KYBERNETIKA — VOLUME *53* (2017), NUMBER 3, PAGES 480–492

DIRECTIONAL QUANTILE REGRESSION IN R

Pavel Boček and Miroslav Šiman

Recently, the eminently popular standard quantile regression has been generalized to the multiple-output regression setup by means of directional regression quantiles in two rather interrelated ways. Unfortunately, they lead to complicated optimization problems involving parametric programming, and this may be the main obstacle standing in the way of their wide dissemination. The presented R package modQR is intended to address this issue. It originates as a quite faithful translation of the authors' moQuantile toolbox for Octave and MATLAB, and provides all the necessary computational support for both the directional multiple-output quantile regression methods to the wide statistical public. The article offers a concise summary of the statistical theory behind modQR, overviews the package in brief, points out its departures from moQuantile, comments on its use and performance, and demonstrates its application.

Keywords: multivariate quantile, regression quantile, halfspace depth, regression depth, depth contour

Classification: 62-04, 65C60, 62H05, 62J99

1. INTRODUCTION

Roughly speaking, the directional regression quantile concepts discussed here first associate each direction in the response space with a (regression) quantile hyperplane and its upper (regression) quantile halfspace, and then define (regression) quantile regions as the convex intersections of the upper (regression) quantile halfspaces over all the directions. Such concepts have been theoretically investigated in several articles including [8, 9, 10, 16], and [19], their computational side has been successfully addressed in [17] and [18], and their practical applications still grow in number; see, e. g., [15] and [20]. Similar ideas also appear in other articles such as [3, 5, 6, 13, 14] and [4].

There already exists a toolbox moQuantile [2] for Octave and MATLAB that implements the algorithms of [17] and [18] in a professional way and makes the directional multiple-output quantile regression accessible to ordinary users. The R package modQR [1] presented here (and already available from CRAN) is its faithful R translation by its authors that primarily aims at those many statisticians using R as their only computing environment.

The codes probably have no close competitors in the general regression case as the concept of multivariate quantile regression is quite novel in the statistical literature.

DOI: 10.14736/kyb-2017-3-0480

However, there exist some R packages on robust regression, data depth, regression depth, multiple-output regression, and single-response quantile regression, all of which touching at least one aspect of the directional multiple-output quantile methodology.

The article proceeds with a brief review of the properties of both the directional (regression) quantile methods of interest because R documentation is not suitable for stating complex mathematical expressions. Both of them are motivated by standard single-output quantile regression, introduced in [12] and surveyed in [11]. The package and its functionality are overviewed next. The interpretation of the results is then briefly discussed and followed with two demo examples. At the end, a telegraphic speed comparison of modQR with moQuantile concludes the presentation.

2. METHODOLOGY

Consider a random sample $(\boldsymbol{Y}_i^\top, \boldsymbol{Z}_i^\top)^\top \in \mathbb{R}^{m+p-1}, i = 1, \ldots, n$, where m-dimensional responses $\boldsymbol{Y}_i$ are coupled with both p-dimensional regressors $\boldsymbol{X}_i = (1, \boldsymbol{Z}_i^\top)^\top$ and weights $w_i = w_i(\boldsymbol{Y}_i, \boldsymbol{Z}_i) > 0,\ i = 1, \ldots, n > m+p-1$. Assume that $(\boldsymbol{Y}_i^\top, \boldsymbol{Z}_i^\top)^\top \in \mathbb{R}^{m+p-1}, i = 1, \ldots, n$, come from a continuous distribution. Next consider only integer parameters $m \geq 2$ and $p \geq 1$ where $p = 1$ corresponds to the location case when $(\boldsymbol{Y}_i', \boldsymbol{Z}_i')' = \boldsymbol{Y}_i$ and all vectors $\boldsymbol{Z}_i$'s can thus be viewed as empty.

These assumptions guarantee that all the directional quantiles considered below are uniquely defined and that the algorithms of [17] and [18] should theoretically work fine almost surely for all but a finite number of quantile levels τ's such as $\tau = i/n,\ i = 0, 1, 2, \ldots, n$, in the location case with unit weights. All the exceptional τ values and almost never occurring data configurations will be ignored in this section for the sake of simplicity and clarity. They will be recalled later in the discussion of practical applications of the package.

Analogous but different entities figuring in both directional approaches are denoted with the same symbol hereinafter to highlight the similarity of the two methods. It should not cause any confusion because the method should always be clear from the context if it is important for the validity of the claims being made.

Both the methods define, for each quantile level $\tau \in (0, 1)$ and each direction $\boldsymbol{u} \in \mathbb{R}^m \setminus \{\boldsymbol{0}\}$, the directional (regression) τ-quantile as one directional (regression) quantile hyperplane $\pi_{\tau\boldsymbol{u}}$ associated with its upper directional (regression) τ-quantile halfspace $\mathcal{H}_{\tau\boldsymbol{u}}^+$:

$$\pi_{\tau\boldsymbol{u}} = \big\{(\boldsymbol{y}^\top, \boldsymbol{z}^\top)^\top \in \mathbb{R}^{m+p-1} : \boldsymbol{b}_{\tau\boldsymbol{u}}^\top \boldsymbol{y} - \boldsymbol{a}_{\tau\boldsymbol{u}}^\top \boldsymbol{x} = 0,\ \boldsymbol{x} = (1, \boldsymbol{z}^\top)^\top\big\},$$
$$\mathcal{H}_{\tau\boldsymbol{u}}^+ = \big\{(\boldsymbol{y}^\top, \boldsymbol{z}^\top)^\top \in \mathbb{R}^{m+p-1} : \boldsymbol{b}_{\tau\boldsymbol{u}}^\top \boldsymbol{y} - \boldsymbol{a}_{\tau\boldsymbol{u}}^\top \boldsymbol{x} \geq 0,\ \boldsymbol{x} = (1, \boldsymbol{z}^\top)^\top\big\}.$$

The difference lies only in the quantile coefficient vector $(\boldsymbol{b}_{\tau\boldsymbol{u}}^\top, \boldsymbol{a}_{\tau\boldsymbol{u}}^\top)^\top \in \mathbb{R}^{m+p-1}$ that is defined as the solution to the same minimization problem up to a method-specific linear constraint:

$$(\boldsymbol{a}_{\tau\boldsymbol{u}}^\top, \boldsymbol{b}_{\tau\boldsymbol{u}}^\top)^\top = \underset{(\boldsymbol{a}^\top, \boldsymbol{b}^\top)^\top}{\operatorname{argmin}} \sum_{i=1}^{n} w_i \rho_\tau(\boldsymbol{b}^\top \boldsymbol{Y}_i - \boldsymbol{a}^\top \boldsymbol{X}_i)$$

subject to $\boldsymbol{b}^\top \boldsymbol{u} = 1$ (Method 1 of [9]) or $\boldsymbol{b} = \boldsymbol{u}$ (Method 2 of [16]) where $\rho_\tau(x) =$

$x\big(\tau - \mathrm{I}(x<0)\big)$ is the well-known quantile check function and

$$\Psi_{\tau\boldsymbol{u}} = \sum_{i=1}^{n} w_i \rho_\tau(r_{\tau\boldsymbol{u},i}) = \sum_{i=1}^{n} w_i \rho_\tau(\boldsymbol{b}_{\tau\boldsymbol{u}}^\top \boldsymbol{Y}_i - \boldsymbol{a}_{\tau\boldsymbol{u}}^\top \boldsymbol{X}_i)$$

stands for the optimal value of the objective function computed from the residuals $r_{\tau\boldsymbol{u},i} = \boldsymbol{b}_{\tau\boldsymbol{u}}^\top \boldsymbol{Y}_i - \boldsymbol{a}_{\tau\boldsymbol{u}}^\top \boldsymbol{X}_i$, $i = 1,\dots,n$. The constraint used by Method 1 is associated with the scalar Lagrange multiplier $\lambda_{\tau\boldsymbol{u}}$ equal to $\Psi_{\tau\boldsymbol{u}}$ while that of Method 2 results in the Lagrange multiplier vector $\mu_{\tau\boldsymbol{u}}^{\boldsymbol{b}}$ linked to $\Psi_{\tau\boldsymbol{u}}$ in a straightforward way: $\Psi_{\tau\boldsymbol{u}} = \mu_{\tau\boldsymbol{u}}^{\boldsymbol{b}\top}\boldsymbol{u}$. Note that Method 2 is more or less the ordinary τ-quantile regression of projections $\boldsymbol{u}^\top\boldsymbol{Y}_i$'s on $\boldsymbol{X}_i$'s.

Only unit weights $w_i = 1$, $i = 1,\dots,n$, are assumed from now on because the weighted case can be transformed to the unweighted one by substitutions $\boldsymbol{Y}_i := w_i\boldsymbol{Y}_i$ and $\boldsymbol{X}_i := w_i\boldsymbol{X}_i$ that change neither the optimal value of the objective function nor the quantile hyperplane coefficients.

The upper quantile halfspaces are important for defining two meaningful, convex, and polyhedral (regression) τ-quantile regions, namely the exact one ($\mathcal{R}_\tau^E$) and the approximate one ($\mathcal{R}_\tau^A$):

$$\mathcal{R}_\tau^E = \cap_{\pi_{\tau\boldsymbol{u}}\in\Pi_\tau} \mathcal{H}_{\tau\boldsymbol{u}}^+$$
$$\text{and}$$
$$\mathcal{R}_\tau^A = \operatorname{convhull}\big\{(\boldsymbol{Y}_i^\top, \boldsymbol{Z}_i^\top)^\top \in \mathbb{R}^{m+p-1} : (\boldsymbol{Y}_i^\top, \boldsymbol{Z}_i^\top)^\top \in \mathcal{R}_\tau^E\big\}$$

where Π_τ denotes the finite set of all distinct directional (regression) τ-quantile hyperplanes passing through exactly $m+p-1$ observations:

$$\Pi_\tau = \big\{\pi_{\tau\boldsymbol{u}} : \boldsymbol{u}\in\mathbb{R}^m,\ \|\boldsymbol{u}\| = 1,\ \pi_{\tau\boldsymbol{u}} \text{ contains } m+p-1 \text{ observations}\big\}.$$

In other words, $\mathcal{R}_\tau^A$ stands for the convex hull of all the observations contained in $\mathcal{R}_\tau^E$ where $\mathcal{R}_\tau^E$ is the intersection of all upper directional (regression) τ-quantile halfspaces with $m+p-1$ observations in their bordering directional (regression) τ-quantile hyperplanes. The borders of $\mathcal{R}_\tau^E$ and $\mathcal{R}_\tau^A$ are referred to as (exact) and approximate τ-quantile contours, respectively. In a general regression case, the $\boldsymbol{z}_0$-cuts $\mathcal{R}_\tau^E(\boldsymbol{z}_0)$ and $\mathcal{R}_\tau^A(\boldsymbol{z}_0)$:

$$\mathcal{R}_\tau^E(\boldsymbol{z}_0) = \big\{\boldsymbol{y}\in\mathbb{R}^m : (\boldsymbol{y}^\top, \boldsymbol{z}_0^\top)^\top \in \mathcal{R}_\tau^E\big\}$$
$$\text{and}$$
$$\mathcal{R}_\tau^A(\boldsymbol{z}_0) = \big\{\boldsymbol{y}\in\mathbb{R}^m : (\boldsymbol{y}^\top, \boldsymbol{z}_0^\top)^\top \in \mathcal{R}_\tau^A\big\},$$

if computed for various $\boldsymbol{z}_0 \in \mathbb{R}^{p-1}$, may provide valuable information about the trend and heteroscedasticity.

It is important to know that Π_τ, $\mathcal{R}_\tau^E$, and $\mathcal{R}_\tau^A$ do not depend on the directional quantile method used and that $\mathcal{R}_\tau^E$ must be non-empty (and thus contain at least one point of $\mathbb{R}^{m+p-1}$) for any $\tau \le 1/(m+p)$. The approximate region $\mathcal{R}_\tau^A$ asymptotically approaches the exact one and can be determined even if $\mathcal{R}_\tau^E$ is difficult to obtain due to the very large number of hyperplanes in Π_τ. It is because the observations in $\mathcal{R}_\tau^E$ are known even before computing its vertices and facets. Unlike density contours, the regions always remain convex, even if the distribution behind the observations is multimodal.

Such a distribution may arise unexpectedly even from very simple mixtures, see, e. g., [7].

Fortunately, parametric linear programming can solve exactly (for all $\boldsymbol{u} \in \mathbb{R}^m \setminus \{\boldsymbol{0}\}$) the minimization problems involved in both Method 1 and Method 2. It says that the space $\mathbb{R}^m \setminus \{\boldsymbol{0}\}$ of all directions can be partitioned into blunt polyhedral cones where the solution has a simple form and the observations with zero residuals do not change. Furthermore, the technique also produces, in each such cone, the Lagrange multipliers associated with the constraint and the residuals. Those multipliers corresponding to positive and negative residuals always equal $-\tau$ and $1-\tau$, respectively. The Lagrange multiplier vector $\mu^{\boldsymbol{r}_0}_{\tau\boldsymbol{u}}$ associated with zero residuals is more interesting as it must have all its data-dependent coordinates in $(-\tau, 1-\tau)$. It can be interpreted like rank scores in standard quantile regression and used for defining halfspace depth of individual observations as in (5.1) of [16].

The finite conic segmentation $\Gamma(\tau) = \{C_q(\tau) : q = 1, \dots N_\tau\}$ of $\mathbb{R}^m$ corresponding to Method 1 consists of non-degenerate closed convex polyhedral cones $C_q(\tau)$ where $\boldsymbol{a}_{\tau\boldsymbol{u}} = \boldsymbol{a}_{q,\tau}/d_{q,\tau}(\boldsymbol{u})$, $\boldsymbol{b}_{\tau\boldsymbol{u}} = \boldsymbol{b}_{q,\tau}/d_{q,\tau}(\boldsymbol{u})$, $\lambda_{\tau\boldsymbol{u}} = \lambda_{q,\tau}/d_{q,\tau}(\boldsymbol{u})$, $(\Psi_{\tau\boldsymbol{u}} = \lambda_{\tau\boldsymbol{u}})$, and $\mu^{\boldsymbol{r}_0}_{\tau\boldsymbol{u}} = \mathbb{V}_{q,\tau}\boldsymbol{u}/d_{q,\tau}(\boldsymbol{u})$ for any $\boldsymbol{0} \neq \boldsymbol{u} \in C_q(\tau)$ where $d_{q,\tau}(\boldsymbol{u}) = \boldsymbol{b}^{\top}_{q,\tau}\boldsymbol{u}$ and $\boldsymbol{b}_{q,\tau} \in \mathbb{R}^m$, $\boldsymbol{a}_{q,\tau} \in \mathbb{R}^p$, $\lambda_{q,\tau} \in \mathbb{R}$, and $\mathbb{V}_{q,\tau} \in \mathbb{R}_{(m+p-1)\times m}$ are constant up to their possible dependence on τ and q. All directions $\boldsymbol{u}$ inside $C_q(\tau)$ thus lead to the same hyperplane coming through $m+p-1$ observations although the hyperplane coefficients may differ by a multiplicative $\boldsymbol{u}$-dependent scaling factor.

Similarly, the finite conic segmentation $\Gamma(\tau) = \{C_q(\tau) : q = 1, \dots, N_\tau\}$ of $\mathbb{R}^m$ corresponding to Method 2 consists of non-degenerate closed convex polyhedral cones $C_q(\tau)$ where $\boldsymbol{a}_{\tau\boldsymbol{u}} = \mathbb{A}_{q,\tau}\boldsymbol{u}$, $\boldsymbol{b}_{\tau\boldsymbol{u}} = \boldsymbol{u}$, $\mu^{\boldsymbol{b}}_{\tau\boldsymbol{u}} = \mu^{\boldsymbol{b}}_{q,\tau}$, (and $\Psi_{\tau\boldsymbol{u}} = \mu^{\boldsymbol{b}\top}_{q,\tau}\boldsymbol{u}$), and $\mu^{\boldsymbol{r}_0}_{\tau\boldsymbol{u}} = \mu^{\boldsymbol{r}_0}_{q,\tau}$ for any $\boldsymbol{u} \in C_q(\tau)$ where $\mathbb{A}_{q,\tau} \in \mathbb{R}_{p\times m}$, $\mu^{\boldsymbol{b}}_{q,\tau} \in \mathbb{R}^m$, and $\mu^{\boldsymbol{r}_0}_{q,\tau} \in \mathbb{R}^p$ may depend on τ and q but not on $\boldsymbol{u}$. Each direction $\boldsymbol{u}$ inside $C_q(\tau)$ thus leads to a different hyperplane containing the same p observations. Any τ-quantile hyperplane passing through $m+p-1$ observations then corresponds to a vertex direction of some $C_q(\tau)$. Nevertheless, such directions may also be associated with τ-quantile hyperplanes having some of their coefficients zero and passing through less than $m+p-1$ observations. It is because two adjacent cones of $\Gamma(\tau)$ can sometimes differ only in the sign of one of the (regression) τ-quantile hyperplane coefficients and not necessarily in the p observations fitted by the hyperplanes associated with their inner directions.

As you probably realize, all the entities $\boldsymbol{b}_{\tau\boldsymbol{u}}$, $\boldsymbol{a}_{\tau\boldsymbol{u}}$, $\Psi_{\tau\boldsymbol{u}}$, $\mu^{\boldsymbol{r}_0}_{\tau\boldsymbol{u}}$, $\Gamma(\tau)$, and N_τ are method-dependent although the method does not explicitly appear in their notation.

The package modQR implements both methods and makes it possible both to find the conic segmentations with all the cone-wise quantile-related characteristics mentioned above and to compute and evaluate the (regression) quantile contours. It is described in the next section.

3. PACKAGE FUNCTIONALITY

The package modQR results from the Octave toolbox moQuantile as its faithful R translation. It has a detailed documentation expanding on the summary presented here.

The package contains seven functions for end users. The names of functions associated solely with Method 1 or Method 2 end with M1u or M2u, respectively. In other words, the functions compContourM2u, getCTechSTM2u, and getCharSTM2u are analogous to compContourM1u, getCTechSTM1u, and getCharSTM1u but relate to the second method. The remaining function evalContour analyses and evaluates any convex polytope given by a set of linear inequalities such as a (regression) quantile region.

Although the following text focuses only on Method 1, everything also applies to Method 2 after changing M1u to M2u, except for the explicitly mentioned differences.

The optimization problem behind Method 1 can be solved with compContourM1u. The function admits up to four input arguments: (1) the scalar quantile level τ, (2) the matrix with all the responses $\boldsymbol{Y}_i$'s in rows, $i = 1, \dots, n$, (3) (optionally) the matrix with all the corresponding regressors $\boldsymbol{X}_i$'s in rows, $i = 1, \dots, n$, and (4) (optionally) the list CTechST of parameters driving the computation (described two paragraphs below). If (3) is missing, then the unit vector of the right length is automatically considered. If (4) is missing, then the default list produced by getCTechSTM1u is used instead, which usually works well for common tasks and small to moderate data sets.

The function compContourM1u expects $\tau \in (0, 0.5)$, $n > m + p - 1$, and $2 \leq m \leq 6$, and it could be used reliably at least when the triple (m, n, p) is lexicographically smaller than $(2, 10000, 10)$, $(3, 500, 5)$, or $(4, 150, 3)$. That is to say that the computation increasingly tends to numerical errors and prohibitive computation times with growing m, n, p, and τ (denoted as M, N, P, and Tau in the documentation).

As for the argument CTechST, its fields determine such things as the amount of the output (BriefOutputI) and if it is stored in the files (OutSaveI), the output file names (OutFilePrefS), if some auxiliary information regarding the progress of the computation is displayed on the screen (ReportI), the function used for computing the output list field CharST (getCharST, equal to getCharSTM1u by default), and also the modifications of the algorithm (D2SpecI, CubRegWiseI, ...) and the storing mechanism (ArchAllFI) employed.

The output list resulting from compContourM1u includes some fields containing error and warning messages (CompErrMsgS, CTechSTMsgS, ProbSizeMsgS, TauMsgS) as well as some information about the problem size (NDQFiles, NumB, MaxLWidth) and computational reliability (NIniNone, NIniBad, NSkipCone). It also includes the field PosVec, which is the vector of length n describing the position of each observation with respect to the τ-quantile (regression) region (its ith coordinate usually equals $0/1/2$ if the ith observation lies in the interior/border/exterior of the region, $i = 1, \dots, n$). Last but not least, it contains the important list CharST (produced, by default, by getCharSTM1u passed to compContourM1u as the field getCharST of CTechST).

The list CharST bears some information about the preprocessing step discarding all the redundant artificially induced directions and about the reliability of the computation (NUESkip, NBZSkip, NAZSkip). If $m \leq 4$, then it also includes the matrix field HypMat with $m{+}p$ columns where only the coefficient vectors $(\boldsymbol{b}_{\tau\boldsymbol{u}}^\top, \boldsymbol{a}_{\tau\boldsymbol{u}}^\top)^\top$ with no zero coordinate of all distinct (regression) τ-quantile hyperplanes passing through $m{+}p{-}1$ observations are stored in rows after being normalized with $\|\boldsymbol{b}_{\tau\boldsymbol{u}}\|$ for Method 1, rounded, and sorted lexicographically. Then both Method 1 and Method 2 should lead to the same HypMat field.

Furthermore, CharST always includes two method-specific matrix fields CharMinMat and CharMaxMat that respectively contain (slightly rounded) minima and maxima of certain directional (regression) τ-quantile characteristics over all the (regression) τ-quantile hyperplanes without any zero coefficient and passing through $m+p-1$ observations. Each row of these matrices contains one such minimum or maximum in the last coordinate and one of the (L_2- or L_∞-normalized) directions $\boldsymbol{u}$ where it is attained in the preceding ones:

Method 1:

CharMaxMat =

$$\begin{pmatrix}
\boldsymbol{u}^\top & \max\|\boldsymbol{b}_{\tau\boldsymbol{u}}\| \\
\boldsymbol{u}^\top & \max\Psi_{\tau\boldsymbol{u}} \\
\boldsymbol{u}^\top & \max\bigl(\Psi_{\tau\boldsymbol{u}}/\|\boldsymbol{b}_{\tau\boldsymbol{u}}\|\bigr) \\
\boldsymbol{u}^\top & \max\bigl\|(a_{\tau\boldsymbol{u}}^{(2)},\dots,a_{\tau\boldsymbol{u}}^{(p)})^\top\bigr\| \\
\boldsymbol{u}^\top & \max\Bigl(\bigl\|(a_{\tau\boldsymbol{u}}^{(2)},\dots,a_{\tau\boldsymbol{u}}^{(p)})^\top\bigr\|/\|\boldsymbol{b}_{\tau\boldsymbol{u}}\|\Bigr) \\
\boldsymbol{u}^\top & \max|a_{\tau\boldsymbol{u}}^{(2)}| \\
\boldsymbol{u}^\top & \max\bigl(|a_{\tau\boldsymbol{u}}^{(2)}|/\|\boldsymbol{b}_{\tau\boldsymbol{u}}\|\bigr) \\
\cdots & \cdots \\
\boldsymbol{u}^\top & \max|a_{\tau\boldsymbol{u}}^{(p)}| \\
\boldsymbol{u}^\top & \max\bigl(|a_{\tau\boldsymbol{u}}^{(p)}|/\|\boldsymbol{b}_{\tau\boldsymbol{u}}\|\bigr)
\end{pmatrix}$$

CharMinMat =

$$\begin{pmatrix}
\boldsymbol{u}^\top & \min\|\boldsymbol{b}_{\tau\boldsymbol{u}}\| \\
\boldsymbol{u}^\top & \min\Psi_{\tau\boldsymbol{u}} \\
\boldsymbol{u}^\top & \min\bigl(\Psi_{\tau\boldsymbol{u}}/\|\boldsymbol{b}_{\tau\boldsymbol{u}}\|\bigr) \\
\boldsymbol{u}^\top & \min\bigl\|(a_{\tau\boldsymbol{u}}^{(2)},\dots,a_{\tau\boldsymbol{u}}^{(p)})^\top\bigr\| \\
\boldsymbol{u}^\top & \min\Bigl(\bigl\|(a_{\tau\boldsymbol{u}}^{(2)},\dots,a_{\tau\boldsymbol{u}}^{(p)})^\top\bigr\|/\|\boldsymbol{b}_{\tau\boldsymbol{u}}\|\Bigr)
\end{pmatrix}$$

Method 2:

CharMaxMat =

$$\begin{pmatrix}
\boldsymbol{u}^\top & \max\Psi_{\tau\boldsymbol{u}} \\
\boldsymbol{u}^\top & \max\|\mu_{\tau\boldsymbol{u}}^{\boldsymbol{b}}\| \\
\boldsymbol{u}^\top & \max\bigl\|(a_{\tau\boldsymbol{u}}^{(2)},\dots,a_{\tau\boldsymbol{u}}^{(p)})^\top\bigr\| \\
\boldsymbol{u}^\top & \max|a_{\tau\boldsymbol{u}}^{(2)}| \\
\cdots & \cdots \\
\boldsymbol{u}^\top & \max|a_{\tau\boldsymbol{u}}^{(p)}|
\end{pmatrix}$$

CharMinMat =

$$\begin{pmatrix}
\boldsymbol{u}^\top & \min\Psi_{\tau\boldsymbol{u}} \\
\boldsymbol{u}^\top & \min\|\mu_{\tau\boldsymbol{u}}^{\boldsymbol{b}}\| \\
\boldsymbol{u}^\top & \min\bigl\|(a_{\tau\boldsymbol{u}}^{(2)},\dots,a_{\tau\boldsymbol{u}}^{(p)})^\top\bigr\|
\end{pmatrix}$$

where $a_{\tau\boldsymbol{u}}^{(i)}$ stands for the ith component of $\boldsymbol{a}_{\tau\boldsymbol{u}}$, $i=1,\dots,p$. The last rows are included only if $p \geq 2$.

The users interested only in the (regression) quantile contours or their cuts will need only HypMat. They need not modify the default function getCharSTM1u if they do not intend to experiment with five- or six-dimensional responses when the HypMat field is not present by default due to its expected large size. That is to say that PosVec and CharST are included in the output list to make the file output and its processing as unnecessary as possible.

The function evalContour can compute and evaluate a polyhedral region or contour either from a user-defined input, or from the output of compContourM1u and compContourM2u. If the region is described by a set of inequalities, say $\mathbb{A}\boldsymbol{z} \leq \boldsymbol{b}$, and contains an interior point $\boldsymbol{v}$, then evalContour takes $\mathbb{A}$, $\boldsymbol{b}$, and (optionally) $\boldsymbol{v}$ as its input (denoted as AAMat, BBVec and IPVec in the documentation, respectively) and produces a list with the matrix (TVVMat) of clearly distinct contour vertices (in rows), the matrix (TKKMat) of clearly distinct elementary contour facets (in rows, described by means of the indices to their corner vertices in TVVMat), the number of clearly distinct contour vertices (NumV), the number of clearly distinct contour facets (NumF), and both the approximate volume (Vol) and area (Area) of the region. Of course, all the output is sensitive to the degree of rounding used for HypMat and TVVMat, and all that happens only if evalContour runs successfully and the output Status field is equal to zero. The information about the interior point improves the speed, accuracy, and reliability of the computation.

All the functions are thoroughly documented in the package and have their close counterparts in the moQuantile toolbox. The differences are nevertheless worth mentioning.

One of them stems from the fact that the SeDuMi optimization toolbox for MATLAB and Octave has not yet been ported to R and had to be replaced. The linear programming problems are therefore solved with function lp contained in the package *lpSolve* that lacks high flexibility, does not seem to support sparse matrices and works only with non-negative variables. The codes thus had to be changed accordingly, and the modification might affect their performance.

Also, all the structures and cell arrays have been changed to lists although their original names have been preserved for maximum similarity.

Next, the output list produced with evalContour contains the field Area unlike its counterpart in Octave. It states the (approximate) surface area of the resulting contour.

Furthermore, the Octave demo examples are translated rather vaguely to remain simple despite the different plotting tools available in R.

Finally, the separate Octave m-functions getCharSTM1u.m and getCharSTM2u.m have become fields of CTechST produced respectively by functions getCTechSTM1u and getCTechSTM2u so that the user could approach and modify them easily.

Both the functions should be comprehensible with the aid of [17] and [18], respectively. The articles describe the underlying algorithms and contain all the boring technical details too spacious to be repeated here.

In fact, all the method-specific functions can be studied side by side because they have been intentionally written in a way highlighting their similarities. Equal line numbers thus correspond to analogous lines (if there are any). Similarly, the lines of all R functions should match those of the corresponding m-functions in the Octave toolbox. The desire for maximum similarity explains why the guidelines for writing R extensions are not followed to the last dot, especially regarding naming conventions and maximum line length.

Potential problems with the computation are also described meticulously in the documentation. They may be caused by the bad choice of τ, bad configuration of data points, bad scale of the data, bad expectations (e. g., by unexpected computational time, quantile crossing, empty or unbounded (regression) contours, the absence of HypMat in CharST for $m \geq 4$), or by using the same file output names in different tasks. Of course, the models can also be designed or interpreted erroneously. The ways to handle all these issues are mentioned in the documentation. In particular, it is recommended to always perturb the data with some random noise of a reasonably small magnitude before the computation. One can also fight the troubles by means of weights, affine equivariance, or a tiny change of τ.

The next section shows how modQR can be used.

4. DEMONSTRATION

The package includes five demo examples ExampleA to ExampleE that focus on the essential functionality and avoid unnecessary or fancy features in order to be short and easy to understand. They should guide the reader through the most common applications, i. e., through the interpretation of the output resulting from evalContour and compContourM1u or compContourM2u (ExampleA) and through the computation and plotting of a few 2D location quantile contours (ExampleB), a 3D location quantile contour (ExampleC), and both parametric (ExampleD) and nonparametric (ExampleE) regression quantile contour cuts. Consequently, this section contains only a short illustrative interpretation of the output and a couple of motivational pictures.

The interpretation of the results generated by compContourM1u and evalContour can be elucidated with the short output of ExampleA. It only analyses $n = 7$ bivariate responses accompanied with one nontrivial regressor ($n = 7$, $m = 2$, $p = 2$, and $\tau = 0.20$) for the sake of illustration as any statistical analysis of such a small data set would be meaningless. The specific regression case leads to the following output list of compContourM1u (where the second column actually follows after the first one):

```
[1] "Method No 1:"

$CTechSTMsgS
[1] ""

$ProbSizeMsgS
[1] ""

$CompErrMsgS
[1] ""

$TauMsgS
[1] ""

$CharST
$CharST$CharMaxMat
            [,1]        [,2]      [,3]
[1,]  0.72125618 -0.69266841 1.5698165
[2,] -0.48469491  0.87468328 1.0504376
[3,] -0.01511641  0.99988574 0.9950834
[4,]  0.24441775  0.96967003 1.4791022
[5,]  0.99996117  0.00881287 1.1463858
[6,]  0.24441775  0.96967003 1.4791022
[7,]  0.99996117  0.00881287 1.1463858

$CharST$CharMinMat
           [,1]       [,2]       [,3]
[1,] -0.6600165 -0.7512511 1.00020333
[2,] -0.6600165 -0.7512511 0.64877013
[3,] -0.6600165 -0.7512511 0.51793198
[4,] -0.9922859 -0.1239703 0.03612446
[5,] -0.4846949  0.8746833 0.03225298

$CharST$NUESkip
[1] 0

$CharST$NAZSkip
[1] 0

$CharST$NBZSkip
[1] 0
```

```
$CharST$HypMat
            [,1]       [,2]        [,3]        [,4]
[1,] -0.94177775  0.3362360 -0.69880593  0.03225298
[2,] -0.64473516 -0.7644060 -0.77413249 -0.11421401
[3,] -0.17895976  0.9838564  0.14075498  0.13219642
[4,] -0.07449065 -0.9972217 -0.60544939 -0.41437394
[5,]  0.17836987  0.9839635 -0.45315479 -0.87225069
[6,]  0.80216550  0.5971017 -0.23850321  1.14638577
[7,]  0.81276492 -0.5825918 -0.52146711  0.17995888
[8,]  0.92072087  0.3902218  0.06224986  0.60158347

$PosVec
     [,1]
[1,]    1
[2,]    1
[3,]    1
[4,]    0
[5,]    2
[6,]    2
[7,]    1

$NDQFiles
[1] 1

$NumB
[1] 9

$MaxLWidth
[1] 1

$NIniNone
[1] 0

$NIniBad
[1] 0

$NSkipCone
[1] 0
```

The analysed problem was evidently small in size (NumB, MaxLWidth, NDQFiles). No warning/error messages (CTechSTMsgS, ProbSizeMsgS, CompErrMsgS, TauMsgS) indicate that the input was correct and that the computation probably ended successfully. Such an expectation is further supported with the facts that there were no problems with finding the initial solution(s) starting the algorithm (NIniNone, NIniBad) and that no almost degenerate cones have been encountered during the computation (NSkipCone).

It is also immediately apparent that two observations lie outside the τ-quantile region, four in its boundary, and the fourth one in its interior (PosVec).

The list field CharST collects the information regarding the (regression) τ-quantile hyperplanes. Obviously, there were no misleading hyperplanes (NUESkip) or potentially problematic hyperplanes with zero coefficients (NAZSkip, NBZSkip) to be excluded from HypMat. (Such hyperplanes would routinely occur for $m > 2$ or in Method 2, for example.) It also appears that eight distinct (regression) τ-quantile hyperplanes with $m + p - 1 = 3$ observations have been found. That is to say that their coefficient vectors $(\boldsymbol{b}_{\tau\boldsymbol{u}}^\top, \boldsymbol{a}_{\tau\boldsymbol{u}}^\top)^\top$ are normalized with $\|\boldsymbol{b}_{\tau\boldsymbol{u}}\|$, rounded, sorted lexicographically, and then recorded in the rows of HypMat. The first row of HypMat thus corresponds to the hyperplane $-0.94177775y_1 + 0.3362360y_2 + 0.69880593 - 0.03225298z = 0$.

Most importantly, the fields CharMaxMat and CharMinMat of CharST respectively contain not only the maxima and minima of some statistics over all the hyperplanes of HypMat (before their coefficients are normalized with $\|\boldsymbol{b}_{\tau\boldsymbol{u}}\|$), but also the directions when the extremes are attained (one direction for each extreme). For example, the first row of CharMaxMat reveals that $\max \|\boldsymbol{b}_{\tau\boldsymbol{u}}\| \doteq 1.5698$ in Method 1 corresponds (e. g.) to $\boldsymbol{u} \doteq (0.7213, -0.6927)^\top$.

The fields of the output list of evalContour follow (actually in one column):

```
$Status
[1] 0

$Vol
[1] 0.4213127

$NumF
[1] 10

$NumV
[1] 7

$TVVMat
           [,1]      [,2]       [,3]
[1,] -0.5768124 0.3290967 -0.7728145
[2,] -0.3984807 0.4661213 -0.4109964
[3,] -0.2681254 0.0276619 -0.4958983
[4,]  0.2290980 0.9483334  0.8623007
[5,]  0.3315653 0.2822571  0.5870709
[6,]  0.8351256 0.1756013 -0.8883929
[7,]  0.8634514 0.4400569  1.0414408
```

```
$TKKMat
      [,1] [,2] [,3]
 [1,]    6    4    1
 [2,]    6    4    7
 [3,]    5    4    7
 [4,]    5    6    7
 [5,]    2    4    1
 [6,]    2    5    4
 [7,]    3    6    1
 [8,]    3    5    6
 [9,]    3    2    1
[10,]    3    2    5

$Area
[1] 4.726156
```

They reveal some properties of the resulting (regression) τ-quantile region after its successful analysis (Status). It has (approximate) volume 0.421 (Vol), (approximate) surface area 4.726 (Area), 7 vertices (NumV), and 10 facets (NumF). The vertices are stored in the rows of TVVMat after being rounded and sorted lexicographically. The coordinates of the first vertex are thus roughly equal to $(-0.577, 0.329, -0.773)^\top$. The facets are stored in the rows of TKKMat. The ninth facet is thus defined by its corner vertices in the first three rows of TVVMat.

The code of ExampleA also shows how HypMat and PosVec might be used to find the input parameters AAMat, BBVec, and IPVec of evalContour, and how the regression τ-quantile contour can be plotted.

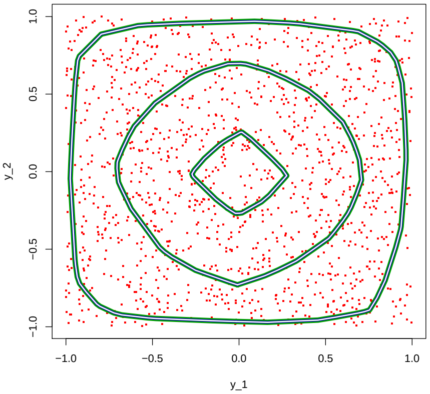

Fig. 1. The figure shows three bivariate location τ-quantile contours, $\tau = 0.3579, 0.1357$, and 0.0135, obtained from $n = 1357$ (red) independent random points coming from the model $(Y_1, Y_2)^\top \sim U([-1,1]^2)$. The contours were computed in three different ways leading to the same graphical output (blue, green, and white). Check ExampleB for all the technical details.

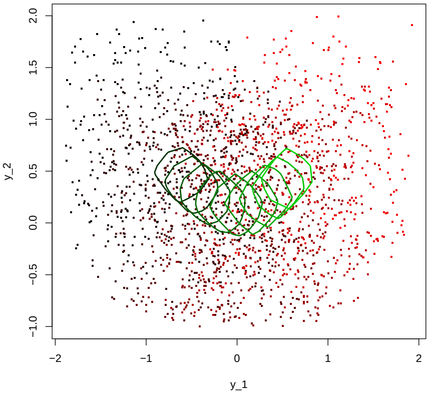

Fig. 2. Given $\tau = 0.35$, $x_0 = -0.8, -0.6, \ldots, 0.8$, and $n = 1\,999$ (red) bivariate random points coming from the model $(Y_1, Y_2)^\top = (X, X^2)^\top + \boldsymbol{\varepsilon}$ with independent $X \sim U([-1,1])$ and $\boldsymbol{\varepsilon} \sim U([-1,1]^2)$, this figure shows (green) x_0-cuts through the local constant (i. e., nonparametric) regression τ-quantile regions obtained for each x_0 with the aid of normal kernel weights corresponding to bandwidth 0.4. The cuts lighten with increasing x_0 and the points darken with decreasing regressor values. Check ExampleE for all the technical details.

The other examples are also worth consulting. They employ only the first method as both Method 1 and Method 2 should lead to the same graphical output. The second method would be used if one changed compContourM1u to compContourM2u and getCTechSTM1u to getCTechSTM2u everywhere in the codes. The graphical output generated by ExampleB and ExampleE is also included here in Figures 1 and 2 to convince the reader to give the multiple-output quantile regression a try. Figure 1 presents a bunch of 2D location quantile contours that sort the observations from the center outwards and identify possible outliers. They have been obtained in three different ways: (a) directly from the original data (green), (b) by using weights and replacing the observations surely in the interior of the computed quantile region with a single pseudo-observation (blue), and (c) by changing τ and deleting the observations surely in the interior of the computed quantile region (white). The last two ways (mentioned in [17] and [18]) compute the (nested) contours from the innermost one outwards.

Figure 2 shows the τ-quantile cuts through several fixed regression values that were obtained by means of the locally constant regression of [8]. They strongly (and rightly) suggest that the observations follow a homoscedastic model with a quadratic trend.

5. SPEED COMPARISON

It may be interesting to know how the R package (modQR) compares with the Octave toolbox (moQuantile) in terms of the computational speed. Table 1 shows the average execution times of default Method 1 (or, compContourM1u) based on 10 runs for

			τ					
			0.010		0.025		0.05	
m	p	n	Octave	R	Octave	R	Octave	R
2	1	249	0.342	0.092	0.560	0.170	0.638	0.235
		2499	2.103	1.962	3.699	2.790	5.835	3.880
	2	249	0.343	0.086	0.510	0.190	0.760	0.266
		2499	2.407	2.234	4.315	3.200	6.572	4.456
	4	249	0.432	0.102	0.672	0.241	0.967	0.344
		2499	3.313	2.814	5.593	4.247	8.831	6.123
	8	249	0.666	0.189	0.895	0.315	1.303	0.484
		2499	4.635	4.132	8.318	6.420	13.106	9.448
3	1	49	2.528	1.731	5.696	5.426	11.717	10.906
		249	23.710	23.730	101.353	119.872	274.294	308.170
	2	49	4.139	3.440	5.222	4.546	12.182	11.567
		249	28.673	33.863	115.514	137.929	329.735	380.401
	4	49	9.800	9.117	9.882	9.307	15.835	15.333
		249	41.382	47.746	166.286	195.949	464.829	525.607

Tab. 1. Average execution times of default Method 1 for uniformly distributed responses and (non-intercept) regressors, obtained on a computer with processor Intel I7-2600 3400GHz and 16GB RAM.

uniformly distributed responses and regressors (except for the first unit regressor) and for some representative values of τ, m, n, and p. The codes were executed on a decent computer (processor Intel I7-2600 3400Ghz, 16GB RAM) with Windows 7, Octave 3.8.2 and R 3.3.1.

Apparently, modQR is faster than moQuantile for small problems and loses its superiority as the problem size grows, at least within the scope of this simulation study.

ACKNOWLEDGMENTS

The research work of Pavel Boček and Miroslav Šiman was supported by the grant GA14-07234S from the Czech Science Foundation.

(Received March 3, 2016)

REFERENCES

[1] P. Boček and M. Šiman: modQR: Multiple-Output Directional Quantile Regression. R package version 0.1.0, 2015.

[2] P. Boček and M. Šiman: Directional quantile regression in Octave and MATLAB. Kybernetika *52* (2016), 28–51. DOI:10.14736/kyb-2016-1-0028

[3] B. Chakraborty: On multivariate quantile regression. J. Statist. Planning Inference *110* (2003), 109–132. DOI:10.1016/s0378-3758(01)00277-4

[4] I. Charlier, D. Paindaveine, and J. Saracco: Multiple-output regression through optimal quantization. ECARES Working Paper 2016-18.

[5] P. Chaudhury: On a geometric notion of quantiles for multivariate data. J. Amer. Stat. Assoc. *91* (1996), 862–872. DOI:10.2307/2291681

[6] Y. Cheng and J. G. De Gooijer: On the uth geometric conditional quantile. J. Statist. Planning Inference *137* (2007), 1914–1930. DOI:10.1016/j.jspi.2006.02.014

[7] Š. Došlá: Conditions for bimodality and multimodality of a mixture of two unimodal densities. Kybernetika *45* (2009) 279–292.

[8] M. Hallin, Z. Lu, D. Paindaveine, and M. Šiman: Local bilinear multiple-output quantile/depth regression. Bernoulli *21* (2015), 1435–1466. DOI:10.3150/14-bej610

[9] M. Hallin, D. Paindaveine, and M. Šiman: Multivariate quantiles and multiple-output regression quantiles: From L_1 optimization to halfspace depth. Ann. Statist. *38* (2010), 635–669. DOI:10.1214/09-aos723

[10] M. Hallin, D. Paindaveine, and M. Šiman: Rejoinder. Ann. Statist. *38* (2010), 694–703. DOI:10.1214/09-aos723rej

[11] R. Koenker: Quantile Regression. Cambridge University Press, New York 2005. DOI:10.1017/cbo9780511754098

[12] R. Koenker and G. J. Bassett: Regression quantiles. Econometrica *46* (1978), 33–50. DOI:10.2307/1913643

[13] V. Koltchinskii: M-estimation, convexity and quantiles. Ann. Statist. *25* (1997), 435–477. DOI:10.1214/aos/1031833659

[14] L. Kong and I. Mizera: Quantile tomography: Using quantiles with multivariate data. Statistica Sinica *22* (2012), 1589–1610. DOI:10.5705/ss.2010.224

[15] I. W. McKeague, S. López-Pintado, M. Hallin, and M. Šiman: Analyzing growth trajectories. J. Developmental Origins of Health and Disease *2* (2011), 322–329. DOI:10.1017/s2040174411000572

[16] D. Paindaveine and M. Šiman: On directional multiple-output quantile regression. J. Multivariate Anal. *102* (2011), 193–212. DOI:10.1016/j.jmva.2010.08.004

[17] D. Paindaveine and M. Šiman: Computing multiple-output regression quantile regions. Comput. Statist. Data Anal. *56* (2012), 840–853. DOI:10.1016/j.csda.2010.11.014

[18] D. Paindaveine and M. Šiman: Computing multiple-output regression quantile regions from projection quantiles. Comput. Statist. *27* (2012), 29–49. DOI:10.1007/s00180-011-0231-y

[19] M. Šiman: On exact computation of some statistics based on projection pursuit in a general regression context. Commun. Statist. – Simulation and Computation *40* (2011), 948–956. DOI:10.1080/03610918.2011.560730

[20] M. Šiman: Precision index in the multivariate context. Commun. Statist. – Theory and Methods *43* (2014), 377–387. DOI:10.1080/03610926.2012.661509

Pavel Boček, Institute of Information Theory and Automation, The Czech Academy of Sciences, Pod Vodárenskou věží 4, 182 08 Praha 8. Czech Republic.
e-mail: bocek@utia.cas.cz

Miroslav Šiman, Institute of Information Theory and Automation, The Czech Academy of Sciences, Pod Vodárenskou věží 4, 182 08 Praha 8. Czech Republic.