Journal of Machine Learning Research 10 (2009) 2473-2505 Submitted 12/08; Revised 8/09; Published 11/09

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

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

Editor: Ingo Steinwart

Abstract

We show how to follow the path of *cross validated* solutions to families of regularized optimization problems, defined by a combination of a parameterized loss function and a regularization term. A primary example is kernel quantile regression, where the parameter of the loss function is the quantile being estimated. 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 real and artificial data. This algorithm allows us to efficiently solve the whole family of bi-level problems. We show how it can be extended to cover other modeling problems, like support vector regression, and alternative in-sample model selection approaches.[1]

1. Introduction

In the standard predictive modeling setting, we are given a *training sample* of n examples $\{\mathbf{x}_1, y_1\}, \ldots, \{\mathbf{x}_n, y_n\}$ 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 employ 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.

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

1. A short version of this paper appeared at ICML 2008 (Rosset, 2008).

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). In regularized linear quantile regression we take the family $\mathcal{F}$ to be all linear combinations characterized by a coefficient vector $\beta \in \mathbb{R}^p$ and the modeling problem is

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

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):

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

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 and Showalter, 1998). There has been a surge of interest in the machine learning community in conditional quantile estimation in recent years, including theoretical analyses of consistency in quantile estimation and connections with support vector machines (Steinwart and Christmann, 2008; Christmann and Steinwart, 2008); methodological work on algorithms for quantile regression and their performance (Meinshausen, 2006; Takeuchi et al., 2006; Mease et al., 2007); and work on practical uses of extreme quantile estimation for data mining applications Perlich et al. (2007). 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 and Zhu, 2007; Hastie et al., 2004; Li et al., 2007; Takeuchi et al., 2009) 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 (a notable exception being Takeuchi et al. 2009), it naturally follows that similar algorithms exist for the case that we fix $\lambda = \lambda_0$ and are interested in generating the curve $\{\hat{\beta}(\tau,\lambda_0) \ : \ 0 < \tau < 1\}$.

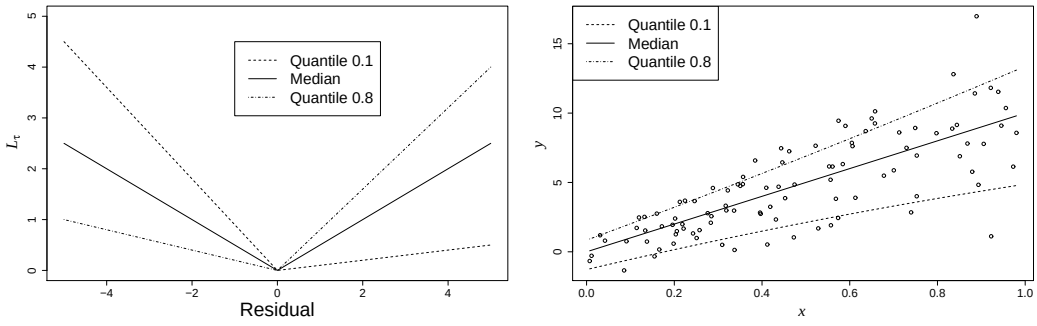

Figure 1: Quantile loss function for some values of τ (left) and an example where the median of Y is linear in X but the quantiles of $P(Y|X)$ are not because the noise is not identically distributed (right).

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 :

1. 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.

2. Weighted support vector machines, where the parameter of the loss function corresponds to reweighting the hinge loss differentially for the two classes, for example as a means for deriving accurate probability estimates (as recently suggested by Wang et al. 2008).

3. Huberized Lasso (Rosset and Zhu, 2007) with ℓ_1 regularization, where *huberizing* adds robustness to the traditional squared error loss, with a tunable parameter.

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

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

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 and Wahba, 1971) implies that the solution of Problem (3) lies in a low dimensional subspace spanned by the representer functions $\{K(\cdot,\mathbf{x}_i), i \in 1,\ldots,n\}$. Following the ideas of Hastie et al. (2004) for the support vector machine, Li et al. (2007) have shown that the λ-path of solutions to Problem (3) when τ is fixed can also be efficiently generated. A similar approach was independently developed by Takeuchi et al. (2009).

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) and (3), where $f^*(\tau) = \hat{f}(\tau, \lambda^*)$ and λ^* solves

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

where L_{CV} is the cross validation loss function (the bi-level formulation for Problem (1) would be identical, with $\hat{\beta}$ replacing $\hat{f}$). We will assume for now that $L_{\text{CV}} = L_\tau$, in order to evaluate the performance in estimating the τth quantile. 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. (2008) for the support vector machine, which is very similar to quantile regression from an optimization perspective (piecewise linear objective with quadratic penalty). 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 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 data set. 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. Our algorithms are based on extensions and generalizations of some of the ideas underlying the path following algorithms for 1-dimensional paths on convex problems (Hastie et al., 2004; Li et al., 2007; Rosset and Zhu, 2007). We concentrate our attention on the quantile regression case (both kernelized and linear), as one where the parameterized-loss problem is well motivated and historically useful, but we also discuss the similarities and differences in algorithms for the other examples we mentioned above. 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. However, this algorithm carries a heavy computational burden. The question of whether it is practical from a computational perspective depends on

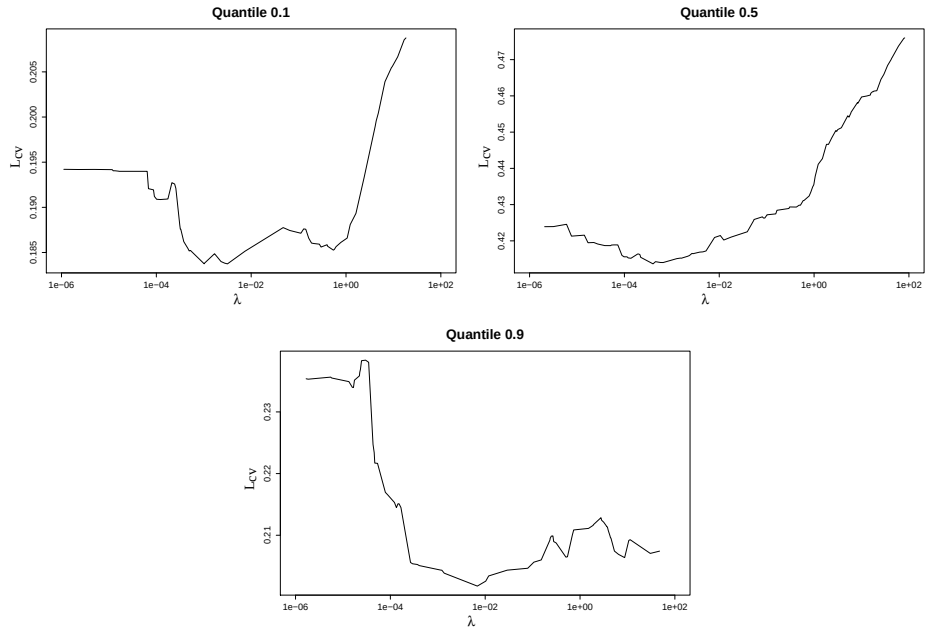

Figure 2: Estimated prediction error curves of Kernel Quantile Regression for some quantiles on one data set. The errors are shown as a function of the regularization parameter λ.

the properties of the modeling problems at hand, and may also benefit greatly from computational tricks and optimization shortcuts which are not the focus of this paper. We demonstrate its ability to successfully generate the complete set of cross-validated solutions on some illuminating simulation problems and on two medium-size real-life data-sets.

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. This is because the optimal solution always corresponds to a situation where either one of the validation points is crossing the non-differentiability *elbow* in the cross validation loss L_{CV}, or the regularization path is going thorough a *knot* in its piecewise linear change. However, due to the non-convexity of the problem, the solutions occasionally "jump" from one such point to another. It turns out that to follow this jumpy behavior we need to follow, not one path of solutions, but about $N+n$ of them, corresponding to all possible candidates for L_{CV} optimizers. 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. In Section 5 we discuss the extension of our methodology to other scenarios, including application of our methodology to SVR. We demonstrate our methods with a simulated and real data study in Section 6, where we show that our approach leads to model-selection that is more efficient than previous approaches, and illustrate the interesting behavior of KQR in practice.

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 $K(\mathbf{x},\tilde{\mathbf{x}}) = \mathbf{x}^\intercal\tilde{\mathbf{x}}$.

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. Similar results were independently generated by Takeuchi et al. (2009), who concentrate on the properties of $\hat{f}(\tau,\cdot)$ with λ fixed (as we elaborate below, these problems are in fact very similar). The representer theorem (Kimeldorf and Wahba, 1971) implies that the solution can be written as

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

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

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

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

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

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 $\lambda_{k-1} < \lambda \le \lambda_k$. 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 $\theta_i = -(1-\tau)$. This observation may be either *entering* the elbow (if it was previously in $\mathcal{L}$ or $\mathcal{R}$), or *exiting* it (if it previously had $\theta_i \in (-(1-\tau),\tau)$).[2] 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).

2. It is clear that the definition of an observation as *entering* or *exiting* the elbow is arbitrary, since an observation which enters at λ_k when we are decreasing λ actually exits at λ_k if we choose to traverse the path while increasing λ. There is also a possibility of more than one observation making this transition at once. With general τ and points in general location, this event has probability zero. As we will see, in the course of our investigation of the paths we are bound to encounter such cases, and we will address this issue when it comes up.

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. As previously mentioned, this has been recognized and used by other authors, including Takeuchi et al. (2009) for quantile regression, and Wang et al. (2008) for weighted hinge loss. More interestingly, the same is also true when both τ, λ are changing together *along a straight line*, that is, 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 following lemma makes this more general result concrete. The proof relies on a study of the KKT conditions in the spirit of Li et al. (2007) and the other references above, and we omit it.

Lemma 1 *Let $\tau(\lambda) = u\lambda + v$, and denote $\hat{f}(\lambda) = \hat{f}(\tau(\lambda),\lambda)$. Then in the range $\Gamma = \{\lambda \geq 0 :\ 0 < \tau(\lambda) < 1\}$ there exist* knots $\lambda_0 < \ldots < \lambda_m$ *such that for $\lambda_{k-1} < \lambda \leq \lambda_k$ we have:*

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

where $h_k(\mathbf{x}) = b_0^k + \sum_{i\in\mathcal{E}_k} b_i^k K(\mathbf{x},\mathbf{x}_i)$, and the direction $\mathbf{b}^k = \begin{pmatrix} b_0^k \\ \vdots \\ b_{|\mathcal{E}_k|}^k \end{pmatrix}$ is the solution of a set of linear equations with $|\mathcal{E}_k|+1$ unknowns:

$$\mathbf{A}^k \mathbf{b}^k = \begin{pmatrix} 0 \\ \mathbf{r}_{\mathcal{E}_k} \end{pmatrix}$$

with

$$\mathbf{A}^k = \begin{pmatrix} 0 & \mathbf{1}^\intercal \\ \mathbf{1} & \mathbf{K}_{\mathcal{E}_k} \end{pmatrix} ,$$

as defined in Li et al. (2007); and $r_j = y_j + u \cdot \left(\sum_{i\in\mathcal{R}_k} K(\mathbf{x}_j,\mathbf{x}_i) - \sum_{i\in\mathcal{L}_k} K(\mathbf{x}_j,\mathbf{x}_i)\right)$ for $j \in \mathcal{E}_k$.

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 $\hat{\theta}_{i_k}(\tau_0,\lambda_k) = \tau_0$, that is, 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 2 *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 $\delta \in [-\varepsilon_k, \nu_k]$, 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

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

And the fit at this knot progresses as

$$\hat{f}(\tau_0+\delta,\lambda_k+c_k\delta)=\frac{1}{\lambda_k+c_k\delta}\left(\lambda_k\hat{f}(\lambda_k,\tau_0)(\mathbf{x})+\delta h_k(\mathbf{x})\right) \tag{7}$$

$$h_k(\mathbf{x})=\tilde{b}_0^k+\sum_{i\in\mathcal{E}_k-i_k}\tilde{b}_i^k K(\mathbf{x},\mathbf{x}_i)+\sum_{i\in\mathcal{L}\cup\mathcal{R}\cup\{i_k\}}K(\mathbf{x},\mathbf{x}_i)\ . \tag{8}$$

Proof For small δ, the *modified* knot should be characterized by $\theta_{i_k}=\tau_0+\delta$, and $\mathcal{L},\mathcal{R},\mathcal{E}$ remaining the same. If we can find a c_k such that this holds for quantile $\tau_0+\delta$ and $\lambda=\lambda_k+c_k\delta$, and also the KKT conditions are maintained, we have our knot for quantile $\tau_0+\delta$. Assume this can be accomplished by moving in a direction h_k as in Equations (7) and (8). For the KKT conditions to hold we need to maintain:

C1. $\hat{f}(\tau_0+\delta,\lambda_k+c_k\delta)(\mathbf{x}_i)=y_i\ ,\ \ \forall i\in\mathcal{E}$

C2. $\hat{\theta}_i(\tau_0+\delta,\lambda_k+c_k\delta)=\hat{\theta}_i(\tau_0,\lambda_k)+\delta\ ,\ \ \forall i\in\mathcal{L}\cup\mathcal{R}\cup\{i_k\}$

C3. $\sum_{i\in\mathcal{E}_k-\{i_k\}}\tilde{b}_i^k=-|\mathcal{L}\cup\mathcal{R}\cup\{i_k\}|$

where C1 maintains the observations in $\mathcal{E}$ at the elbow, C2 maintains the equality requirements on $\hat{\theta}$ for the observations in $\mathcal{L}\cup\mathcal{R}$ (and the one on the boundary), and C3 maintains the constraint that the $\hat{\theta}$'s sum to 0.

We can express C1 in terms of Equations (7) and (8) and condition C2:

$$\begin{aligned}&\hat{f}(\tau_0+\delta,\lambda_k+c_k\delta)(\mathbf{x}_i)=y_i\ ,\ \ \forall i\in\mathcal{E}\\ \Leftrightarrow\quad &h_k(\mathbf{x}_i)=\tilde{b}_0^k+\sum_{j\in\mathcal{E}_k-i_k}\tilde{b}_j^k K(\mathbf{x}_j,\mathbf{x}_i)+\sum_{j\in\mathcal{L}\cup\mathcal{R}\cup\{i_k\}}K(\mathbf{x}_j,\mathbf{x}_i)=c_k y_i\ ,\ \ \forall i\in\mathcal{E}\ ,\end{aligned} \tag{9}$$

where the last term on the RHS of (9) accounts for the changes in the $\hat{\theta}_j$ which C2 implies for $j\in\mathcal{L}\cup\mathcal{R}\cup\{i_k\}$.

Now, by combining C3 and (9) into one set of equations in matrix notation, we get the result of the theorem:

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

and moving in the direction of the solution of this matrix equation as in (7,8) maintains the KKT conditions and the observation at the elbow, hence is a knot on the λ-solution path for quantile $\tau+\delta$ for every (small enough) δ.

■

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*, that is, 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. We defer the details of how these are identified and handled to the detailed algorithm description (Appendix A). When any of these events occurs, the set of knots has to be updated and their directions have to be re-calculated using Lemma 1, Theorem 2 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 3 *When the cross validation loss is the quantile loss (i.e., $L_{CV} = L_\tau$), then any minimizer*[3] *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.* $\hat{f}(\tau, \lambda^*(\tau))(\tilde{\mathbf{x}}_i) = \tilde{y}_i$

Proof Define $\tilde{\mathcal{L}}, \tilde{\mathcal{R}}$ in the obvious way, as the sets of validation observations with negative and positive residuals, respectively, for a given model. Fix τ, and consider the cross validation loss for a given value of λ:

$$\begin{aligned} L_{\text{cv}}(\lambda) \quad &:= \quad \sum_{i=1}^{N} L_{\text{CV}}(\tilde{y}_i, \hat{f}(\lambda)(\tilde{\mathbf{x}}_i)) = \sum_{i \in \tilde{\mathcal{L}}} (1-\tau)(\hat{f}(\lambda)(\tilde{\mathbf{x}}_i) - \tilde{y}_i) + \sum_{i \in \tilde{\mathcal{R}}} \tau \cdot (\tilde{y}_i - \hat{f}(\lambda)(\tilde{\mathbf{x}}_i)) = \\ &= \quad \tau \sum_{i \in \tilde{\mathcal{R}}} y_i - (1-\tau) \sum_{i \in \tilde{\mathcal{L}}} y_i - \tau \sum_{i \in \tilde{\mathcal{R}}} \hat{f}(\lambda_k)(\tilde{\mathbf{x}}_i) + (1-\tau) \sum_{i \in \tilde{\mathcal{L}}} \hat{f}(\lambda_k)(\tilde{\mathbf{x}}_i) + \\ &\quad + \frac{(\lambda - \lambda_k)}{\lambda} \left[-\tau \sum_{i \in \tilde{\mathcal{R}}} (h_k(\tilde{\mathbf{x}}_i) - \hat{f}(\lambda_k)(\tilde{\mathbf{x}}_i)) + (1-\tau) \sum_{i \in \tilde{\mathcal{L}}} (h_k(\tilde{\mathbf{x}}_i) - \hat{f}(\lambda_k)(\tilde{\mathbf{x}}_i)) \right] \end{aligned}$$

where k is such that $\lambda_{k-1} \leq \lambda \leq \lambda_k$ (where the list of $\lambda_\cdot$'s now combines both knots and validation crossings), and we take advantage of the representation in (6). From the last two rows we can see that L_{cv} is monotone in λ as long as $\tilde{\mathcal{L}}, \tilde{\mathcal{R}}$ are fixed (i.e., no validation crossing occurs) and h_k is fixed (i.e., no knot is encountered). Therefore any local (or global) extremum must be at a knot or a validation crossing. ■

Corollary 4 *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 2 and the validation crossing following problem. In each case we

3. 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.

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 pre-determined, 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 the following result, closely related to Theorem 2. Assume there is a validation crossing at λ_v for quantile τ_0, and that validation set observation j_v is the one crossing the elbow, that is,

$$\hat{f}(\tau_0, \lambda_v)(\tilde{\mathbf{x}}_{j_v}) = 0 \, .$$

Let $\mathbf{s}_{\mathcal{E}_v}, K_{\mathcal{E}_v}, \mathbf{b}_{\mathcal{E}_v}$ be defined as in Theorem 2 (with $\{i_v\} = \Phi$ for definition of $\mathbf{s}$). Let $\mathbf{k}_v = (K(X_{\mathcal{E}_v}, \tilde{\mathbf{x}}_{j_v}))$ be a $1 \times |\mathcal{E}_v|$ vector of the kernel evaluations at $\tilde{\mathbf{x}}_{j_v}$ for the elbow observation functionals.

Proposition 5 *λ_v moves linearly as τ changes. That is, there exists a constant d_v such that for quantile $\tau_0 + \delta$ there is a validation crossing in the λ-solution path at $\lambda_v + d_v\delta$, for $\delta \in [-\varepsilon_v, \nu_v]$, a non-empty neighborhood of 0. d_v is determined through the solution of a set of $|\mathcal{E}_v| + 2$ linear equations with $|\mathcal{E}_v| + 2$ unknowns:*

$$\mathbf{B}^v \begin{pmatrix} \tilde{\mathbf{b}}^v \\ d_v \end{pmatrix} = \begin{pmatrix} -(|\mathcal{R}| + |\mathcal{L}|) \\ -\mathbf{s}_{\mathcal{E}_v} \\ -\tilde{s}_{j_v} \end{pmatrix}$$

with

$$\mathbf{B}^v = \begin{pmatrix} 0 & \mathbf{1}^T & 0 \\ \mathbf{1} & \mathbf{K}_{\mathcal{E}_v} & -\mathbf{y}_{\mathcal{E}_v} \\ 1 & \mathbf{k}_v & -\tilde{y}_{j_v} \end{pmatrix} .$$

Furthermore, the solution $\hat{f}(\tau_0 + \delta, \lambda_v + c_v\delta)$ is given by:

$$\hat{f}(\tau_0 + \delta, \lambda_v + c_v\delta) = \frac{1}{\lambda_v + c_v\delta} \left(\lambda_v \hat{f}(\lambda_v, \tau_0)(\mathbf{x}) + \delta h_v(\mathbf{x}) \right)$$
$$h_v(\mathbf{x}) = \tilde{b}_0^v + \sum_{i \in \mathcal{E}_v} b_i^v K(\mathbf{x}, \mathbf{x}_i) + \sum_{i \in \mathcal{L} \cup \mathcal{R}} K(\mathbf{x}, \mathbf{x}_i) \, .$$

The proof relies on following the same steps as the proof of Theorem 2 and is omitted for brevity.

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 (denote $f^k(\delta) = \hat{f}(\tau_0 + \delta, \lambda_k + c_k\delta)$):

$$\begin{aligned} \textstyle\sum_{i=1}^N \quad & L_{\text{cv}}(\tilde{y}_i, f^k(\delta)(\tilde{\mathbf{x}}_i)) = \\ & = -(1 - \tau_0 - \delta) \sum_{i \in \tilde{\mathcal{L}}} (\tilde{y}_i - f^k(\delta)(\tilde{\mathbf{x}}_i)) + (\tau_0 + \delta) \sum_{i \in \tilde{\mathcal{R}}} (\tilde{y}_i - f^k(\delta)(\tilde{\mathbf{x}}_i)) = \\ & = \sum_{i=1}^N L_{\text{cv}}(\tilde{y}_i, f^k(0)(\tilde{\mathbf{x}}_i)) + \delta \sum_i |\tilde{y}_i - f^k(0)(\tilde{\mathbf{x}}_i)| + \frac{\delta}{\lambda_k + c_k\delta} \cdot \qquad (10) \\ & \cdot \left[-(1 - \tau_0 - \delta) \sum_{i \in \tilde{\mathcal{L}}} (c_k f^k(0)(\tilde{\mathbf{x}}_i) - h_k(\tilde{\mathbf{x}}_i)) + (\tau_0 + \delta) \sum_{i \in \tilde{\mathcal{R}}} (c_k f^k(0)(\tilde{\mathbf{x}}_i) - h_k(\tilde{\mathbf{x}}_i)) \right] . \end{aligned}$$

A similar expression can be derived for validation crossings (with f^k, c_k, h_k replaced by f^v, d_v, h_v in the obvious way). 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). The algorithms developed mostly conform to the common-sense approach of maintaining the order of the validation loss scores from smallest to largest; identifying the τ values where elements with neighboring scores *meet* (i.e., obtain identical score); and whenever a meeting occurs, re-calculating only the relevant crossing points, that is, those of the elements that changed order and their immediate neighbors. We also have to re-calculate some of the validation loss scores whenever an *event* happens on the training solution path (like a *knot crossing*).

To calculate the meeting point of two elements with neighboring scores (assume WLOG that they are two knots k, l) we find the zeros of the cubic equation obtained by requiring equality for the two rational functions of the form (10) corresponding to the two elements. Writing the expression in (10) for both k and l, and requiring equality gives us the cubic equation:

$$\begin{aligned} 0 = \quad & \lambda_k\lambda_l(lo_k - lo_l) + \\ & +\delta[(\lambda_k c_l + \lambda_l c_k)(lo_k - lo_l) + \lambda_k\lambda_l(la_k - la_l) + \lambda_l(\tau_0 LR_k - Le_k) - \lambda_k(\tau_0 LR_l - Le_l)] \\ & +\delta^2[c_k c_l(lo_k - lo_l) + (c_k\lambda_l + c_l\lambda_k)(la_k - la_l) + \\ & \quad +c_l(\tau_0 LR_k - Le_k) + \lambda_l LR_k - c_k(\tau_0 LR_l - Le_l) - \lambda_k LR_l] \\ & +\delta^3[c_k c_l(la_k - la_l) + c_l LR_k - c_k LR_l] \,, \end{aligned} \tag{11}$$

where:

$$\begin{aligned} lo_k &= \sum_{i=1}^{N} L_{\mathrm{cv}}(\tilde{y}_i, f^k(0)(\tilde{\mathbf{x}}_i)) \\ la_k &= \sum_i |\tilde{y}_i - f^k(0)(\tilde{\mathbf{x}}_i)| \\ LR_k &= \sum_{i\in\tilde{\mathcal{L}}} (c_k f^k(0)(\tilde{\mathbf{x}}_i) - h_k(\tilde{\mathbf{x}}_i)) + \sum_{i\in\tilde{\mathcal{R}}} (c_k f^k(0)(\tilde{\mathbf{x}}_i) - h_k(\tilde{\mathbf{x}}_i)) \\ Le_k &= \sum_{i\in\tilde{\mathcal{L}}} (c_k f^k(0)(\tilde{\mathbf{x}}_i) - h_k(\tilde{\mathbf{x}}_i)) \,, \end{aligned}$$

with similar expressions for the elements with subscript l derived in the obvious way. The smallest non-negative solution for δ is the one we are interested in.

Figure 3 gives a simple illustration of the process of following the validation loss scores, and identifying their optimum, while updating the directions of the knots and validation crossings as events occur. It shows the set of training and validation loss scores for two knots and the validation loss only for one validation crossing. The training loss is shown in solid lines, and the validation loss in dashed lines. Assuming these are the only three candidates in Theorem 3, the figure shows in bold the lower envelope which defines the optimal cross validation solution at every value of τ. As we can see, in this example the first (left) switch is between two knots as a result of a knot crossing, while the second (right) is a result of a validation crossing becoming optimal. It should be noted,

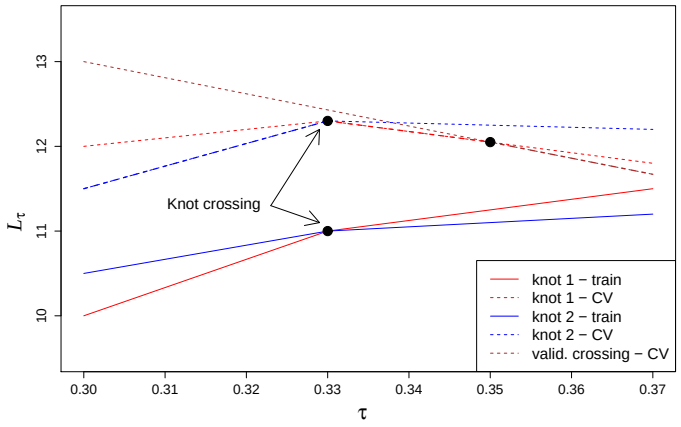

Figure 3: Illustration of the process of lower envelope tracking, in the presence of two knots and one validation crossing being tracked. See text for details.

that the linearity of the solid and dashed lines in Figure 3 is for illustration simplicity, and is not a realistic depiction of the non-linear evolution of the loss as τ varies, as discussed above.

4. Algorithm Overview

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

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 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 and 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 the N validation observations 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

Algorithm 1: Main steps of our 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 2 ;
Find direction of each validation crossing according to Proposition 5;
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 (10);
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 $\tau_{\text{next}} = \min(\tau_4, \tau_5, \tau_6, \tau_{\text{new}})$;
        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_{cv} 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 (10);
        Set $\tau_{\text{now}} = \tau_{\text{next}}$;
    end
    Update the list of knots according to the first event (knot crossing, birth, merge) ;
    Update the directions of affected knots using Theorem 2 ;
end

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 2 or Proposition 5. These calculations involve updating and inverting 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 (Sherman and Morrison, 1949) 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 computational 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. Extensions

In this section we discuss some of the possible extensions of our algorithm. First we discuss the design of algorithms that are similar in spirit for other parameterized loss function problems, in particular for support vector regression (ε-SVR) and *Huberized* least squares regression. We then move on to the use of in-sample model selection criteria instead of cross validation. Finally, we address the issue of possible non-monotonicity in τ, noted by previous authors (Koenker, 2005; Takeuchi et al., 2006). We demonstrate how our algorithm can be naturally extended to amend this situation.

5.1 Support Vector Regression and Weighted Support Vector Machines

One possible view of regularized ε-SVR (Smola and Schölkopf, 2004) is similar to the quantile regression problem for $\tau = 0.5$:

$$\hat{f}(\varepsilon,\lambda) = \arg\min_f \sum_i L_\varepsilon(y_i - f(\mathbf{x}_i)) + \frac{\lambda}{2}\|f\|^2_{\mathcal{H}_K} \tag{12}$$

where the parameterized loss function L_ε is piecewise linear and symmetric around zero, with a *don't care* region of size 2ε:

$$L_\varepsilon(r) = \begin{cases} r-\varepsilon & r \geq \varepsilon \\ 0 & -\varepsilon < r < \varepsilon \\ -r-\varepsilon & r \leq -\varepsilon \end{cases} .$$

The loss is parameterized by ε. From an optimization perspective, this problem is very similar to KQR, with a piecewise linear loss and an RKHS norm penalty. The solution can be represented as in (5), and the KKT conditions for optimality of solutions of (12) in terms of coefficients of representer functions have been formalized and used to design λ-path following algorithms by Gunther and Zhu (2005). For example, if we define for a proposed solution $f(\mathbf{x})$ of ε-SVR:

- $\mathcal{L} = \{i : y_i - f(\mathbf{x}_i) < -\varepsilon\}$ (points on left of *left elbow* of L_ε)
- $\mathcal{E}_\mathcal{L} = \{i : y_i - f(\mathbf{x}_i) = -\varepsilon\}$ (*left elbow*)
- $\mathcal{C} = \{i : |y_i - f(\mathbf{x}_i)| < \varepsilon\}$ (don't care region)
- $\mathcal{E}_\mathcal{R} = \{i : y_i - f(\mathbf{x}_i) = \varepsilon\}$ (*right elbow*)
- $\mathcal{R} = \{i : y_i - f(\mathbf{x}_i) > 0\}$ (right of *right elbow*),

Then the Karush-Kuhn-Tucker (KKT) conditions for optimality of a solution $\hat{f}(\varepsilon,\lambda)$ of Problem (12) can be phrased as:

- $i \in \mathcal{C} \;\Rightarrow\; \hat{\theta}_i = 0$
- $i \in \mathcal{L} \;\Rightarrow\; \hat{\theta}_i = -1$
- $i \in \mathcal{R} \;\Rightarrow\; \hat{\theta}_i = 1$
- $i \in \mathcal{E}_{\mathcal{L}} \;\Rightarrow\; -1 \leq \hat{\theta}_i \leq 0$
- $i \in \mathcal{E}_{\mathcal{R}} \;\Rightarrow\; 0 \leq \hat{\theta}_i \leq 1$
- $\sum_i \hat{\theta}_i = 0.$

Wang et al. (2006) have noted that ε-paths (with fixed λ) can similarly be followed. Because of the fundamental similarity in the optimization setting, all our results regarding behavior of λ-paths and knots as τ changes in quantile regression (e.g., Theorem 2) can be adapted in a reasonably straight forward manner to follow paths and knots of λ-solution paths in ε-SVR, as ε varies.

There is, however, a fundamental difference in the statistical setting between parameterized quantile loss and parameterized ε-SVR loss. While every quantile loss function defines an interesting modeling problem of estimation of a given conditional quantile, there is no such clear motivation for varying ε. Furthermore, there is no obvious way in which the cross validation loss should change with ε, if at all. In most cases, it seems ε is viewed more as another tuning parameter for a single modeling problem (like λ), than a parameter defining a range of loss functions, each of its own independent interest. In this situation, the only motivation for solving the range of bi-level problems parameterized by ε may be as a way to efficiently traverse the entire (ε, λ) solution space in search of a single "best" prediction model. It may therefore be appropriate to use a single cross validation objective L_{cv} independent of ε. If we choose $L_{\text{cv}} = L_{\tau=0.5}$ (the symmetric quantile loss, sometimes called absolute loss), then our observations on the bi-level path following problem (e.g., Theorem 3) would require slight modifications, but the ideas would carry through to the ε-SVR case in a straight forward manner.

An interesting recent development is the proposal of weighted support vector machines for probability estimation by Wang et al. (2008). Their proposed approach calls for fitting weighted versions of the support vector machine, with a range of relative weights applied to the two classes, as a provably valuable approach for estimating probabilities. We omit the details for brevity, but note that like the SVR case above, extending our bi-level approach to this problem is straight forward.

5.2 ℓ_1-regularized Huberized Squared Loss

Rosset and Zhu (2007) suggested the use of robust versions of squared error loss with ℓ_1 regularization, as an approach for combining computational efficiency and robustness to long-tailed error distribution. Huber's loss function is quadratic for small absolute residuals, then continues linearly as the residuals move away from zero, while maintaining differentiability. It is parameterized with a *huberizing point* t:

$$L_t(r) = \begin{cases} r^2 & |r| < t \\ 2t|r| - r^2 & \text{otherwise} \end{cases}.$$

The algorithm proposed in Rosset and Zhu (2007) for λ-path following can be thought of as an extension of the LARS-Lasso algorithm proposed for the Lasso (squared error loss with ℓ_1 penalty) by Efron et al. (2004). The loss function is differentiable, there is no concept of *elbow* (although

there are still knots), the KKT conditions are quite different, and if we also use a differentiable L_{cv}, the cross validation procedure would be affect as well. However, the general reasoning of this paper can still be applied to build bi-level path following algorithms for the Huberized lasso problem, and to choose good t, λ combinations.

5.3 Use of In-sample Model Selection Criteria

Li et al. (2007) follow the literature in proposing two model selection criteria for selecting λ^* for a fixed value of τ, when there is no validation sample. These are Schwartz information criterion (SIC, Schwarz, 1978) and generalized approximate cross validation (GACV, Yuan, 2006). Both of these use the model's effective degrees of freedom (DF) as a complexity measure which penalizes the empirical error. Following Zou et al. (2007), Li et al. (2007) show that an unbiased estimate of DF is the size of the elbow $|\mathcal{E}|$. Thus, they arrive at following SIC and GACV approximations:

$$\text{SIC}(\lambda) = \log\left(\frac{1}{n}\sum_{i=1}^{n} L_\tau(y_i - \hat{f}(\tau,\lambda)(\mathbf{x}_i))\right) + \frac{\log n}{2n}|\mathcal{E}| \quad (13)$$

$$\text{GACV}(\lambda) = \frac{\sum_{i=1}^{n} L_\tau(y_i - \hat{f}(\tau,\lambda)(\mathbf{x}_i))}{n - |\mathcal{E}|}\,. \quad (14)$$

If we were to adopt these measures (or similar ones) for model selection instead of the cross validation approach using an independent validation set, tracking the optimal solution $\lambda^*(\tau)$ requires no extra work besides following the knots of the solution (as described in Sections 2, 4). This is guaranteed by the following simple result:

Proposition 6 *For any fixed τ, the minimizer $\lambda^*(\tau)$ of SIC, GACV and any similar model selection criterion which is monotone in both $\sum_{i=1}^{n} L_\tau(y_i - \hat{f}(\tau,\lambda)(\mathbf{x}_i))$ and $|\mathcal{E}|$, is always at one of the knots of the solution path.*

Proof The loss is monotone between knots (e.g., from looking at Equation 6), while $|\mathcal{E}|$ is fixed. ■

Thus, if we wish to use SIC or similar measures for selecting $\lambda^*(\tau)$, the inner loop of Algorithm 1 (lines 12-21) can be omitted and replaced with a simple tracking of the value of SIC at the knots that are being followed. Since the algorithmic complexity of applying SIC/GACV is reduced compared to cross validation, and given the ongoing debates in the literature on the merits of in-sample versus out-of-sample model selection, it may often be beneficial to apply these in-sample methods in addition, or even instead of, cross validation. We demonstrate and compare performance of the two approaches in Section 6 below.

5.4 Addressing *Quantile Crossings*

The problem of quantile crossing, as formulated by Koenker (2005), is that for any fixed λ (in particular $\lambda = 0$ in the linear quantile regression case, which is the one that Koenker (2005) concentrates on), the prediction $\hat{f}(\tau,\lambda)(\mathbf{x})$ may not be non-decreasing in τ for a fixed $\mathbf{x}$. That is, we may have $\tau_0 < \tau_1$ and $\hat{f}(\tau_0,\lambda)(\mathbf{x}) > \hat{f}(\tau_1,\lambda)(\mathbf{x})$, which can never be true of the corresponding population conditional quantiles, of course.

Takeuchi et al. (2006) address this problem by constraining the solution to comply with the monotonicity requirement over a *finite* set of "interesting" quantiles. Their approach cannot work

in our case, since our algorithm is local in nature and generates the solutions for the complete space of (τ,λ) values. However, we can offer a partial remedy to the quantile crossing problem through observation of the guaranteed sub-optimality of the resulting solutions, and a consequent *envelope tracking* modification. The main motivation is the following:

Proposition 7 *Assume* $\tau_0 < \tau_1$ *and* $\hat{f}(\tau_0,\lambda)(\mathbf{x}) > \hat{f}(\tau_1,\lambda)(\mathbf{x})$ *for some* $\lambda, \mathbf{x}$. *Then either*

$$\mathbb{E}_{Y|X=\mathbf{x}} L_{\tau_0}(Y, \hat{f}(\tau_0,\lambda)(\mathbf{x})) \geq \mathbb{E}_{Y|X=\mathbf{x}} L_{\tau_0}(Y, \hat{f}(\tau_1,\lambda)(\mathbf{x}))\,. \tag{15}$$

or

$$\mathbb{E}_{Y|X=\mathbf{x}} L_{\tau_1}(Y, \hat{f}(\tau_0,\lambda)(\mathbf{x})) \leq \mathbb{E}_{Y|X=\mathbf{x}} L_{\tau_1}(Y, \hat{f}(\tau_1,\lambda)(\mathbf{x})) \tag{16}$$

Thus, we can always improve the predictive quality of either $\hat{f}(\tau_0,\lambda)$ *or* $\hat{f}(\tau_1,\lambda)$ *by eliminating the non-monotonicity.*

Proof In what follows we eliminate the explicit conditioning in the expectations. All expectations are with regard to the distribution $P(Y|X=\mathbf{x})$. Denote by c_0 and c_1 the τ_0 and τ_1 quantiles respectively of $P(Y|X=\mathbf{x})$. By definition, $c_0 \leq c_1$. We also assume $\hat{f}(\tau_0,\lambda)(\mathbf{x}) > \hat{f}(\tau_1,\lambda)(\mathbf{x})$. We hereafter denote these two fitted value by $\hat{f}_0, \hat{f}_1$ respectively for brevity. This gives us three possible scenarios:

1. $\hat{f}_1 \geq c_0$. In this case (15) holds, since:

$$\begin{aligned}
\mathbb{E}L_{\tau_0}(Y,\hat{f}_0) &= \tau_0 \int_{y\geq\hat{f}_0} y-\hat{f}_0 dP(y|\mathbf{x}) + (1-\tau_0)\int_{y<\hat{f}_0} -y+\hat{f}_0 dP(y|\mathbf{x}) \\
&= \tau_0 \int_{y\geq\hat{f}_0} P(Y\geq y|\mathbf{x})dy + (1-\tau_0)\int_{y<\hat{f}_0} P(Y\leq y|\mathbf{x})dy \\
&= \mathbb{E}L_{\tau_0}(Y,\hat{f}_1) + \int_{\hat{f}_1}^{\hat{f}_0} \left[(1-\tau_0)P(Y\leq y|\mathbf{x}) - \tau_0 P(Y\geq y|\mathbf{x})\right] dy \\
&\geq \mathbb{E}L_{\tau_0}(Y,\hat{f}_1)\,,
\end{aligned}$$

   where the inequality on the last line is because $P(Y\leq y|X=\mathbf{x}) \geq \tau_0$ in the range $\hat{f}_1 \leq y \leq \hat{f}_0$ (by our assumption that $c_0 \leq \hat{f}_1 < \hat{f}_0$).

2. $\hat{f}_0 \leq c1$. By the same line of argument in this case (16) holds.

3. If neither of the previous two holds, we must have $\hat{f}_1 < c_0 \leq c_1 < \hat{f}_0$. Following the same steps as in case 1 we write:

$$\mathbb{E}L_{\tau_0}(Y,\hat{f}_0) = \mathbb{E}L_{\tau_0}(Y,\hat{f}_1) + \int_{\hat{f}_1}^{\hat{f}_0} \left[(1-\tau_0)P(Y\leq y|\mathbf{x}) - \tau_0 P(Y\geq y|\mathbf{x})\right] dy \tag{17}$$

$$\mathbb{E}L_{\tau_1}(Y,\hat{f}_0) = \mathbb{E}L_{\tau_1}(Y,\hat{f}_1) + \int_{\hat{f}_1}^{\hat{f}_0} \left[(1-\tau_1)P(Y\leq y|\mathbf{x}) - \tau_1 P(Y\geq y|\mathbf{x})\right] dy\,. \tag{18}$$

   Assume $\mathbb{E}L_{\tau_0}(Y,\hat{f}_0) < \mathbb{E}L_{\tau_0}(Y,\hat{f}_1)$. It implies the integral in (17) is negative which in turn implies that the integral in (18) is also negative, since trivially

$$\frac{\partial}{\partial\tau}\int_{\hat{f}_1}^{\hat{f}_0} \left[(1-\tau)P(Y\leq y|\mathbf{x}) - \tau P(Y\geq y|\mathbf{x})\right] dy < 0\,.$$

   This negativity implies $\mathbb{E}L_{\tau_1}(Y,\hat{f}_0) < EL_{\tau_1}(Y,\hat{f}_1)$.

■

The following is an immediate consequence of Proposition 7 if we take $P(Y|\tilde{\mathbf{x}}_i)$ to be a point mass at $Y = \tilde{y}_i$.

Corollary 8 *If non-monotonicity holds at a validation point, that is,* $\tau_0 < \tau_1$ *and* $\hat{f}(\tau_0,\lambda)(\tilde{\mathbf{x}}_i) > \hat{f}(\tau_1,\lambda)(\tilde{\mathbf{x}}_i)$*, then either*

$$L_{\tau_0}(\tilde{y}_i, \hat{f}(\tau_0,\lambda)(\tilde{\mathbf{x}}_i)) \geq L_{\tau_0}(\tilde{y}_i, \hat{f}(\tau_1,\lambda)(\tilde{\mathbf{x}}_i))$$

or

$$L_{\tau_1}(\tilde{y}_i, \hat{f}(\tau_0,\lambda)(\tilde{\mathbf{x}}_i)) \leq L_{\tau_1}(\tilde{y}_i, \hat{f}(\tau_1,\lambda)(\tilde{\mathbf{x}}_i)) \ .$$

Thus, we can improve our holdout performance at either quantile τ_0 *or* τ_1 *by appropriately enforcing monotonicity.*

We conclude that eliminating non-monotonicity can improve both predictive performance and cross validation performance. In terms of practical implications, it is easy to see (though not trivial to implement) how our algorithm can be extended to identify quantile crossings. When these occur, at least one knot will be moving in the 'wrong direction', that is, the expression in (7) will be decreasing in δ. The algorithm will then have to keep careful tabs on the upper and lower limits of the fit at every λ as τ changes (the quantile-crossing gap). Discussion of the details and the appropriate way to resolve the non-monotonicity given this envelope is left for future work.

6. Experiments

Our methodology offers a new approach for generating the full set of cross-validated kernel quantile regression models. There are several interesting aspects of the modeling problem in general and our algorithm in particular that should be studied through a data-based study.

First, to evaluate the new algorithm, the efficiency of the algorithm should be compared to alternative approaches that allow generation of complete set of solutions and cross-validation. This includes the naive *grid-based* search whereby the KQR problem is solved using standard approaches (Takeuchi et al., 2006) for a grid of values in the (τ,λ) space, and a good regularization parameter is chosen for each value of τ by cross-validation; and the method of Li et al. (2007), which can be used to generate the complete λ-path at a grid of τ-values and cross validate each path separately. As Li et al. (2007) demonstrated clearly, their λ-path method is far superior to the grid-based approach in terms of computation, and so we concentrate on comparison to the λ-path approach only, and show that our algorithm compares favorably to it in generating the full set of bi-level solutions.

Second, we may also be interested in studying properties of the modeling problem, not necessarily tied to the new algorithm. Cross-validation based selection of regularization should be compared to *in-sample* approaches such as SIC (Schwarz, 1978) and GACV (Yuan, 2006). As noted above, all of these can be implemented in our framework. It is obvious that given the same amount of data for model fitting, it is better to use holdout data for model selection. However, the fair comparison should be between integrating the validation set into the training set and implementing an in-sample model selection approach, and using a smaller training set in a cross-validation framework.

Another interesting question about the modeling approach regards the ability of KQR to deal with skewed and non-homogeneous error distributions, and still generate reasonable estimates of the underlying quantiles.

We address all of these aspects in this section.

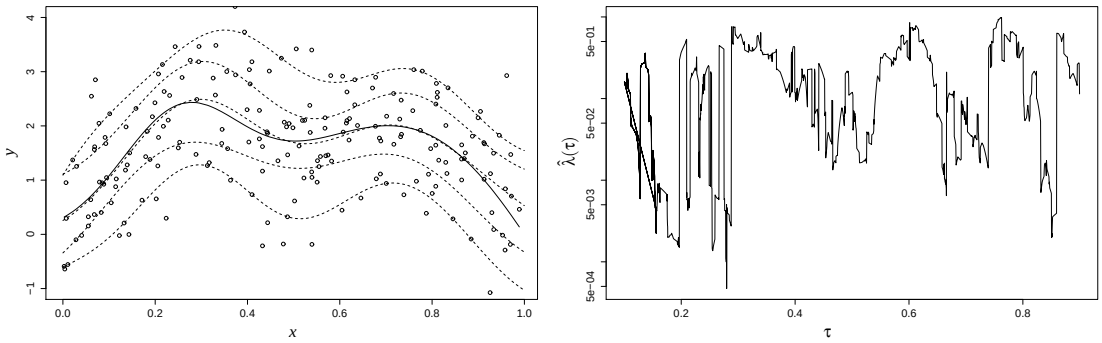

Figure 4: Left: The function $f(x)$ (solid), data points drawn from it with i.i.d normal error, and our cross-validated estimates of quantiles $0.1, 0.25, 0.5, 0.75, 0.9$ (dashed lines, from bottom to top). Right: Evolution of optimal regularization parameter $\hat{\lambda}(\tau)$, as τ varies.

6.1 Simulations

Our simulation setup starts with univariate data $x \in [0,1]$ and a "generating" function $f(x) = 2 \cdot \left(\exp(-30 \cdot (x-0.25)^2) + \sin(\pi \cdot x^2)\right)$ (see Figure 4). We then let $Y = f(x) + \varepsilon$, where the errors ε are independent, with a distribution that can be either:

1. $\varepsilon \sim N(0,1)$, that is, i.i.d standard normal errors
2. $\varepsilon + (x+1)^2 \sim \exp(1/(x+1)^2)$, 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 5).

Figure 4 demonstrates the results of the algorithm with i.i.d normal errors, 200 training samples and 200 validation samples and a Gaussian kernel with parameter $\sigma = 0.2$. In the left panel, we see that the quantile estimates all capture the general shape of the true curve, with some "smoothing" due to regularization. In the right panel we see the evolution of the optimal regularization parameter $\hat{\lambda}(\tau)$ as τ varies. We see the expected "jumpy" behavior of the optimal parameter, but we do not see a clear tendency to be smaller for quantiles closer to $1/2$. This is somewhat surprising when we think in terms of bias and variance (or approximation error and estimation error) in learning. Values of τ closer to $1/2$ typically create learning problems that are "easier", that is, variance is smaller (Koenker, 2005), and this should in principle allow us to build more complex models (reduce regularization), and decrease bias. A confounding factor in this analysis is the fact that the scale of quantile error need not be comparable for different quantiles. In particular, we may expect that loss magnitude would be larger for quantiles close to 0.5, where both types of errors get penalized equally. If that is the case, then having the similar regularization parameter may in fact imply *less* regularization for τ close to 0.5 compared to extreme quantiles. Another interesting observation is that while $\lambda^*(\tau)$ may be jumpy, both the empirical and the validation loss may vary smoothly. This smoothness is in fact guaranteed for the validation loss L_{cv}, since it is easily seen that the "jumps" are points where validation loss is equal at two knots or validation crossings.

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 KQR algorithm

NTRAIN	NSTEPS	TIME(BI-LEVEL)	TIME(LI ET AL.)	BREAK-EVEN RESOLUTION
200	29238	931 SEC.	2500 SEC.	3000
100	12269	99 SEC.	900 SEC.	900
50	2249	23 SEC.	480 SEC.	400

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 $\tau = 0.9$, as a function of the number of training observations NTRAIN. These results are based on applying Li et al. (2007) at 8000 different values of τ. The last column shows what resolution would give similar running times to both approaches (see text for details).

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 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 $O(n^3)$. Since 8000 is typically an unnecessarily fine grid for practical applications, we offer in the last column an evaluation of the comparative efficiency of the two methods in terms of the number of distinct τ values that can be fitted with the Li et al. (2007) approach in roughly the same running time as our approach. It is clear from these results that if just a small number of τ values (say, 10) are sufficient to address the complete problem, our approach does not carry a computational benefit.

Next, we demonstrate the ability of KQR to capture the quantiles with "strange" errors from model 2. Figure 5 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 of the lower quantiles, and at smaller values of x is easier, because the distribution $P(Y|X = x)$ has long right tails everywhere and has much larger variance when x is big.

6.2 Baseball Data and California Housing

As discussed in Perlich et al. (2007), estimating conditional quantiles is often a modeling task that is well grounded in practical applications. In the context of house prices, we can think of estimating a high (but not extreme[4]) conditional quantile as the seller's search for a favorable bargaining position in negotiations. Similarly for salaries, estimating a high conditional quantile can serve as a measure of what an employee can expect to receive optimistically (but still realistically), given his characteristics and performance.We therefore demonstrate KQR on two well studied data sets that correspond to such modeling problems: baseball salaries as a function of a player's home runs and years of experience (He et al., 1998) and the California housing data set (Pace and Barry, 1997),

4. Extreme quantile estimation is also of interest in some contexts, but we do not demonstrate it here due to the inherent statistical difficulty and questionable results, see some discussion in Conclusion section.

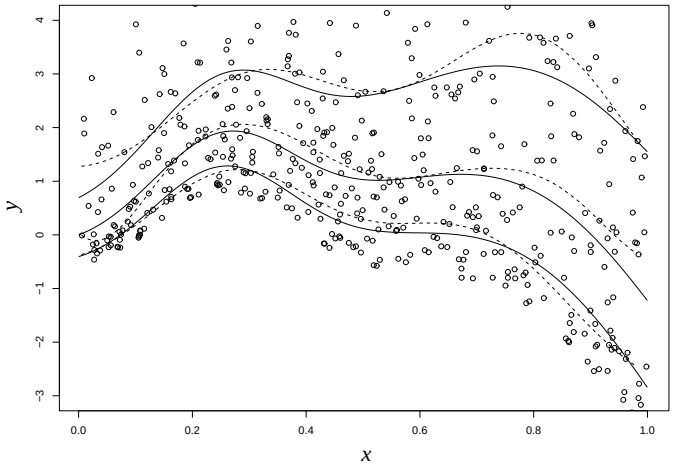

Figure 5: Quantiles of $P(Y|X)$ (solid), and their estimates (dashed) for quantiles $(0.25, 0.5, 0.75)$ with the exponential error model.

which describes the median prices of houses in neighborhoods of California along with nine explanatory demographic variables. We seek to demonstrate predictive performance, fitted models and the relative performance of different model selection approaches.

For our experiments, we use a Gaussian kernel, with the parameter $\gamma = 1$ chosen based on experimentation, to give flexible but not overly jumpy fits. We demonstrate the fit and accuracy of model selection using CV compared to using SIC. For CV, we used 50 of the 263 players in the data set for validation (selection of $\hat{\lambda}(\tau)$) and 50 more for testing the accuracy of the resulting model. Thus, 163 examples were used for training. For SIC, we used 213 (training+validation) as the training set, and applied Equation (13) for selecting $\hat{\lambda}(\tau)$. Both approaches were evaluated using the 50 test observations. In Figure 6 we show the resulting fit in both approaches, for three different quantiles. As expected, compensation seems to be monotone in performance (home runs) but not in experience (salary tends to increase as players gain experience, but then decreases as they get older and performance deteriorates). As we can see, the model-selected surfaces are quite similar between CV and SIC, though this need not be the case, as we should keep in mind that SIC is choosing between models trained on more data. In terms of accuracy on the test set (shown above each plot), The results are also very comparable. When comparing the two approaches we should also keep in mind the reduced complexity of applying SIC, and the existing literature on instability of CV-based model selection, though this is not evident in our results.

For the California housing data set, we use only longitude and latitude as the two explanatory variables in fitting KQR, to facilitate meaningful visualization of results. We model the log of the median price, since the actual median fluctuates widely over the data. We use 500 observations for training and 50 as validation for CV, 550 as training for SIC, and 500 additional observations for testing. Figure 7 shows the results. It is clear that visualization is hampered by the fact that California is far from being rectangular, so one corner of the plots (latitude 34N, longitude 122W) is well inside the ocean, while the other (latitude 40N, longitude 115W) is well inland from the California border. The wild extrapolation of the fit in that direction is therefore not informative.

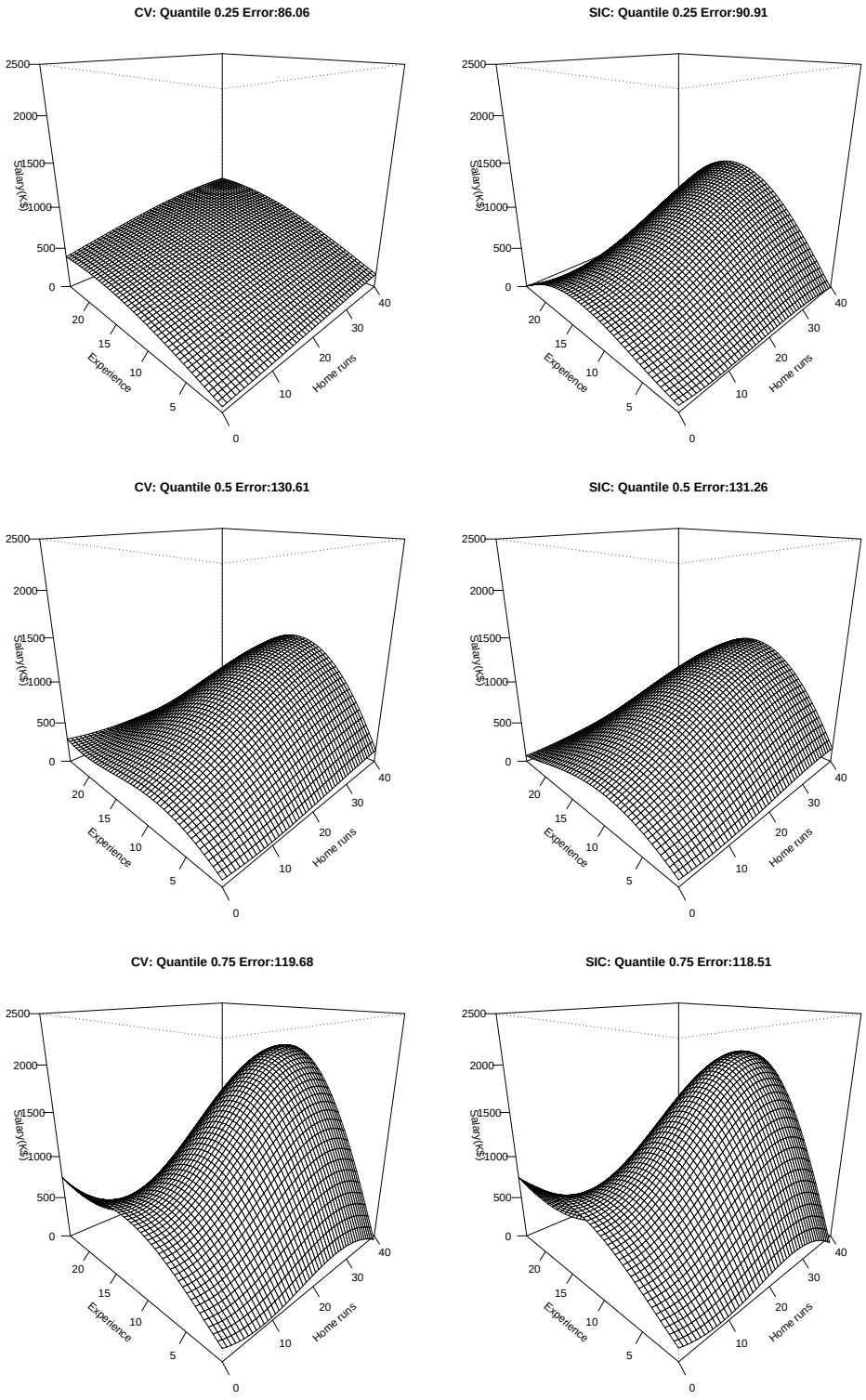

Figure 6: Models selected using CV (left) and SIC (right) on the Baseball data, for three different quantiles.

On each plot the fit at San Francisco (red circle), Los Angeles (blue square) and Sacramento (green triangle) are marked. We can see that the selected fits using CV and SIC are quite similar, with possible exception to the more jumpy fit selected by SIC for quantile 0.5. The valid insights that seem to arise out of these plots relate to the lower house values in the central valley of California compared to the coastal area, and the reduced house values in the Sacramento area compared to near by the Bay Area.

7. Conclusions and Future Work

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 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. Characterization of a family of non-convex optimization problems 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.

We have shown how our approach can be extended to other modeling problems with a parameterized loss function, such as SVR, and to other versions of KQR, including using in-sample model selection criteria and enforcing monotonicity on the resulting quantiles.

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, and more extensive empirical studies.

It is of particular interest to us to investigate the bias-variance tradeoff in loss function selection. As we have mentioned, modeling with the quantile loss function L_τ leads to estimation of the τth quantile of $P(Y|x)$ in the *decision theoretic* sense that the population optimizer of the loss function is this quantile (see Equation 2). However, this by no means guarantees that a model learned from finite data using L_τ (with or without regularization) will do well in predicting the τth quantile. In particular, there is no guarantee that a model built using a different loss function (say, L_η, $\eta \neq \tau$) will not do better in predicting this quantile. This can be thought of in terms of bias and variance, where the model generating quantile η is similar enough to the one for quantile τ (i.e., bias is small), but it is "easier" to learn with L_η, that is, variance is smaller, which would typically be the case if η is closer to $1/2$ than τ (Koenker, 2005). A detailed investigation of this question is outside the scope of the current work, but will be a natural extension.

A particularly important and difficult type of quantile estimation problems pertains to estimation of *extreme* quantiles (e.g., $\tau = 0.01$ or $\tau = 0.99$) which can serve as approximations for expected extreme values of the function being estimated. These problems are typically very difficult *statistically*, that is, hard because of the scarcity of information implicit in trying to estimate events we rarely observe. However they are not expected to be particularly difficult algorithmically. That is, our (and others') KQR approaches can estimate these models, but it is not clear how useful the results are. These observations are verified by our limited experiments (results not shown), which yield very "jumpy" and unstable models for extreme quantiles.

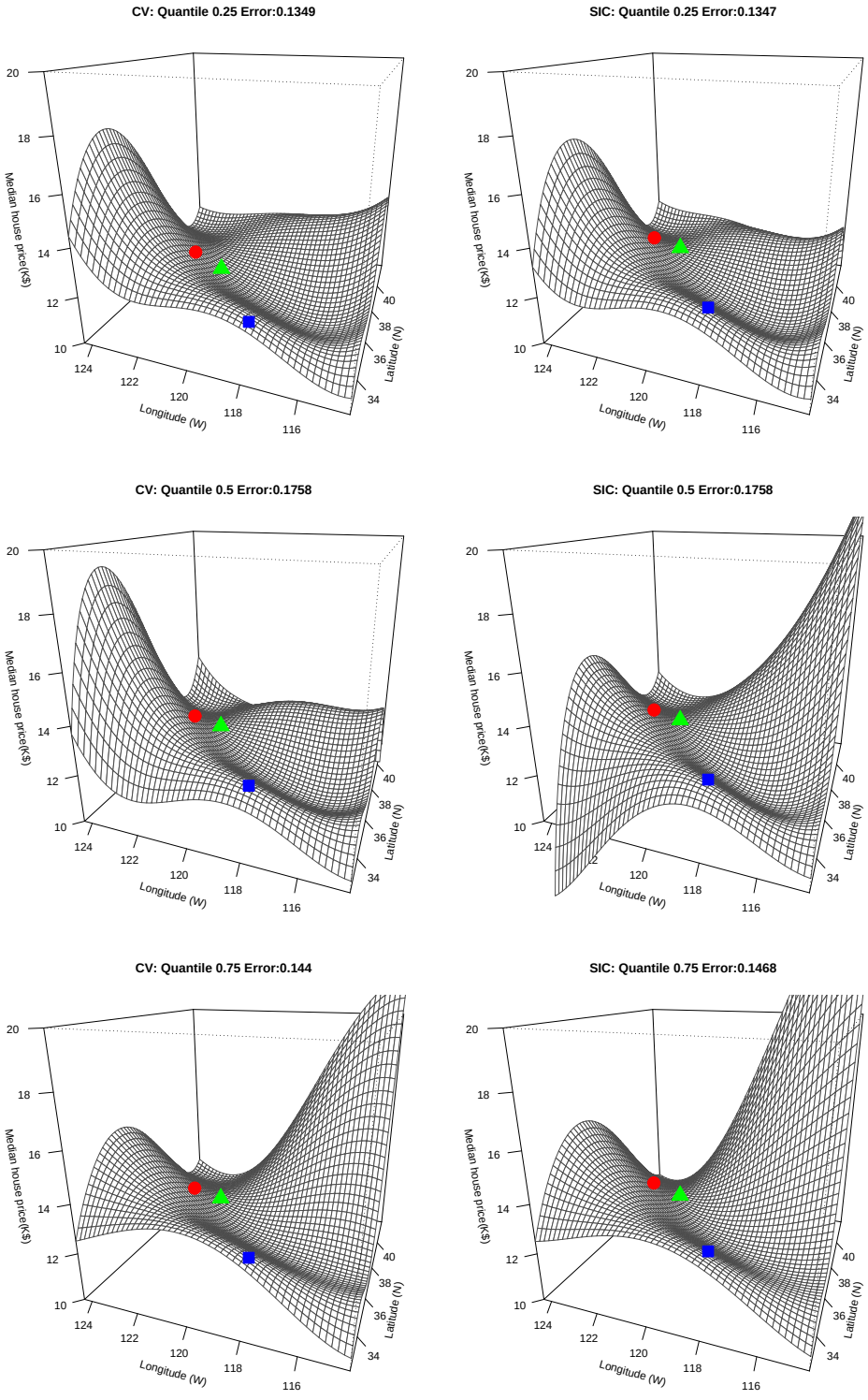

Figure 7: Synthetic maps of the models selected using CV (left) and SIC (right) on the CA housing data, for three different quantiles. The fits at San Francisco (red circle), Los Angeles (blue square) and Sacramento (green triangle) are marked on each map.

Acknowledgments

The author thanks Ramesh Natarajan for useful early discussions, Ji Zhu for help with the KQR code of Li et al. (2007), and the action editor, two anonymous referees and Ronny Luss for their useful comments.

Appendix A. Pseudo-code of Algorithm

Algorithm 2 and its accompanying procedures describe our implementation in some detail. This pseudo code is meant to complete the implementation details given in the paper. We use mathematical notation rather than programming commands as much as possible, to make understanding easier. Given the complexities and intricacies involved in the complete implementation, it seems unrealistic and probably non-useful to give an exhaustive description. Rather, we concentrate on clarifying the general flow of the algorithm and the mathematical problems it solves at each step. We also emphasize the aspects of the algorithm not covered in technical detail in the main text, such as the differentiation of different types of events (knot crossing, knot merging, knot splitting). Where the text offers the technical content, we simply refer to that point. For example, Theorem 2 describes the direction calculation and also implicitly the accounting orf the identities of the sets $\mathcal{E}, \mathcal{L}, \mathcal{R}$ required for it. We thus simply refer back to it where relevant in the algorithm.

Some further comments on the pseudo code:

- We assume the training and validation data are "global variables" known to all procedures.
- We use $\hat{f}$ and $\hat{\theta}$ interchangeably, given formula (5).
- Some of the elements are not described in the most efficient implementation, which would require a lot more accounting and data management. For example, the search for the minima in the function UpdateValidList does not have to be done from scratch on every call, but a list can be maintained, and only the necessary items updated.
- The pseudo-code glosses over numerical issues which plague the actual implementation. In particular, all equalities must have "tolerance" in the practical implementation due to machine rounding errors. This obviously creates a problem when events on the path are bunched together close enough that two distinct events fall within this tolerance.
- We avoid repetition of similar procedures. Thus the call to function *KnotSplit* at end of Algorithm 2 is replaced with a brief explanation of its near-identity to the function *KnotCross* which is already given.

Algorithm 2: Algorithm description

Input: The entire λ-solution path for quantile τ_0 characterized by its m knots $\boldsymbol{\lambda} = \lambda_1, ..., \lambda_m$; the solution directions $\mathbf{g} = g_1, ..., g_m$ as defined in (6); and $\mathbf{i} = i_1, ..., i_m$ the observations which "hit the elbow" at every knot

Output: Cross-validated solutions $f^*(\tau)$ for $\tau \in [\tau_0, \tau_{\text{end}}]$ described as set of intervals in the variable OPT

`/* Initialization: find validation crossings, calculate cross validation loss at knots and validation crossings, sort by it, find future meetings of neighbors on the list, where the order changes */`

Set M = InitializeValidList($\tau_0, \boldsymbol{\lambda}, \mathbf{g}$);

Set $OPT = (\tau_0, f^*(\tau_0), h^*)$ where $f^*(\tau_0) = \hat{f}(\tau_0, M.\lambda_1)$, $h^* = M.h_1$ are from the first (smallest loss) entry in M;

Set $\tau_{\text{now}} = \tau_0$;

Let T be the list of the fits $\mathbf{f} = (\hat{f}(\tau_0, \lambda_1), ..., \hat{f}(\tau_0, \lambda_m))$, regularization values $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_m)$, rates $\mathbf{c} = (c_1, ..., c_m)$, and directions $\mathbf{h} = (h_1, ..., h_m)$ as defined in Theorem 2; `/* Main loop */`

while $\tau_{\text{now}} < \tau_{\text{end}}$ **do**

  Update $\{\tau_{\text{new}}, k_{\text{new}}, i_{\text{new}}, \text{type}\}$ = FindEvent(T);

  Update $T.\lambda_k = T.\lambda_k + (\tau_{\text{new}} - \tau_{\text{now}})T.c_k$ for $k = 1, ..., m$;

  Update $T.f_k$ according to (6);

  while $\tau_{\text{now}} < \tau_{\text{new}}$ **do**

    Set $\tau_{\text{keep}} = \tau_{\text{now}}$;

    (M, change, τ_{now}) = UpdateValidList($M, \tau_{\text{keep}}, \tau_{\text{new}}$);

    if *change=TRUE* **then**

      $OPT = concatenate(OPT, (\tau_{\text{now}}, f^*(\tau_{\text{now}}), h^*))$ where $f^*(\tau_{\text{now}}) = f(\tau_{\text{now}}, M.\lambda_1)$, $h^* = M.h_1$ are from the first (smallest loss) entry in M;

    end

  end

  if *type* = *cross* **then** `/* Knot crossing of knots` k_{new}, $k_{\text{new}}+1$ `*/`

    Set T = KnotCross($T, k_{\text{new}}, \tau_{\text{now}}$);

  else if *type* = *merge* **then** `/* Knot merge of` k_{new}, $k_{\text{new}}+1$, $k_{\text{new}}+2$ `*/`

    Remove knots k_{new}, $k_{\text{new}+2}$ from T;

    Update sets $\mathcal{E}, \mathcal{R}, \mathcal{L}$ for the remaining knot (Move the observation which defined the two removed knots from $\mathcal{E}$ to $\mathcal{L}$ or $\mathcal{R}$);

    else `/* Knot split of knot` k_{new} `with observation` i_{new} `*/`

      `/* Function` ***`KnotSplit`*** `would be identical to` ***`KnotCross`*** `---identify two observations at border and find legal directions---except that a split situation yields three such directions, hence three knots, while a cross situation yields two */`

end

Procedure "`IntializeValidList`": Initialization of bi-level candidate list

Input: Initial value τ_0, vector of knot values $\boldsymbol{\lambda}$, corresponding directions $\mathbf{g}$

Output: A list $M = \{(r_k, \lambda_k, l_k, h_k, \tau_k) : k = 1, ..., m+v\}$ sorted by $l_1 \leq l_2 \leq .... \leq l_{m+v}$, where:

r_k is an indicator in $\{knot, valx\}$ whether this is a knot or a validation crossing

λ_k is its "location" on the path

l_k is its cross validation loss

h_k is its direction

τ_k is knot meeting point

Find knot directions $h_1, ..., h_m$ according to Theorem 2;

`/* Identify all validation crossings in the solution path for` τ_0 `*/`

Set $V = \Phi$ the empty set;

for $k = 1, ..., m$ *knots and* $i = 1, ..., N$ *validation observations* **do**

  if $\hat{f}(\tau_0, \lambda_{k-1})(\tilde{\mathbf{x}}_i) > \tilde{y}_i$ *and* $\hat{f}(\tau_0, \lambda_k)(\tilde{\mathbf{x}}_i) < \tilde{y}_i$ *or vice versa* **then**

    Set $\tilde{\lambda} = \lambda_k \frac{\hat{f}(\tau_0, \lambda_k)(\tilde{\mathbf{x}}_i) - g_k(\tilde{\mathbf{x}}_i)}{\tilde{y}_i - g_k(\tilde{\mathbf{x}}_i)}$;

    Find the validation crossing direction $\tilde{h}(\mathbf{x})$ according to Proposition 5;

    Add the entry $(\tilde{\lambda}, \tilde{h})$ characterizing the validation crossing to the set V;

  end

end

`/* Sort knots and validation crossings by their loss */`

Denote the number of validation crossings by $v = |V|$;

for $k = 1, ..., m$ **do**

  Calculate knot validation loss: $l_k = \sum_{i=1}^{N} L_{\tau_0}(\hat{f}(\tau_0, \lambda_k)(\tilde{\mathbf{x}}_i), \tilde{y}_i)$;

end

for $k = 1, ..., v$ **do**

  Calculate validation crossing loss: $l_{m+k} = \sum_{i=1}^{N} L_{\tau_0}(\hat{f}(\tau_0, \tilde{\lambda}_k)(\tilde{\mathbf{x}}_i), \tilde{y}_i)$;

end

Create list $M = \{(r_k, \lambda_k, l_k, h_k, \tau_k) : k = 1, ..., m+v\}$ sorted by $l_1 \leq l_2 \leq .... \leq l_{m+v}$, where:

  r_k is an indicator whether this is a knot or a validation crossing with possible values *knot* and *valx* respetively

  λ_k is its "location" on the path

  l_k is its cross validation loss

  h_k is its direction

  τ_k is knot meeting point, defined below;

`/* Identify mtg pts of neighboring knots or valid. crossings */`

for $k = 1, ..., m+v-1$ **do**

  Let $\tau_k = \tau_0 + \delta_k$ where δ_k is the minimal positive solution of the Problem (11) with $l = k+1$;

end

Procedure "FindEvent": Find the next event on the path as τ changes

Input: The list T of knots and their directions

Output: Event type in $\{$*cross, merge, birth*$\}$, τ_{new} where next event happens, k_{new} the knot where this event happens, i_{new} the observation involved in the event (if a birth)

/* Next knot crossing or knot merging */

Set $\tilde{\tau}_k = \frac{T.\lambda_k - T.\lambda_{k+1}}{T.c_{k+1} - T.c_k}$, $k = 1, ..., m-1$;

Set $k_{\text{new}} = \arg\min_{k=1,...,m-1}\{\tilde{\tau}_k : \tilde{\tau}_k > 0\}$;

Set $\tau_{\text{new}} = \tilde{\tau}_{k_{\text{new}}}$ **if** $\tilde{\tau}_{k_{\text{new}}} = \tilde{\tau}_{k_{\text{new}}+1}$ **then**

    type=*merge* ; /* Knots merging $3 \Rightarrow 1$ */

else

    type=*cross*; /* Two knots crossing */

end

/* Next observation-knot crossing = knot birth */

for $k = 1, ..., m$ **do**

    Set $i'_k = \arg\min_{i=1,...,n}\{\frac{T.\lambda_k(\hat{f}(T.\lambda_k)(\mathbf{x}_i) - y_i)}{T.c_k(y_i - h_k(\mathbf{x}_i))} : \frac{T.\lambda_k(\hat{f}(T.\lambda_k)(\mathbf{x}_i) - y_i)}{T.c_k(y_i - h_k(\mathbf{x}_i))} > 0\}$;

    Set τ'_k to be the minimum attained;

end

Set $k' = \arg\min_k \tau'_k$;

if $\tau'_{k'} < \tau_{\text{new}}$ **then**

    Set $\tau_{\text{new}} = \tau'_{k'}$, $k_{\text{new}} = k'$, type=*birth* ;

end

Procedure "`UpdateValidList`": Find the next validation event on the path as τ changes, and update the list if necessary

Input: Validation candidate list M, current value τ_{keep}, next event on main path τ_{new}
Output: Logical indicator *change* whether optimum changed, Updated list M, and τ_{now} where next validation event happens

Set change = FALSE ;
`/* Pair of validation crossings can disappear, or a new a validation crossing can appear, or a regular order change in the elements in` M `can occur. We first identify the next order change in the list M */`
Set $\tau_{\text{now}} = \min_{k=1,\ldots,m+v-1} M.\tau_k$;
Set $k_{\text{now}} = \arg\min_{k=1,\ldots,m+v-1} M.\tau_k$;
`/* Now find the next time a validation observation hits a knot` $\Rightarrow$ `new validation crossing */`
For $i = 1,\ldots N$ and $k = 1\ldots,m$ set $\Delta\tau(i,k) = M.\lambda_k \frac{\hat{f}(\tau_{\text{keep}}, M.\lambda_k)(\tilde{\mathbf{x}}_i) - \tilde{y}_i}{\tilde{y}_i M.c_k - M.h_k(\tilde{\mathbf{x}}_i)}$;
Set $\tilde{\tau} = \tau_{\text{keep}} + \min_{i=1,\ldots,N,k=1,\ldots,m}\{\Delta\tau(i,k) : \Delta\tau(i,k) > 0\}$;
Set $(\tilde{i},\tilde{k}) = \arg\min_{i=1,\ldots,N,k=1,\ldots,m}\{\Delta\tau(i,k) : \Delta\tau(i,k) > 0\}$;
if $\tau_{\text{now}} > \tau_{\text{new}}$ *and* $\tilde{\tau} > \tau_{\text{new}}$ **then** `/* No validation event before` τ_{new} `*/`
    $\tau_{\text{now}} = \tau$new;
    return;
else if $\tau_{\text{now}} > \tilde{\tau}$ **then** `/* New validation xing appears---add it to list */`
    Set $\tilde{\lambda} = (\tilde{\tau} - \tau_{\text{keep}}) M.c_{\tilde{k}} + M.\lambda_{\tilde{k}}$;
    Set $\tilde{l} = \sum_{i=1}^{N} L_{\tilde{\tau}}(\hat{f}(\tilde{\tau},\tilde{\lambda})(\tilde{\mathbf{x}}_i), \tilde{y}_i)$;
    Set the validation xing direction $\tilde{h}(\mathbf{x})$ according to Proposition 5;
    Find the location k' in the sorted list of the cross validation loss $\tilde{l}$ and insert the element $(r = valx, \tilde{\lambda}, \tilde{l}, \tilde{h})$ into M at location k' ;
    Recalculate $\tau_{k'-1}, \tau_{k'}$ in M according to (11);
    Set $\tau_{\text{now}} = \tilde{\tau}$;
else
    `/* If two validation crossings meet knot---the two disappear */`
    Set $(merged, M)$ = CheckMerge(M, k_{now});
    if *merged=FALSE* **then** `/* Usual situation: swap elements, update meetings */`
        Swap elements k_{now} and $k_{\text{now}} + 1$ in M;
        Recalculate $\tau_{k_{\text{now}}-1}, \tau_{k_{\text{now}}}, \tau_{k_{\text{now}}+1}$ in M according to (11);
        if $k_{\text{now}} = 1$ **then** `/* First element changed` $\Rightarrow$ `change of optimum */`
            changed = TRUE;
        end
    end
end

Procedure "CheckMerge": Find out if the validation event is in fact two validation crossings of same observation meeting a knot and merging

Input: Validation candidate list M, index of event point k

Output: Logical indicator *merge* whether a merge occurred, updated list M

merge = FALSE;
/* With observations in general location, $\tau_k = \tau_{k+1}$ in M implies immediately that we have a merge. Two of $k, k+1, k+2$ are validation crossings of the same observation, the knot is the third involved in the crossing. If we do not assume that, more checks are required! */
if $M.\tau_k = M.\tau_{k+1}$ **then**
    merge = TRUE ;
    /* Find out which one of the entries $k, k+1, k+2$ in M is a knot, delete the other two */
    if $M.r_k =$ knot **then**
        remove entries $k+1, k+2$ from M;
    else if $M.r_{k+1} =$ knot **then**
        remove entries $k, k+2$ from M ;
    else /* $M.r_{k+2} =$ knot */
        remove entries $k, k+1$ from M ;
    Update $M.\tau_{k-1}, M.\tau_k$ according to (11);
end

Procedure "KnotCross": Update directions when knots cross

Input: Knot list T, index of first of crossing knots k, current quantile τ

Output: Updated list T

/* Identify i_1, i_2, the ''knot'' observations at the two knots */

Find i_1, i_2 s.t. $\theta_{i_j} \in \{\tau, -(1-\tau)\}$ and $y_{i_j} = \hat{f}(\tau, T.\lambda_{k+j-1})(\mathbf{x}_{i_j})$ for $j \in \{1,2\}$;

Calculate the sets $\mathcal{E}, \mathcal{R}, \mathcal{L}$ as defined in the text for the meeting knots (leaving out the "border observations" i_1, i_2);

Set $u = k$ **for** rel $= 1,2$ **do** /* try releasing each border observation to $\mathcal{E}$ or $\mathcal{L}$ or $\mathcal{R}$ as appropriate */

  Add observation i_{rel} to $\mathcal{E}$ and calculate direction h, c according to Theorem 2;

  /* Check sign and magnitude of $b_{i_{\text{rel}}}$ for consistency (to maintain $\theta_{i_{\text{rel}}} \in [-(1-\tau), \tau]$ as τ increases) */

  if $b_{i_{\text{rel}}} \leq 1$ *and* $\hat{\theta}_{i_{\text{rel}}} = \tau$ **then**

    Update entry u in T with this direction h, c, set $u = u+1$;

  else if $b_{i_{\text{rel}}} \geq -1$ *and* $\hat{\theta}_{i_{\text{rel}}} = -(1-\tau)$ **then**

    Update entry u in T with this direction h, c, set $u = u+1$;

  end

  if $\hat{\theta}(i_{\text{rel}}) = \tau$ **then**

    Add i_{rel} to $\mathcal{R}$;

  else /* $\hat{\theta}(i_{\text{rel}}) = -1-\tau$ */

    Add i_{rel} to $\mathcal{L}$;

  Calculate direction h and c according to Theorem 2;

  /* Check sign and magnitude of $h(\mathbf{x}_{i_{\text{rel}}})$ for sign consistency */

  if $h(\mathbf{x}_{i_{\text{rel}}}) < 0$ *and* $\hat{\theta}_{i_{\text{rel}}} = \tau$ **then**

    Update entry u in T with this direction h, c, set $u = u+1$;

  else if $h(\mathbf{x}_{i_{\text{rel}}}) > 0$ *and* $\hat{\theta}_{i_{\text{rel}}} = -(1-\tau)$ **then**

    Update entry u in T with this direction h, c, set $u = u+1$;

  end

end

References

M. Buchinsky. Changes in the u.s. wage structure 1963-1987: Application of quantile regression. *Econometrica*, 62(2):405–458, Mar. 1994.

A. Christmann and I. Steinwart. Consistency of kernel-based quantile regression. *Applied Stochastic Models in Business and Industry*, 24(2):171–183, 2008. ISSN 1524-1904.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32 (2):407–499, 2004.

E. Eide and M.H. Showalter. The effect of school quality on student performance: A quantile regression approach. *Economics Letters*, 58(3):345–350, Mar. 1998.

L. Gunther and J. Zhu. Efficient computation and model selection for the support vector regression. *Neural Computation*, 19(6), 2005.

T. Hastie, R. Tibshirani, and J. Friedman. *Elements of Statistical Learning*. Springer, 2001.

T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. The entire regularization path of the support vector machine. *JMLR*, 5:1391–1415, Oct 2004.

X. He, P. Ng, and S. Portnoy. Bivariate quantile smoothing splines. *Journal of the Royal Statistical Society, Ser. B*, 60:537–550, 1998.

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

R. Koenker. *Quantile Regression*. New York : Cambridge University Press, 2005.

G. Kunapuli, K.P. Bennett, J. Hu, and J.-S. Pang. Bilevel model selection for support vector machines. In Pierre Hansen and Panos Pardalos, editors, *Data Mining and Mathematical Programming [CRM Proceedings and Lecture Notes]*, volume 45. American Mathematical Society, 2008.

Y. Li, Y. Liu, and J. Zhu. Quantile regression in reproducing kernel hilbert spaces. *JASA*, 102(477), 2007.

D. Mease, A.J. Wyner, and A. Buja. Boosted classification trees and class probability/quantile estimation. *JMLR*, 8:409–439, Oct 2007.

N. Meinshausen. Quantile regression forests. *JMLR*, 7:983–999, Jun 2006.

R. K. Pace and R. Barry. Sparse spatial autoregressions. *Statistics and Probability Letters*, 33: 291–297, 1997.

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

S. Rosset. Bi-level path following for cross validated solution of kernel quantile regression. In *Proceedings of the 25th international conference on Machine Learning*, 2008.

S. Rosset and J. Zhu. Piecewise linear regularized solution paths. *Annals of Statistics*, 35(3), 2007.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

M. Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and their Geometric Applications*. Cambridge University Press, 1995.

J. Sherman and W.J. Morrison. Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix. *Annals of Mathematical Statistics*, 20:621, 1949.

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

I. Steinwart and A. Christmann. How SVMs can estimate quantiles and the median. In *Neural Information Processing Systems 20*, pages 305–312, 2008.

I. Takeuchi, Q.V. Le, T.D. Sears, and A.J. Smola. Nonparametric quantile estimation. *JMLR*, 7: 1231–1264, Jul 2006.

I. Takeuchi, K. Nomura, and T. Kanamori. Nonparametric conditional density estimation using piecewise-linear solution path of kernel quantile regression. *Neural Computation*, 21(2):533–559, 2009.

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

J. Wang, X. Shen, and Y. Liu. Probability estimation for large margin classifiers. *Biometrika*, 95 (1):149–167, 2008.

M. Yuan. GACV for quantile smoothing splines. *Computational Statistics and Data Analysis*, 5: 813–829, 2006.

H. Zou, T. Hastie, and R. Tibshirani. On the degrees of freedom of the Lasso. *Annals of Statistics*, 35:2173–2192, 2007.