Bi-Level Path Following for Cross Validated Solution of Kernel Quantile Regression

Saharon Rosset SAHARON@POST.TAU.AC.IL
Department of Statistics and Operations Research, The Raymond and Beverly Sackler School of Mathematical Sciences, Tel Aviv University, Israel

Abstract

Modeling of conditional quantiles requires specification of the quantile being estimated and can thus be viewed as a parameterized predictive modeling problem. Quantile loss is typically used, and it is indeed parameterized by a quantile parameter. In this paper we show how to follow the path of cross validated solutions to regularized kernel quantile regression. Even though the bi-level optimization problem we encounter for every quantile is non-convex, the manner in which the optimal cross-validated solution evolves with the parameter of the loss function allows tracking of this solution. We prove this property, construct the resulting algorithm, and demonstrate it on data. This algorithm allows us to efficiently solve the whole family of bi-level problems.

1. Introduction

In the standard predictive modeling setting, we are given a *training sample* of n examples $\{\mathbf{x}_1, y_1\}, ..., \{\mathbf{x}_n, y_n\}$ drawn i.i.d from a joint distribution $P(X, Y)$, with $\mathbf{x}_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$ for regression, $y_i \in \{0, 1\}$ for two-class classification. We aim to utilize these data to build a model $\hat{Y} = \hat{f}(X)$ to describe the relationship between X and Y, and later use it to predict the value of Y given new X values. This is often done by defining a family of models $\mathcal{F}$ and finding (exactly or approximately) the model $f \in \mathcal{F}$ which minimizes an *empirical loss function*: $\sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i))$. Examples of such algorithms include linear and logistic regression, empirical risk minimization in classification and others.

Appearing in *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, 2008.

If $\mathcal{F}$ is complex, it is often desirable to add *regularization* to control model complexity and overfitting. The generic regularized optimization problem can be written as:

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) + \lambda J(f)$$

where $J(f)$ is an appropriate model complexity penalty and λ is the regularization parameter. Given a loss and a penalty, selection of a good value of λ is a *model selection* problem. Popular approaches that can be formulated as regularized optimization problems include all versions of support vector machines, ridge regression, the Lasso and many others. For an overview of predictive modeling, regularized optimization and the algorithms mentioned above, see for example Hastie et al. (2001).

In this paper we are interested in a specific setup where we have a family of regularized optimization problems, defined by a parameterized loss function and a regularization term. A major motivating example for this setting is regularized quantile regression (Koenker, 2005):

$$\hat{\beta}(\tau, \lambda) = \arg\min_{\beta} \sum_{i=1}^{n} L_\tau(y_i - \beta^\mathsf{T} \mathbf{x}_i) + \lambda \|\beta\|_q^q \quad (1)$$
$$\text{for } 0 < \tau < 1,\ 0 \leq \lambda < \infty$$

where L_τ, the parameterized quantile loss function, has the form:

$$L_\tau(r) = \begin{cases} r\tau & r \geq 0 \\ -r(1-\tau) & r < 0 \end{cases}$$

and is termed *τ-quantile loss* because its population optimizer is the appropriate quantile (Koenker, 2005):

$$\arg\min_{c} E(L_\tau(Y - c)|X) = \text{quantile } \tau \text{ of } P(Y|X) \quad (2)$$

Because quantile loss has this optimizer, the solution of the quantile regression problems for the whole range

$0 < \tau < 1$ has often been advocated as an approach to estimating the full conditional probability of $P(Y|X)$ (Koenker, 2005; Perlich et al., 2007). Much of the interesting information about the behavior of $Y|X$ may lie in the details of this conditional distribution, and if it is not *nicely behaved* (i.i.d Gaussian noise being the most commonly used concept of nice behavior), just estimating a conditional mean or median is often not sufficient to properly understand and model the mechanisms generating Y. The importance of estimating a complete conditional distribution, and not just a central quantity like the conditional mean, has long been noted and addressed in various communities, like Econometrics, Education and Finance (Koenker, 2005; Buchinsky, 1994; Eide & Showalter, 1998). There has also been a surge of interest in the Machine Learning community in conditional quantile estimation in recent years (Meinshausen, 2006; Takeuchi et al., 2006). Figure 1 shows a graphical representation of L_τ for several values of τ, and a demonstration of the conditional quantile curves in a univariate regression setting, where the linear model is correct for the median, but the noise has a non-homoscedastic distribution.

On the penalty side, we typically use the ℓ_q norm of the parameters with $q \in \{1, 2\}$. Adding a penalty can be thought of as shrinkage, complexity control or putting a prior to express our expectation that the β's should be small.

As has been noted in the literature (Rosset & Zhu, 2007; Hastie et al., 2004; Li et al., 2007) if $q \in \{1, 2\}$ and if we fix $\tau = \tau_0$, we can devise *path following* (AKA parametric programming) algorithms to efficiently generate the 1-dimensional curve of solutions $\{\hat{\beta}(\tau_0, \lambda) \ : \ 0 \leq \lambda < \infty\}$. Although it has not been explicitly noted by most of these authors, it naturally follows that similar algorithms exist for the case that we fix $\lambda = \lambda_0$ and are interested in generating the curve $\{\hat{\beta}(\tau, \lambda_0) \ : \ 0 < \tau < 1\}$.

In addition to parameterized quantile regression, there are other modeling problems in the literature which combine a parameterized loss function problem with the existence of efficient path following algorithms. These include Support vector regression (SVR, Smola and Schölkopf (2004), see Gunther and Zhu (2005) for path following algorithm) with ℓ_1 or ℓ_2 regularization, where the parameter ϵ determines the width of the *don't care* region around 0.

An important extension of the ℓ_2-regularized optimization problem is to *non-linear* fitting through kernel embedding (Schölkopf & Smola, 2002). The kernelized version of problem (1) is:

$$\hat{f}(\tau, \lambda) = \arg\min_f \sum_i L_\tau(y_i - f(\mathbf{x}_i)) + \frac{\lambda}{2}\|f\|^2_{\mathcal{H}_K} \quad (3)$$

where $\|\cdot\|_{\mathcal{H}_K}$ is the norm induced by the positive-definite kernel K in the Reproducing Kernel Hilbert Space (RKHS) it generates. The well known *representer theorem* (Kimeldorf & Wahba, 1971) implies that the solution of problem (3) lies in a low dimensional subspace spanned by the representer functions $\{K(\cdot, \mathbf{x}_i), i \in 1, ..., n\}$. Following the ideas of Hastie et al. (2004) for the support vector machine, Li et al. (2007) have shown that λ-path of solutions to problem (3) when τ is fixed can also be efficiently generated.

It is important to note the difference in the roles of the two parameters τ, λ. The former defines a family of loss functions, in our case leading to estimation of different quantiles. Thus we would typically want to build and use a model for every value of τ. The latter is a regularization parameter, controlling model complexity with the aim of generating a better model and avoiding overfitting, and is not part of the prediction objective (at least as long as we avoid the Bayesian view). We would therefore typically want to generate a set of models $\beta^*(\tau)$ (or $f^*(\tau)$ in the kernel case), by selecting a good regularization parameter $\lambda^*(\tau)$ for every value of τ, thus obtaining a family of good models for estimating the range of conditional quantiles, and consequently the whole conditional distribution.

This problem, of model selection to find a good regularization parameter, is often handled through *cross-validation*. In its simplest form, cross-validation entails having a second, independent set of data $\{\tilde{\mathbf{x}}_i, \tilde{y}_i\}_{i=1}^N$ (often referred to as a *validation set*), which is used to evaluate the performance of the models and select a good regularization parameter. For a fixed τ, we can write our model selection problem as a *Bi-level programming* extension of problems (1, 3), where $f^*(\tau) = \hat{f}(\tau, \lambda^*)$ and λ^* solves:

$$\begin{aligned} \min_{\lambda} \quad & \sum_{i=1}^{N} L_\tau(\tilde{y}_i, \hat{f}(\tau, \lambda)^\mathsf{T}\tilde{\mathbf{x}}_i) \\ & \text{s.t.} \quad \hat{f}(\tau, \lambda) \text{ solves problem (3)} \end{aligned} \quad (4)$$

The objective of this minimization problem is not convex as a function of λ. A similar non-convex optimization problem has been tackled by Kunapuli et al. (2007). The fundamental difference between their setting and ours is that they had a single bi-level optimization problem, while we have a family of such problems, parameterized by τ. This allows us to take advantage of internal structure to solve the bi-level

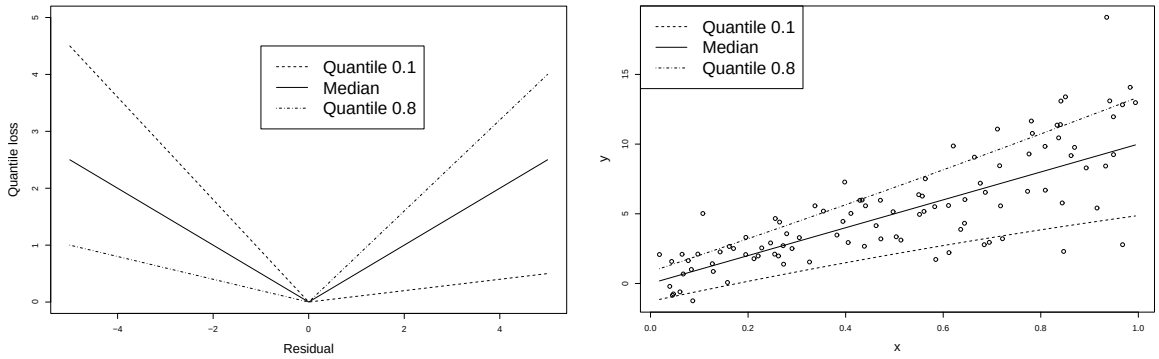

Figure 1. Quantile loss function for some values of τ (left) and an example where the median of Y is linear in X but the quantiles of $P(Y|X)$ are not because the noise is not identically distributed (right).

problem for all values of τ *simultaneously* (or more accurately, in one run of our algorithm).

The concept of a parameterized family of bi-level regularized quantile regression problems is demonstrated in Figure 2, where we see the cross-validation curves of the objective of (4) as a function of λ for several values of τ on the same dataset. As we can see, the optimal level of regularization varies with the quantile, and correct choice of the regularization parameter can have a significant effect on the success of our quantile prediction model.

The main goal of this paper is to devise algorithms for following the bi-level optimal solution path $f^*(\tau)$ as a function of τ, and demonstrate their practicality. We show that this non-convex family of bi-level programs can be solved exactly, as the optimum among the solutions of $O(n+N)$ standard (convex) path-following problems, with some additional twists. This result is based on a characterization of the evolution of the solution path $\hat{f}(\tau,\cdot)$ as τ varies, and on an understanding of the properties of optimal solutions of the bi-level problem, which can only occur at a limited set of points. We combine these insights to formulate an actual algorithm for solving this family of bi-level programs via path-following.

The rest of this paper is organized as follows. In Section 2 we discuss the properties of the quantile regression solution paths $\hat{f}(\tau,\lambda)$ and their evolution as τ changes. We then discuss in Section 3 the properties of the bi-level optimization problem (4) and demonstrate that the solutions change predictably with τ. Bringing together all our insights leads us to formulate an algorithm in Section 4, which allows us to follow the path of solutions $\{f^*(\tau)\ ,\ 0<\tau<1\}$ and only requires following a large but manageable number of solution paths of problem (3) simultaneously. We demonstrate our methods with a simulated data study in Section 5, where we demonstrate the computational efficiency of our approach and the ability of KQR to capture non-standard conditional distributions $P(Y|X)$.

This paper is a short version of Rosset (2008), and we defer the proofs, some of the technical details and much of the discussion to that version.

2. Quantile regression solution paths

We concentrate our discussion on the kernel quantile regression (KQR) formulation in (3), with the understanding that it subsumes the linear formulation (1) with ℓ_2 regularization by using the *linear* kernel $K(\mathbf{x},\tilde{\mathbf{x}})=\mathbf{x}^\mathsf{T}\tilde{\mathbf{x}}$.

We briefly survey the results of Li et al. (2007) regarding the properties of $\hat{f}(\tau,\cdot)$, the optimal solution path of (3), with τ fixed. The representer theorem (Kimeldorf & Wahba, 1971) implies that the solution can be written as:

$$\hat{f}(\tau,\lambda)(\mathbf{x})=\frac{1}{\lambda}\left[\hat{\beta}_0+\sum_{i=1}^{n}\hat{\theta}_i K(\mathbf{x},\mathbf{x}_i)\right] \qquad (5)$$

For a proposed solution $f(\mathbf{x})$ define:

- $\mathcal{E}=\{i: y_i-f(\mathbf{x}_i)=0\}$ (points on *elbow* of L_τ)
- $\mathcal{L}=\{i: y_i-f(\mathbf{x}_i)<0\}$ (left of elbow)
- $\mathcal{R}=\{i: y_i-f(\mathbf{x}_i)>0\}$ (right of elbow)

Then Li et al. (2007) show that the Karush-Kuhn-Tucker (KKT) conditions for optimality of a solution $\hat{f}(\tau,\lambda)$ of problem (3) can be phrased as:

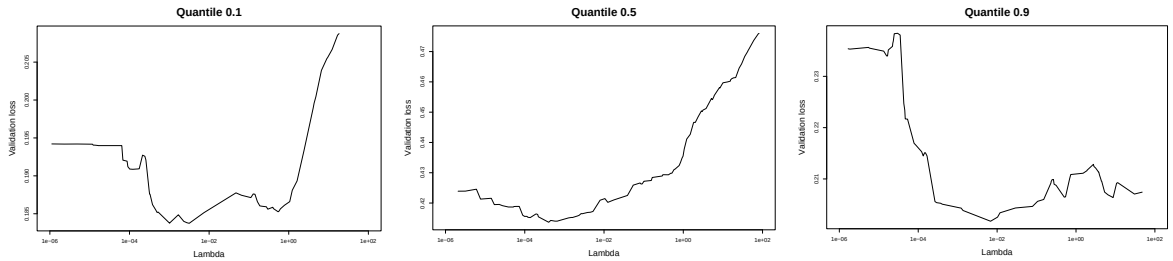

Figure 2. Estimated prediction error curves of Kernel Quantile Regression for some quantiles on one dataset. The errors are shown as a function of the regularization parameter λ

- $i \in \mathcal{E} \;\Rightarrow\; -(1-\tau) \leq \hat{\theta}_i \leq \tau$
- $i \in \mathcal{L} \;\Rightarrow\; \hat{\theta}_i = -(1-\tau)$
- $i \in \mathcal{R} \;\Rightarrow\; \hat{\theta}_i = \tau$
- $\sum_i \hat{\theta}_i = 0$

with some additional algebra they show that for a fixed τ, there is a series of *knots*, $0 = \lambda_0 < \lambda_1 < ... < \lambda_m < \infty$ such that for $\lambda \geq \lambda_m$ we have $\hat{f}(\tau, \lambda) =$ constant and for $\lambda_{k-1} < \lambda \leq \lambda_k$ we have:

$$\hat{f}(\tau,\lambda)(\mathbf{x}) = \frac{1}{\lambda}\left(\lambda_k \hat{f}(\tau,\lambda_k)(\mathbf{x}) + (\lambda - \lambda_k) h_k(\mathbf{x})\right) \quad (6)$$

where $h_k(\mathbf{x}) = b_0^k + \sum_{i \in \mathcal{E}_k} b_i^k K(\mathbf{x}, \mathbf{x}_i)$ can be thought of as the *direction* in which the solution is moving for the region $\lambda_{k-1} < \lambda \leq \lambda_k$. The knots λ_k are points on the path where an observation passes between $\mathcal{E}$ and either $\mathcal{L}$ or $\mathcal{R}$, that is $\exists i \in \mathcal{E}$ such that exactly $\theta_i = \tau$ or $\theta_i = -(1-\tau)$. These insights lead Li et al. (2007) to an algorithm for incrementally generating $\hat{f}(\tau, \lambda)$ as a function of λ for fixed τ, starting from $\lambda = \infty$ (where the solution only contains the intercept β_0).

Although Li et al. (2007) suggest it is a topic for further study, it is in fact a reasonably straight forward extension of their results to show that a similar scenario holds when we fix λ and allow τ only to change, and also when both τ, λ are changing together *along a straight line*, i.e., a 1-dimensional subspace of the (τ, λ) space (this has been observed by Wang et al. (2006) for SVR, which is very similar from an optimization perspective). The explicit result is given in (Rosset, 2008), but we omit it here for brevity, given its marginal novelty.

Armed with this result, we next show the main result of this section: that the knots themselves move in a (piecewise) straight line as τ changes, and can therefore be *tracked* as τ and the regularization path change. Fix a quantile τ_0 and assume that λ_k is a knot in the λ-solution path for quantile τ_0. Further, let i_k be the observation that is passing in or out of the elbow at knot λ_k. Assume WLOG that $\hat{\theta}_{i_k}(\tau_0, \lambda_k) = \tau_0$, i.e. it is on the boundary between $\mathcal{R}_k$ and $\mathcal{E}_k$. Let $\tilde{\mathbf{K}}_{\mathcal{E}_k}$ be the matrix $\mathbf{K}_{\mathcal{E}_k}$ with the i_k column removed, and $\tilde{\mathbf{b}}^k = \mathbf{b}^k$ with index i_k removed. Let $s_i = \sum_{j \in \mathcal{R} \cup \mathcal{L} \cup \{i_k\}} K(\mathbf{x}_i, \mathbf{x}_j)$ for $i \in \mathcal{E}_k$. Let $\mathbf{s}_{\mathcal{E}_k}$ be the vector of all these values.

Theorem 1 *Any knot λ_k moves linearly as τ changes. That is, there exists a constant c_k such that for quantile $\tau_0 + \delta$ there is a knot in the λ-solution path at $\lambda_k + c_k\delta$, for $\delta \in [-\epsilon_k, \nu_k]$, a non-empty neighborhood of 0. c_k is determined through the solution of another set of $|\mathcal{E}_k|+1$ linear equations with $|\mathcal{E}_k|+1$ unknowns:*

$$\mathbf{B}^k \begin{pmatrix} \tilde{\mathbf{b}}^k \\ c_k \end{pmatrix} = \begin{pmatrix} -(|\mathcal{R}| + |\mathcal{L}| + 1) \\ -\mathbf{s}_{\mathcal{E}_k} \end{pmatrix}$$

with

$$\mathbf{B}^k = \begin{pmatrix} 0 & \mathbf{1}^\mathsf{T} & 0 \\ \mathbf{1} & \tilde{\mathbf{K}}_{\mathcal{E}_k} & -\mathbf{y}_{\mathcal{E}_k} \end{pmatrix}$$

And the fit at this knot progresses as:

$$\hat{f}(\tau_0 + \delta, \lambda_k + c_k\delta) = \quad (7)$$
$$= \frac{1}{\lambda_k + c_k\delta}\left(\lambda_k \hat{f}(\lambda_k, \tau_0)(\mathbf{x}) + \delta h_k(\mathbf{x})\right)$$
$$h_k(\mathbf{x}) = \tilde{b}_0^k + \sum_{i \in \mathcal{E}_k - i_k} \tilde{b}_i^k K(\mathbf{x}, \mathbf{x}_i) + \quad (8)$$
$$+ \sum_{i \in \mathcal{L} \cup \mathcal{R} \cup \{i_k\}} K(\mathbf{x}, \mathbf{x}_i)$$

This theorem tells us that we can in fact track the knots in the solution efficiently as τ changes. We still have to account for various types of *events* that can change the direction the knot is moving in. The value θ_i for a point in $\mathcal{E}_k - \{i_k\}$ can reach τ or $-(1-\tau)$, or a point in $\mathcal{L} \cup \mathcal{R}$ may reach the elbow $\mathcal{E}$. These events correspond to *knots crossings*, i.e., the knot λ_k is encountering another knot (which is tracking the other event). There are also *knot birth* events, and *knots*

merge events, which are possible but rare, and somewhat counter-intuitive. The details of how these are identified and handled can be found in the detailed algorithm description (Rosset, 2008). When any of these events occurs, the set of knots has to be updated and their directions have to be re-calculated using Theorem 1 and the new identity of the sets $\mathcal{E}, \mathcal{L}, \mathcal{R}$ and the observation i_k. This in essence allows us to map the whole 2-dimensional solution surface $\hat{f}(\tau, \lambda)$.

3. The bi-level optimization problem

Our next task is to show how our ability to track the knots as τ changes allows us to track the solution of the bi-level optimization problem (4), as τ changes. The key to this step is the following result.

Theorem 2 *Any minimizer*[1] *of (4) is always either at a knot in the λ-path for this τ or a point where a validation observation crosses the elbow. In other words, one of the two following statements must hold:*

- *λ^* is a knot: $\exists i \in \{1...n\}$ s.t. $\hat{f}(\tau, \lambda^*(\tau))(\mathbf{x}_i) = y_i$ and $\theta_i \in \{\tau, -(1-\tau)\}$, or*
- *λ^* is a* validation crossing*: $\exists i \in \{1...N\}$ s.t. $\hat{f}(\tau, \lambda^*(\tau))(\tilde{\mathbf{x}}_i) = \tilde{y}_i$*

Corollary 1 *Given the complete solution path for $\tau = \tau_0$, the solutions of the bi-level problem (4) for a range of quantiles around τ_0 can be obtained by following the paths of the knots and the validation crossings only, as τ changes.*

To implement this corollary in practice, we have two main issues to resolve: 1. How do we follow the paths of the validation crossings? 2. How do we determine which one of the knots and validation crossings is going to be optimal for every value of τ?

The first question is easy to answer when we consider the similarity between the knot following problem we solve in Theorem 1 and the validation crossing following problem. In each case we have a set of *elbow* observations whose fit must remain fixed as τ changes, but whose $\hat{\theta}$ values may vary; sets $\mathcal{L}, \mathcal{R}$ whose $\hat{\theta}$ are changing in a pre-determined manner with τ, but whose fit may vary freely; and one special observation which *characterizes* the knot or validation crossing. The only difference is that in a knot this is a *border* observation from the training set, so both its fit and its $\hat{\theta}$ are predetermined, while in the case of validation crossing it is a *validation* observation, whose fit must remain fixed (at the *elbow*), but which does not even have a $\hat{\theta}$ value. Taking all of this into account, it is easy to show a result similar to Theorem 1 for the validation crossings. We refer the reader to Rosset (2008) for the details.

The second question we have posed requires us to explicitly express the validation loss (i.e., L_τ on the validation set) at every knot and validation crossing in terms of δ, so we can compare them and identify the optimum at every value of δ. Using the representation in (7) we can write the *validation loss* for a knot k :

$$\sum_{i=1}^{N} \quad L_\tau(\tilde{y}_i, \hat{f}(\tau_0 + \delta, \lambda_k + c_k\delta)(\tilde{\mathbf{x}}_i)) = \ldots = \\ = \frac{\text{quadratic in } \delta}{\lambda_k + c_k\delta} \tag{9}$$

see Rosset (2008) for details, and note that similar expressions can naturally be derived for validation crossings. These are rational functions of δ with quadratic expressions in the numerator and linear expressions in the denominator. Our cross-validation task can be re-formulated as the identification of the minimum of these rational functions among all knots and validation crossings, for every value of τ in the current *segment*, where the directions h_k, h_v of all knots and validation crossings are fixed (and therefore so are the coefficients in the rational functions). This is a *lower-envelope* tracking problem, which has been extensively studied in the literature (Sharir and Agarwal (1995) and references therein).

To calculate the meeting point of two elements with neighboring scores we find the zeros of the cubic equation obtained by requiring equality for two rational functions of the form (9). The smallest non-negative solution for δ is the one we are interested in.

4. Algorithm overview

Bringing together all the elements from the previous sections, we now give (Algorithm 1) a succinct overview of the resulting algorithm. Since there is a multitude of details, we defer a detailed pseudo-code description of our algorithm to Rosset (2008).

The algorithm follows the knots of the λ-solution path as τ changes using the results of Section 2, and keeps track of the cross-validated solution using the results of Section 3. Every time an *event* happens (like a knot crossing), the direction in which two of the knots are moving has to be changed, or knots have to be added or deleted. Between these events, the evolution of the cross-validation objective at all knots and validation crossings has to be sorted and followed. Their order is maintained, and updated whenever crossings occur

[1] In pathological cases there may be a "segment" of minimizers. In this case it can be shown that such a segment will always be flanked by points described in the theorem.

Algorithm 1 Main steps of our bi-level path following algorithm

Input: The entire λ-solution path for quantile τ_0; the bi-level optimizer $\lambda^*(\tau_0)$
Output: Cross-validated solutions $f^*(\tau)$ for $\tau \in [\tau_0, \tau_{\text{end}}]$
Initialization: identify all knots and validation crossings in the solution path for τ_0;
Find direction of each knot according to Theorem 1
Find direction of each validation crossing
Create a list M of knots and validation crossings sorted by their validation loss
Let $\lambda^*(\tau_0)$ be the one at the bottom of the list M, and $f^*(\tau_0)$ accordingly
Calculate future meeting of each pair of neighbors in M by solving the cubic equation implied by (9)
Set $\tau_{\text{now}} = \tau_0$
while $\tau_{\text{now}} < \tau_{\text{end}}$ **do**
  Find value $\tau_1 > \tau_{\text{now}}$ where first knot crossing occurs
  Find value $\tau_2 > \tau_{\text{now}}$ where first knot merge occurs
  Find value $\tau_3 > \tau_{\text{now}}$ where first knot birth occurs
  Set $\tau_{\text{new}} = \min(\tau_1, \tau_2, \tau_3)$
  while $\tau_{\text{now}} < \tau_{\text{new}}$ **do**
    Find value $\tau_4 > \tau_{\text{now}}$ where first future meeting (order change) in M occurs
    Find value $\tau_5 > \tau_{\text{now}}$ where first validation crossing birth occurs
    Find value $\tau_6 > \tau_{\text{now}}$ where first validation crossing cancelation occurs
    Set $\tau_{\text{next}} = \min(\tau_4, \tau_5, \tau_6)$
    Update $\lambda^*(\tau), f^*(\tau)$ for $\tau \in (\tau_{\text{now}}, \tau_{\text{next}})$ as the evolution of the knot or validation crossing attaining the minimal L_τ in M (i.e., the one at $\lambda^*(\tau_{\text{now}})$)
    Update M according to the first event (order change, birth, cancelation)
    Update the future meetings of the affected elements using (9)
    Set $\tau_{\text{now}} = \tau_{\text{next}}$
  end while
  Update the list of knots according to the first event (knot crossing, birth, merge)
  Update the directions of affected knots using Theorem 1
end while

between them.

4.1. Approximate computational complexity

Looking at Algorithm 1, we should consider the number of steps of the two loops and the complexity of the operations inside the loops. Even for a "standard" λ-path following problem for fixed τ, it is in fact impossible to rigorously bound the number of steps in the general case, but it has been argued and empirically demonstrated by several authors that the number of knots in the path behaves as $O(n)$, the number of samples (Rosset & Zhu, 2007; Hastie et al., 2004; Li et al., 2007). In our case the outer loop of Algorithm 1 implements a 2-dimensional path following problem, that can be thought of as following $O(n)$ 1-dimensional paths traversed by the knots of the path. It therefore stands to reason (and we confirm it empirically below) that the outer loop typically has $O(n^2)$ steps where *events* happen. The events in the inner loop, in turn, have to do with validation observations, of which there are N, meeting the $O(n)$ knots. So a similar logic would lead us to assume that the number of meeting events (counted by the inner loop) should be at most $O(nN)$ total for the whole running of the algorithm (i.e., many iterations of the outer loop may have no events happening in the inner loop). Each iteration of either loop requires a re-calculation of up to three directions (of knots or validation crossings), using Theorem 1. These calculations involve updating and inversion of matrices that are roughly $|\mathcal{E} \times \mathcal{E}|$ in size (where $\mathcal{E}$ is the number of observations in the elbow). However note that only one row and column are involved in the updating, leading to a complexity of $O(n + |\mathcal{E}|^2)$ for the whole direction calculation operation, using the Sherman-Morrison formula for updating the inverse. In principle, $|\mathcal{E}|$ can be equal to n, although it is typically much smaller for most of the steps of the algorithm, on the order of $\sqrt{n}$ or less. So we assume here that the loop cost is between $O(n)$ and $O(n^2)$.

Putting all of these facts and assumptions together, we can estimate the algorithm's complexity's *typical* dependence on the number of observations in the training and validation set as ranging between $O(n^2 \cdot \max(n, N))$ and $O(n^3 \cdot \max(n, N))$. Clearly, this estimation procedure falls well short of a formal "worst case" complexity calculation, but we offer it as an intuitive guide to support our experiments below and get an idea of the dependence of running time on the amount of data used.

We have not considered the complexity of the lower envelope tracking problem in our analysis, because it is expected to have a much lower complexity (number of

order changes $O(\max(n, N)\log(\max(n, N)))$ and each order change involves $O(1)$ work).

5. Experiments

We demonstrate two main aspects of our methodology: The computational behavior as a function of the amount of training data used; and the ability of KQR to capture the form of the conditional distribution $P(Y|X)$, both in standard (i.i.d error) situations and when the error is not homoscedastic and asymmetric. We limit the experiments to simple simulated data, where we know the truth and can understand the behavior of KQR. This is due to shortage of space, and since our main contribution is not a new method that should be compared to competitors on real data, but rather a new algorithmic approach for an existing method.

Our simulation setup starts from univariate data $x \in [0, 1]$ and a "generating" function $f(x) = 2 \cdot \big(\exp(-30 \cdot (x - 0.25)^2) + \sin(\pi \cdot x^2)\big)$ (see Figure 3). We then let $Y = f(x) + \epsilon$, where the errors ϵ are independent, with a distribution that can be either:

1. $\epsilon \sim N(0, 1)$, i.e., i.i.d standard normal errors
2. $\epsilon + (x+1)^2 \sim \exp(1/(x+1)^2)$, which gives us errors that are still independent and have mean 0, but are asymmetric and have non-constant variance, with small signal-to-noise ratio on the higher values of x (see Figure 4).

Figure 3 demonstrates the results of the algorithm with i.i.d normal errors, 200 training samples and 200 validation samples and Gaussian kernel with parameter $\sigma = 0.2$. We see that the quantile estimates all capture the general shape of the true curve, with some "smoothing" due to regularization.

Next we consider the computational complexity of the algorithm, and its dependence on the number of training samples (with 200 validation samples). We compare it to the 1-dimensional KQR algorithm of Li et al. (2007), who have already demonstrated that their algorithm is significantly more efficient than grid-based approaches for generating 1-dimensional paths for fixed τ. Table 1 shows the number of steps of the main (outer) loop of Algorithm 1 and the total run time of our algorithm for generating the complete set of cross-validated solutions for $\tau \in [0.1, 0.9]$ as a function of the number of training samples (with validation sample fixed at 200). Also shown is the run time for the algorithm of Li et al. (2007), when we use it on a grid of 8000 evenly spaced τ values in $[0.1, 0.9]$ and find

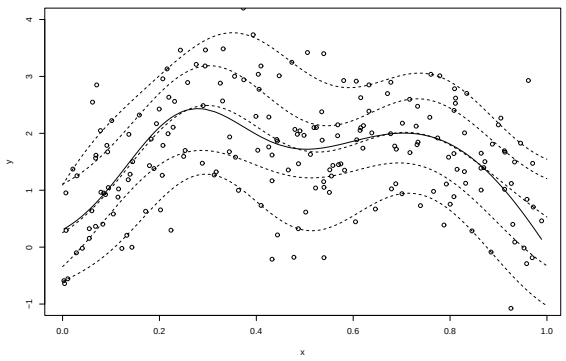

Figure 3. The function $f(x)$ (solid), data points drawn from it with i.i.d normal error, and our cross-validated estimates of quantiles $0.1, 0.25, 0.5, 0.75, 0.9$ (dahsed lines, from bottom to top).

Table 1. Number of steps and run times of our algorithm and of Li et al. (2007), for the whole path from $\tau = 0.1$ to $\tau = 0.9$.

NTRAIN	NSTEPS	TIME(BI-LEVEL)	TIME(LI ET AL.)
200	29238	931 SEC.	2500 SEC.
100	12269	99 SEC.	900 SEC.
50	2249	23 SEC.	480 SEC.

the best cross validated solution by enumerating the candidates as identified in Section 3. Our conjecture that the number of knots in the 2-dimensional path behaves like $O(n^2)$ seems to be consistent with these results, as is the hypothesized overall time complexity dependence of $O(n^3)$.

Finally, we demonstrate the ability of KQR to capture the quantiles with "strange" errors from model 2. Figure 4 shows a data sample generated from this model and the $(0.25, 0.5, 0.75)$ quantiles of the conditional distribution $P(Y|X)$ (solid), compared to their cross-validated KQR estimates (dashed), using 500 samples for learning and 200 for validation (more data is needed for learning because of the very large variance at values of x close to 1). As expected, we can see that estimation is easier of the lower quantiles and at smaller values of x, because the distribution $P(Y|X = x)$ has long right tails everywhere and has much larger variance when x is big.

6. Conclusions and extensions

In this paper we have demonstrated that the family of bi-level optimization problems (4) defined by the family of loss functions L_τ can be solved via a *path*

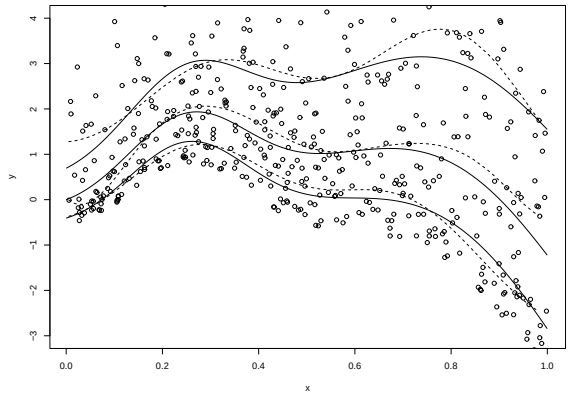

Figure 4. Quantiles of $P(Y|X)$ (solid), and their estimates (dashed) for quantiles $(0.25, 0.5, 0.75)$ with the exponential error model.

following approach which essentially maps the whole surface of solutions $\hat{f}(\tau, \lambda)$ as a function of both τ and λ and uses insights about the possible locations of the bi-level optima to efficiently find them. This leads to a closed-form algorithm for finding $f^*(\tau)$ for all quantiles. We see two main contributions in this work: a. Identification and solution of a family of non-convex optimization problem of great practical interest which can be solved using solely convex optimization techniques; and b. Formulation of a practical algorithm for generating the full set of cross-validated solutions for the family of kernel quantile regression problems.

Our algorithm as presented here can easily be adapted to bi-level path following of cross validated solutions of SVR, as the size ϵ of the *don't-care* region changes (see Rosset (2008) for details). However, it should be noted that the statistical motivation for solving quantile regression for multiple quantiles does not really carry through to ϵ-SVR, as the parameter ϵ and the loss function it parameterizes do not have a natural interpretation in the spirit of τ.

There are many other interesting aspects of our work, which we have not touched on here, including: development of further optimization shortcuts to improve algorithmic efficiency; investigation of the range of applicability of our algorithmic approach beyond KQR and SVR; analysis of the use of various kernels for KQR and how the kernel parameters and kernel properties interact with the solutions; implementation of in-sample model selection approaches such as SIC (Koenker, 2005; Li et al., 2007) instead of cross validation in our framework (which should require minimal changes).

References

Buchinsky, M. (1994). Changes in the u.s. wage structure 1963-1987: Application of quantile regression. *Econometrica*, *62*, 405–458.

Eide, E., & Showalter, M. (1998). The effect of school quality on student performance: A quantile regression approach. *Economics Letters*, *58*, 345–350.

Gunther, L., & Zhu, J. (2005). Efficient computation and model selection for the support vector regression. *Neural Computation*, *19*.

Hastie, T., Rosset, S., Tibshirani, R., & Zhu, J. (2004). The complete regularization path of the support vector machine. *JMLR*, *5*, 1391–1415.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *Elements of statistical learning*. Springer.

Kimeldorf, G., & Wahba, G. (1971). Some results on chebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, *33*, 82–95.

Koenker, R. (2005). *Quantile regression*. New York : Cambridge University Press.

Kunapuli, G., Bennett, K. P., Hu, J., & Pang, J.-S. (2007). Bilevel model selection for support vector machines. CRM Proceedings and Lecture Notes. American Mathematical Society, to appear.

Li, Y., Liu, Y., & Zhu, J. (2007). Quantile regression in reproducing kernel hilbert spaces. *JASA*, *102*.

Meinshausen, N. (2006). Quantile regression forests. *JMLR*, *7*, 983–999.

Perlich, C., Rosset, S., Lawrence, R., & Zadrozny, B. (2007). High quantile modeling for customer wallet estimation with other applications. *Proceedings of the Twelfth International Conference on Data Mining, KDD-07*.

Rosset, S. (2008). Bi-level path following for cross validated solution of kernel quantile regression. In preparation, evolving draft available at www.tau.ac.il/~saharon/papers/cvpath.pdf.

Rosset, S., & Zhu, J. (2007). Piecewise linear regularized solution paths. *Annals of Statistics*, *35*.

Schölkopf, B., & Smola, A. (2002). *Learning with kernels*. MIT Press, Cambridge, MA.

Sharir, M., & Agarwal, P. K. (1995). *Davenport-schinzel sequences and their geometric applications*. Cambridge University Press.

Smola, A., & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, *14*, 199–222.

Takeuchi, I., Le, Q., Sears, T., & Smola, A. (2006). Nonparametric quantile estimation. *JMLR*, *7*, 1231–1264.

Wang, G., Yeung, D.-Y., & Lochovsky, F. H. (2006). Two-dimensional solution path for support vector regression. *Proceedings of the 23rd international conference on Machine learning* (pp. 993–1000).