Integrating Uncertainty Awareness into Conformalized Quantile Regression

Raphael Rossellini
University of Chicago

Rina Foygel Barber
University of Chicago

Rebecca Willett
University of Chicago

Abstract

Conformalized Quantile Regression (CQR) is a recently proposed method for constructing prediction intervals for a response Y given covariates X, without making distributional assumptions. However, existing constructions of CQR can be ineffective for problems where the quantile regressors perform better in certain parts of the feature space than others. The reason is that the prediction intervals of CQR do not distinguish between two forms of uncertainty: first, the variability of the conditional distribution of Y given X (i.e., aleatoric uncertainty), and second, our uncertainty in estimating this conditional distribution (i.e., epistemic uncertainty). This can lead to intervals that are overly narrow in regions where epistemic uncertainty is high. To address this, we propose a new variant of the CQR methodology, Uncertainty-Aware CQR (UACQR), that explicitly separates these two sources of uncertainty to adjust quantile regressors differentially across the feature space. Compared to CQR, our methods enjoy the same distribution-free theoretical coverage guarantees, while demonstrating in our experiments stronger conditional coverage properties in simulated settings and real-world data sets alike.

1 INTRODUCTION

While machine learning approaches in recent years have frequently succeeded in producing models that produce predictions with low error on average, there are many instances in which these models can provide a false sense of precision. Prediction intervals are one of the most natural ways to quantify uncertainty, providing bounds on the true response that hold with some desired probability, frequently 90%. To this end, conformal prediction has emerged as an attractive paradigm for creating statistically-valid prediction intervals for black-box machine learning models with no distributional assumptions. Specifically, suppose we have observed n samples $(X_i, Y_i) \in \mathbb{R}^p \times \mathbb{R}$, and we desire a prediction interval for Y_{n+1} based on X_{n+1}. If the $n+1$ samples are exchangeable (e.g., if the pairs (X_i, Y_i) are drawn i.i.d. from an arbitrary shared distribution), conformal prediction allows one to construct an interval $\hat{C}_n(X_{n+1})$ such that

$$\mathbb{P}\left\{Y_{n+1} \in \hat{C}_n(X_{n+1})\right\} \geq 1-\alpha \tag{1}$$

A remaining challenge is the ability to provide prediction intervals that hold conditionally on the test point's covariates,

$$\mathbb{P}\left\{Y_{n+1} \in \hat{C}_n(X_{n+1}) \mid X_{n+1} = x\right\} \geq 1-\alpha,$$

which is not ensured by conformal prediction under the exchangeability assumption alone. Unfortunately, it is impossible for any distribution-free approach to guarantee conditional coverage in a non-trivial way when the covariates follow a continuous distribution (Vovk et al., 2005; Lei and Wasserman, 2014; Barber et al., 2021).

In spite of this impossibility result, there have been numerous methodological advances that allow conformal prediction to provide better conditional coverage properties empirically. A particularly notable advancement has been Conformalized Quantile Regression (CQR) (Romano et al., 2019), which uses quantile regressors as inputs into the conformal prediction paradigm to create prediction intervals that adapt to the covariates. Part of the appeal is that the true conditional quantile functions are a sufficient ingredient to achieving conditional coverage. If we let $q_{Y|X}(x;\alpha_{\text{lo}})$, $q_{Y|X}(x;\alpha_{\text{hi}})$ be the *true* $\alpha_{\text{lo}}, \alpha_{\text{hi}}$ conditional quantiles for $Y \mid X = x$, where $\alpha_{\text{lo}} = \frac{\alpha}{2}$ and $\alpha_{\text{hi}} = 1 - \frac{\alpha}{2}$, then by construction $\mathbb{P}\left\{Y \in [q_{Y|X}(X;\alpha_{\text{lo}}), q_{Y|X}(X;\alpha_{\text{hi}})] \mid X = x\right\} \geq 1-\alpha$.

Proceedings of the 27[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

Thus, if one can find accurate quantile regressors via machine learning, CQR holds potential to create prediction intervals that approximately have conditional coverage while maintaining the marginal validity of previous conformal prediction methods. Specifically, for any $a \in (0,1)$, let $\hat{q}_{Y|X}(x,a)$ denote an estimate of the conditional a-quantile of $Y \mid X = x$ computed using training data. CQR generates prediction intervals of the form

$$\hat{C}_n(X_{n+1}) = [\hat{q}_{Y|X}(X_{n+1};\alpha_{\text{lo}}) - t, \hat{q}_{Y|X}(X_{n+1};\alpha_{\text{hi}}) + t]$$

where calibration data is selected to choose t.

In practice, however, our estimate $\hat{q}_{Y|X}(x,a)$ may be less accurate in particularly challenging regions of the feature space. CQR struggles in this setting since it simply inflates the initial estimates by a constant, non-adaptive, additive adjustment $t \in \mathbb{R}$.

Others have implemented adaptive variants. For example, the CQR-r variant from Sesia and Candès (2020) multiplies t by the interval width $w(x) = \hat{q}_{Y|X}(x;\alpha_{\text{hi}}) - \hat{q}_{Y|X}(x;\alpha_{\text{lo}})$, which yields:

$$\hat{C}_n(X_{n+1}) = [\hat{q}_{Y|X}(X_{n+1};\alpha_{\text{lo}}) - t \cdot w(X_{n+1}), \\ \hat{q}_{Y|X}(X_{n+1};\alpha_{\text{hi}}) + t \cdot w(X_{n+1})]$$

The issue here is that this is scaling t by the wrong type of uncertainty. $w(x)$ is a proxy for *aleatoric uncertainty*—the irreducible uncertainty inherent to a data generating process, i.e., the amount of variability in the conditional distribution of $Y \mid X$. However, the gap between the upper and lower quantile regressors already accounts for aleatoric uncertainty. The additive adjustments should ideally reflect the *epistemic uncertainty*—the scale of the estimation error of the quantile regressors at each X:

$$|\hat{q}_{Y|X}(x,a) - q_{Y|X}(x,a)|, \ a \in \{\alpha_{\text{lo}}, \alpha_{\text{hi}}\}$$

The aleatoric uncertainty may be associated with epistemic uncertainty in many contexts, but there is no guarantee that there must be a correspondence between the two. In medical studies, for instance, a poorly-represented demographic could have a narrower range possible of outcomes than a well-represented one, meaning that this subpopulation has high epistemic uncertainty yet low aleatoric uncertainty. Our work centers on the idea that, for this type of setting, the quantile-based prediction interval should be inflated in a way that scales with how well sampled each demographic is (epistemic uncertainty), rather than with the estimated range of outcomes for each demographic (aleatoric uncertainty).

1.1 Our contribution

We propose a new family of methods for conformalizing quantile regression that incorporate epistemic uncertainty. We call them *UACQR-S (Uncertainty-Aware CQR via Scaling)* and *UACQR-P (Uncertainty-Aware CQR via Percentiles)* [1] UACQR-S creates prediction intervals of a similar form to CQR-r, but its scaling function directly estimates the epistemic, rather than aleatoric, uncertainty in $Y \mid X$ at each value of X. In contrast, UACQR-P is constructed based on an ensemble of quantile regression estimates (e.g., obtained via bootstrapping the training sample), and outputs a prediction interval that is defined as a calibrated percentile of the B different estimates at each point. Our methods offer the same distribution-free guarantee of validity as existing CQR constructions, while offering empirical improvements in terms of conditional coverage.

To minimize computational concerns, we provide recipes for implementing UACQR-S and UACQR-P that have the same computational cost of CQR when using any one of a broad range of machine learning paradigms, namely bagging (which includes Random Forests) and epoch-based optimization (e.g., neural networks). These two classes cover a large proportion of successful modern machine learning architectures, so our proposed methods hold potential for widespread applications.

2 EXISTING METHODS

We review split conformal prediction (Vovk et al., 2005) in more depth, and define a generic meta-algorithm that captures all existing variants of CQR.

Suppose we are given a data set $(X_1, Y_1), \ldots, (X_n, Y_n)$ and a new test point X_{n+1} for which we would like to predict the response, Y_{n+1}. We limit ourselves to assuming only that $(X_1, Y_1), \ldots, (X_{n+1}, Y_{n+1})$ are exchangeable. In its most basic version, split conformal prediction operates as follows. We split the n data points into a training set and calibration set of sizes $n_0 + n_1 = n$. After fitting a predictive model $\hat{\mu}$ to the training set $\{(X_i, Y_i) : i = 1, \ldots, n_0\}$, we then define

$$\widehat{C}_n(X_{n+1}) = \hat{\mu}(X_{n+1}) \pm \hat{t}, \tag{2}$$

where $\hat{t}$ is the $\lceil (1-\alpha)(n_1+1) \rceil$-th smallest element of the calibration set residuals $\{|Y_i - \hat{\mu}(X_i)| : i = n_0 + 1, \ldots, n\}$.

Split conformal prediction is much more general than this simple construction, however, and in particular it can be used to produce prediction intervals of

[1] We provide code implementing UACQR at https://github.com/rrross/UACQR

any "shape" rather than restricting ourselves only to constant-width type intervals of the form $\hat{\mu}(x) \pm$ (constant). Gupta et al. (2022) describe a formulation based on nested sets: suppose that, given the training data $\{(X_i, Y_i) : i = 1, \ldots, n_0\}$, we construct a family of nested prediction bands $\hat{C}_{n_0,t}$ (where $\hat{C}_{n_0,t}(x) \subseteq \mathbb{R}$ is a prediction set or prediction interval at each x), indexed over some set $t \in \mathcal{T} \subseteq \mathbb{R}$. This family is required to be nested, with $\hat{C}_{n_0,t}(x) \subseteq \hat{C}_{n_0,t'}(x)$ for $t < t'$, and also that $\inf_{t\in\mathcal{T}} \hat{C}_{n_0,t}(x) = \emptyset$ and $\sup_{t\in\mathcal{T}} \hat{C}_{n_0,t}(x) = \mathbb{R}$. Then, using the calibration set, define

$$\hat{t} = \inf\left\{ t : \sum_{i=n_0+1}^{n} \mathbf{1}_{Y_i \in \hat{C}_{n_0,t}(X_i)} \geq (1-\alpha)(n_1+1) \right\}, \tag{3}$$

and return the prediction set $\hat{C}_n(X_{n+1}) := \hat{C}_{n_0,\hat{t}}(X_{n+1})$. To summarize, the training points ($i = 1, \ldots, n_0$) define a nested family of prediction bands $\{\hat{C}_{n_0,t}\}$, and then the calibration set ($i = n_0 + 1, \ldots, n_1$) is used to choose a particular $\hat{t}$ that selects one prediction band from this family to then provide at least $1-\alpha$ coverage of the final prediction interval. Vovk et al. (2005) and Gupta et al. (2022) show any construction of this form offers the distribution-free prediction guarantee (1). To see how this general formulation applies to the simple construction (2) given above, we can simply take $\hat{C}_t(x) = \hat{\mu}(x) \pm t$ for $t \in \mathcal{T} = \mathbb{R}$ to recover the prediction set given in (2).

To relate this construction to the problem of quantile regression, we change the notation to provide prediction sets that are specifically designed to be intervals. Let $\hat{q}_{\mathrm{lo}}(X, t)$ and $\hat{q}_{\mathrm{hi}}(X, t)$ be quantile regressors for $Y \mid X$, fitted using the training data $\{(X_i, Y_i) : i = 1, \ldots, n_0\}$, where $t \mapsto \hat{q}_{\mathrm{lo}}(x, t)$ is decreasing, while $t \mapsto \hat{q}_{\mathrm{hi}}(x, t)$ is increasing. We consider the nested family of sets given by $\hat{C}_{n_0,t}(x) = [\hat{q}_{\mathrm{lo}}(x, t), \hat{q}_{\mathrm{hi}}(x, t)]$, and consequently, split conformal prediction returns the interval

$$\hat{C}_n(X_{n+1}) = \hat{C}_{n_0,\hat{t}}(X_{n+1}) = [\hat{q}_{\mathrm{lo}}(X_{n+1}, \hat{t}), \hat{q}_{\mathrm{hi}}(X_{n+1}, \hat{t})] \tag{4}$$

where, for this specific construction, the definition (3) of $\hat{t}$ simplifies to

$$\inf\left\{ t : \sum_{i=n_0+1}^{n} \mathbf{1}_{\hat{q}_{\mathrm{lo}}(X_i,t) \leq Y_i \leq \hat{q}_{\mathrm{hi}}(X_i,t)} \geq (1-\alpha)(n_1+1) \right\} \tag{5}$$

(To satisfy the earlier conditions, we also assume that $\inf_{t\in\mathcal{T}} \hat{q}_{\mathrm{lo}}(x, t) = -\infty$ and $\sup_{t\in\mathcal{T}} \hat{q}_{\mathrm{lo}}(x, t) = +\infty$, and $\inf_{t\in\mathcal{T}} \hat{q}_{\mathrm{hi}}(x, t) = -\infty$ and $\sup_{t\in\mathcal{T}} \hat{q}_{\mathrm{hi}}(x, t) = +\infty$.)

Next we review CQR and several related existing methods. Our presentation reformulates each method using the general notation we have just defined.

Conformalized Quantile Regression (CQR) For any $a \in (0, 1)$, let $\hat{q}_{Y|X}(x, a)$ denote an estimate (fitted on the training data set) of the conditional a-quantile of Y given $X = x$. Romano et al. (2019) proposed fitting quantile regressors for preset upper and lower quantiles, α_{lo} and α_{hi} (e.g., $\alpha_{\mathrm{lo}} = \alpha/2$ and $\alpha_{\mathrm{hi}} = 1 - \alpha/2$), and then using conformal prediction to select a constant additive adjustment to expand or contract these intervals to ensure marginal coverage. This produces a prediction interval of the form

$$\hat{C}_n(X_{n+1}) = [\hat{q}_{Y|X}(X_{n+1}, \alpha_{\mathrm{lo}}) - \hat{t}, \hat{q}_{Y|X}(X_{n+1}, \alpha_{\mathrm{hi}}) + \hat{t}].$$

By choosing

$$\hat{q}_{\mathrm{lo}}(x, t) = \hat{q}_{Y|X}(x, \alpha_{\mathrm{lo}}) - t, \quad \hat{q}_{\mathrm{hi}}(x, t) = \hat{q}_{Y|X}(x, \alpha_{\mathrm{hi}}) + t,$$

indexed by $t \in \mathbb{R}$, we can see that the CQR interval is equal to the general construction (4) above.

CQR-r and CQR-m Sesia and Candès (2020) propose a variant of CQR, which they call CQR-r. The prediction interval is given by

$$\begin{aligned} \hat{C}_n(X_{n+1}) = [&\hat{q}_{Y|X}(X_{n+1}; \alpha_{\mathrm{lo}}) - t \cdot w(X_{n+1}), \\ &\hat{q}_{Y|X}(X_{n+1}; \alpha_{\mathrm{hi}}) + t \cdot w(X_{n+1})] \\ w(x) := &\hat{q}_{Y|X}(x; \alpha_{\mathrm{hi}}) - \hat{q}_{Y|X}(x; \alpha_{\mathrm{lo}}) \end{aligned}$$

which we can interpret as scaling the inflation of the interval by the *aleatoric* uncertainty in $Y \mid X$. This method corresponds to the general construction (4) above obtained by using

$$\begin{aligned} \hat{q}_{\mathrm{lo}}(x, t) &= \hat{q}_{Y|X}(x, \alpha_{\mathrm{lo}}) - t \cdot w(x), \\ \hat{q}_{\mathrm{hi}}(x, t) &= \hat{q}_{Y|X}(x, \alpha_{\mathrm{hi}}) + t \cdot w(x), \end{aligned}$$

where $t \in \mathbb{R}$. Sesia and Candès (2020)'s CQR-r construction is a variant of the method proposed by Kivaranovic et al. (2020) (called "CQR-m" by Sesia and Candès (2020)), where the scaling factor instead compares the (estimated) upper and lower quantiles to the (estimated) median:

$$\begin{aligned} \hat{q}_{\mathrm{lo}}(x, t) &= \hat{q}_{Y|X}(x, \alpha_{\mathrm{lo}}) - t \cdot w_{\mathrm{lo}}(x), \\ \hat{q}_{\mathrm{hi}}(x, t) &= \hat{q}_{Y|X}(x, \alpha_{\mathrm{hi}}) + t \cdot w_{\mathrm{hi}}(x), \\ w_{\mathrm{lo}}(x) &:= \hat{q}_{Y|X}(x, 0.5) - \hat{q}_{Y|X}(x, \alpha_{\mathrm{lo}}), \\ w_{\mathrm{hi}}(x) &:= \hat{q}_{Y|X}(x, \alpha_{\mathrm{hi}}) - \hat{q}_{Y|X}(x, 0.5), \end{aligned}$$

again with $t \in \mathbb{R}$, leading to the prediction interval

$$\begin{aligned} \hat{C}_n(X_{n+1}) = [&\hat{q}_{Y|X}(X_{n+1}, \alpha_{\mathrm{lo}}) - \hat{t} \cdot w_{\mathrm{lo}}(X_{n+1}), \\ &\hat{q}_{Y|X}(X_{n+1}, \alpha_{\mathrm{hi}}) + \hat{t} \cdot w_{\mathrm{hi}}(X_{n+1})]. \end{aligned}$$

Distributional Conformal Prediction (DCP) An alternative style of approach is proposed by Chernozhukov et al. (2021), which uses conformal prediction to select the target upper and lower quantiles for

quantile regression to achieve the desired coverage. After fitting the conditional t-quantile $\hat{q}_{Y|X}(x,t)$ for every $t \in \mathcal{T} \subseteq [0,1]$, the prediction set is given by

$$\hat{C}_n(X_{n+1}) = [\hat{q}_{Y|X}(X_{n+1}, \hat{t}), \hat{q}_{Y|X}(X_{n+1}, 1-\hat{t})],$$

which corresponds to the general construction (4) above by using the following, with $t \in \mathcal{T} \subseteq [0,1]$:

$$\hat{q}_{\text{lo}}(x,t) = \hat{q}_{Y|X}(x,t), \quad \hat{q}_{\text{hi}}(x,t) = \hat{q}_{Y|X}(x,1-t).$$

2.1 Limitations of existing methods

To understand the motivation behind our new methods, we briefly consider the limitations of this existing range of constructions. Consider the "baseline" prediction interval $\hat{C}_{\text{base}}(x) = [\hat{q}_{Y|X}(x,\alpha/2), \hat{q}_{Y|X}(x,1-\alpha/2)]$, given by running quantile regression on the training set. In many settings, this interval may more closely resemble the oracle one in regions with larger quantities of training data and/or more smoothly varying quantiles, and less so in other regions. If we use a method such as a neural network to compute this baseline, typically the baseline interval undercovers on average over the calibration (and test) data, as demonstrated in Romano et al. (2019), for example.

For each of the existing methods summarized above, this initial interval $\hat{C}_{\text{base}}$ appears in the nested family of sets: specifically, it is recovered by taking $t = 0$ for CQR, CQR-r, and CQR-m (assuming $\alpha_{\text{lo}} = \alpha/2$ and $\alpha_{\text{hi}} = 1-\alpha/2$), and by taking $t = \alpha/2$ for DCP. Now, if $\hat{C}_{\text{base}}$ undercovers on the calibration set, conformal prediction leads us to choose a larger value of $\hat{t}$, thus inflating $\hat{C}_{\text{base}}$ in order to achieve a marginal coverage guarantee (1).

Let us now consider how this inflation behaves. CQR provides an *equal* amount of inflation of $\hat{C}_{\text{base}}(x)$ for every x, while CQR-r and CQR-m provide a *higher* amount of inflation of $\hat{C}_{\text{base}}(x)$ at values x with higher aleatoric uncertainty (i.e., higher variability in the conditional distribution of $Y \mid X$), and generally this is the case for DCP as well.

In contrast, it would be more efficient to inflate $\hat{C}_{\text{base}}(x)$ primarily at those values x where the initial estimate undercovers—that is, regions with high error in $\hat{q}_{Y|X}(x,a)$ as an estimate of the true conditional a-quantile for $a = \alpha/2$ and/or $a = 1-\alpha/2$. In other words, we would do best to inflate $\hat{C}_{\text{base}}(x)$ more in regions of high *epistemic* uncertainty, which we define as reducible uncertainty (e.g., uncertainty that could have been reduced if we had more data), following the definitions in Senge et al. (2014) and Hüllermeier and Waegeman (2021).

Interestingly, the main empirical conclusion of Sesia and Candès (2020) is that CQR-r and the related CQR-m, which provide varying amounts of inflation at different values x, were unable to provide smaller intervals on average than CQR, whose construction has constant inflation at each x. We might be tempted to conclude from this that incorporating local adaptivity into the inflation of CQR is not beneficial. However, an alternative hypothesis explored in this paper is that the issue lies with CQR-r and CQR-m inflating the baseline interval proportional to the aleatoric uncertainty, which may be completely different from the epistemic uncertainty; thus, we instead conjecture that it is important to be locally adaptive to the right source of uncertainty (that is, epistemic rather than aleatoric).

2.2 Additional Related Work

One of our main proxies for epistemic uncertainty uses the bagging structure of Quantile Regression Forests. Kim et al. (2020) and Gupta et al. (2022) propose using bagging structure for a different purpose: to avoid data splitting for conformal. These papers leverage that each bagged estimator is only trained on a subset of the training data in order to conformalize them by just using out-of-bag residuals, albeit with modified coverage guarantees. Gupta et al. (2022) adapt this strategy to conformalizing quantile estimators. A characteristic of their algorithm is that the prediction intervals will be smaller when there is consensus among the ensemble of quantile regressors and larger when there is disagreement. This variation among the ensemble members is termed "instability." As we discuss in Section 3.1, we leverage instability as one way to estimate epistemic uncertainty. In this sense, previous work has implicitly taken epistemic uncertainty into account – but not yet in our split conformal setting. In addition, the inflation of Gupta et al. (2022)'s intervals when using unstable quantile regressors may also depend on aleatoric uncertainty in subtle ways, while our method more directly separates the two.

Prior conformal methods have incorporated uncertainty estimates. For example, Lei et al. (2018) scale the $\hat{t}$ in (2) by a non-negative multiplicative factor. In that framework, there are many possible choices of scaling factor, including an estimate of the conditional standard deviation and an estimate of the epistemic uncertainty in the conditional mean estimator. A positive of UACQR is that when using quantile regressors the appropriate choice of scaling factor is simply the epistemic uncertainty of the quantile regressors. In contrast, for Lei et al. (2018), the optimal choice may incorporate both aleatoric and epistemic uncertainty.

Izbicki et al. (2020) propose a conformal procedure for conditional density estimates to provide prediction sets that not may not be intervals. As with CQR,

they estimate aleatoric uncertainty directly and then conformalize the estimates. One can generalize our uncertainty-aware framework to this setting with likely similar benefits when appropriate.

3 UNCERTAINTY-AWARE CQR

Our proposal is to integrate uncertainty awareness into CQR via the general construction (4), by explicitly incorporating an estimate of epistemic uncertainty into the construction. We now present two specific proposals that achieve this aim.

UACQR-S Our first proposed method is similar in construction to the approach of CQR-r and CQR-m, where the baseline quantiles are inflated proportional to a scaling factor. (Here the "S" in UACQR-S denotes *scaling*.) We define

$$\hat{q}_{\mathrm{lo}}(x,t) = \hat{q}_{Y|X}(x,\alpha_{\mathrm{lo}}) - t\cdot\hat{g}_{\mathrm{lo}}(x),$$
$$\hat{q}_{\mathrm{hi}}(x,t) = \hat{q}_{Y|X}(x,\alpha_{\mathrm{hi}}) + t\cdot\hat{g}_{\mathrm{hi}}(x),$$

indexed by $t \in \mathbb{R}$, where $\hat{q}_{Y|X}(x,\alpha_{\mathrm{lo}})$ (respectively, $\hat{q}_{Y|X}(x,\alpha_{\mathrm{hi}})$) is some baseline initial estimator of the conditional α_{lo}- (respectively, α_{hi}-) quantile, while $\hat{g}_{\mathrm{lo}}(x)$ (respectively, $\hat{g}_{\mathrm{hi}}(x)$) estimates the standard deviation of this quantile estimate. (To give a concrete example, if we construct $\hat{q}^b_{Y|X}(x,a)$'s via bootstrapping subsamples of the training data for each $a = \alpha_{\mathrm{lo}}, \alpha_{\mathrm{hi}}$, then $\hat{q}_{Y|X}(x,a)$ might be obtained by averaging the B bootstrapped estimates, while $\hat{g}_{\mathrm{lo}}(x)$ and $\hat{g}_{\mathrm{hi}}(x)$ might be computed via the sample standard deviations of the $\hat{q}^b_{Y|X}(x,a)$'s at each value of x.)

Applying our general construction (4) then yields the prediction interval

$$\hat{C}_n(X_{n+1}) = [\hat{q}_{Y|X}(X_{n+1},\alpha_{\mathrm{lo}}) - \hat{t}\cdot\hat{g}_{\mathrm{lo}}(X_{n+1}), \\ \hat{q}_{Y|X}(X_{n+1},\alpha_{\mathrm{hi}}) + \hat{t}\cdot\hat{g}_{\mathrm{hi}}(X_{n+1})], \quad (6)$$

where $\hat{t}$ is computed as in (5).

Examining this construction for UACQR-S, we can observe that CQR-r and CQR-m can be written in the same format—for example, by taking $\hat{g}_{\mathrm{lo}}(x) = \hat{q}_{Y|X}(x,0.5) - \hat{q}_{Y|X}(x,\alpha_{\mathrm{lo}})$ for CQR-m. However, for CQR-r and CQR-m the corresponding scaling factors $\hat{g}_{\mathrm{lo}}$ and $\hat{g}_{\mathrm{hi}}$ correspond to the aleatoric uncertainty in our initial quantile regression, while our particular choice of the scaling factors $\hat{g}_{\mathrm{lo}}$ and $\hat{g}_{\mathrm{hi}}$ is aimed at estimating the epistemic uncertainty, thus providing a more useful local adaptivity when inflating the prediction interval.

UACQR-P Our second variant is UACQR-P, where the "P" denotes *percentile*. To begin, we compute an ensemble of B estimates of $q_{Y|X}(x,\alpha_{\mathrm{lo}})$ and of $q_{Y|X}(x,\alpha_{\mathrm{hi}})$, denoted by $\hat{q}^b_{Y|X}(x,a)$ for each $b \in \{1,\ldots,B\}$ and each $a \in \{\alpha_{\mathrm{lo}},\alpha_{\mathrm{hi}}\}$. (For instance, we may obtain these by running a quantile regression procedure on B different subsets of the training set.)

Let $\hat{q}^{(b)}_{Y|X}(x,a)$ refer to the b-th order statistic for each $a = \alpha_{\mathrm{lo}}, \alpha_{\mathrm{hi}}$, so that the estimates are now sorted, with $\hat{q}^{(1)}_{Y|X}(x,a) \le \cdots \le \hat{q}^{(B)}_{Y|X}(x,a)$. We also define $\hat{q}^{(0)}_{Y|X}(x,a) = -\infty$ and $\hat{q}^{(B+1)}_{Y|X}(x,a) = +\infty$. Then defining

$$\hat{q}_{\mathrm{lo}}(x,t) = \hat{q}^{(B+1-t)}_{Y|X}(x,\alpha_{\mathrm{lo}}), \quad \hat{q}_{\mathrm{hi}}(x,t) = \hat{q}^{(t)}_{Y|X}(x,\alpha_{\mathrm{hi}}),$$

over the index set $t \in \mathcal{T} = \{0,1,\ldots,B,B+1\}$, applying our general construction (4) yields the prediction interval

$$\hat{C}_n(X_{n+1}) = [\hat{q}^{(B+1-\hat{t})}_{Y|X}(X_{n+1},\alpha_{\mathrm{lo}}), \hat{q}^{(\hat{t})}_{Y|X}(X_{n+1},\alpha_{\mathrm{hi}})], \quad (7)$$

where $\hat{t}$ is computed as in (5).

One useful property of this construction is that it preserves the smoothness properties of the underlying base estimators—in the Appendix D, we show that if each of the bootstrapped estimates $\hat{q}^b_{Y|X}(x,a)$ is Lipschitz as a function of x, then each order statistic $\hat{q}^{(b)}_{Y|X}(x,a)$ is Lipschitz as well.

3.1 Strategies for estimating epistemic uncertainty

In UACQR-P, epistemic uncertainty is estimated indirectly, via producing an ensemble of quantile estimates $\{\hat{q}^b_{Y|X}(x,a)\}_{b=1,\ldots,B}$ at each $a = \alpha_{\mathrm{lo}}, \alpha_{\mathrm{hi}}$. For UACQR-S, in contrast, we require explicit estimates of epistemic uncertainty, $\hat{g}_{\mathrm{lo}}(x)$ and $\hat{g}_{\mathrm{hi}}(x)$. Here we outline a few potential strategies for generating these ensembles or these uncertainty estimates.

Bootstrapping/bagging/ensembling. We can generate a collection $\{\hat{q}^b_{Y|X}(x,a)\}_{b=1,\ldots,B}$ by training each $\hat{q}^b_{Y|X}$ on a different subset of the training data, e.g., by bootstrapping (Efron and Tibshirani, 1993) the training set. The empirical distribution of this collection approximates the distribution of $\hat{q}_{Y|X}$ under different draws of the training data. For UACQR-S, we can compute

$$\hat{q}_{Y|X}(x,\alpha_{\mathrm{lo}}) = \frac{1}{B}\sum_{b=1}^{B}\hat{q}^b_{Y|X}(x,\alpha_{\mathrm{lo}})$$

and

$$\hat{g}_{\mathrm{lo}}(x) = \left[\frac{1}{B}\sum_{b=1}^{B}\left(\hat{q}^b_{Y|X}(x,\alpha_{\mathrm{lo}}) - \hat{q}_{Y|X}(x,\alpha_{\mathrm{lo}})\right)^2\right]^{1/2},$$

and $\hat{q}_{Y|X}(x, \alpha_{\text{hi}})$ and $\hat{g}_{\text{hi}}(x)$ are defined analogously. Of course, we might also choose to use a different aggregation procedure, e.g., a median or quantile of the bootstrapped estimates, rather than the mean, which corresponds to bagging (Breiman, 1996). In Appendix A, we discuss how we implement our methods with Quantile Regression Forests (Meinshausen, 2006) even with their unique aggregation procedure. For neural networks, performance may improve by training each $\hat{q}^b_{Y|X}$ on all training samples, only varying the random initialization of weights for each b (Lakshminarayanan et al., 2017).

Epochs of training. Previous work has indicated that neural networks can fit "easy" regions of the feature space well in early epochs while "difficult" regions are not fit until later epochs. For instance, Mangalam and Prabhu (2019) show that deep neural networks learn examples which are learnable by shallow networks first, suggesting that the number of epochs until a training sample is accurately learned may reflect its epistemic uncertainty. Therefore, a heuristic for epistemic uncertainty is to measure how much a neural network's predictions change across epochs. Here, $\{\hat{q}^b_{Y|X}(x, a)\}_{b=1,\ldots,B}$ would refer to the model's predictions after each epoch b. This heuristic may be more suitable for UACQR-S than UACQR-P, since ensemble members corresponding to later epochs may be higher quality than those of earlier epochs. Huang et al. (2017) explores using cyclic learning rates to make these ensemble members more exchangeable.

A parametric approach. Some quantile regression models may provide a distribution for its estimates under parametric assumptions, such as linear quantile regression (Koenker, 1994) or Bayesian learning approaches which place priors on model parameters (Neal, 2012). Specifically, for UACQR-S, the quantile estimates $\hat{q}_{Y|X}(x, \alpha_{\text{lo}})$ and $\hat{q}_{Y|X}(x, \alpha_{\text{hi}})$ would be fitted via some parametric model, and the scaling parameters $\hat{g}_{\text{lo}}(x)$ and $\hat{g}_{\text{hi}}(x)$ would then be defined as the standard errors of these quantile estimates, as calculated according to the parametric model.

3.2 Theoretical guarantee

Theorem 1. *Assume the data points* $(X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$ *are exchangeable. Then the UACQR-P and UACQR-S prediction intervals constructed in* (7) *and* (6)*, respectively, satisfy the marginal coverage guarantee* $\mathbb{P}\{Y_{n+1} \in \hat{C}_n(X_{n+1})\} \geq 1 - \alpha$.

Proof. This result follows immediately from the properties of conformal prediction (Vovk et al., 2005; Gupta et al., 2022), once we observe that UACQR-P and UACQR-S can be formulated as a special case of the nested conformal prediction construction, as detailed in (4). □

Sesia and Candès (2020) provide an asymptotic guarantee of conditional coverage under consistent quantile estimators for CQR, CQR-r, and CQR-m. This result can directly be extended to UACQR-S. We focus on comparing our methods to baselines under less idealized conditions: finite data and potentially misspecified models.

4 SIMULATION CASE STUDY

In this section, we demonstrate our methods on simulated data in which we can highlight the differences between aleatoric and epistemic uncertainty. We generate simulated data from the distribution

$$X \sim \text{Beta}(1.2, 0.8), \quad Y \mid X \sim \mathcal{N}(\sin(X^{-3}), X^4),$$

with sample size $n = 100$ (and dimension $p = 1$). We compare our UACQR-P and UACQR-S methods against the existing CQR and CQR-r methods, and repeat the experiment for 150 independent trials. For all four methods, the base quantile regression method is a neural network—we note that this base estimator exhibits substantial overfitting in this simulation. Details of the implementation of all four methods are given in Appendix B. We set $\alpha = 0.1$ to create 90% prediction intervals.

Results. Figure 1 shows the resulting prediction bands constructed by each of the four methods, for one trial of the experiment. We also show the oracle prediction interval, given by $[q_{Y|X}(x, \alpha/2), q_{Y|X}(x, 1-\alpha/2)]$ (where these now refer to the *true* conditional quantiles of $Y \mid X$).

Examining the oracle prediction interval, in each chart, we can see the distinction between aleatoric and epistemic uncertainty. *Aleatoric uncertainty* is high near $X = 1$, since the variance of $Y \mid X$ is higher and thus the oracle prediction interval is relatively wide, and low near $X = 0$, where the variance of $Y \mid X$ is low and thus the oracle prediction interval is overly narrow. *Epistemic uncertainty*, in contrast, is relatively low near $X = 1$, since we have many data points that allow our neural net to fit the region well, and much higher near $X = 0$, since the Beta distribution for the feature X ensures we have very few training samples in this region. In addition, the highly non-smooth conditional mean function is particularly challenging to estimate near $X = 0$.

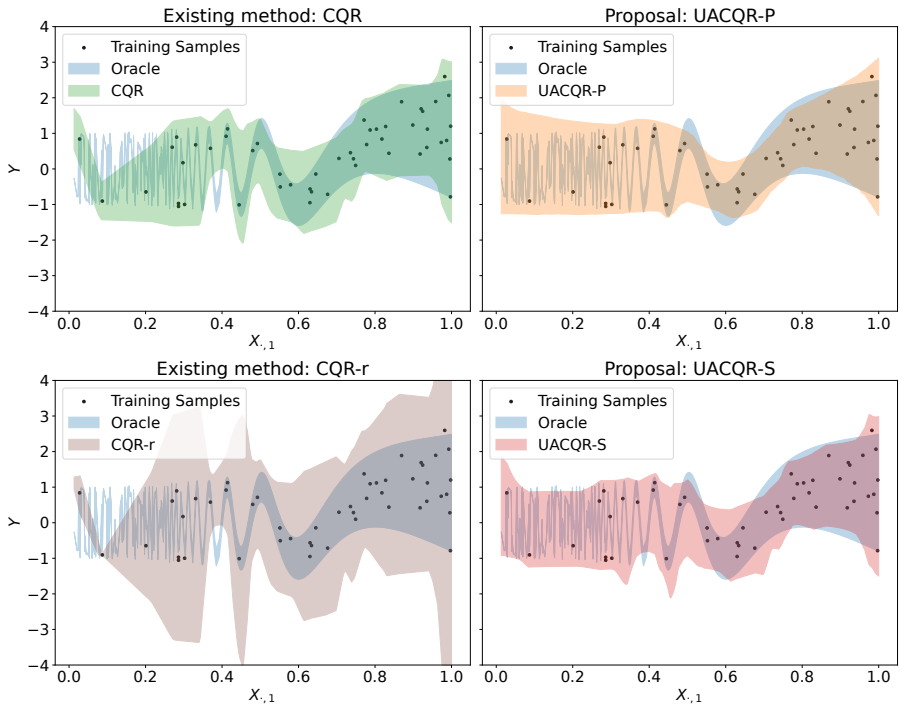

Figure 1: A comparison of the prediction intervals for existing and proposed methods under one random draw from the data generating process. While existing methods will have low conditional coverage in the high epistemic uncertainty region near $X = 0$, our proposals adapt to the toughness of this region. Each conformal procedure is run on the same fitted neural net. For UACQR we use the epoch-based heuristic for epistemic uncertainty. We limit the y-axis to [-4,4] for visual clarity.

As a consequence, CQR and CQR-r show prediction intervals that are too narrow near $X = 0$, which is compensated for by the overly large inflations of the intervals near $X = 1$. In contrast, UACQR-P and UACQR-S correctly inflate the prediction intervals more in the high-uncertainty region near $X = 0$, and they do not overly inflate the intervals in the more well-informed region near $X = 1$.

In Figure 2, we show the results of the experiment averaged over 150 independent trials, where the figure displays the average conditional coverage, $\mathbb{P}\left\{Y_{n+1} \in \hat{C}_n(X_{n+1}) \mid X_{n+1} = x\right\}$, as a function of the test feature value $x \in [0, 1]$. While all four methods achieve *marginal* coverage (1), averaged over X_{n+1} (as guaranteed by the theory), we observe very different empirical trends in the conditional coverage. Specifically, CQR and CQR-r are undercovering in the region of high epistemic uncertainty (near $X = 0$), and overcovering in regions of lower epistemic uncertainty (higher values of X). In particular, we can also see that the adaptivity of CQR-r to aleatoric uncertainty is counterproductive here, with unnecessarily wide prediction intervals near $X = 1$.

In contrast, UACQR-P and UACQR-S maintain relatively even coverage across the range of X values. Of course, these prediction intervals do not fit accurately to the highly non-smooth, high-frequency trends in the conditional mean of $Y \mid X$ near $X = 0$, but the wider prediction intervals in this region of high uncertainty compensate appropriately for our inability to estimate this challenging conditional mean. This ability to provide coverage in the high epistemic uncertainty area (near $X = 0$) does not come at the expense of overly wide intervals in the low epistemic uncertainty region (larger values of X); both of the UACQR methods provide close to 90% coverage across all values of X.

5 PERFORMANCE ON DATA SETS

We measure the performance of our proposals and baselines on data sets from Romano et al. (2019) and Sesia and Candès (2020) in Table 1. Six of the twelve data sets included in these studies had a response of 0 for over 25% of samples. This zero-inflation means that methods such as CQR and CQR-r, which inflate the prediction interval symmetrically at the left and right endpoint, are not as well suited for this data. In order to enable a fair comparison with these existing methods, then, we restrict our attention to

Table 1: Real data set results, outperformance bolded, underlined if significant

	Average Test Interval Score Loss on 20 Runs (Standard Error)			
Dataset	UACQR-P	UACQR-S	CQR	CQR-r
`bike`	**1.436** (0.013)	1.573 (0.008)	1.586 (0.008)	1.564 (0.009)
`bio`	**1.882** (0.011)	1.900 (0.010)	1.900 (0.010)	1.900 (0.011)
`cbc`	1.294 (0.026)	1.276 (0.017)	**1.274** (0.017)	1.275 (0.017)
`community`	**2.103** (0.045)	2.155 (0.033)	2.157 (0.034)	2.156 (0.033)
`concrete`	**0.834** (0.018)	0.881 (0.015)	0.882 (0.015)	0.864 (0.015)
`forest`	**2.436** (0.015)	2.461 (0.010)	2.462 (0.010)	2.461 (0.010)
`homes`	**0.878** (0.012)	0.916 (0.011)	0.930 (0.010)	0.911 (0.011)
`imdb_wiki`	**1.837** (0.017)	1.926 (0.010)	1.928 (0.010)	1.927 (0.010)
`star`	0.214 (0.001)	**0.213** (0.001)	**0.213** (0.001)	**0.213** (0.001)

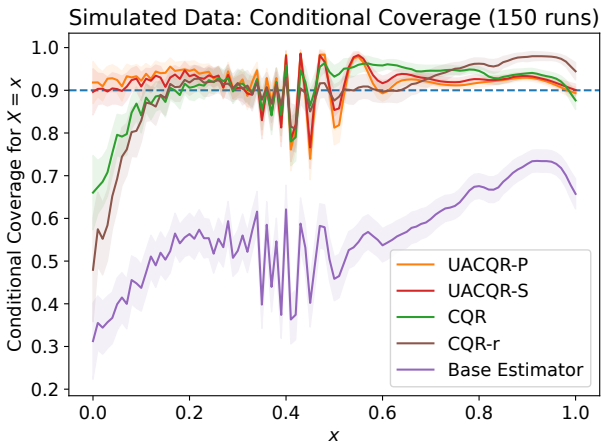

Figure 2: A comparison of the conditional coverage of existing and proposed methods across 150 random draws, with same setting as Figure 1. All existing methods severely undercover near $X = 0$, where epistemic uncertainty is high, while our proposals maintain conditional coverage even in this challenging region.

the remaining six data sets: `bike` (Dua and Graff, 2017; Fanaee-T and Gama, 2013), `bio` (Dua and Graff, 2017), `community` (Dua and Graff, 2017; Redmond and Baveja, 2002), `concrete` (Dua and Graff, 2017; Yeh, 1998), `homes` (Kaggle, 2016), and `star` (Achilles et al., 2008).

We also tested our methods on three new data sets that map images to a continuous response in order to measure the performance of our methods on a wider breadth of tasks. For `cbc` (Alam and Islam, 2019), we estimate the complete blood count from blood smear images. For `forest` (Demir et al., 2018), from aerial images we estimate the percent of area without forest cover. For `imdb_wiki` (Rothe et al., 2015, 2018), we estimate the year in which celebrity photos from Wikipedia were taken. For all three data sets, we pass the images through a pre-trained ResNet-18 model (He et al., 2016) to extract 512 continuous features. We then train our quantile regressors using these features.

We used Quantile Regression Forests for these data sets, since most are tabular and tree-based methods frequently beat neural networks on tabular data (Grinsztajn et al., 2022). We note the average performance across 20 runs in Table 1, bolding outperformance and underlining when statistically significant at the 5% significance level. Outside simulation settings, we cannot measure conditional coverage, so our metric will instead be the average test interval score loss (defined in Appendix B). The motivation is that interval score loss is a proper scoring rule for prediction intervals, so its expectation is minimized by the true conditional quantiles (Gneiting and Raftery, 2007). For users who care about the size of prediction intervals over conditional coverage, we show in Table 3 that our methods also perform well in terms of average interval width.

Since we use random train-test splits in these experiments, we ensure exchangeability of the data, and therefore all methods will have above 90% coverage on average. Moreover, we use a randomized conformal procedure, detailed in Appendix C, that ensures all methods average 90% coverage almost exactly (see Appendix E.2). Our choice of hyperparameter (`min_samples_leaf`) for each method was determined by a cross-validation procedure that minimized the interval score loss for each conformal method.

In general, it is not necessarily surprising that UACQR-P frequently outperforms UACQR-S here. As we observed in Figure 1, prediction intervals from UACQR-P may vary more smoothly with x than those from UACQR-S, as the epistemic uncertainty scaling factors may be noisy. In addition, UACQR-P is invariant to monotonically increasing transformations of the response since it is a function of ranks, while UACQR-S is not. Therefore, it is possible that the relative per-

formance of UACQR-S could improve under certain transformations. An additional factor is that the machine learning quantile regressors frequently achieve close to 90% marginal coverage even before conformal prediction (see Table 1 in Romano et al. (2019)), leading to a small conformal adjustment ($\hat{t}$) for each of UACQR-S, CQR, and CQR-r. This regime will lead to these three methods performing similarly, regardless of the scaling factor used. In contrast, for our simulation, we chose hyperparameters that encouraged overfitting and yielded far below 90% coverage without conformal. Accordingly, UACQR-S performs much better relatively in this simulation.

6 DISCUSSION

In this work, we have shown that locally adaptive calibration can be beneficial for conformalizing a quantile regression method, as long as the local adaptivity is capturing the correct information—epistemic, rather than aleatoric, uncertainty, or in other words, the variability of the quantile regression estimates, rather than the variability of the response Y itself.

While our methods have demonstrated strong empirical performance on simulations and real-world data sets alike, there are a few caveats. First, the performance of our methods is gated by the ability to estimate epistemic uncertainty well. Thus, for bootstrapping and heuristic approaches alike, our methods may be sensitive to the sample size n and our ensemble size B. In addition, our methods are more suitable for high-variance, low-bias quantile regressors, since we estimate epistemic uncertainty by estimating model variance. A second potential limitation is that, depending on the specific methods used to compute the initial collection of B estimates, UACQR may present users with a computational-statistical trade-off. An additional limitation is the lack of a theoretical framework for understanding the performance characteristics of UACQR. We suspect that a general theory for this may not be possible but that it is feasible to provide theoretical guarantees in special cases, which may inform a qualitative understanding of the behavior of the method in more general settings.

On the other hand, if epistemic uncertainty is estimated reasonably well, then (as we see in our experiments), UACQR has the potential to offer prediction intervals with better conditional coverage properties. This demonstrates the benefits of a more locally adaptive approach when implementing conformal prediction, and may offer insights for improving the empirical performance of conformal prediction on problems beyond quantile regression – including classification tasks.

7 ACKNOWLEDGEMENTS

R.F.B. was partially supported by the National Science Foundation via grant DMS-2023109, and by the Office of Naval Research via grant N00014-20-1-2337. R.M.W. was partially supported by the National Science Foundation via grants DMS-1930049 and NSF DMS-2023109, and by the Department of Energy via grant DE-AC02-06CH113575.

References

Achilles, C., Bain, H. P., Bellott, F., Boyd-Zaharias, J., Finn, J., Folger, J., Johnston, J., and Word, E. (2008). Tennessee's student teacher achievement ratio (STAR) project. Accessed: July, 2019.

Alam, M. M. and Islam, M. T. (2019). Machine learning approach of automatic identification and counting of blood cells. *Healthcare Technology Letters*, 6(4):103–108.

Barber, R. F., Candes, E. J., Ramdas, A., and Tibshirani, R. J. (2021). The limits of distribution-free conditional predictive inference. *Information and Inference: A Journal of the IMA*, 10(2):455–482.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Chernozhukov, V., Fernández-Val, I., and Galichon, A. (2010). Quantile and probability curves without crossing. *Econometrica*, 78(3):1093–1125.

Chernozhukov, V., Wüthrich, K., and Zhu, Y. (2021). Distributional conformal prediction. *Proceedings of the National Academy of Sciences*, 118(48):e2107794118.

Demir, I., Koperski, K., Lindenbaum, D., Pang, G., Huang, J., Basu, S., Hughes, F., Tuia, D., and Raskar, R. (2018). Deepglobe 2018: A challenge to parse the earth through satellite images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Efron, B. (1981). Nonparametric standard errors and confidence intervals. *Canadian Journal of Statistics*, 9(2):139–158.

Efron, B. and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. Number 57 in Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, Boca Raton, Florida, USA.

Fanaee-T, H. and Gama, J. (2013). Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, pages 1–15.

Feldman, S., Bates, S., and Romano, Y. (2021). Improving conditional coverage via orthogonal quantile regression. *Advances in neural information processing systems*, 34:2060–2071.

Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

Grinsztajn, L., Oyallon, E., and Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on tabular data? *arXiv preprint arXiv:2207.08815*.

Gupta, C., Kuchibhotla, A. K., and Ramdas, A. (2022). Nested conformal prediction and quantile out-of-bag ensemble methods. *Pattern Recognition*, 127:108496.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., and Weinberger, K. Q. (2017). Snapshot ensembles: Train 1, get m for free. In *International Conference on Learning Representations*.

Hüllermeier, E. and Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110:457–506.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr.

Izbicki, R., Shimizu, G., and Stern, R. (2020). Flexible distribution-free conditional predictive bands using density estimators. In Chiappa, S. and Calandra, R., editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3068–3077. PMLR.

Kaggle (2016). House sales in King County, USA. `https://www.kaggle.com/harlfoxem/housesalesprediction/metadata`. Accessed: August, 2019.

Kim, B., Xu, C., and Barber, R. (2020). Predictive inference is free with the jackknife+-after-bootstrap. *Advances in Neural Information Processing Systems*, 33:4138–4149.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kivaranovic, D., Johnson, K. D., and Leeb, H. (2020). Adaptive, distribution-free prediction intervals for deep networks. In *International Conference on Artificial Intelligence and Statistics*, pages 4346–4356. PMLR.

Koenker, R. (1994). Confidence intervals for regression quantiles. In Mandl, P. and Hušková, M., editors, *Asymptotic Statistics*, pages 349–359, Heidelberg. Physica-Verlag HD.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30.

Lei, J., G'Sell, M., Rinaldo, A., Tibshirani, R. J., and Wasserman, L. (2018). Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523):1094–1111.

Lei, J. and Wasserman, L. (2014). Distribution-free prediction bands for non-parametric regression. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, pages 71–96.

Mammen, E. (1991). Nonparametric regression under qualitative smoothness assumptions. *The Annals of Statistics*, 19(2):741–759.

Mangalam, K. and Prabhu, V. U. (2019). Do deep neural networks learn shallow learnable examples first? In *ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena*.

Meinshausen, N. (2006). Quantile regression forests. *Journal of Machine Learning Research*, 7(35):983–999.

Neal, R. M. (2012). *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media.

Redmond, M. and Baveja, A. (2002). A data-driven software tool for enabling cooperative information sharing among police departments. *European Journal of Operational Research*, 141(3):660–678.

Romano, Y., Patterson, E., and Candès, E. J. (2019). Conformalized quantile regression. In *NeurIPS*.

Rothe, R., Timofte, R., and Gool, L. V. (2015). Dex: Deep expectation of apparent age from a single image. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*.

Rothe, R., Timofte, R., and Gool, L. V. (2018). Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision*, 126(2-4):144–157.

Senge, R., Bösner, S., Dembczyński, K., Haasenritter, J., Hirsch, O., Donner-Banzhoff, N., and Hüllermeier, E. (2014). Reliable classification: Learning classifiers that distinguish aleatoric and epistemic uncertainty. *Information Sciences*, 255:16–29.

Sesia, M. and Candès, E. J. (2020). A comparison of some conformal quantile regression methods. *Stat*, 9(1):e261.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Vovk, V., Gammerman, A., and Shafer, G. (2005). *Algorithmic Learning in a Random World*. Springer-Verlag, Berlin, Heidelberg.

Yeh, I.-C. (1998). Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete research*, 28(12):1797–1808.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model.
   Yes, see Section 2 and Section 3.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm.
   Yes, see Appendix A.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries.
   Yes

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results.
   Yes, see Section 3.2
   (b) Complete proofs of all theoretical results.
   Yes, see Section 3.2 and Appendix D
   (c) Clear explanations of any assumptions.
   Yes, see sections above.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL).
   Yes, code for reproducibility is included in supplementary material.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen).
   Yes, these details are included in Appendix A, Appendix B, and in provided source code
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times).
   Yes, see Section 5.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider).
   Yes, see Appendix B.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets.
   Yes, see end of Appendix B.
   (b) The license information of the assets, if applicable.
   Not applicable.
   (c) New assets either in the supplemental material or as a URL, if applicable.
   Not applicable.
   (d) Information about consent from data providers/curators.
   Not applicable.
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content.
   Not applicable.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots.
   Not applicable.
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable.
   Not applicable.
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation.
   Not applicable.

A FAST HEURISTIC ALTERNATIVES TO BOOTSTRAPPING

We propose several ways to implement UACQR-P and UACQR-S without needing to refit the base learned model B times.

A.1 Quantile Regression Forests

We first describe what happens when one runs Quantile Regression Forests (QRF) (Meinshausen, 2006) on a dataset to learn the a-quantile. The model is composed of N decision trees, each trained with a bootstrapped sample of points. To make a prediction, we start by feeding the point's covariates X_i through each of the decision trees and note which leaves X_i resides in. We then take the weighted a-quantile of the responses of the training samples in these leaves, where the weights correspond to multiplicity of the training samples in this collection. See Figure 3 for reference.

Instead of refitting B full QRFs on bootstrapped training sets, we instead leverage the bootstrapping already done in fitting a single QRF. Namely, we estimate the a-quantile in each leaf that X_i resides in and use these N estimates as an ensemble of estimates. Observe that this is equivalent to running UACQR with $B = N$ on QRF models with only one tree in each model. This entails setting $\{\hat{q}^b_{Y|X}(x, a)\}_{b=1,\ldots,B}$ to be the estimate of the a-quantile in each tree.

For UACQR-S, we can use the standard deviation of this set for $\hat{g}_{\text{lo}}(x)$ when $a = \alpha_{\text{lo}}$ and similarly for $\hat{g}_{\text{hi}}(x)$. In our implementations, we set $\hat{q}_{Y|X}(x, a)$ to be the output of the full QRF, with $B = N$ trees, for the a-quantile. The reason is that this allows the base estimator to be QRF as originally designed, with the heuristic only being used for estimating $\hat{g}_{\text{lo}}(x), \hat{g}_{\text{hi}}(x)$.

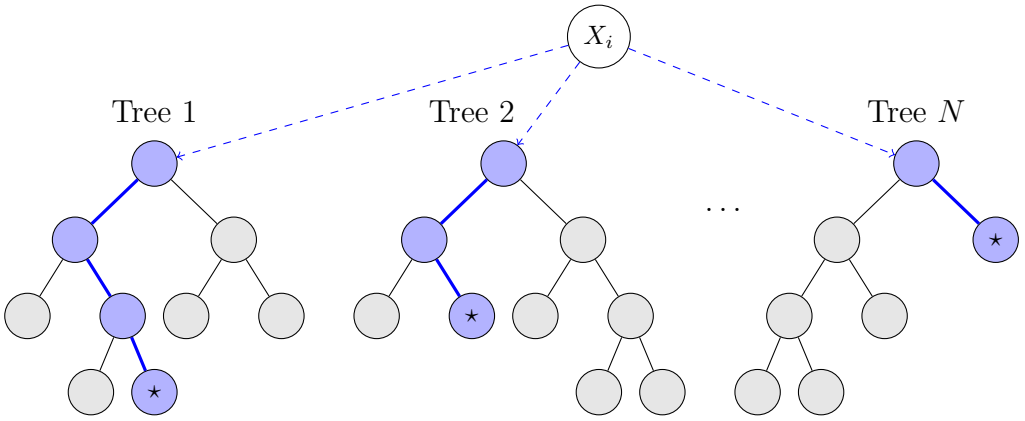

Figure 3: Illustration of a Random Forest, which serves as a visual aid for the ensuing discussion on existing and proposed methods. **QRF** outputs the empirical $\alpha_{\text{lo}}, \alpha_{\text{hi}}$ quantiles of the collection of all responses in all the leaves associated with X_i (marked with a star for each tree). **CQR** then calibrates a constant additive adjustment to these to get $1 - \alpha$ coverage. Our sped-up **UACQR-P** calculates the empirical $\alpha_{\text{lo}}, \alpha_{\text{hi}}$ quantiles in *each* leaf associated with X_i (marked with a star) and then calibrates how to aggregate these quantile estimates to get $1 - \alpha$ coverage.

A.2 Epoch-based Optimization

Previous work has indicated that neural networks can fit "easy" regions of the feature space well in early epochs while "difficult" regions are not fit until later epochs. For instance, Mangalam and Prabhu (2019) show that deep neural networks learn examples which are learnable by shallow networks first, suggesting that the number of epochs until a training sample is accurately learned may reflect its epistemic uncertainty.

Using this idea as a heuristic, one way to approximate epistemic uncertainty is to measure how much a neural network's predictions change across epochs. Here, $\{\hat{q}^b_{Y|X}(x, a)\}_{b=1,\ldots,B}$ would refer to the model's predictions after each epoch b. We can use the standard deviation of this set for $\hat{g}_{\text{lo}}(x)$ when $a = \alpha_{\text{lo}}$ and similarly for $\hat{g}_{\text{hi}}(x)$.

There are two ways to implement this epoch-based heuristic. First, during training, one can store the model's predictions on the held-out samples after each epoch. This may not be feasible if we receive the data in an online fashion and need to train the quantile regressors without access yet to the calibration and test data. Second, one can copy the neural network's parameters after each epoch, and then we can later reload each of these copies during calibration and test to see how the model's predictions for the held-out data changed across epochs. This is more memory intensive, however we did not encounter any problems when using this second approach for our experiments.

One wrinkle, though, is that the user may view the model trained for B epochs as the highest quality estimate. In this case, for UACQR-S we may consider letting $\hat{q}_{Y|X}(x,a) = \hat{q}^B_{Y|X}(x,a)$ for $a \in \{\alpha_{\text{lo}}, \alpha_{\text{hi}}\}$, i.e., the lower and upper bounds of our prediction intervals, pre-conformal, are the fully trained models. This is in fact what we do for our experiments.

It is not quite as easy to privilege the fully-trained model with UACQR-P, and therefore this heuristic may be more suitable for UACQR-S. A subject of future research may be to take weighted percentiles of $\{\hat{q}^b_{Y|X}(x,a)\}_{b=1,\ldots,B}$ for UACQR-P, where weights increase with b. Alternatively, in the bootstrapping literature there is the bias-corrected percentile method (Efron, 1981) that, in their setting, incorporates an estimator that uses all of the training samples into the confidence interval construction. In spite of this intuition, our experimental results suggest using UACQR-P over UACQR-S when using the epoch-based heuristic. This outperformance, though, may have depended on using an extensive hyperparameter tuning procedure and always training for at least 100 epochs. In terms of computation and memory, both methods are interchangeable here. See Appendix B.2 for further discussion.

B METHODOLOGICAL DETAILS

All of our experiments were done on a computing cluster, with no extra memory allocations or extra computing nodes requested. We use 150 trials for simulations and 20 trials for the real-world data set experiments. In each trial, we have a new draw of the data (training, calibration, and test) and a different random initialization of the model.

B.1 Simulation

We use a neural network with 2 hidden layers, with 100 nodes in each, and ReLU activations for each hidden layer. The neural network outputs three responses, each with the different loss function. The first two are trained with quantile loss, with the target quantiles being 0.05 and 0.95. The third response is trained with squared error loss. We find that adding the squared error loss output dimension, while not theoretically necessary for quantile regression, improves finite sample performance. We use batch normalization (Ioffe and Szegedy, 2015) and min-max normalization of both the inputs and the responses. The parameters of the neural network are initialized using PyTorch's default setting. We train the neural network for 1000 epochs with a batch size of 2 and an initial learning rate of 0.001. To make batch normalization more stable in the presence of such a small batch size, we use the running moment estimates after the first epoch, instead of continuing to estimate the moments for each batch. We use a step learning rate scheduler, with a decay rate of 0.999 every 10 epochs, and the Adam optimizer (Kingma and Ba, 2014). These hyperparameter choices were made to encourage overfitting, since if the neural network learned the conditional quantiles well then the benefits of any of the conformal methods would be minimal. For implementing both UACQR methods, we use the epoch-based heuristic to avoid refitting and set $B = 999$. For UACQR-S, CQR, and CQR-r, we let the base estimator $\hat{q}_{Y|X}(x,a)$ be the fully trained network $\hat{q}^B_{Y|X}(x,a)$.

Quantile crossing is a phenomenon in which a quantile regressor may predict a lower value for an upper quantile than for a lower quantile. This tends to occur out-of-sample, when extrapolating. A common post-processing mechanism to address quantile crossing is isotonization: when estimating two quantiles, in the event of crossing, set the upper and lower quantile outputs equal to the average of the two (Mammen, 1991; Chernozhukov et al., 2010). We ran our experiments with and without this post-processing and received nearly identical results. The likelihood of quantile crossing was reduced in our simulation setting, since the distribution of X was bounded.

B.2 Real-world data sets

We use Quantile Regression Forests for quantile regression with these data sets. We use default hyperparameter settings, with the exception of `min_samples_leaf`, i.e. the minimum number of samples a leaf can contain. As discussed in Appendix A, the way in which we use Quantile Regression forests is different for UACQR-P from the rest. Since with UACQR-P we calculate the target quantiles in each leaf, there is a heightened need for `min_samples_leaf` to be greater than 1 for our implementation of UACQR-P to have decent estimations in each leaf. We end up choosing this hyperparameter via a cross-validation procedure outlined below. In general, we observe that the optimal `min_samples_leaf` for UACQR-P is usually 5 and 1 for the rest of the methods. For both UACQR methods, we use $B = 100$. For UACQR-S, CQR, and CQR-r, we let the base estimator $\hat{q}_{Y|X}(x, a)$ be the full QRF, with $B = N = 100$ trees, using the standard QRF implementation.

To ensure exchangeability, we sample the training, calibration, and test sets without replacement from the full data sets. We use 40% of the points for training, 40% for calibration, and 20% for testing. For computability, for the `imdb_wiki`, `bio`, and `homes` data sets, we use only use 25%, 25%, and 50%, respectively, of the full data sets in any run. For clarity, with `bio` in each run 10% of all data points are used for training, 10% for calibration, 5% for testing, and 75% not at all.

To select hyperparameters, once we do our initial train/calibration/test splits, we further divide the training set again with a 40%/40%/20% breakdown for the purposes of cross-validation. We then perform each of the conformal procedures using this sub-splitting of the training data with the various hyperparameter settings. We note the hyperparameter setting for each method that minimizes interval score loss on the 20% sub-split of the training data. We then use these settings for each method on the initial train/calibration/test splits to measure final performance. In short, we selected hyperparameters without using oracle knowledge or data-snooping, ensuring realistic results.

For the `cbc` data set, we increase the number of samples by also including horizontal and vertical flips of the images, increasing the total number of sample from 360 to 1,080. For the `imdb_wiki` data set, we note that we only use the Wikipedia images to reduce the number of samples to a more manageable size. In addition, we only use images taken before 2005, since otherwise the distribution of when photos were taken would skew heavily toward more recent years.

As done in Romano et al. (2019), we divide the responses by the mean absolute value response of the training set. This ensures that the rows in Table 1 each have similar magnitude.

We do not need to worry about quantile crossing for the experiments with Quantile Regression Forests, since each quantile estimate corresponds to the sample quantile of the same set of samples. We undo quantile crossing for our neural network experiment on real-world data.

In all experiments, including the additional ones in Appendix E.2, for UACQR-S we calculate $\hat{g}_{\text{lo}}(x), \hat{g}_{\text{hi}}(x)$ using standard deviations, as described in Section 3.1. However, we have observed that UACQR-S's performance on these real data experiments often improves if we use interquartile range instead, likely due to the metric's robustness to outliers.

B.3 Evaluation Metrics

Interval Score Loss For real-world data sets, we mainly evaluate our methods with interval score loss:

$$\begin{aligned}\ell^{\text{is}}(y, \hat{q}_{\text{lo}}(x,\hat{t}), \hat{q}_{\text{hi}}(x,\hat{t}); \alpha) = \hat{q}_{\text{hi}}(x,\hat{t}) - \hat{q}_{\text{lo}}(x,\hat{t}) \\ + \frac{2}{\alpha} \cdot (\hat{q}_{\text{lo}}(x,\hat{t}) - y)\mathbf{1}_{y<\hat{q}_{\text{lo}}(x,\hat{t})} \\ + \frac{2}{\alpha} \cdot (y - \hat{q}_{\text{hi}}(x,\hat{t}))\mathbf{1}_{y>\hat{q}_{\text{hi}}(x,\hat{t})}\end{aligned} \tag{8}$$

The first two terms can be viewed as the interval width, and the remaining terms can be viewed as penalties for not covering by the added distance needed to have covered. Therefore, by minimizing interval score loss, we are balancing the goal of minimizing interval widths with the goal of not undercovering.

As mentioned in Section 5, interval score loss is a proper scoring rule (Gneiting and Raftery, 2007), so it is minimized in expectation by the true $\frac{\alpha}{2}, 1 - \frac{\alpha}{2}$ quantiles. In fact, interval score loss is proportional to the sum

of pinball losses for the $\frac{\alpha}{2}, 1-\frac{\alpha}{2}$ quantiles. Therefore, it is ideally used for measuring the performance of $1-\alpha$ prediction intervals that are intended to be *symmetric*, with equal probability of miscovering above and below. This matches our use case.

Additional metrics Feldman et al. (2021) propose additional metrics that may be useful for measuring conditional coverage properties. Chief among their suggestions is the dependence between a coverage indicator variable $\mathbf{1}_{Y_i \in \hat{C}_n(X_i)}$ and interval width $\hat{q}_{\text{hi}}(X_i, \hat{t}) - \hat{q}_{\text{lo}}(X_i, \hat{t})$, usually as measured by the correlation between the two random variables. If there is dependence, then we may view this as a violation of conditional coverage, since a necessary condition of conditional coverage is $\mathbf{1}_{Y_i \in \hat{C}_n(X_i)}$ being independent of *all* measurable functions of X_i. With this in mind, we see that achieving zero correlation on their metric is only a necessary but not sufficient condition for conditional coverage, as even degenerate constructions can achieve zero correlation, such as any prediction interval with constant width. Feldman et al. (2021) provide additional metrics, but each is also susceptible to a degenerate construction scoring well. In contrast, it may be very difficult or impossible to minimize interval score loss in expectation, depending on regularity conditions, without using the true $\frac{\alpha}{2}, 1-\frac{\alpha}{2}$ conditional quantiles.

C RANDOMIZED CUTOFFS FOR EXACT COVERAGE

The coverage guarantee obtained by methods within the conformal prediction framework is generally written as a lower bound, i.e., $\mathbb{P}\{Y_{n+1} \in \hat{C}_n(X_{n+1})\} \geq 1-\alpha$. Examining the construction for the nested split conformal case specifically, given in (3) and (4), we can see that the true probability of coverage might be higher than $1-\alpha$, for two reasons: first, due to rounding error (if $(1-\alpha)(n_1+1)$ is not an integer), and second, due to ties among the conformity scores, which in this case are given by the values

$$t_i = \inf\{t \in \mathcal{T} : Y_i \in \hat{C}_{n_0,t}(X_i)\}.$$

These issues may be problematic when comparing the empirical performance of various methods—e.g., if we see one method has wider prediction intervals than another, is this a genuine difference or might it simply be due to overcoverage arising from rounding error or from ties? For this reason we use a modified version of the procedure for our real data experiments, as detailed next.

To alleviate this issue of overcoverage, Vovk et al. (2005) proposes a randomization strategy, referred to as "smoothed conformal predictors" in that work. Here we show calculations for how to specialize Vovk et al. (2005)'s randomization strategy to the specific setting of split conformal via nested sets, as constructed in (4).

Let $k = \lceil (1-\alpha)(n_1+1) \rceil$, and let $\delta = k - (1-\alpha)(n_1+1)$, capturing the rounding error. Let t_i be defined as above for each $i = n_0+1, \ldots, n$ in the calibration set, and define the order statistics $t_{(1)} \leq \cdots \leq t_{(n_1)}$ of this list. Define

$$T_0 = \#\{i \leq k-1 : t_{(i)} = t_{(k-1)}\} \text{ and } T_1 = \#\{i \geq k : t_{(i)} = t_{(k)}\},$$

which capture information about the number of ties for $(k-1)$st or kth place. Then we split into two cases: if $t_{(k-1)} < t_{(k)}$, then define

$$\hat{C}_n(X_{n+1}) = \begin{cases} (\hat{q}_{\text{lo}}(X_{n+1}, t_{(k-1)}), \hat{q}_{\text{hi}}(X_{n+1}, t_{(k-1)})), & \text{with probability } \frac{\delta}{T_0+1}, \\ [\hat{q}_{\text{lo}}(X_{n+1}, t_{(k-1)}), \hat{q}_{\text{hi}}(X_{n+1}, t_{(k-1)})], & \text{with probability } \frac{\delta T_0}{T_0+1}, \\ (\hat{q}_{\text{lo}}(X_{n+1}, t_{(k)}), \hat{q}_{\text{hi}}(X_{n+1}, t_{(k)})), & \text{with probability } \frac{(1-\delta)T_1}{T_1+1}, \\ [\hat{q}_{\text{lo}}(X_{n+1}, t_{(k)}), \hat{q}_{\text{hi}}(X_{n+1}, t_{(k)})], & \text{with probability } \frac{1-\delta}{T_1+1}, \end{cases}$$

while if instead $t_{(k-1)} = t_{(k)}$, then define

$$\hat{C}_n(X_{n+1}) = \begin{cases} (\hat{q}_{\text{lo}}(X_{n+1}, t_{(k)}), \hat{q}_{\text{hi}}(X_{n+1}, t_{(k)})), & \text{with probability } \frac{T_1+\delta}{T_0+T_1+1}, \\ [\hat{q}_{\text{lo}}(X_{n+1}, t_{(k)}), \hat{q}_{\text{hi}}(X_{n+1}, t_{(k)})], & \text{with probability } \frac{T_0+1-\delta}{T_0+T_1+1}. \end{cases}$$

Note that we might have $k = n_1+1$ (i.e., the order statistic $t_{(k)}$ is not defined, or rather, is taken to be $+\infty$). This happens in the case where $\alpha(n_1+1) < 1$. In this case, we take $t_{(n_1+1)} := +\infty$ and all the above definitions are still valid. (Observe that this scenario would fall into the first case, $t_{(k-1)} < t_{(k)}$, since $t_{(k-1)} = t_{(n_1)}$ is finite while $t_{(k)}$ is set to be $+\infty$.)

D LIPSCHITZ PROPERTY FOR UACQR-P

As mentioned earlier, the UACQR-P construction preserves the smoothness properties of the underlying base estimators. Here we will verify that if each of the bootstrapped estimates $\hat{q}^b_{Y|X}(x,a)$ is L-Lipschitz (as a function of x), then this property is preserved by the function $\hat{q}^{(b)}_{Y|X}(x,a)$, for each b. In particular, the left- and right-endpoint functions of the prediction band $\hat{C}_n = \{\hat{C}_n(x)\}$ will then be L-Lipschitz as well. The following basic result verifies this claim.

Proposition 1. *Let $f_1, \ldots, f_B : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is a normed space. For any $b \in \{1, \ldots, B\}$, define the function $f_{(b)} : \mathcal{X} \to \mathbb{R}$ as*

$$f_{(b)}(x) = \inf\left\{t \in \mathbb{R} : \sum_{i=1}^{B} \mathbf{1}_{f_i(x) \le t} \ge b\right\}.$$

(In other words, for each x, $f_{(1)}(x) \le \cdots \le f_{(B)}(x)$ are the order statistics of $f_1(x), \ldots, f_B(x)$.)

Then, if f_b is L-Lipschitz for all b, it holds that $f_{(b)}$ is L-Lipschitz for all b.

Proof. Suppose $f_{(b)}$ is not L-Lipschitz. Then there must exist some $x_0, x_1 \in \mathcal{X}$ with $|f_{(b)}(x_0) - f_{(b)}(x_1)| > L\|x_0 - x_1\|$. Without loss of generality, we can assume $f_{(b)}(x_1) > f_{(b)}(x_0)$.

By definition of $f_{(b)}(x_0)$ as the bth order statistic of $f_1(x_0), \ldots, f_B(x_0)$, we have

$$\sum_{i=1}^{B} \mathbf{1}_{f_i(x_0) \le f_{(b)}(x_0)} \ge b.$$

Similarly, by definition of $f_{(b)}(x_1)$ as the bth order statistic of $f_1(x_1), \ldots, f_B(x_1)$, we have

$$\sum_{i=1}^{B} \mathbf{1}_{f_i(x_1) \ge f_{(b)}(x_1)} \ge B - b + 1.$$

Combining these two facts, we must have at least one $i \in \{1, \ldots, B\}$ for which it holds that

$$f_i(x_0) \le f_{(b)}(x_0) \text{ and } f_i(x_1) \ge f_{(b)}(x_1) > f_{(b)}(x_0) + L\|x_0 - x_1\|.$$

This is a contradiction, since f_i is L-Lipschitz. □

E ADDITIONAL EXPERIMENTS

E.1 Simulation

In Figure 2 we saw that UACQR methods are able to maintain conditional coverage near $X = 0$, where epistemic uncertainty was high, without overcovering significantly when epistemic uncertainty was low, near $X = 1$. In Figure 4, we see that this adaptivity usually yields more precise intervals on average. Near $X = 0$, the UACQR methods provide wider intervals than the existing methods in order to achieve conditional coverage. This does not yield wider intervals on average everywhere else, though, with UACQR methods providing intervals as narrow as other methods when epistemic uncertainty is low. This further demonstrates the locally adaptive properties of our methods in this simulation setting.

E.2 Real-world data sets

In this section, we test several alternative implementations of our real data experiments, to verify that our results persist across different choices made in the implementation. We also verify the marginal coverage properties, as guaranteed by the theory and the construction of our train/calibration/test splits.

As before, we bold outperformance and underline when statistically significant. Our measure for statistical significance for each data set is a two-sample t-test between the best performing method and the second best performing method, with no multiplicity corrections.

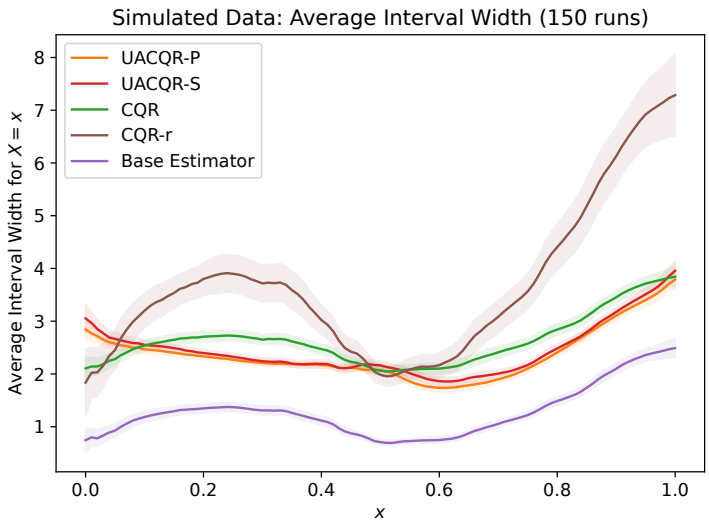

Figure 4: In the same setting as Figure 2, we see that the two UACQR methods have on average shorter interval widths than CQR and CQR-r for most values of X. When epistemic uncertainty is high ($X = 0$), our methods have wider intervals, allowing them to achieve conditional coverage in this region.

Table 2: Real data results on 20 runs: marginal coverage

	Average Test Coverage (Standard Error)			
Dataset	UACQR-P	UACQR-S	CQR	CQR-r
`bike`	0.904 (0.002)	0.900 (0.002)	0.900 (0.002)	0.900 (0.002)
`bio`	0.900 (0.001)	0.900 (0.002)	0.901 (0.002)	0.900 (0.002)
`cbc`	0.896 (0.006)	0.897 (0.006)	0.893 (0.006)	0.894 (0.006)
`community`	0.905 (0.005)	0.902 (0.003)	0.901 (0.003)	0.899 (0.003)
`concrete`	0.902 (0.006)	0.896 (0.006)	0.893 (0.006)	0.896 (0.005)
`forest`	0.902 (0.003)	0.902 (0.003)	0.902 (0.003)	0.902 (0.003)
`homes`	0.901 (0.002)	0.901 (0.001)	0.901 (0.001)	0.902 (0.002)
`imdb_wiki`	0.897 (0.005)	0.896 (0.005)	0.893 (0.005)	0.894 (0.005)
`star`	0.898 (0.003)	0.898 (0.004)	0.898 (0.004)	0.898 (0.004)

In Table 2, we provide the average test coverage for the experimental results already provided in Section 5. We confirm that we have the desired marginal coverage of 90%, with slight variations due only to using 20 runs.

In Table 3, we see that our methods also usually provide more precise intervals on average, as measured by average test interval width. The one exception is the `star` data set. We see that these instances of outperformance for UACQR-P are statistically significant in seven of the eight cases.

In Table 4, we measure performance again in this same setting as Section 5, but this time we use the Feldman et al. (2021) metric to measure conditional coverage properties. Specifically, as discussed in Appendix B, we provide the absolute value of the correlation on test data between a coverage indicator variable $\mathbf{1}_{Y_i \in \hat{C}_n(X_i)}$ and interval width $\hat{q}_{\text{hi}}(X_i, \hat{t}) - \hat{q}_{\text{lo}}(X_i, \hat{t})$. Our cross-validation procedure on this experiment chooses hyperparameters that minimize this metric, as opposed to interval score loss. We can see that UACQR-P provides the best results on four of the nine data sets, including in a significant way on one of them. UACQR-S and CQR-r each perform uniquely the best on two data sets, and they tie for best on the final data set. Observe that most of these differences are not statistically significant. As we discussed in Appendix B, this metric may be minimized even under degenerate constructions. Therefore, we interpret these results as still underscoring the strong conditional coverage properties of UACQR-P, by performing better or similarly well on this metric as any other while doing better almost always on the more discriminating interval score loss metric.

Table 3: Real data results on 20 runs: interval width

	Average Test Interval Width (Standard Error)			
Dataset	UACQR-P	UACQR-S	CQR	CQR-r
bike	**1.184** (0.007)	1.456 (0.006)	1.484 (0.006)	1.440 (0.006)
bio	**1.530** (0.007)	1.654 (0.004)	1.656 (0.004)	1.653 (0.004)
cbc	**0.827** (0.013)	1.004 (0.012)	0.998 (0.013)	1.002 (0.012)
community	**1.487** (0.028)	1.774 (0.021)	1.779 (0.021)	1.774 (0.022)
concrete	**0.678** (0.015)	0.734 (0.009)	0.732 (0.008)	0.711 (0.008)
forest	**2.269** (0.012)	2.361 (0.008)	2.363 (0.008)	2.363 (0.008)
homes	**0.633** (0.005)	0.699 (0.004)	0.758 (0.004)	0.714 (0.005)
imdb_wiki	**1.558** (0.018)	1.799 (0.011)	1.796 (0.011)	1.798 (0.011)
star	0.176 (0.001)	0.176 (0.001)	**0.175** (0.001)	0.176 (0.002)

Table 4: Real data results on 20 runs on Feldman et al. (2021)'s metric

	Correlation between Coverage and Width (Standard Error)			
Dataset	UACQR-P	UACQR-S	CQR	CQR-r
bike	**0.038** (0.005)	0.130 (0.008)	0.217 (0.007)	0.119 (0.007)
bio	0.028 (0.004)	0.029 (0.005)	0.027 (0.004)	**0.024** (0.004)
cbc	**0.059** (0.012)	0.072 (0.009)	0.062 (0.011)	0.073 (0.012)
community	0.063 (0.009)	**0.062** (0.009)	0.066 (0.012)	0.063 (0.008)
concrete	**0.105** (0.014)	0.113 (0.014)	0.197 (0.016)	0.112 (0.014)
forest	0.054 (0.007)	**0.047** (0.007)	0.051 (0.008)	**0.047** (0.007)
homes	0.029 (0.005)	0.027 (0.004)	0.096 (0.004)	**0.019** (0.004)
imdb_wiki	0.079 (0.012)	**0.073** (0.015)	0.078 (0.014)	0.078 (0.014)
star	**0.125** (0.011)	0.132 (0.014)	0.128 (0.013)	0.132 (0.015)

In Table 5, we show results after taking a log transformation (specifically, replacing Y_i with $\log(1 + Y_i)$ for all data points, to allow for zero values). We do this instead of the usual mean normalization, so the units in Table 5 are different from other tables. All other details of the implementation remain the same. We observe that UACQR-P continues to show the best performance in terms of interval score loss on almost all data sets, with the exception of `star` and `imdb_wiki`.

In Table 6, we show the results after running the original experiment without the log transformation, except that we do not use randomization for handling ties/rounding (as detailed in Appendix C); instead, we use the original definition of the split nested conformal method given in (4) earlier. Here we see that the UACQR-P method is generally producing the lowest interval score loss, with the exception of the `star` and `cbc` data sets, same as with Table 1.

In Table 7, we show the results when the methods are run using a neural network, rather than Quantile Regression Forests, as the base algorithm. The model architecture is similar to what is described in Appendix B.1, except that we used an extensive hyperparameter grid search over number of epochs, learning rate, batch size, weight decay penalty, dropout rate (Srivastava et al., 2014), and size of the hidden layers. For each method, we report the average test interval score loss (across 20 trials) after choosing for each trial the best hyperparameter choices among 25 randomly drawn options using our cross-validation procedure. Again, we see that the UACQR-P method generally provides the lowest interval score loss.

Table 5: Real data results on 20 runs with log transformation.

	Average Test Interval Score Loss (Standard Error)			
Dataset	UACQR-P	UACQR-S	CQR	CQR-r
bike	**2.048** (0.012)	2.212 (0.009)	2.224 (0.009)	2.201 (0.01)
bio	**1.843** (0.014)	1.88 (0.008)	1.879 (0.008)	1.879 (0.008)
cbc	**1.582** (0.051)	1.602 (0.034)	1.599 (0.034)	1.598 (0.033)
community	**0.378** (0.006)	0.379 (0.005)	0.379 (0.005)	0.379 (0.005)
concrete	**0.984** (0.019)	1.027 (0.024)	1.035 (0.021)	1.021 (0.025)
forest	**0.649** (0.002)	0.656 (0.001)	0.656 (0.001)	0.656 (0.001)
homes	**0.829** (0.005)	0.851 (0.005)	0.855 (0.004)	0.846 (0.005)
imdb_wiki	3.409 (0.061)	3.393 (0.032)	3.394 (0.032)	**3.392** (0.032)
star	0.212 (0.001)	**0.211** (0.001)	**0.211** (0.001)	**0.211** (0.001)

Table 6: Real data results on 20 runs with no randomized cutoffs.

	Average Test Interval Score Loss (Standard Error)			
Dataset	UACQR-P	UACQR-S	CQR	CQR-r
bike	**1.447** (0.013)	1.573 (0.008)	1.586 (0.008)	1.564 (0.009)
bio	**1.880** (0.012)	1.900 (0.010)	1.900 (0.010)	1.900 (0.011)
cbc	1.278 (0.023)	1.275 (0.016)	**1.273** (0.016)	**1.273** (0.016)
community	**2.112** (0.042)	2.156 (0.033)	2.157 (0.034)	2.156 (0.033)
concrete	**0.842** (0.019)	0.882 (0.015)	0.881 (0.015)	0.864 (0.015)
forest	**2.439** (0.014)	2.461 (0.010)	2.462 (0.010)	2.461 (0.010)
homes	**0.880** (0.012)	0.916 (0.011)	0.930 (0.010)	0.911 (0.011)
imdb_wiki	**1.841** (0.018)	1.926 (0.010)	1.927 (0.010)	1.926 (0.010)
star	0.214 (0.001)	**0.213** (0.001)	**0.213** (0.001)	**0.213** (0.001)

Table 7: Real data results on 20 runs with neural network as base algorithm.

	Average Test Interval Score Loss (Standard Error)			
Dataset	UACQR-P	UACQR-S	CQR	CQR-r
bike	**1.204** (0.015)	1.301 (0.018)	1.322 (0.020)	1.317 (0.018)
bio	**2.020** (0.016)	2.073 (0.015)	2.076 (0.016)	2.074 (0.017)
cbc	1.291 (0.040)	1.293 (0.027)	1.265 (0.020)	**1.256** (0.021)
community	**2.372** (0.050)	2.546 (0.054)	2.573 (0.044)	2.598 (0.053)
concrete	**1.832** (0.028)	**1.832** (0.024)	1.839 (0.025)	1.842 (0.026)
forest	**2.639** (0.030)	2.731 (0.023)	2.815 (0.026)	2.830 (0.024)
homes	**0.834** (0.011)	0.918 (0.013)	0.927 (0.015)	0.912 (0.011)
imdb_wiki	2.062 (0.040)	2.037 (0.033)	**1.954** (0.028)	2.002 (0.026)
star	**0.240** (0.003)	0.245 (0.003)	0.250 (0.004)	0.258 (0.004)