant estimators from Yu and Jones (1998) base this decision on a (usually, global-in-x) bandwidth, while the k-nearest neighbor (kNN) estimator from Bhattacharya and Gangopadhyay (1990) selects the k observations whose X-parts are closest to x. Our method is close in spirit to kNN, but has the advantage over kNN that the number of observations used to estimate $q_\alpha(x)$ may depend on x.

When the sample size n is small to moderate ($n \leq 300$, say), the estimated reference curves $x \mapsto \widehat{q}_\alpha^{N,n}(x)$ typically are not smooth. To improve on this, Charlier et al. (2014a,b)

introduced the following bootstrapped version of the estimator in (2.2). For some positive integer B, generate B samples of size n from the original sample $\{(X_i', Y_i)'\}_{i=1,\ldots,n}$ with replacement. From each of these bootstrap samples, the stochastic gradient algorithm provides an "optimal" grid, using these bootstrapped samples to perform the iterations. The bootstrapped estimator of conditional quantile is then

$$\bar{q}_{\alpha,B}^{N,n}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{q}_{\alpha}^{(b)}(x), \tag{2.3}$$

where $\hat{q}_{\alpha}^{(b)}(x) = \hat{q}_{\alpha}^{(b),N,n}(x)$ denotes the estimator in (2.2) computed on the basis of the b^{th} optimal grid. We stress that, when computing $\hat{q}_{\alpha}^{(b)}(x)$, the original sample is used in (S2); the bootstrapped samples are only used to provide the B different grids. As shown in Charlier et al. (2014a,b), the bootstrapped reference curves are much smoother than the original ones. Of course, B should be chosen large enough to make the bootstrap useful, but also small enough to keep the computational burden under control. We usually choose $B = 50$ when X is univariate.

2.4 Selecting the grid size N

Both for the original estimators $\widehat{q}_{\alpha}^{N,n}(x)$ and for their bootstrapped version $\bar{q}_{\alpha,B}^{N,n}(x)$, an appropriate value of N should be identified. If N is too small, then reference curves will have a large bias, while if N is too large, then they will show much variability. This leads to the usual bias/variance trade-off that is to be achieved when selecting the value of a smoothing parameter in nonparametric statistics.

Charlier et al. (2014b) proposed the following data-driven method to choose N. Let $\{x_1, \ldots, x_J\}$ be a set of x-values at which we want to estimate $q_\alpha(x)$ (the x_j's are for instance chosen equispaced on the support of X) and let $\mathcal{N}$ be a finite collection of N-values (this represents the values of N one allows for and should typically be chosen according to the sample size n). Ideally, we would like to select the optimal value of N as

$$N_{\alpha;\text{opt}}^{-} = \arg\min_{N\in\mathcal{N}} \text{ISE}_{\alpha}^{-}(N), \quad \text{with} \quad \text{ISE}_{\alpha}^{-}(N) = \frac{1}{J}\sum_{j=1}^{J}\left(\bar{q}_{\alpha,B}^{N,n}(x_j) - q_\alpha(x_j)\right)^2. \tag{2.4}$$

This, however, is infeasible, since the population quantiles $q_\alpha(x_j)$ are unknown. This is why we draw $\widetilde{B}$ extra bootstrap samples (still of size n) from the original sample and consider

$$\hat{N}_{\alpha;\text{opt}}^{-} = \arg\min_{N\in\mathcal{N}} \widehat{\text{ISE}}_{\alpha}^{-}(N), \quad \text{with} \quad \widehat{\text{ISE}}_{\alpha}^{-}(N) = \frac{1}{J}\sum_{j=1}^{J}\left(\frac{1}{\widetilde{B}}\sum_{\tilde{b}=1}^{\widetilde{B}}\left(\bar{q}_{\alpha,B}^{N,n}(x_j) - \hat{q}_{\alpha}^{(\tilde{b})}(x_j)\right)^2\right), \tag{2.5}$$

where $\hat{q}_{\alpha}^{(\tilde{b})}(x_j)$, for $\tilde{b} = 1\ldots, \tilde{B}$, is obtained by projecting the original sample in (S2) on the quantization grid resulting from this $\tilde{b}^{\text{th}}$ new bootstrap sample (again, the bootstrap sample is

only used to perform the iterations of the algorithm, the original sample being still used in both the initial grid and in (S2)).

As shown in Charlier et al. (2014b) through simulations, both $N \mapsto \overline{\mathrm{ISE}}_\alpha(N)$ and $N \mapsto \overline{\widehat{\mathrm{ISE}}}_\alpha(N)$ are essentially convex in N and lead to roughly the same minimizers. This therefore provides a feasible way to select a reasonable value of N for the estimator $\bar{q}^{N,n}_{\alpha,B}(x)$ in (2.3). Note that this also applies to $\hat{q}^{N,n}_\alpha(x)$ by simply taking $B = 1$ in the procedure above.

If quantiles are to be estimated at various orders α, (2.5) will provide an optimal N-value for each α. It may then happen, in principle, that the resulting reference curves cross, which is of course undesirable. One way to guarantee that no such crossings occur is to identify a common N-value for the various α's. In such a case, N will be chosen as

$$\hat{N}^{-}_{\mathrm{opt}} = \arg\min_{N\in\mathcal{N}} \overline{\widehat{\mathrm{ISE}}}(N), \qquad \text{with } \overline{\widehat{\mathrm{ISE}}}(N) = \textstyle\sum_\alpha \overline{\widehat{\mathrm{ISE}}}_\alpha(N), \tag{2.6}$$

where the sum is computed over the various α-values considered.

3 The QuantifQuantile package

This section provides a description of the various functions offered by the R package **QuantifQuantile**. We first detail the three functions that allow to estimate conditional quantiles through quantization (Section 3.1). Then we describe a function computing optimal quantization grids (Section 3.2).

3.1 Conditional quantile estimation

QuantifQuantile is composed of three main functions that each provide estimated conditional quantiles in (2.2)-(2.3). These functions work in a similar way but address different values of d (recall that d is the dimension of the covariate vector X) :

- The function `QuantifQuantile` is suitable for $d = 1$.
- The function `QuantifQuantile.d2` addresses the case $d = 2$.
- Finally, `QuantifQuantile.d` can deal with an arbitrary value of d.

Combined with the `plot` function, the first two functions provide reference curves and reference surfaces, respectively. No graphical outputs can be obtained from the third function.

The three functions share the same arguments :

- `X`: a $d \times n$ real array (required by all three functions). The columns of this matrix are the X_i's, $i = 1, \ldots, n$.

- `Y`: an $n \times 1$ real array (required by all three functions). This vector collects the Y_i's, $i = 1, \ldots, n$.
- `alpha`: an $r \times 1$ array with components in $(0, 1)$ (optional for all three functions). This vector contains the orders for which $q_\alpha(x)$ should be estimated. The default is (0.05, 0.25, 0.5, 0.75, 0.95).
- `x`: a $d \times J$ real array (optional for `QuantifQuantile` and `QuantifQuantile.d2`, required by `QuantifQuantile.d`). The columns of this matrix are the x_j's at which the quantiles $q_\alpha(x_j)$ are to be estimated. If `x` is not provided when calling `QuantifQuantile`, then it is set to a vector of $J = 100$ equispaced values between the minimum and the maximum of the X_i's. If this argument is not provided when calling `QuantifQuantile.d2`, then the default for `x` is a matrix whose $J = 20^2 = 400$ column vectors are obtained as follows: 20 equispaced values are considered between the minimum and maximum values of the $(X_i)_1$'s and similarly for the second component. The 400 column vectors of the default `x` are obtained by considering all combinations of those 20 values for the first component with the 20 values for the second one[1].
- `testN`: an $m \times 1$ vector of pairwise distinct positive integers (optional for all three functions). The entries of this vector are the elements of the set $\mathcal{N}$ in (2.5)-(2.6), hence are the N-values for which the $\widehat{\mathrm{ISE}}^{-}_{\alpha}$ quantity considered will be evaluated. The default is $(35, 40, \ldots, 55)$ but it is strongly recommended to adapt it according to the sample size n at hand.
- `p`: a real number larger than or equal to one (optional for all three functions). This is the parameter p to be used when performing optimal quantization in L_p-norm. The default is 2.
- `B`: a positive integer (optional for all three functions). This is the number of bootstrap replications B to be used in (2.3). The default is 50.
- `tildeB`: a positive integer (optional for all three functions). This is the number of bootstrap replications $\widetilde{B}$ to be used when determining $\hat{N}^{-}_{\alpha;\mathrm{opt}}$ or $\hat{N}^{-}_{\mathrm{opt}}$. The default is 20.
- `same_N`: Boolean by default (optional for all three functions). If `same_N=TRUE`, then a common value of N (that is, $\hat{N}^{-}_{\mathrm{opt}}$ in (2.6)) will be selected for all α's. If `same_N=FALSE`, then optimal values of N will be chosen independently for the various of α (which will provide several $\hat{N}^{-}_{\alpha;\mathrm{opt}}$, as in (2.5)). The default is `TRUE`.

All three functions return the following list of objects, which is of class `QuantifQuantile` :

[1]Since the number J of points in a default value of x obtained in this fashion would increase exponentially with the dimension d, we did not adopt the same approach for $d \geq 3$

- `hatq_opt`: an $r \times J$ real array (where r is the number of α-values considered). If `same_N=TRUE`, then the entry (i, j) of this matrix is $\bar{q}_{\alpha_i,B}^{\hat{N}^-_{\text{opt}},n}(x_j)$. If `same_N=FALSE`, then it is rather $\bar{q}_{\alpha_i,B}^{\hat{N}^-_{\alpha_i;\text{opt}},n}(x_j)$.
- `N_opt`: a positive integer (if `same_N=TRUE`) or an $r \times 1$ array of positive integers (if `same_N=FALSE`). Depending on `same_N`, this provides the value of $\hat{N}^-_{\text{opt}}$ or the vector $(\hat{N}^-_{\alpha_1;\text{opt}}, \ldots, \hat{N}^-_{\alpha_r;\text{opt}})$.
- `hatISE_N`: an $r \times m$ real array. The entry (i, j) of this matrix is $\widehat{\text{ISE}}^-_{\alpha_i}(N_j)$. Plotting this for fixed α or plotting its average over the various α, in both cases over `testN`, allows to assess the global convexity of these ISEs. Hence, it can be used to indicate whether or not the choice of `testN` was adequate. This will be illustrated in the examples below.
- `hatq_N`: an $r \times J \times m$ real array. The entry (i, j, ℓ) of this matrix is $\bar{q}_{\alpha_i,B}^{N_\ell,n}(x_j)$, where N_ℓ is the ℓ^{th} entry of the argument `testN`. From this output, it is easy by fixing the third entry to get the matrix of the $\bar{q}_{\alpha_i,B}^{N,n}(x_j)$ values for any N in `testN`.
- The arguments `X`, `Y`, `x`, `alpha`, and `testN` are also reported in this response list.

The `QuantifQuantile` class response can be used as argument of the functions `plot` (only for $d \leq 2$), `summary` and `print`. The `plot` function draws the observations and plots the estimated conditional quantile curves ($d = 1$) or surfaces ($d = 2$) — for $d = 2$, the **rgl** package is used, which allows to change the perspective in a dynamic way. In order to illustrate the selection of N, the package further contains the function `plot.select.N` by plotting (against N) the $\widehat{\text{ISE}}^-_{\alpha}$ and $\widehat{\text{ISE}}^-$ quantities in (2.5) or in (2.6), depending on the choice `same_N=FALSE` or `same_N=TRUE`, respectively; see the examples below for details.

3.2 Computing optimal grids

Besides the functions that allow to estimate conditional quantiles and plot the corresponding reference curves/surfaces, **QuantifQuantile** provides a function that computes optimal quantization grids. This function, called `choice.grid` admits the following arguments :

- `X`: a $d \times n$ real array (required). The columns of this matrix are the X_i's, $i = 1, \ldots, n$, for which the optimal quantization grid should be determined. Each point of `X` is used as a stimulus in the stochastic gradient algorithm to get an optimal grid.
- `N`: a positive integer (required). The size of the desired quantization grid.
- `ng`: a positive integer (optional). The number of desired quantization grids. The default is 1.
- `p`: a real number larger than or equal to one (optional). This is the parameter p used in the quantization error. The default is 2.

Let us detail the parameter `ng`. In some cases, it may be necessary to have several quantization grids. For example, in the use of this function inside `QuantifQuantile` and its multidimensional versions, we need `B + tildeB` grids. If `ng` > 1, the different grids are obtained using as stimuli a resampling version of `X` (the X_t's in Section 2.2).

The output is a list containing the following elements :

- `init_grid`: a d×`N`×`ng` real array. The entry (i, j, ℓ) of this matrix is the i^{th} component of the j^{th} point of the ℓ^{th} initial grid.
- `opti_grid`: a d×`N`×`ng` real array. The entry (i, j, ℓ) of this matrix is the i^{th} component of the j^{th} point of the ℓ^{th} optimal grid provided by the algorithm.

4 Illustrations

In this section, we provide illustrate the use of the functions described above on several examples. Since we want to provide graphical representations, we restrict to `QuantifQuantile` and `QuantifQuantile.d2` (Sections 4.1 and 4.2, respectively). We conclude this section with an illustration of the function `choice.grid` (Section 4.3).

4.1 Example 1 : one-dimensional covariate

We generate a random sample of size $n = 300$, where X is uniformly distributed on the interval $(-2, 2)$ and where Y is obtained by adding to X^2 an independent standard normal error term :

```
R> set.seed(644936)
R> n <- 300
R> X <- runif(n,-2,2)
R> Y <- X^2 + rnorm(n)
```

We test the number N of quantizers between 10 and 30 by steps of 5 and we do not change the default values of the other arguments. We then run the function `QuantifQuantile` with these arguments and we stock the response in `res`.

```
R> testN <- seq(10,30,by=5)
R> res <- QuantifQuantile(X,Y,testN=testN)
R> cat(paste("N_opt=",res$N_opt))

N_opt= 15
```

We first check that this choice of `testN` was adequate. For this purpose, we use the function `plot.select.N` that plots `hatISEmean_N` (obtained by taking the mean of `hatISE_N` over `alpha`) against the various N-values in `testN`.

```
R> plot.select.N(res)
```

Figure 1a provides the resulting graph. We conclude that `testN` was well chosen since `hatISEmean_N` is larger for smaller and larger values of N than `N_opt`. We then plot the corresponding estimated conditional quantiles curves in Figure 1b. The default colors of the points and of the curves are changed by using the `col.plot` argument. This argument is a vector of size 1+`length(alpha)`, whose first component fixes the color of the data points and whose remaining components determine the colors of the various reference curves.

```
R> col.plot <- c("grey","red","orange","green","orange","red")
R> plot(res,col.plot=col.plot,xlab="X",ylab="Y")
```

It is natural to make the grid `testN` finer. Of course, the more N-values we test, the longer it takes. This is why we adopted this two-stage approach, where the goal of first stage was to get a rough approximation of the optimal N-value. In the second stage, we can then refine the grid only in the vicinity of the value `N_opt` obtained in the first stage.

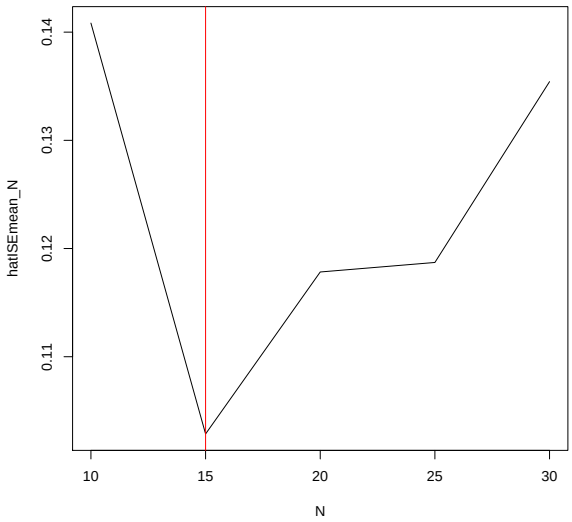
```
R> testN <- c(seq(10,20,by=1),seq(25,30,by=5))
R> res_step1 <- QuantifQuantile(X,Y,testN=testN)
R> cat(paste("N_opt=",res_step1$N_opt))
R> plot.select.N(res_step1)
R> plot(res_step1,col.plot=col.plot,xlab="X",ylab="Y")
```

```
N_opt= 17
```

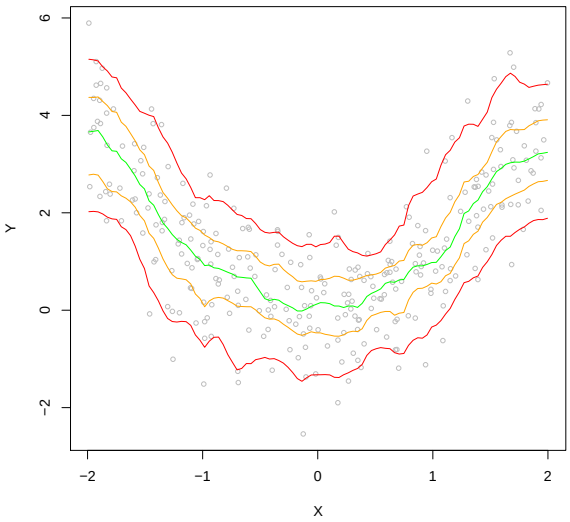
The resulting graphs are plotted in Figure 1c and 1d respectively. We observe that the value of `N_opt` is made more precise, since we now get `N_opt`=17 instead of 15. The resulting estimated conditional quantiles curves in Figure 1d are very similar to the ones in Figure 1b.

So far, we used the default value `same_N=TRUE`, which leads to selecting an N-value that is common to all α's. For the sake of comparison, we now explore the results for `same_N=FALSE`.

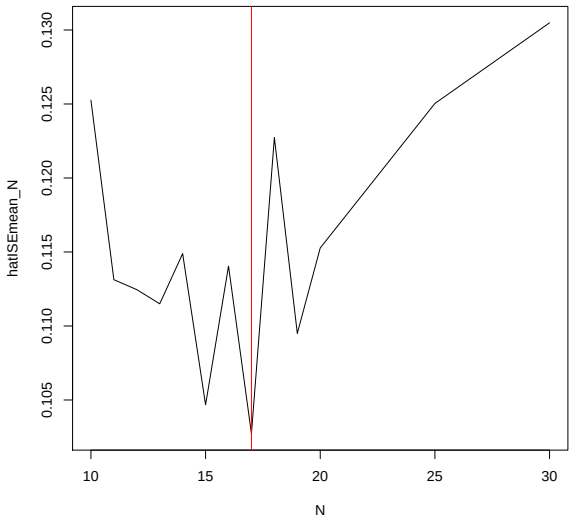
```
R> testN <- c(seq(10,30,by=5))
R> res2 <- QuantifQuantile(X,Y,testN=testN,same_N=FALSE)
R> cat(paste("N_opt=",res2$N_opt))
R> plot.select.N(res2)
R> plot(res2,col.plot=col.plot,xlab="X",ylab="Y")
R> testN <- c(seq(10,20,by=1),seq(25,30,by=5))
R> res2_step1 <- QuantifQuantile(X,Y,testN=testN,same_N=FALSE)
R> cat(paste("N_opt=",res2_step1$N_opt))
R> plot.select.N(res2_step1)
R> plot(res2_step1,col.plot=col.plot,xlab="X",ylab="Y")
```



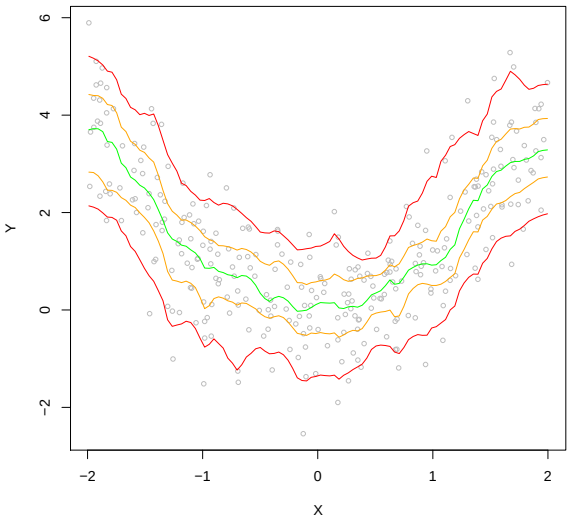
(a)



(b)



(c)



(d)

Figure 1: For a sample of size $n = 300$ with $X \sim U(-2, 2)$ and $Y = X^2 + \varepsilon$, where ε is a standard normal error term (independent of X), this figure provides (a) the plot of $N \mapsto \widehat{\overline{\text{ISE}}}_\alpha(N)$ with N from 5 to 30 by steps of 5, and (b) the resulting reference curves. The panels (c)-(d) provide the corresponding plots when taking N from 5 to 20 by step of 1, along with 25 and 30.

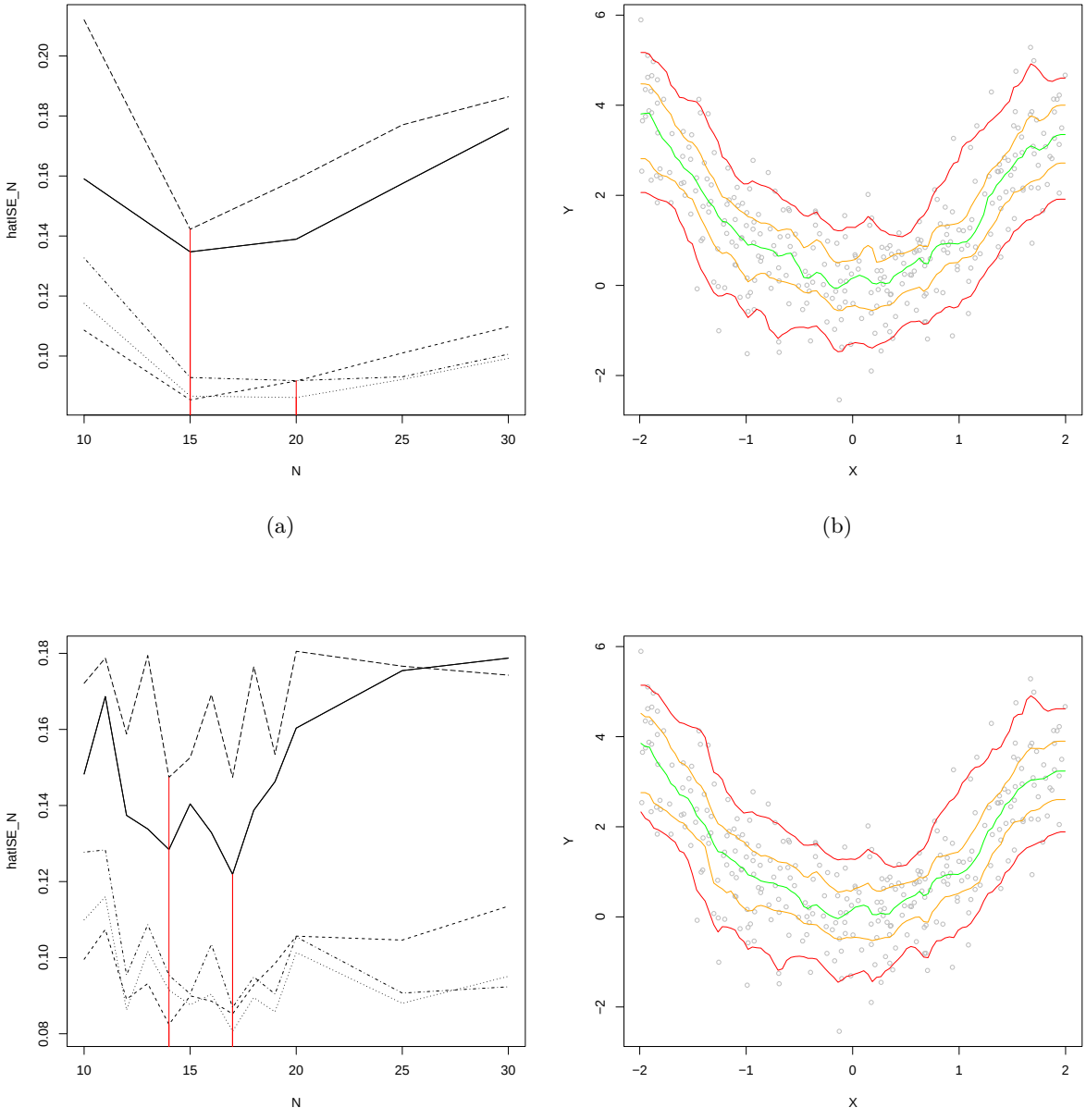


Figure 2: The same results as in Figure 1, but when selecting optimal values of N separately for each α.

```
N_opt= 15 N_opt= 15 N_opt= 20 N_opt= 20 N_opt= 15
N_opt= 17 N_opt= 14 N_opt= 17 N_opt= 17 N_opt= 14
```

The results are provided in Figure 2. Comparing the left panels of Figures 1 and 2, we see that when choosing N by step of five, we find `N_opt`=15 with `same_N=TRUE` and `N_opt`= 15 or 20 (depending on `alpha`) for `same_N=FALSE`. When we refine the grid `testN` and we choose N by step of one, we find analogously `N_opt`=17 and `N_opt`=14 or 17, respectively. In the present setup, thus, both methods provide relatively close optimal N-values, which explains why the corresponding estimated reference curves are so similar (see the right panels of Figures 1 and 2).

4.2 Example 2 : two-dimensional covariate

We now generate a sample of size $n = 1,000$ where $X = (X_1, X_2)'$ is uniformly distributed on the square $(-2, 2)^2$ and where Y is obtained by adding to $X_1^2 + X_2^2$ an independent standard normal error term.

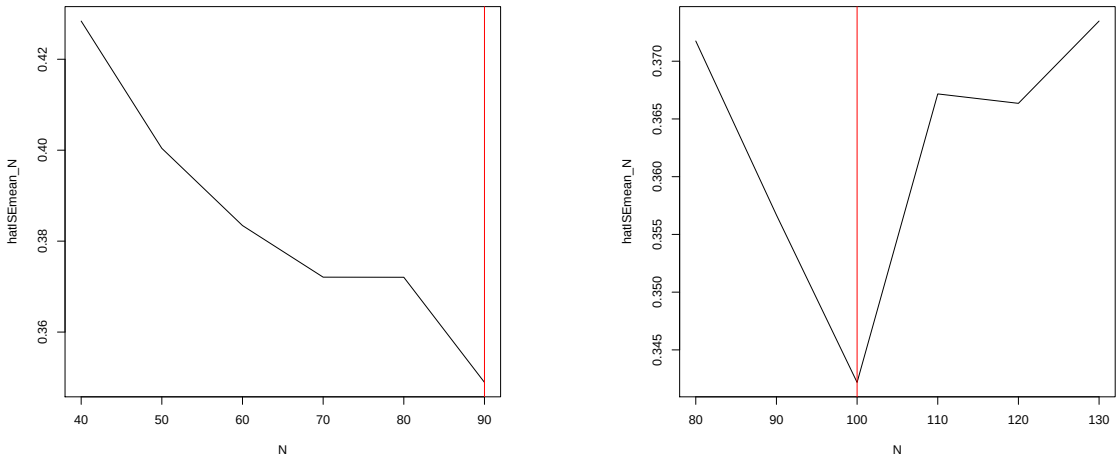
```
R> set.seed(642516)
R> n <- 1000
R> X <- matrix(runif(n*2,-2,2),ncol=n)
R> Y <- apply(X^2,2,sum) + rnorm(n)
```

We test N between 40 and 90 by steps of 10. We change the values of `B` and `tildeB` to reduce the computational burden, that is heavier when $d = 2$ than when $d = 1$. We keep the default values of all other arguments when running the function `QuantifQuantile.d2`. We first investigate whether or not the choice of `testN` was adequate, using again the function `plot.select.N`.

```
R> testN <- seq(40,90,by=10)
R> B <- 20
R> tildeB <- 15
R> res <- QuantifQuantile.d2(X,Y,testN=testN,B=B,tildeB=tildeB)
R> plot.select.N(res)
```

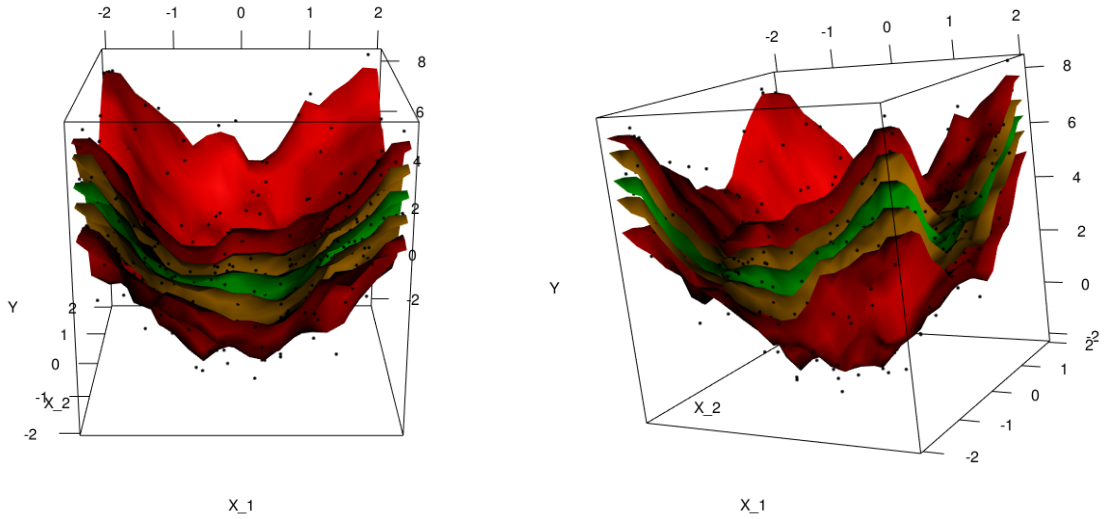
Figure 3a provides the resulting graph. We observe here that `testN` was not well chosen since `hatISEmean_N` becomes smaller and smaller as `N_opt` increases. We then adapt the choice of `testN` accordingly and rerun the procedure, which provides Figure 3b.

```
R> testN <- seq(80,130,by=10)
R> res <- QuantifQuantile.d2(X,Y,testN=testN,B=B,tildeB=tildeB)
R> plot.select.N(res)
```



(a) (b)

Figure 3: For a sample of size $n = 1,000$ with $X = (X_1, X_2)' \sim U((-2,2)^2)$ and $Y = X_1^2 + X_2^2 + \varepsilon$, where ε is a standard normal error term (independent of X), this figure plots $N \mapsto \widehat{\text{ISE}}^{-}(N)$ (a) with N from 40 to 90 by steps of 10, and (b) with N from 80 to 130 by steps of 10.



(a) (b)

Figure 4: For a sample of size $n = 1,000$ with $X = (X_1, X_2)' \sim U((-2,2)^2)$ and $Y = X_1^2 + X_2^2 + \varepsilon$, where ε is a standard normal error term (independent of X), this figure plots (with two different views) some estimated conditional quantile surfaces obtained with the plot function.

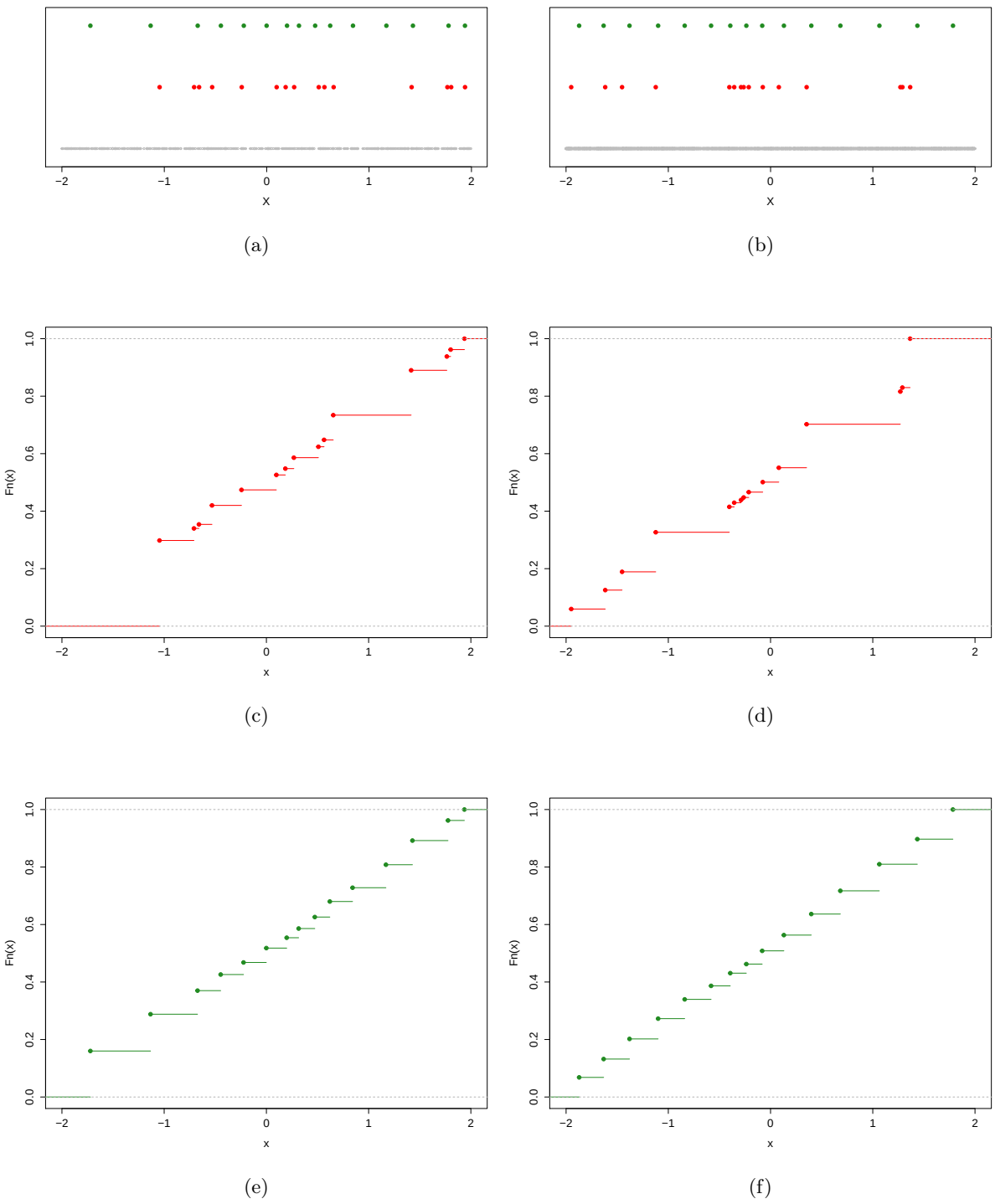
We then plot the corresponding estimated conditional quantile surfaces in Figure 4.

```
R> col.plot <- c("black", "red","orange","green","orange","red")
R> plot(res,col.plot=col.plot,xlab="X_1",ylab="X_2",zlab="Y")
```

4.3 Illustration of choice.grid

In this section, we illustrate the function `choice.grid` in the univariate and bivariate settings. First, we generate 500 points from the uniform distribution on $(-2, 2)$, which defines the data matrix `X`. For `N`=15 and `ng`=1, this function returns a single (randomized) initial grid and the corresponding optimal grid. The top left graph of Figure 5 represents the observations (in grey), the initial grid (in red), and the optimal grid (in green). The middle and bottom left graphs of Figure 5 plot the empirical cumulative distribution functions (ecdfs) of the observations projected on the initial grid (middle, in red) and on the optimal grid (bottom, in green) respectively.

```
R> set.seed(643625)
R> n <- 500
R> X <- runif(n,-2,2)
R> N <- 15
R> ng <- 1
R> res <- choice.grid(X,N,ng)
R> # Plots of the initial and optimal grids
R> plot(X,rep(0,n),col="grey",cex=0.5,ylim=c(-0.1,1.1),yaxt="n",ylab="")
R> points(res$init_grid,rep(0.5,N),col="red",pch=16,cex=1.2)
R> points(res$opti_grid,rep(1,N),col="forestgreen",pch=16,cex=1.2)
R> # To plot the ecdf
R> projX_init <- array(0, dim=c(n, 1))
R> projX_opti <- array(0, dim=c(n, 1))
R> i_init <- array(0, dim = c(n, 1))
R> i_opti <- array(0, dim = c(n, 1))
R> for (i in 1:n) {
R>   RepX <- rep(X[i], N)
R>   diff_init <- sqrt((RepX - res$init_grid)^2)
R>   diff_opti <- sqrt((RepX - res$opti_grid)^2)
R>   i_init[i] <- which.min(diff_init)
R>   i_opti[i] <- which.min(diff_opti)
R>   projX_init[i] <- res$init_grid[i_init[i]]
R>   projX_opti[i] <- res$opti_grid[i_opti[i]]
R> }
R> plot(ecdf(projX_init),main="",col="red",xlim=c(-2,2))
R> plot(ecdf(projX_opti),main="",col="forestgreen",xlim=c(-2,2))
```



(a) (b) (c) (d) (e) (f)

Figure 5: The left panels correspond to a random sample size $n = 500$ from the uniform distribution on $(-2, 2)$, while the right ones are for $n = 5,000$. The top figures represent the n observations (in grey), the initial grids (in red), and the optimal grids (in green), provided by choice.grid. The middle and bottom figures plot the empirical cdf of the observations projected onto the initial grids and the optimal grids, respectively.

Since the parent distribution is uniform over $(-2, 2)$, the optimal quantization grid is the equispaced grid on that interval. In the light of this result, we clearly observe that the optimal grid is much closer to the population optimal grid than the initial one. The ecdfs of the projected observations tell the same story. The fact that the optimal grid provided by our function does not closely approximate the population optimal grid is due to the moderate sample size n considered ($n = 500$). Recall indeed that the CLVQ algorithm performs exactly n iterations. For the sake of comparison, the right panels of Figure 5 provide the same result as in the left panels but for $n = 5,000$ instead of $n = 500$. Clearly, the optimal grid is much more satisfactory than for $n = 500$.

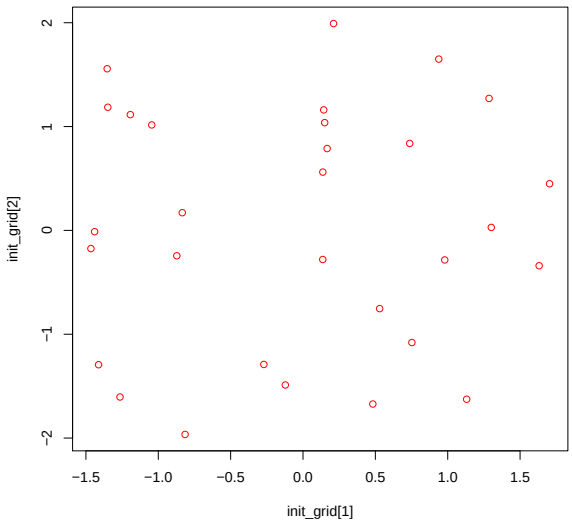
We conclude this paper with an illustration of `choice.grid` in the bivariate case. We generate 2,000 points from the uniform distribution over the square $(-2, 2)^2$, and we choose `N`=30 and `ng=1`. The resulting initial and optimal grids are plotted in the left panels of Figure 6 (top and bottom, respectively). The right panels are obtained similarly from 20,000 points instead of 2,000.

```
R> set.seed(345689)
R> n <- 2000
R> X <- matrix(runif(n*2,-2,2),nc=n)
R> N <- 30
R> ng <- 1
R> res <- choice.grid(X,N,ng)
R> col=c("red","forestgreen")
R> l=c("init_grid[1]","init_grid[2]","opti_grid[1]","opti_grid[2]")
R> plot(res$init_grid[1,,1],res$init_grid[2,,1],col=col[1],xlab=l[1],ylab=l[2])
R> plot(res$opti_grid[1,,1],res$opti_grid[2,,1],col=col[2],xlab=l[3],ylab=l[4])
```

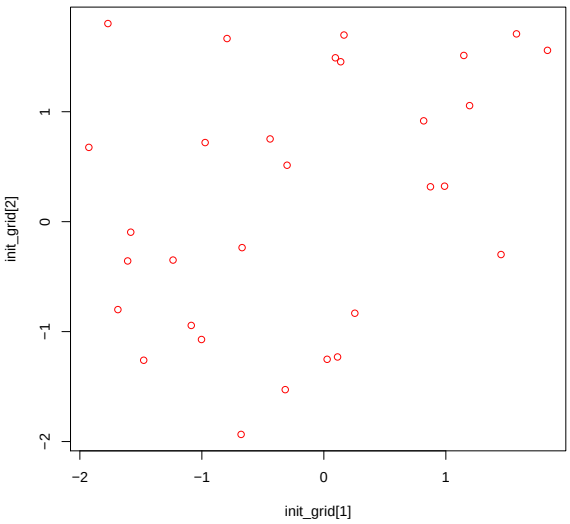
As in the univariate case, we observe an improvement when going from the initial grid to the "optimal" one, that is, the one provided by the function (here as well, the population optimal grid should be uniformly spread over the support of the underlying distribution). Clearly, the "optimal" grid for $n = 2,000$ is quite satisfactory, but the one obtained with n=20,000 is of course much better.

5 Conclusion

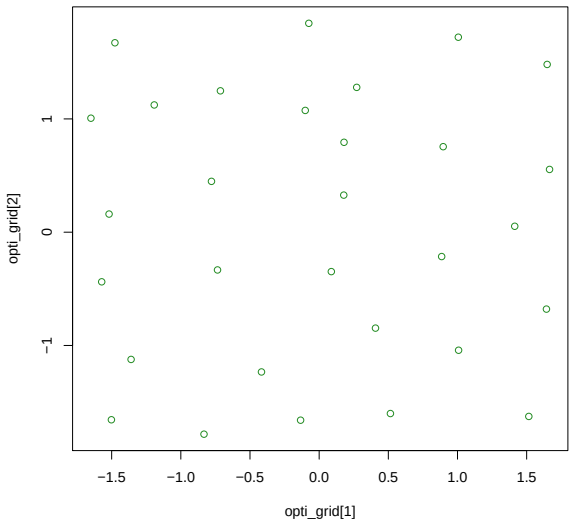
In this paper, we described the **QuantifQuantile** package that we developed for the quantization-based method introduced in Charlier et al. (2014a,b). The package allows the user to compute the corresponding estimators in a quite straightforward way, since the `QuantifQuantile` function and its higher dimensional versions essentially only require providing the covariate and



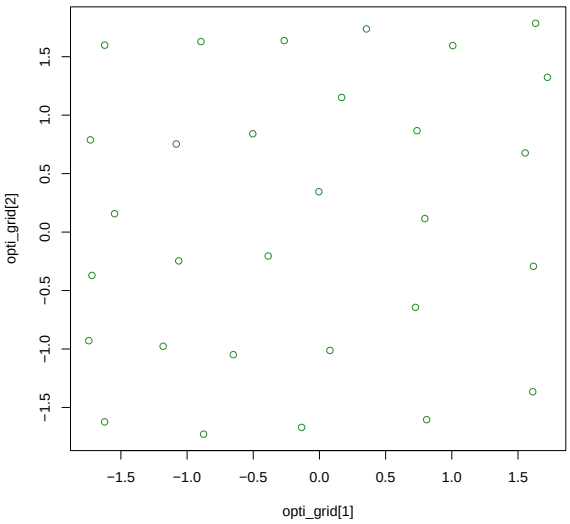
(a)



(b)



(c)



(d)

Figure 6: The left panels correspond to a random sample size $n = 2,000$ from the uniform distribution over $(-2, 2)^2$, while the right ones are for $n = 20,000$. The top and bottom figures represent the initial grids (in red) and the optimal grids (in green), respectively.

response as arguments. Even if the choice of the tuning parameter N is crucial, the function `plot.select.N` serves as guide for the user to change adequately the parameter `testN` of the functions. Moreover, a graphical illustration is directly provided with the `plot` function when the dimension of the covariate is at most 2. Finally, this package also contains a function that provides optimal quantization grids, which might be useful in other contexts, too.

References

Bhattacharya, P. K. and A. K. Gangopadhyay (1990). Kernel and nearest-neighbor estimation of a conditional quantile. *Ann. Statist. 18*(3), 1400–1415.

Charlier, I., D. Paindaveine, and J. Saracco (2014a). Conditional quantile estimation through optimal quantization. *J. Statist. Plann. Inference,* to appear.

Charlier, I., D. Paindaveine, and J. Saracco (2014b). Conditional quantile estimator based on optimal quantization: from theory to practice. *Submitted*.

Fan, J., T.-C. Hu, and Y. Truong (1994). Robust nonparametric function estimation. *Scandinavian Journal of Statistics 21*(4), 433–446.

Gannoun, A., S. Girard, C. Guinot, and J. Saracco (2002). Reference curves based on non-parametric quantile regression. *Statistics in Medicine 21*(4), 3119–3135.

Graf, S. and H. Luschgy (2000). *Foundations of quantization for probability distributions*, Volume 1730 of *Lecture Notes in Mathematics*. Berlin: Springer-Verlag.

Koenker, R. and G. Bassett, Jr. (1978). Regression quantiles. *Econometrica 46*(1), 33–50.

Pagès, G. (1998). A space quantization method for numerical integration. *J. Comput. Appl. Math. 89*(1), 1–38.

Yu, K. and M. C. Jones (1998). Local linear quantile regression. *J. Amer. Statist. Assoc. 93*(441), 228–237.

Yu, K., Z. Lu, and J. Stander (2003). Quantile regression: applications and current research areas. *J. R. Stat. Soc. Ser. D Statistician 52*(3), 331–350.