CLOSING THE COVERAGE GAP IN SIMULTANEOUS QUANTILE REGRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Simultaneous quantile regression builds on quantile regression by minimizing the expected pinball loss to approximate all quantiles at once. This enables the production of a complete implicit cumulative distribution of a prediction interval, or any subset of this function, without assuming any prior distribution shape. Existing implementations of this technique show under-covered distribution tails compared to the observed distribution of the data. Such calibration errors cause the underestimation of risk or uncertainty of a model. Contrary to this observation, regular quantile regression calibrates well on the same dataset and model configurations. In this work, we investigate the cause of the gap in coverage between simultaneous quantile regression and regular quantile regression and identify a category of solutions to the problem. Our empirical evaluation shows that input quantiles need to be dynamically scaled to result in non-linear tail coverage. Our own scaling approaches work just as well as previous literature, while using fewer parameters. We further show that while tail coverage could also be recalibrated with conformalized quantile regression, the post-processing produces a better distributional fit with scaled quantile models.

1 INTRODUCTION

The ability to quantify and assess the uncertainty of a prediction is very important in many regression applications. There is increasing attention to uncertainty, or probabilistic modelling, in application fields such as medical risk prediction, weather, or financial forecasts (Akrami et al., 2022; Price et al., 2025; Eggen et al., 2026), such that even general deep learning advances, e.g., Time Series Foundation Models (Das et al., 2024; Ansari et al., 2025), use quantile regression (QR) to produce a prediction interval. Prediction intervals formed by a probabilistic distribution carry more information than a single-value prediction, supporting decision-making processes (Visser et al., 2024) and may foster trust by communicating uncertainty.

Probabilistic distributions can be evaluated using different criteria than non-probabilistic regression tasks in neural networks, and they require several metrics for comparison. We are interested not only in how well the overall distribution fits the data, but also in the coverage of observed values by the modelled distribution. If adequate coverage is reached (i.e. the distribution is calibrated), then we can prefer the distribution with the least possible width (i.e. to optimize the sharpness of the distribution) (Gneiting et al., 2007).

Probabilistic modelling encompasses a range of techniques that can express a distribution or samples of a distribution, in order to quantify the uncertainty of a prediction. One categorization of such techniques is a parametric or non-parametric method. The aforementioned QR (Koenker & Bassett, 1978) is a non-parametric method, popular for not requiring prior assumptions of distributional shape, resulting in superior calibration across tasks or domains. The downside of QR is that it predicts a set of a priori determined quantiles and not the complete underlying distribution of the data. In other words, QR provides a categorical set of quantiles instead of a continuous distribution.

A recent improvement on QR is called implicit quantile network (IQN)(Dabney et al., 2018), simultaneous quantile regression (SQR)(Tagasovska & Lopez-Paz, 2019), or, very recently, also any-quantile regression (AQR) (Smyl et al., 2026a). Its core idea is to learn an inverse cumulative distribution function (CDF), also called the quantile function, by training a network with randomized

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

quantiles as an input. There are some inherent differences between IQN, SQR, and AQR, mainly related to how quantiles are used as an input to the networks. SQR uses the raw quantiles as input, representing the closest approach to that core idea, which causes us to use this name in this paper unless a distinction becomes necessary. SQR presents a bridge between parametric and non-parametric techniques, and enables training a model without knowing the resolution of the quantile distribution beforehand. Modelling all quantiles at once also presents a useful regularization effect where neighboring quantiles more directly influence each other (Tagasovska & Lopez-Paz, 2019), further explored by Narayan et al. (2023).

There are also downsides to SQR. The literature often observes SQR having a narrow conditional distribution that does not adequately cover the observed distribution of the data. Different works attribute this observation to various parts of the approach itself (Dabney et al., 2018; Chung et al., 2021; Narayan et al., 2023; Smyl et al., 2026a). Regular QR does not have an overly sharp distribution, but instead is often reported as a best-fitting method across tasks. This distinction is surprising, as QR and SQR use the same loss function and generally fit the same tasks. The only difference is in the quantile training procedure and the output size of the network.

It can not be overstated that the tails of the prediction interval are essential in the application fields of SQR, as a financial or weather forecast is expected to accurately capture the risk of any possible change in the forecast. Further, for simple applications of QR, it is very common to predict the median and the uncertainty interval spanned only by the tails of the distribution. If the tail prediction is unreliable, SQR can not be seen as a viable improvement upon QR. In order to meet the full potential of SQR as a flexible bridge between parametric and non-parametric techniques, SQR would need to be as well calibrated as QR. For this purpose, we are investigating SQR's overly sharp distribution in detail and providing solutions to avoid the collapse of SQR's tails. In the following, we will first lay out the problem, then examine related literature, deconstruct the cause, and evaluate solutions by empirical testing on seven distinct real datasets.

1.1 Problem description

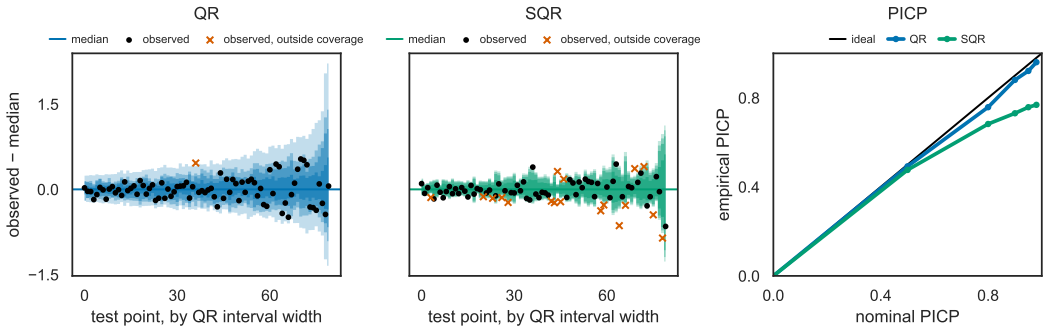


Figure 1: Comparison of QR and SQR distributions under the same training conditions in 80 random test samples from the King County dataset. Results are ordered by coverage width and centred on the median of the approximated distribution. Color shading indicates interval width up to the 1-99% prediction interval. Red crosses show the undercovered parts of the sample.

Let a dataset $\boldsymbol{D}$ consist of N pairs $\{\boldsymbol{x}, y\}$ of target observations y and predictive features $\boldsymbol{x} \in \mathbb{R}^d$. In QR, given a neural network $\hat{y} = f(\boldsymbol{x})$, where $\hat{y}$ is an approximation of the observed y in which we try to optimize a loss function L in $arg\min_f \mathbb{E}[L(f(\boldsymbol{x}), y)]$. While QR was initially just used to model one quantile, nowadays QR with neural networks is implemented by choosing a set of quantiles (Takeuchi et al., 2006), e.g., $\boldsymbol{t} = [0.01, 0.025, ..., 0.975, 0.99] \in [0, 1]^T$ and using the corresponding $\hat{y}$ as the output of the model $f : \mathbb{R}^d \to \mathbb{R}^T$. Usually, pinball loss (also called quantile loss) is chosen as L, as it is a proper scoring rule (Gneiting & Raftery, 2007):

$$L_\tau(y, \hat{y}) = \begin{cases} \tau(y - \hat{y}) & \text{if } y - \hat{y} \geq 0, \\ (1 - \tau)(\hat{y} - y) & \text{else.} \end{cases} \tag{1}$$

In the multiple quantile training of QR, the loss is then taken as the average across the set of quantiles $\frac{1}{T} \sum_{j=1}^{T} L_{\boldsymbol{t}(j)}(y, \hat{y}_{\boldsymbol{t}(j)})$.

SQR instead draws $\tau \sim U(0,1)$ for each sample during training and inserts it as an additional model input $f(\boldsymbol{x}, \tau)$. This way, we do not need to predefine a set of quantiles $\boldsymbol{t}$, so $f : \mathbb{R}^{d+1} \to \mathbb{R}$ during training. This way, we create an inverse CDF. The optimization problem of f for SQR is thus:

$$\arg\min_{f} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{\tau_i \sim U(0,1)} L_{\tau_i}(f(\boldsymbol{x}_i, \tau_i), y_i). \tag{2}$$

During exploitation or evaluation of the network, we can then query for a set of quantiles by expanding the dimensionality of the query $f : \mathbb{R}^{d \times T} \to \mathbb{R}^T$, enabling the approximation of a CDF with sufficient T. Put differently, the SQR implicitly models a full CDF, but needs to be queried at quantile levels. Note that this formulation is agnostic to the structure of $\boldsymbol{x}$.

Under identical training and architecture conditions, SQR shows a narrower prediction interval than QR, in which the tails of the distribution partially collapse towards the median. Figure 1 shows this effect compared to regular QR while trained on one of the datasets used in the empirical part of this work. The figure also shows the prediction interval coverage probability (PICP) in the third plot of Figure 1 compared to the ideal coverage probability. SQR is clearly undercovering the tails of the distribution.

1.2 Related work

The idea of SQR was first published by Dabney et al. (2018) as IQN, a reinforcement learning approach, and then within the probabilistic regression context by Tagasovska & Lopez-Paz (2019) as SQR. Dabney et al. (2018) reported insufficient performance with a linear τ and instead proposed a cosine embedding to enrich the representation of τ in the model. However, they compare performance on Atari games, without including a prediction interval coverage evaluation. On the other hand, while Tagasovska & Lopez-Paz (2019) did not embed τ, they only tested SQR by training and evaluating it on two opposing quantiles and used a static scaling factor (of 12; not reported in their paper, but present in the published code). Subsequent work in the probabilistic regression domain preferred testing on SQR over IQN and did not embed τ in the same way again.

Chung et al. (2021) introduced a calibration-sharpness loss on SQR, meant to use the probabilistic evaluation criteria of coverage (calibration) and interval width (sharpness) as an explicit loss to optimize SQR. They also observed that regular SQR has a sharp distribution and reported a wider distribution using their loss.

In Narayan et al. (2023), the authors used a partially monotonic network on top of SQR to produce an inverse CDF. In their analysis, they compare QR to SQR and show that SQR regularizes quantiles, revealing QR to be more efficient at extreme quantiles. They also experiment with using a β-distribution instead of Uniform for the random sampling of τ and observe that it enables their models to represent skewed data more closely. They do not analyse or compare QR and SQR on prediction interval coverage. However, (Narayan et al., 2023) discusses the phenomenon of quantile collapse without linking it to an experiment, defining it as a flat distribution of $f(\tau)$ over some τ. They argue that quantile collapse may occur in regions of sparse data, or when a model has too much flexibility in the input, causing the data to act sparse with respect to that flexibility.

A similar line of thought can be found in what is called the extreme QR literature (e.g., (Velthoen et al., 2023)). Here, quantile approximation with QR fails in extreme quantile values, when $\tau \to 1, 0$ as there are so few data points that the pinball loss becomes flat and invalid. However, this can not explain the distinction between SQR and QR as described in section 1.1, as QR has enough data points to produce a calibrated distribution. Otherwise, quantile collapse has been observed to be caused by smoothed pinball loss (e.g. huber pinball) (Lopez-Martin et al., 2021).

Erdmann et al. (2026) also examined a partially monotonic network with SQR. They reported that they needed to increase the learning rate during training in the first couple of epochs to widen their prediction distribution. In our own testing, we found that the initial widening will collapse again if training is continued.

Lastly, Smyl et al. (2026a) benchmark architectures in an electricity demand forecasting task, featuring an extension on SQR they call AQR. They compare 3 quantile injection schemes using quantiles drawn from a beta or uniform distribution, but report only coverage error for the overall model

comparison. They report using the beta distribution to mitigate diminishing gradients caused by a saturating activation function. AQR uses an additive shift and multiplicative modulation based on linear embeddings of τ, which they argue is less sensitive to otherwise non-normalized data than a simple concatenation. When they reported average- and extreme coverage, a linear QR came out on top, but followed closely by their AQR approach. They conclude that AQR has overall standout performance except in the extreme quantiles. In subsequent work Smyl et al. (2026b) use similar methods and again report performance issues with extreme quantiles. There, they argue that the relatively small magnitude of pinball loss near the tail end of the distribution may cause the coverage issue.

As described above, collapsed tails or extreme value miscalibration in SQR have been observed before. However, those works do not investigate these observations with a specific focus on quantile collapse, nor show coverage comparisons with the goal of evaluating extensions on SQR. The literature also does not cover a general explanation or solution to the problem.

1.3 Contribution

This paper is focused on a thorough study of the prediction interval tail collapse in SQR. Building up on the gap left by current literature, we argue that the collapsing tails of SQR are caused by the affine initialization of the network in τ, combined with a gradient update that would need to accumulate nonlinearity in τ across samples, but is biased against regions of low data density. This results in an overly smooth function that is approximately affine with respect to τ. In our observation, large enough perturbations of τ will enable the function of τ to be non-linear in τ, solving the problem. To empirically test this hypothesis, we benchmark our own proposals for τ-scalers and two existing approaches in the literature that previously remained untested in prediction interval coverage comparisons. While we show the set of τ-scalers to successfully close the coverage gap from SQR to QR, we also document a number of approaches that do not perturb τ sufficiently. Our benchmark across seven independent tabular datasets reveals that approaches like IQN, AQR, and other τ perturbations reach QR-level calibration. In this paper, we examine how τ enters a multi-layered perceptron (MLP), which should be independent of the structure of $\boldsymbol{x}$. We leave extensions in architecture or data type for future work.

The main goal of this work is to answer the questions of why SQR shows a too-sharp distribution and in what way the distribution can be calibrated to match or surpass QR. Our contributions can thus be summarized as:

- We offer our observations and propose an explanation for the cause of the narrow SQR distribution in comparison to QR.
- We compile a benchmark and compose a family of solutions originating from our own ideas and previous literature.
- We arrive at a final recommendation regarding optimal SQR construction for prediction interval coverage.

2 Cause for collapsing tails

We know that regular QR shows better coverage under the same data, training, and architectural conditions. Thus, the phenomenon should have a different cause. In QR, we estimate conditional quantile values $\hat{y}_\tau$ for a predefined set of probability levels $\tau \in [0, 1]$ with independent parameters in the output layer for each quantile. As such, pinball optimizes each τ such that it approximates the observed distribution for that τ. The optimization objective is separable in τ and consistent for each τ. The difference from QR to SQR then is that we now share weights between conditional quantile estimates, so instead of $f(\boldsymbol{x}) = \hat{\boldsymbol{y}} \in \mathbb{R}^T$, we have $f(\boldsymbol{x}, \tau) = \hat{y}_\tau$. In this section, expectations and distributions are conditional on $\boldsymbol{x}$. We simplify $f(\boldsymbol{x}, \tau) = \hat{y}_\tau$ to $\hat{y}$ and omit the conditioning.

The function $f(\boldsymbol{x}, \tau)$ has a secondary objective, which is the approximation of the distribution, or rather the curvature of τ across any point of $\hat{y}$. A key insight here is that this is the failure point for SQR. As can be seen in Figure 2, the CDF stays approximately linear for regular SQR during training, implying the underlying distribution is uniform. In practice, this causes a sharper distribution with lower coverage.

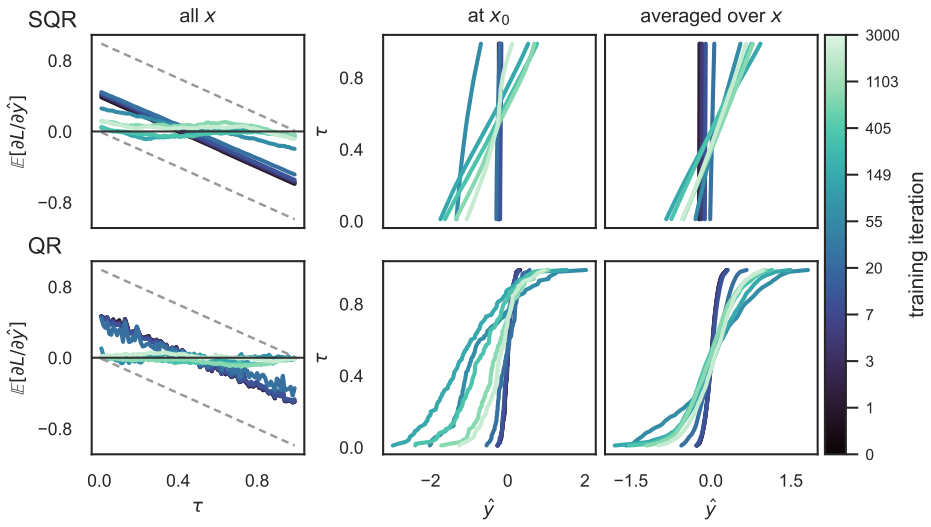


Figure 2: Training of QR and SQR across samples in the elections dataset. The first row is SQR and the second row is QR. The first column shows $\mathbb{E}[\partial L/\partial \hat{y}]$ over the samples as a function of τ. The other two columns are $\hat{y}$ for all τ across a single sample x, and averaged across all samples x. The dashed lines are the limits of the loss derivative given by $[-\tau, 1-\tau]$.

In contrast to SQR, QR learns to estimate a CDF close to the normal distribution. We can see in the tails of the CDF of QR that a small change in τ causes large changes in $\hat{y}$. For SQR, f seems to start almost constant in τ and train to be affine in τ.

Neural network activations Given that we are using a neural network to approximate τ, we expect that it should be able to produce non-linear output dependent on τ. The observation in Figure 2 does not seem to agree with this assertion.

Assuming a simple neural network with one hidden layer, width m, ReLU activation σ, and standardized input $\boldsymbol{x} \in \mathbb{R}^d$,

$$\hat{y} = f(\boldsymbol{x}, \tau) = \sum_{j=1}^{m} c_j \sigma(\boldsymbol{w}_j^\top \boldsymbol{x} + b_j + v_j \tau). \tag{3}$$

If we initialize with He (He et al., 2015) $\boldsymbol{w}_{jk}, v_j \sim \mathcal{N}(0, 2/(d+1)), c_j \sim \mathcal{N}(0, 2/m), b_j = 0$. Now, this initialization causes an activation variance of $\frac{2}{d+1}||\boldsymbol{x}||^2$ for $\boldsymbol{w}_j^\top \boldsymbol{x}$, which is ≈ 2 if standardized $||x||^2 \to d$. In contrast, the variance of $v_j\tau$, with $\tau \sim U(0,1)$, is determined by $\mathbb{E}[v_j^2]\mathbb{E}[\tau^2] = \frac{2}{3(d+1)}$. Thus, on initialization, the contribution of τ to the activation is about $3(d+1)$ smaller. This causes $\hat{y}$ to vary much less in τ than in $\boldsymbol{x}$. f is likely affine in τ at initialization, as discontinuities in the ReLU activation functions occur only as long as $v_j\tau$ for $\tau \in [0,1]$ cause a switch in sign across $\boldsymbol{w}_j^\top \boldsymbol{x} + b_j + v_j\tau$. This is unlikely, given the variance ratio favouring the impact of $\boldsymbol{w}_j^\top \boldsymbol{x}$ on the activation. Any nonlinearities on top of this are likely unstructured at initialization. Expressive curvature in τ would require a relatively high $|v_j|$, which is only reachable through appropriate gradient updates.

Loss derivative Figure 2 shows the expected partial derivative of the loss for QR and SQR to behave relatively similarly, albeit noisier in initial states for QR compared to SQR. This is explained by QR having a unique initialization for each quantile level, while SQR shares the initialization. What is also visible in the figure of the derivative is that SQR seems to not reach an optimized state in its tails.

We can examine the partial derivative of equation 1 where,

$$\frac{\partial L_\tau}{\partial \hat{y}} = \mathbf{1}\{y < \hat{y}\} - \tau = \begin{cases} -\tau & y > \hat{y} \\ 1 - \tau & y < \hat{y}. \end{cases} \tag{4}$$

The magnitude of $\frac{\partial L_\tau}{\partial \hat{y}}$ depends only on τ. Further, $\frac{\partial^2 L_\tau}{\partial \tau \partial \hat{y}} = -1$, so τ in $\frac{\partial L_\tau}{\partial \hat{y}}$ changes uniformly across errors. Still, both QR and SQR optimize the expected pinball loss. We can set $\mathbb{E}[\mathbf{1}(y < \hat{y}) - \tau] = F(\hat{y}) - \tau = 0$ which results in $\hat{y}^* = F^{-1}(\tau)$, the conditional quantile function. $F^{-1}(\tau)$ is non-linear for any distribution apart from the uniform distribution. However, we argue that the problem for SQR lies in the derivative of a sample. In a single sample, $\frac{\partial L_\tau}{\partial \hat{y}}$ across τ depends on $\hat{y}$. At initialization, SQR is affine in τ. QR does not have a $\hat{y}$ that is constrained to be affine in τ as QR has different weight activations for each τ. SQR uses shared weights for its prediction, thus has to accumulate non-linearity through the expectation of the derivative. Exactly this accumulation is biased towards areas of high data density.

Impact of data-sample density on learning Expected pinball loss weighs regions with higher data density more heavily compared to regions with lower density (Narayan et al., 2023) (also see Appendix A). Intuitively, SQR's gradient is pushed equally by every data sample missed or covered in the distribution of the data. The data distribution is likely to be approximately normal, and most samples are in the bulk of the distribution. As such, the gradient is influenced mainly by this region. This causes the smoothing effect in τ documented by Narayan et al. (2023). If we again look at Figure 2, we can observe that, even for QR, $\hat{y}$ approximately linear in τ in the section close to the median. In SQR, this section is the one that mainly drive the gradient update, and may cause smoothing that extends linearly into the tails.

Looking at the problem in a simpler way, the change in τ in either the loss or the activations is too linear to evoke a nonlinear mapping. Thus it seems appropriate to introduce a way for τ to be scaled non-linearly in the network.

3 Solution approaches

Scaling τ seems to evoke a nonlinear mapping for it in f. In our empirical testing, methods that achieved this showed coverage performance comparable to QR. We identify two approaches from existing literature that achieve such a nonlinear mapping. Further, we simplify this idea to propose our own solutions.

IQN (Dabney et al., 2018) introduces a quantile embedding step using a cosine function. Given f_b as a MLP backbone and f_h as a MLP head, with $f = f_h(f_b())$,

$$\boldsymbol{y}_\tau = f_h(f_b(\boldsymbol{x}) \odot \phi(\tau)), \tag{5}$$

where $\phi(\tau) = \sigma(\boldsymbol{w}^\top cos(\pi k \tau) + \boldsymbol{b})$, and k is a set $[0, \ldots, K-1]$, an embedding dimension in which the default was $K = 64$ for IQN. So the weights $\boldsymbol{w} \in \mathbb{R}^{64 \times h}$, h denoting the size of f_h.

AQR (Smyl et al., 2026a) implements another quantile embedding approach in its MLP blocks. With l_1, l_2 being linear layers matching the output size of $f_b(\boldsymbol{x})$,

$$\boldsymbol{y}_\tau = f_h(f_b(\boldsymbol{x}) \odot (1 + l_1(\tau)) + l_2(\tau)). \tag{6}$$

Further, the $U(0,1)$ sampler was replaced with $\beta(0.3, 0.3)$, similar to (Narayan et al., 2023). Adding 1 to the multiplicative component causes it to be initialized around 1, preventing the injection from diminishing the activation signal from $f_b(\boldsymbol{x})$.

IQN features a circular expansion of τ combined with an element-wise multiplication. AQR, in comparison, features embeddings of τ acting as parameters of a linear regression on $f_b(\boldsymbol{x})$ and the β sampler. Noteworthy here is that AQR is explicitly not a non-linear rescaling of τ, but still produces coverage comparable to QR in our testing. So, it seems not a necessary condition for τ to be scaled by a non-linear function to cause high enough expressiveness in f.

3.1 Our quantile scalers

In both previous approaches, we embed τ with a type of neural network layer and let it interact within the network. Another way to provide a non-linear mapping of τ in f is to scale τ non-

linearly before we compute f. Intuitively, τ should also benefit from a distribution that is close to its ideal shape in this case, provided by the inverse CDF of the normal distribution Φ^{-1}, or the logit function $ln\frac{\tau}{1-\tau}$ (also see Appendix C). So instead of a $f(\boldsymbol{x}, \tau)$, we simply train $f(\boldsymbol{x}, \Phi^{-1}(\tau))$, or $f(\boldsymbol{x}, ln\frac{\tau}{1-\tau})$ to scale τ in the activations of f.

Similar to IQN and AQR, we can also embed a scaled τ. Let s be a linear layer acting on $\boldsymbol{x}$, and $\zeta()$ the softplus activation function, then

$$f(\boldsymbol{x}, \tau) = f_h(f_b(\boldsymbol{x}), \tau, \zeta(s(\boldsymbol{x})) * \Phi^{-1}(\tau)) \tag{7}$$

is another approach that nonlinearly embeds τ in a network. In the case of the pure scaling approach, we would only use as many parameters as standard SQR. Both approaches add fewer parameters to SQR compared to IQN or AQR (at width 64, that is 129 on top of SQR for our embedding approach, 192 more for AQR, and 4096 more for IQN). So, if performance can match the other approaches and QR, it would represent a more parsimonious way to close the gap with regular SQR.

3.2 Conformalized Quantile Regression

Instead of architecturally closing the gap in coverage between QR and SQR, one may also argue that post-processing of coverage may be sufficient. While QR approaches generally are able to show accurate calibration on a wide variety of tasks, they do not impose guarantees on the coverage of their prediction interval. Conformalized Quantile Regression (CQR) (Romano et al., 2019) post-processes QR intervals such that they are guaranteed to meet the expected coverage in finite samples. Details regarding CQR are found in Appendix B. In our experiments, we evaluate it as a post-processing step to test the merit of our scaling approaches against an approach that guarantees coverage.

4 Experiments

The following section lists the evaluation of the described solution approaches. Our goal is to find an adaptation of SQR that enables it to match the coverage performance of QR. As such, we use QR as a baseline and measure coverage and other probabilistic metrics. Importantly, we are aiming to match QR's coverage and evaluate measures such as sharpness after coverage is matched. We present the metrics used to evaluate this goal in Appendix E.

We refer to Appendix G for detailed information regarding the model setup, training procedure, and hyperparameter optimization. To empirically test adaptations for SQR, we use seven distinct regression datasets and retrain with five different seeds. Information on the datasets, statistics, and preprocessing can be found in Appendix F. For brevity, tables in this section average results across datasets. We leave results per dataset in Appendix K.

4.1 General approaches

Because both QR and SQR support a wide variety of potential MLP set-ups, we evaluated other set-ups to test potential alternative explanations. However, in our observations, only τ-scaling equalized SQR coverage with QR. Table 2 in Appendix D introduces and lists the approaches worth mentioning. We aim to rule out other causes for collapsing tails, such as missing capacity in the network, or to test other methods from existing literature.

The standardized results of these approaches averaged over all seven datasets are shown in the first half of Table 1. Even in the summary, it is visible that none of the employed approaches is able to approach QR's performance in calibration (ACE), while many are clearly sharper and display similar Pinball. This mirrors the relationship of regular SQR to QR. Winkler score, weighting larger intervals higher in the evaluation, also shows QR to perform overall better. An improvement in coverage is shown by the constant $\tau-$scaling, likely reinforcing the representation of τ in the output. Otherwise, β samplers and reweighting of loss also show improvements over baseline SQR, indicating that extreme τ loss impact in the optimization is at least partially responsible for tail collapse. Approaches targeting the capacity of the network underlying SQR do not show any results, reinforcing the point that the optimization process itself fails.

Table 1: The average and averaged standard deviation (in parentheses) of standardized results across all datasets. In turn, the results per dataset consist of the average and standard deviation of five seeded training runs. This table is split by dividing lines into baselines, general solutions, and scaling solutions. All metrics except PICP_{95} are better when lower. PICP_{95} is optimized at 0.95. Best solutions are in bold. Lower sharpness is only considered best when ACE is within standard deviation of QR. For general solutions, $U(0, 1)$ was used as the SQR sampler, unless otherwise specified. In the scaling solutions, SQR_s are the pure scaling approaches, and SQR_e are the embedded τ-scalers. SQR scalers and AQR all draw τ from a β-distribution.

Approach	ACE	$\text{Winkler}_{\text{avg}}$	Sharpness	PICP_{95}	Pinball
QR	0.021 (0.010)	2.220 (0.143)	2.238 (0.086)	0.934 (0.012)	**0.081 (0.003)**
SQR	0.066 (0.012)	2.660 (0.162)	1.580 (0.078)	0.866 (0.017)	0.084 (0.003)
SQR_{head}	0.071 (0.013)	2.854 (0.218)	1.493 (0.078)	0.856 (0.020)	0.086 (0.004)
SQR $\beta(0.5, 0.5)$	0.046 (0.007)	2.477 (0.152)	1.787 (0.073)	0.895 (0.015)	0.084 (0.003)
SQR_{wide}	0.077 (0.011)	2.815 (0.188)	1.508 (0.073)	0.845 (0.017)	0.086 (0.003)
SQR (8 layers)	0.065 (0.013)	2.640 (0.183)	1.586 (0.084)	0.870 (0.014)	0.084 (0.003)
SQR (loss reweight)	0.046 (0.007)	2.482 (0.167)	1.754 (0.096)	0.894 (0.014)	0.084 (0.004)
SQR cali. loss	0.085 (0.017)	2.900 (0.214)	1.574 (0.051)	0.840 (0.016)	0.088 (0.004)
SQR lr 1e-2, 5 epochs	0.065 (0.010)	2.711 (0.205)	1.564 (0.057)	0.861 (0.015)	0.085 (0.004)
SQR lr 1e-1, 1 epoch	0.107 (0.031)	3.271 (0.483)	1.476 (0.085)	0.807 (0.036)	0.094 (0.007)
SQR (SiLU)	0.056 (0.011)	2.599 (0.157)	1.643 (0.074)	0.876 (0.016)	0.084 (0.003)
SQR (constant τ-scale)	0.036 (0.008)	2.414 (0.144)	1.831 (0.090)	0.908 (0.013)	0.083 (0.003)
IQN	0.022 (0.009)	**2.205 (0.123)**	2.448 (0.100)	0.941 (0.012)	**0.081 (0.003)**
AQR	0.023 (0.006)	2.258 (0.118)	**2.223 (0.118)**	**0.943 (0.014)**	0.082 (0.003)
SQR_e (Φ^{-1})	0.018 (0.012)	2.210 (0.128)	2.252 (0.111)	0.940 (0.011)	**0.081 (0.003)**
SQR_e (logit)	**0.015 (0.009)**	**2.205 (0.117)**	2.380 (0.102)	**0.943 (0.010)**	**0.081 (0.003)**
SQR_s (Φ^{-1})	0.020 (0.010)	2.224 (0.125)	2.312 (0.083)	0.936 (0.009)	**0.081 (0.003)**
SQR_s (logit)	0.019 (0.010)	2.209 (0.127)	2.412 (0.102)	0.942 (0.010)	0.082 (0.003)

4.2 Quantile Scaling Approaches

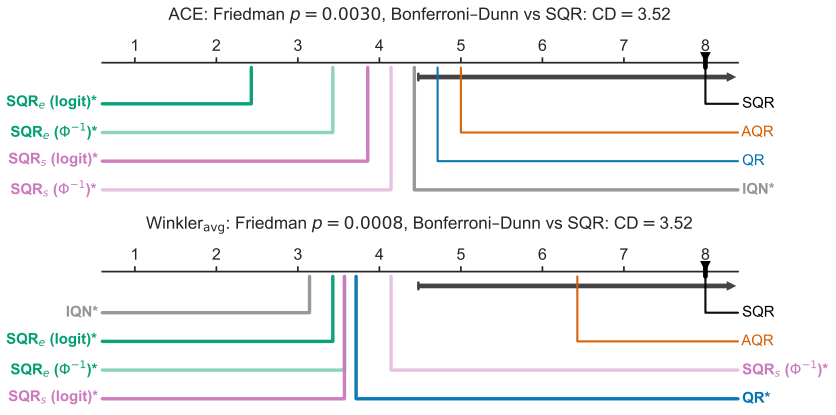


Figure 3: Critical distance diagram with a Bonferroni-Dunn test comparing all 8 approaches against SQR, over 7 datasets with $\alpha = 0.05$. Methods are ordered by their average rank over the datasets. The grey bar spans the Critical Distance (CD) from SQR, with approaches outside differing significantly from SQR (marked by $*$). CD ranges over the axis and was truncated with an arrow in the visualization.

We show a summary of the evaluation results for approaches with dynamic scaling of τ across datasets in the second half of Table 1. Our SQR scaling approaches are referred to as the τ-scaling approach SQR_s and the τ-embedding approach SQR_e, as explained in Section 3.1. It should be noted that the results of SQR_s and SQR_e were obtained with sampling $\tau \sim \beta(0.5, 0.5)$. We obtained similar results for $U(0, 1)$, which are reported in Appendix K for brevity. All approaches seem to

successfully close the coverage gap towards QR, while matching performance in Winkler score and pinball. To build on the problem shown in Figure 2, we show how the tested approaches solve the training in Figure 6 (Appendix I). There are trends between approaches visible in these results, but they fall within plausible sampling variation due to individual performance differences across datasets. We offer thoughts on the scaling approaches' individual differences for each dataset in Appendix I as well.

Independent of individual differences in magnitude, ranking provides a way to differentiate performance of approaches across datasets, as the unexplained variance of their targets may differ. To statistically test the differences between baseline and τ-scaling approaches, we ranked the average performance across seeds (in ACE and average Winkler score) of each approach per dataset and compared this ranking with SQR's ranking in a Bonferroni-Dunn test as a post-hoc to a significant Friedman test (Demšar, 2006). The result of this test is shown in a critical distance diagram in Figure 3. The null hypothesis to be tested here is that each approach has the same average rank across datasets as SQR. The area of this statistical indistinguishability is marked by the grey bar for the critical distance around SQR. We can see that most of the approaches are significantly different from SQR. Markedly, AQR in both and QR in ACE are not significantly different from SQR. This is the case as the given test only evaluates differences in ranking between datasets, not differences in overall metric magnitude (QR's ACE is smaller than SQR's by 4 standard deviations reported in Table 1). SQR ranked last in each dataset, and the other τ-scaling approaches showed better performance than AQR and QR across datasets. Seen together with the observed metrics in Table 1, there is a performance difference relative to SQR, but it does not carry over in the ranking due to the small sample size of seven datasets. For the same reason, the ranking between approaches should be taken as suggestive as well. Overall, the ranking shows that the gap is consistent and not confounded by a subset of the datasets, as SQR is last in all of them.

We also apply CQR post-calibration correction on all main approaches. The full results can be examined in Table 6 in Appendix H. The coverage of all approaches was successfully adapted to an almost perfect (because the calibration datasets are not a perfect representation of the test sets) score. Visible trends are differences in Winkler score and overall Sharpness, both metrics that penalize large tails. Pure SQR has the worst Winkler score and Sharpness (where it ties with IQN but shows higher variability), indicating that its distribution may now be too large.

5 Conclusion

In this paper, we show how SQR undercovers the tails of the conditional distribution compared to QR, leading to a coverage performance gap between the two methods. We hypothesise that the optimization process for SQR favors an affine and smooth function in τ and show examples of this effect. To close the performance gap, we propose to scale or additionally embed τ to increase the expressiveness of f in τ. Comparing τ-scaling to QR, we aim to achieve similar or better coverage with similar or non-increased sharpness. The evaluation of our own approaches, in addition to IQN and AQR, agrees with this hypothesis by matching the coverage to QR, even though only pure SQR and QR were hyperparameter optimized. SQR_s also saves on the number of parameters added, by only using as many as regular SQR.

We further evaluated CQR as a post-processor after applying τ-scaling approaches. We show that a better initial fit also improves post-processing results. However, approaches with already good ACE gained little more coverage but large increases in interval width in some datasets, so applying CQR on already well-calibrated models may not always be beneficial. CQR also requires a calibration set that should be representative of the rest of the data, decreasing the size of training or validation partitions. We share more of our thoughts regarding the discussion of the results in Appendix J.

Overall, we find all listed τ-scaling approaches to lift SQR up to the coverage level of QR and recommend their usage in any modern quantile regression task. Given our findings and additions to SQR, we are confident that in terms of coverage, SQR can now serve as a powerful replacement for QR, extending a discrete quantile mapping to a learned inverse CDF.

AI USE STATEMENT

In this work, we used generative AI tools to help develop the conceptual framework, to provide feedback on research methodology, to propose and refine hypotheses, to design and give feedback on experiments, to implement methods with code, to create code that cleans datasets, to help formulate mathematical claims, to write code that generated a parametric synthetic data generator for preliminary testing and as a discussion partner regarding the interpretation of results. The providing of critical ingredients for proving mathematical claims, assisting in the writing of proofs, assisting with translations, or supporting qualitative and thematic data analysis do not apply to this work. Additionally, we used generative AI tools to create or edit software code, to generate code for figures, suggest experimental parameters, to help in producing the reproduction artifact, to summarize and identify relevant literature, help with brainstorming, and to help with sourcing/searching for information. We further used generative AI to supply readability suggestions and to spot spelling and grammar errors.

We have reviewed all AI-assisted work. Software method implementations were always generated using reference repositories from the literature. No AI model was ever run unsupervised, and the code was verified and tested by the authors afterwards. Other help (e.g., figures, data cleaning) was kept reproducible by only letting an AI model generate the code for the figure and not the figure itself. All results of the paper were produced by executing code and never taken from a generative model output. All AI suggestions and claims in methodology and theory discussions were reviewed and validated by the authors with their sources before they made it into the paper. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

In order to facilitate reproducibility of this work, we include all experimental details in the Appendix. This includes metrics (in Appendix E), datasets (in Appendix F), model architecture, setup, and optimization procedure (i.e., training and hyperparameter optimization) (all in Appendix G). This also includes information regarding the Python packages used for the code and the physical specs of the PC used. The datasets are all freely available, and the dataset description includes references to each dataset. Details regarding architectural differences in the τ-scaling solution approaches should be made clear based on the descriptions in section 3. Code that can be used to reproduce the results of the paper is further made available as supplementary material with an anonymous link here: `https://zenodo.org/records/22814547`

REFERENCES

Haleh Akrami, Anand A. Joshi, Sergül Aydöre, and Richard M. Leahy. Deep Quantile Regression for Uncertainty Estimation in Unsupervised and Supervised Lesion Detection. *The journal of machine learning for biomedical imaging*, 1:008, 2022. ISSN 2766-905X.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami, Shubham Kapoor, Danielle C. Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From Univariate to Universal Forecasting, October 2025.

Herman J Bierens and Donna K Ginther. Integrated conditional moment testing of quantile regression models. *Empirical Economics*, 26(1):307–324, 2001.

Youngseog Chung, Willie Neiswanger, Ian Char, and Jeff Schneider. Beyond Pinball Loss: Quantile Methods for Calibrated Uncertainty Quantification. In *Advances in Neural Information Processing Systems*, volume 34, pp. 10971–10984. Curran Associates, Inc., 2021.

Will Dabney, Georg Ostrovski, David Silver, and Remi Munos. Implicit Quantile Networks for Distributional Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1096–1105. PMLR, July 2018.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting, April 2024.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

Sivert Eggen, Tord Johan Espe, Kristoffer Grude, Morten Risstad, and Rickard Sandberg. Financial time series uncertainty: A review of probabilistic ai applications. *Journal of Economic Surveys*, 40(2):915–953, 2026.

Niklas Erdmann, Lars Bentsen, Roy Stenbro, Heine Nygard Riise, Narada Dilp Warakagoda, and Paal E. Engelstad. Multi-Horizon Time Series Forecasting of Non-Parametric CDFs with Deep Lattice Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(1):255–264, March 2026. ISSN 2374-3468. doi: 10.1609/aaai.v40i1.36986.

Sebastian Felix Fischer, Matthias Feurer, and Bernd Bischl. Openml-ctr23–a curated tabular regression benchmarking suite. In *AutoML Conference 2023 (Workshop)*, 2023.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437.

Tilmann Gneiting and Roopesh Ranjan. Comparing density forecasts using threshold- and quantile-weighted scoring rules. *Journal of Business & Economic Statistics*, 29(3):411–422, 2011. doi: 10.1198/jbes.2010.08110.

Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E. Raftery. Probabilistic Forecasts, Calibration and Sharpness. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 69(2): 243–268, April 2007. ISSN 1369-7412. doi: 10.1111/j.1467-9868.2007.00587.x.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3): 90–95, 2007. doi: 10.1109/MCSE.2007.55.

Kaggle. House sales in king county, usa, 2016. URL `https://www.kaggle.com/datasets/harlfoxem/`.

Roger Koenker and Gilbert Bassett. Regression Quantiles. *Econometrica*, 46(1):33–50, 1978. ISSN 0012-9682. doi: 10.2307/1913643.

Manuel Lopez-Martin, Antonio Sanchez-Esguevillas, Luis Hernandez-Callejo, Juan Ignacio Arribas, and Belen Carro. Additive Ensemble Neural Network with Constrained Weighted Quantile Loss for Probabilistic Electric-Load Forecasting. *Sensors*, 21(9):2979, January 2021. ISSN 1424-8220. doi: 10.3390/s21092979.

Taman Narayan, Serena Lutong Wang, Kevin Robert Canini, and Maya Gupta. Expected Pinball Loss For Quantile Regression And Inverse CDF Estimation. *Transactions on Machine Learning Research*, May 2023. ISSN 2835-8856.

Craig A Olson. A comparison of parametric and semiparametric estimates of the effect of spousal health insurance coverage on weekly hours worked by wives. *Journal of Applied Econometrics*, 13(5):543–565, 1998.

R. Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997a.

R. Kelley Pace and Ronald Barry. Quick computation of spatial autoregressive estimators. *Geographical analysis*, 29(3):232–247, 1997b.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703, 2019. URL http://arxiv.org/abs/1912.01703.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R. Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, Remi Lam, and Matthew Willson. Probabilistic weather forecasting with machine learning. *Nature*, 637(8044): 84–90, January 2025. ISSN 1476-4687. doi: 10.1038/s41586-024-08252-9.

Prashant Rana. Physicochemical Properties of Protein Tertiary Structure. UCI Machine Learning Repository, 2013. DOI: https://doi.org/10.24432/C5QW3H.

Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized Quantile Regression. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Slawek Smyl, Boris N. Oreshkin, Paweł Pełka, and Grzegorz Dudek. Any-quantile probabilistic forecasting of short-term electricity demand: Fusing uncertainties from diverse sources. *Information Fusion*, 127:103637, March 2026a. ISSN 1566-2535. doi: 10.1016/j.inffus.2025.103637.

Slawek Smyl, Paweł Pełka, and Grzegorz Dudek. Probabilistic Multi-Regional Solar Power Forecasting with Any-Quantile Recurrent Neural Networks, February 2026b.

Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, volume 32, Red Hook, NY, USA, December 2019. Curran Associates Inc.

Ichiro Takeuchi, Quoc V Le, Quoc Le, Timothy D Sears, Tim Sears, Alexander J Smola, and Alex Smola. Nonparametric Quantile Estimation. *JMLR*, 7:1231–1264, 2006.

The pandas development team. pandas-dev/pandas: Pandas, February 2020. URL https://doi.org/10.5281/zenodo.3509134.

Jasper Velthoen, Clément Dombry, Juan-Juan Cai, and Sebastian Engelke. Gradient boosting for extreme quantile regression. *Extremes*, 26(4):639–667, December 2023. ISSN 1572-915X. doi: 10.1007/s10687-023-00473-x.

L. R. Visser, T. A. AlSkaif, A. Khurram, J. Kleissl, and W. G. H. J. M. van Sark. Probabilistic solar power forecasting: An economic and technical evaluation of an optimal market bidding strategy. *Applied Energy*, 370:123573, September 2024. ISSN 0306-2619. doi: 10.1016/j.apenergy.2024.123573.

Vladimir Vovk, Alexander Gammerman, and Craig Saunders. Machine-learning applications of algorithmic randomness. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 444–453. Morgan Kaufmann, 1999.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer, 2005.

Tracy Sellers Warwick Nash. Abalone, 1994. URL https://archive.ics.uci.edu/dataset/1.

Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6 (60):3021, 2021. doi: 10.21105/joss.03021.

Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman (eds.), *Proceedings of the 9th Python in Science Conference*, pp. 56 – 61, 2010. doi: 10.25080/Majora-92bf1922-00a.

Robert L. Winkler. A decision-theoretic approach to interval estimation. *Journal of the American Statistical Association*, 67(337):187–191, 1972. doi: 10.1080/01621459.1972.10481224.

APPENDIX

A EXTENDED EXPLANATION FOR COLLAPSING TAILS

The impact of data-sample density In Narayan et al. (2023), the authors show how expected pinball loss smooths sample quantile outputs when there are no samples available for a respective quantile. The cost of pinball loss varies per sample point and has a steepness depending on how many neighboring sample points there are (compared to other such functions where it would be the distance to the target). In other words, the cost of deviating from a point with many sample points in the neighborhood has a large steepness, while deviations in the tails carry a comparatively small steepness. Thus, it is easier to learn in areas with high sample density, which are usually the body or centre of the distribution. It also means that f will be more flexible in areas with more data points and smoother in areas of fewer data points.

To arrive at this conclusion, we can follow the description by Narayan et al. (2023): If we optimize $f(\boldsymbol{x}, \tau)$ for τ alone with in terms of $\boldsymbol{y}$, is equivalent to a single-variable unconditional regressor $f(\tau)$ with data points $\{\tau(i), y_i\}_{i=1}^n$, where y_i are ordered data samples where i is the ith smallest sample, and $\tau(i) = \frac{i-1}{n-1}$. Now we assume that we start at $f(\tau(i)) = y_i$. We choose an error ϵ such that $f(\tau(i)) - \epsilon > y_{i-1}$. According to the definition of our dataset, y_i has $\tau(i)(n-1)$ data points below it, and $(1-\tau(i))(n-1)$ data points above it. Decreasing $f(\tau(i))$ by ϵ causes us to have a loss $\tau(i)(1-\tau(i))(n-1)\epsilon$ for the points above y_i and recover $(1-\tau(i))\tau(i)(n-1)\epsilon$. While these are equal and cancel out, we lose $\tau(i)\epsilon$ for moving away from y_i (as per the definition of pinball). The slope between y_{i-1} and y_i is then $-\tau(i)$. But for each data point we cross, we lose $\tau(i)\epsilon$ instead of gaining $(1-\tau(i))\epsilon$, thus increasing the steepness of our slope by 1. In other words, the optimization incentive for a deviation of the predicted value from the true value is $-\tau(i)$ minus the number of data samples between y_i and $f(\tau(i))$ when $y_i > f(\tau(i))$. Here, the positive case is the inversion.

B CQR METHOD DETAILS

Conformal prediction (Vovk et al., 1999; 2005) can be used to construct prediction intervals from an arbitrary regression model that are guaranteed to meet the expected coverage in finite samples. Conformalized Quantile Regression (CQR) (Romano et al., 2019) extends this idea and applies it to QR as a post-processing of the prediction intervals with reportedly better coverage performance while keeping intervals small. The method here is to take an additional split of the data, such that $D = \mathcal{I}_1, \mathcal{I}_2$ after training a QR model, and compute an error E for each interval of paired quantiles $[\tau, 1-\tau]$ like so,

$$E_i = max(f(\boldsymbol{x}, \tau_i) - \boldsymbol{y}, \boldsymbol{y} - f(\boldsymbol{x}, 1-\tau_i)) \tag{8}$$

Here we assume that we only need to iterate over $T = \{\tau \in \boldsymbol{t} : \tau < 0.5\}$ as the corresponding quantiles are covered with $1-\tau$. The prediction intervals given by $C(\tau_i) = [f(\boldsymbol{x}, \tau_i), f(\boldsymbol{x}, 1-\tau_i)]$ can then be conformalized in a post-processing step:

$$C(\tau_i) = [f(\boldsymbol{x}, \tau_i) - Q_{1-\alpha}(E, \mathcal{I}_2), f(\boldsymbol{x}, 1-\tau_i) + Q_{1-\alpha}(E, \mathcal{I}_2)] \tag{9}$$

Here, $Q_{1-\alpha}(E, \mathcal{I}_2)$ is the $(1-\alpha)(1+1/|\mathcal{I}_2|)$-th empirical quantile of the ordered set of $\{E_i : i \in \mathcal{I}_2\}$ where $\alpha = 2\tau_i$ is the nominal miscoverage of the interval being adjusted. So for each interval, we have a corresponding set of Es in which we can find the right correction. As CQR just needs an arbitrary set of quantiles to be a valid post-processing step, it can be applied to any SQR adaptation, only requiring an additional hold-out partition of the dataset. The overall adaptation still depends on the predicted quantile values of a given QR model, so we hypothesize that better models support better post-processing results. Overall, given that the stated goal of this work is more accurate

calibration for SQR, this represents an excellent post-processing step, applicable to all best solution approaches.

Notably, this is the symmetric version of CQR, targeting both sides of the tail equally. Romano et al. (2019) also detail an asymmetric version in their work. However, this is reported to produce wider prediction intervals and needs a larger calibration dataset $\mathcal{I}_2$. While it would still make sense to use this version for a very heavy-tailed dataset, the datasets employed in this study are all relatively symmetric in variance. This also points towards the downsides of CQR: A priori knowledge of the data distribution may be required. Additionally, we need to use an additional partition on the dataset, leading to fewer samples for training, validation, or testing.

C Notes on chosen scaling functions

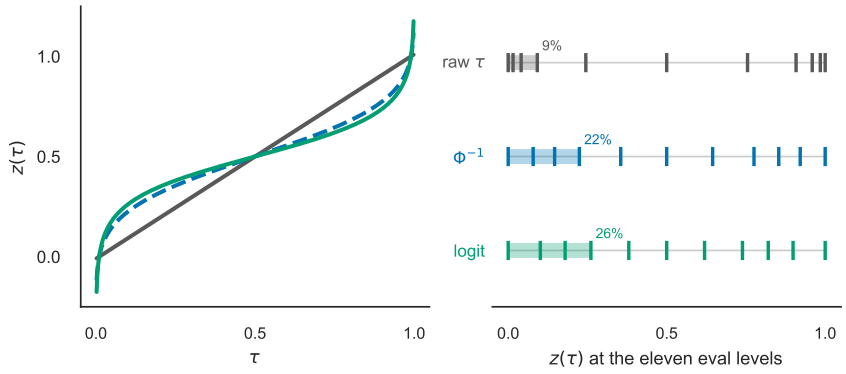


Figure 4: Scaling impact of Φ^{-1} and logit on τ. The general scaling function is represented by $z(\tau)$.

Based on our proposal for a more expressive network f in τ, it seems natural to choose functions that scale τ in a non-linear way, especially close to 0 or 1. As can be seen in Figure 4, this is exactly what these functions do. In the right part of the figure, we can see the percentage of shared spread of the overall interval from 0 to 1 that the first 3 quantiles $[0.01, 0.025, 0.05]$ occupy. Extending this observation to interval size means that around half of the variability of τ is dedicated to the tail intervals. This presents a different way of thinking about how to solve the problem: many more samples of τ are in the tail intervals, thus biasing training in that direction. The reason why this still performs better than just a sampler is likely that not only does f get more examples, but examples are numerically more spread out, making them more distinct from each other within the training process of the network.

D General Approaches

The general approaches shown in Table 2 share other possible solutions to the predictive interval coverage gap between QR and SQR and briefly describe the reasoning behind testing them.

E Evaluation metrics

In our evaluation, we make a distinction between metrics that are marginally affected by coverage error and metrics that only measure coverage error. Our main aim is to identify methods that show high coverage while, secondarily, maintaining sharpness and fit. Our chosen coverage metrics are ACE and $PICP$, which are always examined first. If these are close to QR values, we assume a method covers the data well. Then, we examine other metrics and argue about further performance differences.

Metrics are evaluated over the test partition, and a set of 11 quantiles, $T = [0.01, 0.025, 0.05, 0.10, 0.25, 0.50, 0.75, 0.90, 0.95, 0.975, 0.99]$. The exception to this is $PICP$, which we only evaluate for tail intervals, and Sharpness, which is the interval width across the

Table 2: Compendium of other proposed solutions to widen SQR tails. These only lead to at most partial closing of the coverage gap between QR and SQR.

Approach	Description
Beta sampling $\tau \sim \beta(a, a)$	If tails are not widening, more examples of the extreme values are needed to help them widen.
Reweighting loss with $\beta(a, a)$ density at τ	Data sampling with $\beta(a, a)$ may either influence exposure or loss weight by extension. This explicitly tests if changes in loss weight may help with wider tails.
More parameter width or depth for SQR	SQR may be parameter-bottlenecked compared to QR, which can be ruled out by empirical testing of deeper models, or models with the same amount of parameters as QR (SQR_{wide} in the results).
Calibration and sharpness loss	The custom loss introduced in Chung et al. (2021) may help with SQR calibration by making calibration error explicit for the network.
Warm-up learning rate	Increasing the learning rate in the first epochs reportedly widens the tails. (Erdmann et al., 2026)
Injecting τ at the head or backbone of the network	The injection point may improve the representation of τ.
Smooth activation with SiLU	The standard MLP in Tagasovska & Lopez-Paz (2019) uses ReLU. A smooth activation does not improve the structuring of τ.
Constant τ-scaling	Constant τ-scaling (Tagasovska & Lopez-Paz, 2019) widens the tail, but does not add curvature.

smallest and largest quantile. Additionally, we also average metrics (and take their standard deviation) across five trained models with the same hyperparameters, but different seeds.

Pinball Adapting equation 1, we can average pinball across quantiles and sample size to arrive at a scoring rule:

$$\text{PB} \;=\; \frac{1}{NT}\sum_{i=1}^{N}\sum_{j=1}^{T} L_\tau(y_i, \hat{y}_{ij})\,. \tag{10}$$

Notably, pinball is equivalent to mean absolute error for $\tau = 0.5$ and approximates CRPS (Gneiting & Ranjan, 2011):

$$CRPS = \int_{-\infty}^{\infty} (F(u) - \mathbf{1}(y \leq u))^2 du = 2\int_0^1 L_\tau(y, f(\tau))d\tau, \tag{11}$$

where $u \in \mathbb{R}$ is a threshold on the outcome space, and F the predicted CDF (of which f is the inverse). Equation 10 is a discrete average over the same integrand over T, albeit clustered around the tails more heavily than CRPS due to the distribution of the chosen set of quantiles. Pinball is potentially insensitive to coverage errors, displaying small values as for $|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_{ij}| \to 0$, $L_{\tau_j} \to 0$ irrespective of τ. Note that pinball can still estimate coverage performance when it is clear that a given model does not undercover the target.

Winkler interval score Winkler interval score (Winkler, 1972) is a technique that scores prediction interval width in addition to a penalty for observations outside of the interval weighted by $2/\alpha$. This is, in essence, a reweighted pinball loss, with higher stress on tail intervals. For a $(1-\alpha)$ prediction interval $\alpha = 2\tau$. By evaluating $f(\boldsymbol{x}, \tau)$, we can then list the predicted interval values of $\hat{y}_l, \hat{y}_u$, and evaluate the interval score of

$$W_\alpha \;=\; \begin{cases} (\hat{y}_u - \hat{y}_l) + \frac{2}{\alpha}(\hat{y}_l - y) & \text{if } y < \hat{y}_l \\ (\hat{y}_u - \hat{y}_l) & \text{if } \hat{y}_l \leq y \leq \hat{y}_u \\ (\hat{y}_u - \hat{y}_l) + \frac{2}{\alpha}(y - \hat{y}_u) & \text{if } y > \hat{y}_u. \end{cases} \tag{12}$$

We report the Winkler score as the average across all intervals A and the number of samples N, similar to equation 10, such that $W_{avg} \;=\; \frac{1}{NA}\sum_{i=1}^{N}\sum_{a=1}^{A} W_{ia}$. Similar to pinball, optimized

$W_{avg} \to 0$ when $|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_{ij}| \to 0$. If the evaluation is well calibrated, the Winkler score can show performance differences in wider intervals compared to pinball, due to reweighting according to the interval size.

PICP and ACE Prediction interval coverage percentage (PICP) counts the number of times the predicted interval has captured the observed data. A well-calibrated interval is indicated by $PICP_{1-\alpha} \to 1 - \alpha$.

$$PICP_{1-\alpha} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}(\hat{y}_{li} \leq y_i \leq \hat{y}_{ui}) \tag{13}$$

Comparing all interval coverage with the ideal interval coverage and averaging that deviation then results in average coverage error (ACE), being a direct summarization of all PICP levels in $\mathcal{A}$. ACE is optimized when $ACE \to 0$.

$$ACE = \frac{1}{|\mathcal{A}|} \sum_{\alpha \in \mathcal{A}} |(1 - \alpha) - PICP_{1-\alpha}| \tag{14}$$

Sharpness We define sharpness here as the width of the largest of the prediction intervals, averaged across all samples.

$$S \;=\; \frac{1}{N} \sum_{i=1}^{N} \Big(\max_{\tau} \hat{y}_{i\tau} - \min_{\tau} \hat{y}_{i\tau} \Big) \tag{15}$$

F DATASETS

In total, seven regression datasets are used for the empirical evaluation in this study. These are named Protein, Wages, Working hours, King County, California, Abalone, and Elections. Except for protein (Rana, 2013), we selected all datasets from a curated benchmark suite (Fischer et al., 2023). The main collection criterion for datasets was to choose real observational data with a large number of samples to rule out that SQR would simply need more examples to train. Each dataset was split into 90 % training and 10 % testing. Of the training partition, 20 % was held out and used for validation and calibration. Over the whole dataset, this is 72% training, 9 % validation, 9 % recalibration, and 10% testing. All categorical features were transformed with one-hot encoding. Further, all datasets are standardized with z-score normalization $x' = \frac{(x-\mu)}{\sigma}$, computed with observable statistics from the training dataset for all other splits as well. This standardization was kept for the results in order to compare across datasets. Lastly, we also discarded all duplicate rows in the datasets. Table 3 shows general information regarding the target, number of samples, features, and statistics of all datasets.

Table 3: General information and statistics of all seven datasets. The number of features is reported after one-hot encoding.

Dataset	Target	Samples	Features	Mean	Median	STD	Skew	Kurtosis
Abalone	shell rings	4177	10	9.93	9	3.22	1.11	2.33
California	median value (USD)	20640	8	206856	179700	115396	0.978	0.328
Elections	log votes per capita	3107	6	−0.576	−0.566	0.198	−1.02	8.41
King County	sale price (USD)	21613	132	540088	450000	367127	4.02	34.6
Protein	RMSD	44019	9	7.8	5.09	6.14	0.553	−1.16
Wages	wage (USD/week)	26027	12	600	515	461	6.63	167
Work Hours	work h/week	22265	25	25.6	35	18.7	−0.342	−1.35

In the following, we will briefly lay out each dataset:

Abalone The target here is predicting shell rings to determine the age of an abalone. As counting shell rings manually is time-consuming manual labor, eight more easily measured features are used as substitutes instead (Warwick Nash, 1994). These are sex, height, length, diameter, whole weight, shucked weight, viscera weight, and shell weight.

California This dataset was sampled for the 1990 U.S. census, predicting median house value for California districts in 100 thousand dollars. The eight features assisting the prediction are median income in the sampled block group, median house age in the block group, average number of rooms per household, average number of bedrooms per household, block group population, average number of household members, block group latitude, and block group longitude (Pace & Barry, 1997a).

Wages This was extracted from the 1988 March Current Population Survey. Hereby we want to predict the wages of a male sample of the survey given six features of their personal information. Specifically, their years of schooling, their years of potential work experience, their ethnicity, whether they are living in a standard metropolitan statistical area, in which region, and whether they are working part-time (Bierens & Ginther, 2001).

Work hours Referenced as health_insurance in the collection, reflecting the aim to relate health insurance to women's working hours in the original study (Olson, 1998). The target for this dataset is women's work hours, predicted with 10 other features: whether the woman is covered by her husband's health insurance, whether the woman has health insurance through her job, whether the husband has health insurance through her job, her years of education, ethnicity, and whether she is Hispanic separately, years of work experience, number of kids under age of 6, number of kids 6-18 years old, husband's income in thousands of dollars, and region of living.

King County Prices of sold houses between May 2014 and May 2015 in King County, USA. originally released on Kaggle (Kaggle, 2016). It includes 21 features (4 categorical, which cause the large one-hot expansion), namely the number of bedrooms, bathrooms, and floors, the square feet of the living area and the loft, the view, whether there is a waterfront, its condition, an ordinal grading of the property, square feet above and below ground, year built and renovated, the square foot area of living space and loft of the 15 nearest neighbors, location in latitude and longitude, zip code, the day, month, and year.

Elections Referenced as space_ga in the collection, originating from Pace & Barry (1997b). The data takes the votes from the 1980 presidential election and predicts the number of votes per population, given the population amount, education levels, number of owned houses, income, and spatial coordinates.

Protein Dataset from CASP 5-9, on the physicochemical properties of protein tertiary structure. The target is the residue-level RMSD of a protein tertiary structure (Rana, 2013). In total, it encompasses nine additional protein-specific features.

G Model setup and optimization

Generally, every model was set-up in a backbone and a head f_b, f_h as described in Section 3. The approaches shared a 2-hidden-layer MLP as backbone with ReLU activation. The prediction head was generally a 1-hidden-layer MLP with different embedding layers or output sizes depending on the approach. Otherwise, we used the Adam optimizer for all models, a batch size of 256, and 300 epochs of training. Instead of early stopping, we save the best validation results of each model trained during training, and the best epoch is used for evaluation. Training and hyperparameter optimization are done on the lowest Pinball (see equation 10).

Only the QR and pure SQR approaches were specifically hyperparameter optimized on every dataset in order to strengthen results and show their robustness in a non-optimized setting. We optimized the learning rate and hidden parameter count for both SQR and QR via a grid search across three seeds. The range of the learning rate was $[1e-4, 3e-4, 1e-3, 3e-3, 1e-2, 3e-2, 1e-1]$, while the hidden parameter size of f_b and f_h was swept over $[32, 64, 128]$ for each, with the option of no hidden layer for f_h. We show the optimized hyperparameters for QR and SQR per dataset in Tables 4 and 5 respectively.

All modifications to SQR used a standard configuration of width=64 for all hidden layers and a learning rate of 1×10^{-3}.

Dataset	lr	number of neurons in f_b	number of neurons in f_h
Abalone	3×10^{-3}	32	32
California	3×10^{-3}	128	128
Wages	3×10^{-4}	32	128
Working hours	1×10^{-3}	32	32
King county	1×10^{-3}	64	64
Protein	3×10^{-3}	128	128
Elections	3×10^{-2}	64	-

Table 4: Optimized QR hyperparameters. In elections, QR was optimized to not have a hidden head layer at all.

Dataset	lr	number of neurons in f_b	number of neurons in f_h
Abalone	1×10^{-2}	64	128
California	3×10^{-3}	128	64
Wages	1×10^{-3}	128	128
Working hours	3×10^{-3}	32	128
King county	1×10^{-3}	32	32
Protein	3×10^{-3}	128	64
Elections	1×10^{-2}	128	32

Table 5: Optimized SQR hyperparameters.

We ran all experiments on a laptop with an AMD Ryzen 7 7735HS CPU and 16 GB of RAM. All code was written in Python, using the PyTorch (Paszke et al., 2019), NumPy (Harris et al., 2020), and pandas (The pandas development team, 2020; Wes McKinney, 2010) packages. The plots displayed in this work with matplotlib (Hunter, 2007) and seaborn (Waskom, 2021).

H CQR RESULTS

In this section, we show the complete post-processed CQR results of all main (τ-scaling) approaches in Table 6. We can see that coverage metrics are almost unified at a lower bound, determined by the representativeness of the calibration partition compared to the test partition. Differences seem to be visible in Winkler score and sharpness, which both may penalize coverage that is too wide. In the overall results in Table 1, the best Winkler Score and Sharpness were similar in value, but here it is SQR_e (Φ^{-1}), while without post-processing, sharpness was best for AQR, and coverage best for SQR_e (logit). It seems then that other approaches' coverage adjustments resulted in harder penalties here. This may be related to the way the approaches are able to model the CDF, but perhaps also to the CDF that the datasets demand.

Table 6: Averages across the standardized metric results of all datasets and seeds for the τ-scaling approaches after recalibration with CQR. SQR_s are the pure scaling approaches, and SQR_e are the embedded τ-scalers. SQR scalers and AQR all draw τ from a β-distribution.

Approach	ACE	$\text{Winkler}_{\text{avg}}$	Sharpness	PICP_{95}	Pinball
QR	**0.010 (0.005)**	2.205 (0.131)	2.437 (0.130)	**0.951 (0.009)**	**0.081 (0.003)**
SQR	0.011 (0.003)	2.326 (0.133)	2.549 (0.157)	**0.949 (0.007)**	0.082 (0.003)
IQN	0.011 (0.004)	2.202 (0.112)	2.549 (0.101)	0.952 (0.008)	**0.081 (0.003)**
AQR	0.011 (0.004)	2.235 (0.107)	2.489 (0.084)	0.953 (0.009)	0.082 (0.003)
SQR_e (Φ^{-1})	**0.010 (0.004)**	**2.201 (0.118)**	**2.428 (0.126)**	0.953 (0.008)	**0.081 (0.003)**
SQR_e (logit)	0.011 (0.004)	2.204 (0.116)	2.458 (0.125)	**0.951 (0.010)**	**0.081 (0.003)**
SQR_s (Φ^{-1})	0.011 (0.004)	2.218 (0.119)	2.474 (0.125)	**0.951 (0.009)**	**0.081 (0.003)**
SQR_s (logit)	**0.010 (0.003)**	2.210 (0.121)	2.506 (0.116)	0.953 (0.009)	**0.081 (0.003)**

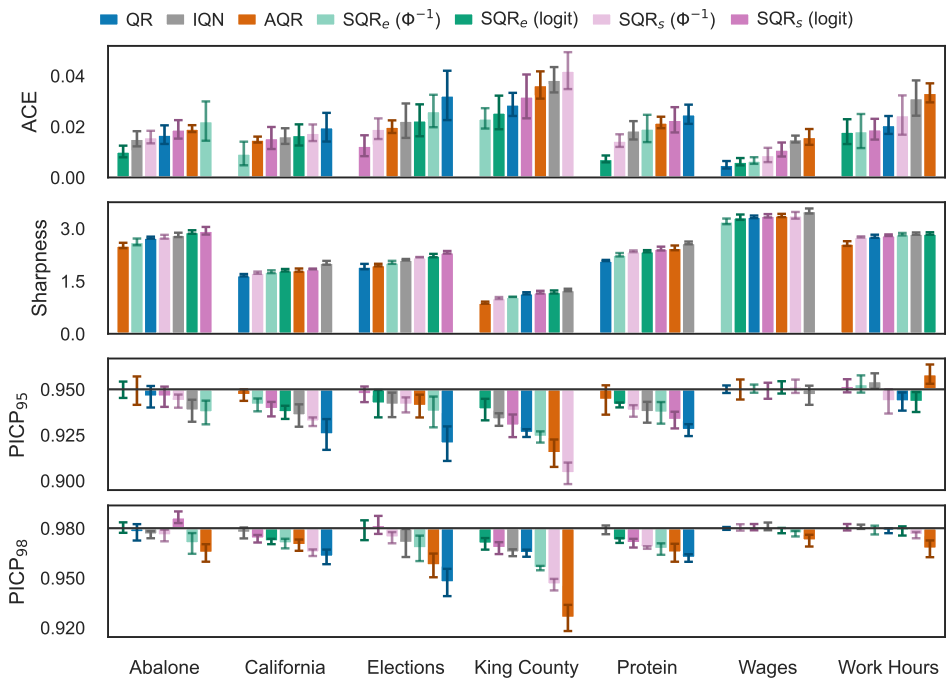


Figure 5: ACE, sharpness, PICP$_{95}$ and PICP$_{98}$ for the best-performing τ-scaling approaches and respective approaches from the literature for each dataset compared to QR. Error bars denote standard error.

I Comparing scaling approaches

There is observable performance variability across datasets. However, most observed differences between scaling approaches are trends, as the differences themselves are often too small compared to variances within models. Figure 5 shows approaches per dataset (excluding pure SQR), with the error bars representing the standard error across five seeds. While the ordering changes across datasets, general trends are visible when comparing performance. The ranking in Figure 3 shows us this for ACE and Winkler, where AQR seems to be generally outcompeted. Together with Figure 5, we can now see that this is because it shows a sharper distribution overall. Overall, IQN seems to be the opposite, with a very high width in distribution and high coverage. Our own τ-scalers seem more in between this tradeoff, although based on overall ranking, SQR$_e$ (Φ^{-1}) displays the overall most favorable balance between coverage and sharpness. Curiously, SQR$_e$ (Φ^{-1}) was also the sharpest approach after CQR post-processing.

Comparing metric trends across datasets instead of approaches, the worst ACE and PICP are visible in King County, with the worst sharpness in Wages. These are the second and first (respectively) highest datasets in skewness and kurtosis.

In Figure 6, we can see how the τ-scaling approaches train and optimize a CDF in the Election dataset, just like how we showed QR and SQR in Figure 2. In comparison to SQR, all τ-scaling approaches reach a CDF that is non-linear and approximates a normal CDF. Interesting to see that, when looking at the optimization over a single x, each approach reaches this differently. IQN shows more variation in the curves, resembling QR, likely caused by the cosine embedding. AQR seems to reach non-linearity by a combination of piecewise linear functions, reflecting its two linear τ embedding layers acting on the internal state of the network, similar to a linear regression. SQR$_e$ and SQR$_s$ show a smoother development, likely informed by the shape of the τ embedding. Neither seems to be forced to only approximate the normal CDF, implying that these techniques may transfer to more non-normal settings.

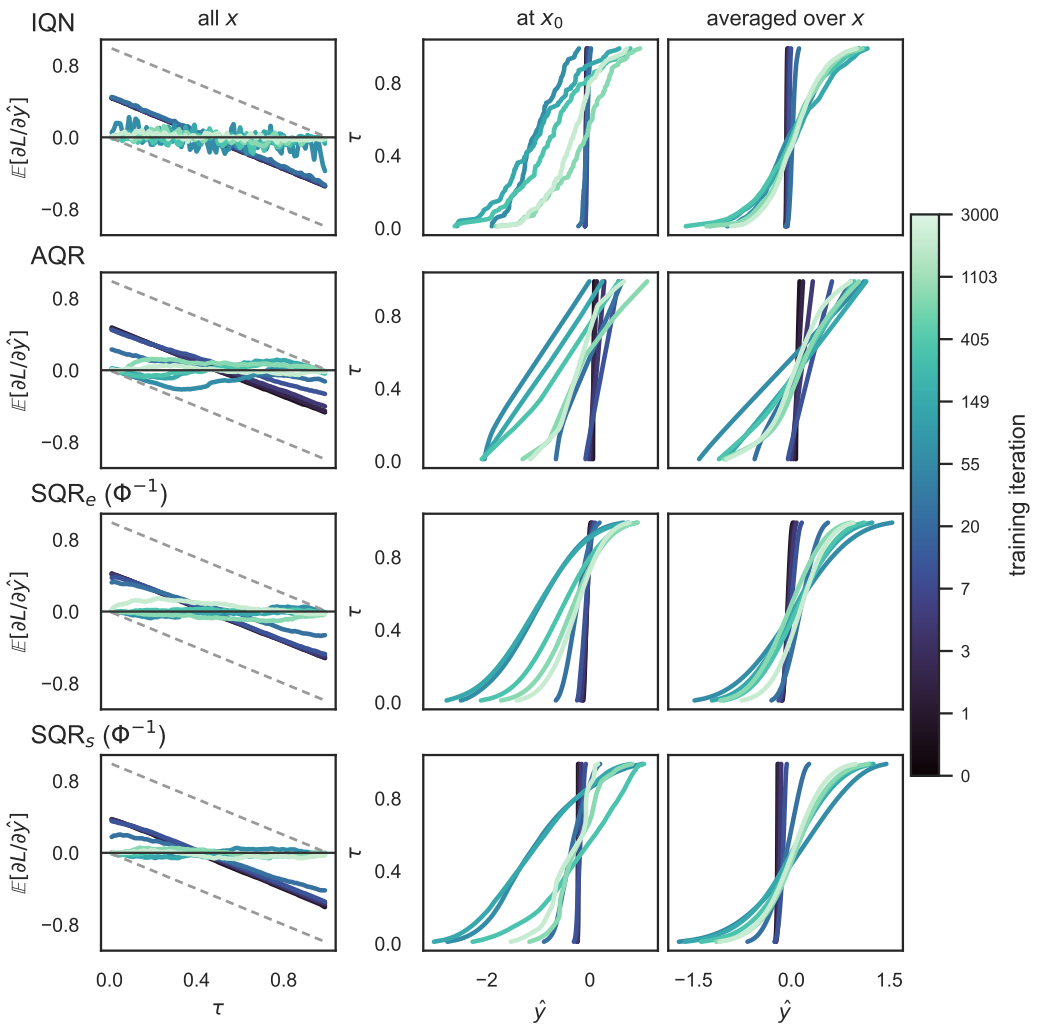


Figure 6: Training of τ-scaling approaches across samples in the elections dataset. The first column shows $\mathbb{E}[\partial L/\partial \hat{y}]$ over the samples as a function of τ. The other two columns are $\hat{y}$ for all τ across a single sample x, and averaged across all samples x. The dashed lines are the limits of the loss derivative given by $[-\tau, 1-\tau]$.

J Discussion of Results

This section represents an extension towards the short discussion of results in the conclusion of this work (section 5).

In this work, we evaluate the coverage of QR and SQR, identify, and lay out a hypothesis regarding SQR's missing tail coverage. We successfully show how our proposed SQR τ-scaling modifications close the coverage gap between SQR and QR. Further, IQN and AQR were previously not compared to standard SQR and QR in their coverage performance, enabling this work to validate their performance for the purpose of closing the coverage gap as well. Overall, the results also seem to support our hypothesis regarding the cause of the missing tails in regular SQR.

The comparison between τ-scaling methods in this paper is, however, limited by every τ-scaling approach showing overall similar performance to each other. This results in visibly high variance within approach performance (in seeds) and between datasets. The conclusions drawn from the comparison should be interpreted as suggestive trends alone. It may further be the case that some

of the approaches tested would show better performance if they were hyperparameter-optimized similarly to the baseline.

AQR deserves a closer discussion here. Compared to all other τ-scalers shown in this work, it does not clearly apply a nonlinear scaling of τ. This is the case as both embedding layers use linear and affine injections. This is also well visible in Figure 6. Our own ablation experiments comparing AQR to using only one of the embedding layers or not using β as a sampler showed that all three components are necessary for AQR's performance. It stands to reason that τ seems to need a sufficient amount of perturbation or scaling during training. While we have empirically shown that such scaling closes the coverage gap to QR, we leave a theoretical proof regarding how much change in τ is needed for future research.

This work also has limitations in its analysis regarding highly non-normal distributions, which may warrant further research. The datasets used for evaluation do not have highly non-normal distributions, except Wages and King County, which show heightened skewness and kurtosis. In either dataset, SQR$_e$ approaches showed high coverage together with QR. We may expect that SQR$_s$ approaches have difficulties with varying distribution shapes, as arguably a simple τ-scaling likely imposes a prior on the training. However, a prior may also pose an advantage if the data distribution is known. The version of CQR used here applies its correction symmetrically on each side of the interval, also limiting the interpretation of our results to approximately normal data.

Lastly, it should be noted that while this work did refer to the phenomenon of quantile collapse in the related work section, it is unclear whether the effect we are solving here is related. This is the case, as quantile collapse is often documented in the context of QR, but we are investigating cases where QR works well, but SQR does not.

K Results per dataset

In the following section, we present the complete standardized results per model and dataset. Models were trained repeatedly with the same configurations but using the seeds $[1, 2, 3, 4, 5]$. The tables also include ablation results for SQR$_e$ and SQR$_s$ where $\tau \sim U(0, 1)$ instead of $\tau \sim \beta(0.5, 0.5)$. We can see that either sampler does close the gap between SQR and QR, but there is a slight trend towards better results with the β sampler.

Table 7: Results from the abalone dataset. The average and standard deviation (in parentheses) of five seeded training runs. The best metric values are in bold. Sharpness is only made bold when ACE is within standard deviation (or better) of QR's value. SQR_{eU} and SQR_{sU} are ablation results of the τ-scaler approaches that sample $\tau \sim U(0,1)$.

Approach	ACE	$\text{Winkler}_{\text{avg}}$	Sharpness	PICP_{95}	Pinball
QR	0.017 (0.008)	2.530 (0.190)	2.738 (0.063)	0.946 (0.013)	0.095 (0.006)
SQR	0.060 (0.007)	3.052 (0.198)	1.860 (0.107)	0.870 (0.014)	0.099 (0.004)
SQR_{head}	0.064 (0.011)	3.062 (0.234)	1.862 (0.112)	0.871 (0.025)	0.099 (0.005)
SQR $\beta(0.5, 0.5)$	0.049 (0.005)	2.746 (0.182)	2.029 (0.108)	0.890 (0.011)	0.097 (0.005)
SQR_{wide}	0.075 (0.010)	3.159 (0.182)	1.735 (0.095)	0.844 (0.009)	0.100 (0.004)
SQR (8 layers)	0.061 (0.009)	3.170 (0.199)	1.860 (0.161)	0.860 (0.007)	0.102 (0.006)
SQR (loss reweight)	0.050 (0.009)	2.756 (0.155)	2.051 (0.169)	0.891 (0.014)	0.097 (0.006)
SQR cali. loss	0.076 (0.017)	3.220 (0.325)	1.763 (0.069)	0.842 (0.016)	0.100 (0.007)
SQR lr 1e-2, 5 epochs	0.065 (0.011)	3.082 (0.331)	1.849 (0.084)	0.861 (0.013)	0.099 (0.006)
SQR lr 1e-1, 1 epoch	0.088 (0.009)	3.531 (0.331)	1.684 (0.062)	0.827 (0.017)	0.105 (0.007)
SQR (SiLU)	0.053 (0.005)	2.906 (0.124)	2.042 (0.203)	0.889 (0.014)	0.099 (0.005)
SQR (constant τ-scale)	0.032 (0.005)	2.696 (0.102)	2.162 (0.138)	0.912 (0.012)	0.096 (0.004)
IQN	0.015 (0.007)	2.516 (0.119)	2.817 (0.138)	0.938 (0.013)	0.096 (0.005)
AQR	0.019 (0.003)	2.515 (0.148)	**2.515 (0.171)**	0.949 (0.017)	**0.094 (0.004)**
SQR_{e} (Φ^{-1})	0.022 (0.017)	2.491 (0.181)	2.617 (0.213)	0.937 (0.014)	0.095 (0.005)
SQR_{eU} (Φ^{-1})	0.014 (0.003)	2.501 (0.194)	2.577 (0.127)	0.942 (0.008)	**0.094 (0.006)**
SQR_{e} (logit)	**0.010 (0.005)**	2.495 (0.136)	2.899 (0.111)	**0.950 (0.010)**	0.095 (0.005)
SQR_{eU} (logit)	0.012 (0.005)	2.468 (0.138)	2.835 (0.128)	0.951 (0.007)	**0.094 (0.005)**
SQR_{s} (Φ^{-1})	0.016 (0.006)	2.491 (0.177)	2.764 (0.122)	0.944 (0.008)	0.095 (0.006)
SQR_{sU} (Φ^{-1})	0.024 (0.008)	2.611 (0.230)	2.651 (0.088)	0.932 (0.007)	0.097 (0.006)
SQR_{s} (logit)	0.019 (0.008)	**2.466 (0.157)**	2.934 (0.243)	0.946 (0.012)	0.095 (0.005)
SQR_{sU} (logit)	0.013 (0.005)	2.504 (0.154)	3.081 (0.202)	0.956 (0.015)	0.095 (0.005)

Table 8: Results from the California dataset. The average and standard deviation (in parentheses) of five seeded training runs. The best metric values are in bold. Sharpness is only made bold when ACE is within standard deviation (or better) of QR's value. SQR_{eU} and SQR_{sU} are ablation results of the τ-scaler approaches that sample $\tau \sim U(0,1)$.

Approach	ACE	$\text{Winkler}_{\text{avg}}$	Sharpness	PICP_{95}	Pinball
QR	0.020 (0.013)	**1.710 (0.086)**	1.668 (0.076)	0.925 (0.019)	**0.061 (0.002)**
SQR	0.053 (0.012)	2.080 (0.080)	1.176 (0.106)	0.881 (0.015)	0.063 (0.001)
SQR_{head}	0.065 (0.010)	2.226 (0.061)	1.080 (0.048)	0.855 (0.013)	0.064 (0.001)
SQR $\beta(0.5, 0.5)$	0.027 (0.003)	1.864 (0.078)	1.446 (0.061)	0.918 (0.005)	0.062 (0.001)
SQR_{wide}	0.078 (0.012)	2.306 (0.158)	1.053 (0.066)	0.838 (0.015)	0.065 (0.002)
SQR (8 layers)	0.043 (0.010)	2.000 (0.083)	1.246 (0.068)	0.892 (0.013)	0.062 (0.001)
SQR (loss reweight)	0.026 (0.007)	1.876 (0.035)	1.436 (0.084)	0.917 (0.013)	0.062 (0.001)
SQR cali. loss	0.052 (0.005)	2.114 (0.061)	1.300 (0.043)	0.881 (0.007)	0.065 (0.002)
SQR lr 1e-2, 5 epochs	0.069 (0.022)	2.209 (0.154)	1.107 (0.063)	0.849 (0.027)	0.064 (0.003)
SQR lr 1e-1, 1 epoch	0.081 (0.006)	2.491 (0.105)	1.117 (0.030)	0.840 (0.007)	0.070 (0.002)
SQR (SiLU)	0.040 (0.003)	1.972 (0.052)	1.254 (0.049)	0.890 (0.005)	0.062 (0.001)
SQR (constant τ-scale)	0.031 (0.010)	1.876 (0.018)	1.394 (0.095)	0.910 (0.011)	0.062 (0.001)
IQN	0.016 (0.007)	1.732 (0.060)	2.022 (0.122)	0.936 (0.014)	**0.061 (0.002)**
AQR	0.015 (0.003)	1.752 (0.030)	1.814 (0.102)	**0.947 (0.007)**	0.062 (0.001)
SQR_{e} (Φ^{-1})	**0.009 (0.010)**	1.741 (0.056)	1.768 (0.090)	0.941 (0.008)	**0.061 (0.001)**
SQR_{eU} (Φ^{-1})	0.028 (0.015)	1.803 (0.082)	**1.599 (0.084)**	0.923 (0.011)	0.062 (0.002)
SQR_{e} (logit)	0.017 (0.009)	1.738 (0.051)	1.810 (0.079)	0.937 (0.008)	**0.061 (0.001)**
SQR_{eU} (logit)	0.023 (0.011)	1.755 (0.062)	1.714 (0.075)	0.932 (0.006)	**0.061 (0.002)**
SQR_{s} (Φ^{-1})	0.018 (0.007)	1.728 (0.056)	1.740 (0.072)	0.932 (0.005)	**0.061 (0.002)**
SQR_{sU} (Φ^{-1})	0.024 (0.010)	1.794 (0.088)	1.637 (0.050)	0.925 (0.007)	**0.061 (0.002)**
SQR_{s} (logit)	0.016 (0.010)	1.734 (0.063)	1.845 (0.025)	0.939 (0.009)	**0.061 (0.002)**
SQR_{sU} (logit)	0.024 (0.009)	1.782 (0.036)	1.765 (0.126)	0.930 (0.008)	**0.061 (0.001)**

Table 9: Results from the elections dataset. The average and standard deviation (in parentheses) of five seeded training runs. The best metric values are in bold. Sharpness is only made bold when ACE is within standard deviation (or better) of QR's value. SQR_{eU} and SQR_{sU} are ablation results of the τ-scaler approaches that sample $\tau \sim U(0,1)$.

Approach	ACE	$\text{Winkler}_{\text{avg}}$	Sharpness	PICP_{95}	Pinball
QR	0.032 (0.022)	2.057 (0.186)	1.914 (0.172)	0.920 (0.021)	0.075 (0.004)
SQR	0.075 (0.006)	2.456 (0.057)	1.398 (0.076)	0.844 (0.008)	0.078 (0.003)
SQR_{head}	0.125 (0.036)	3.013 (0.378)	1.145 (0.072)	0.783 (0.038)	0.083 (0.005)
SQR $\beta(0.5, 0.5)$	0.072 (0.011)	2.529 (0.223)	1.446 (0.043)	0.854 (0.028)	0.081 (0.004)
SQR_{wide}	0.104 (0.013)	2.793 (0.207)	1.244 (0.032)	0.805 (0.020)	0.081 (0.003)
SQR (8 layers)	0.066 (0.007)	2.371 (0.144)	1.451 (0.057)	0.860 (0.013)	0.077 (0.002)
SQR (loss reweight)	0.066 (0.009)	2.416 (0.264)	1.440 (0.054)	0.864 (0.022)	0.079 (0.004)
SQR cali. loss	0.112 (0.023)	3.134 (0.369)	1.313 (0.034)	0.801 (0.023)	0.090 (0.006)
SQR lr 1e-2, 5 epochs	0.101 (0.014)	2.721 (0.298)	1.279 (0.047)	0.817 (0.017)	0.080 (0.006)
SQR lr 1e-1, 1 epoch	0.088 (0.016)	2.941 (0.421)	1.410 (0.053)	0.826 (0.024)	0.087 (0.006)
SQR (SiLU)	0.067 (0.012)	2.440 (0.200)	1.441 (0.036)	0.852 (0.016)	0.079 (0.004)
SQR (constant τ-scale)	0.027 (0.009)	2.115 (0.168)	**1.783 (0.102)**	0.927 (0.018)	0.077 (0.003)
IQN	0.022 (0.015)	1.952 (0.161)	2.110 (0.046)	0.941 (0.015)	0.075 (0.003)
AQR	0.020 (0.006)	1.978 (0.111)	1.958 (0.081)	0.941 (0.014)	**0.074 (0.002)**
SQR_{e} (Φ^{-1})	0.026 (0.014)	1.943 (0.132)	2.035 (0.097)	0.938 (0.019)	**0.074 (0.003)**
SQR_{eU} (Φ^{-1})	0.037 (0.032)	2.049 (0.191)	1.927 (0.111)	0.921 (0.026)	0.076 (0.004)
SQR_{e} (logit)	0.022 (0.014)	1.959 (0.063)	2.228 (0.108)	0.942 (0.017)	0.075 (0.002)
SQR_{eU} (logit)	0.026 (0.006)	**1.935 (0.097)**	2.178 (0.084)	0.941 (0.018)	**0.074 (0.003)**
SQR_{s} (Φ^{-1})	0.019 (0.009)	2.014 (0.109)	2.181 (0.013)	0.941 (0.009)	0.075 (0.003)
SQR_{sU} (Φ^{-1})	0.020 (0.011)	1.996 (0.071)	2.150 (0.150)	0.937 (0.008)	0.075 (0.002)
SQR_{s} (logit)	**0.012 (0.009)**	1.970 (0.102)	2.322 (0.081)	0.947 (0.010)	0.075 (0.003)
SQR_{sU} (logit)	0.014 (0.006)	1.950 (0.109)	2.331 (0.087)	**0.950 (0.014)**	**0.074 (0.003)**

Table 10: Results from the King County dataset. The average and standard deviation (in parentheses) of five seeded training runs. The best metric values are in bold. Sharpness is only made bold when ACE is within standard deviation (or better) of QR's value. SQR_{eU} and SQR_{sU} are ablation results of the τ-scaler approaches that sample $\tau \sim U(0,1)$.

Approach	ACE	$\text{Winkler}_{\text{avg}}$	Sharpness	PICP_{95}	Pinball
QR	0.029 (0.010)	1.219 (0.051)	1.153 (0.068)	0.926 (0.005)	**0.042 (0.001)**
SQR	0.123 (0.013)	1.937 (0.037)	0.590 (0.017)	0.778 (0.013)	0.049 (0.001)
SQR_{head}	0.115 (0.011)	2.161 (0.159)	0.565 (0.018)	0.790 (0.017)	0.051 (0.002)
SQR $\beta(0.5, 0.5)$	0.090 (0.005)	1.728 (0.055)	0.671 (0.029)	0.822 (0.011)	0.047 (0.001)
SQR_{wide}	0.135 (0.013)	2.018 (0.060)	0.573 (0.008)	0.763 (0.012)	0.050 (0.001)
SQR (8 layers)	0.114 (0.010)	1.832 (0.110)	0.605 (0.018)	0.789 (0.012)	0.047 (0.001)
SQR (loss reweight)	0.092 (0.004)	1.700 (0.069)	0.670 (0.011)	0.822 (0.009)	0.047 (0.001)
SQR cali. loss	0.187 (0.009)	2.278 (0.054)	0.493 (0.016)	0.712 (0.010)	0.052 (0.001)
SQR lr 1e-2, 5 epochs	0.105 (0.005)	1.811 (0.059)	0.607 (0.009)	0.801 (0.008)	0.047 (0.001)
SQR lr 1e-1, 1 epoch	0.185 (0.033)	2.807 (0.583)	0.532 (0.061)	0.711 (0.031)	0.060 (0.008)
SQR (SiLU)	0.101 (0.008)	1.801 (0.089)	0.621 (0.020)	0.807 (0.012)	0.047 (0.001)
SQR (constant τ-scale)	0.080 (0.011)	1.576 (0.105)	0.692 (0.026)	0.841 (0.015)	0.045 (0.001)
IQN	0.038 (0.011)	1.248 (0.071)	1.248 (0.078)	0.933 (0.008)	0.043 (0.001)
AQR	0.036 (0.012)	1.323 (0.064)	0.886 (0.067)	0.915 (0.017)	0.043 (0.001)
SQR_{e} (Φ^{-1})	**0.023 (0.009)**	1.230 (0.075)	**1.051 (0.009)**	0.924 (0.007)	**0.042 (0.001)**
SQR_{eU} (Φ^{-1})	0.033 (0.015)	1.270 (0.077)	1.004 (0.092)	0.914 (0.017)	0.043 (0.001)
SQR_{e} (logit)	0.026 (0.015)	**1.214 (0.052)**	1.191 (0.097)	**0.939 (0.013)**	0.043 (0.001)
SQR_{eU} (logit)	**0.023 (0.007)**	1.219 (0.063)	1.166 (0.052)	0.931 (0.006)	**0.042 (0.001)**
SQR_{s} (Φ^{-1})	0.042 (0.016)	1.296 (0.061)	1.018 (0.065)	0.904 (0.013)	0.044 (0.001)
SQR_{sU} (Φ^{-1})	0.034 (0.018)	1.320 (0.099)	1.043 (0.060)	0.913 (0.014)	0.044 (0.001)
SQR_{s} (logit)	0.032 (0.019)	1.251 (0.054)	1.185 (0.086)	0.930 (0.014)	0.043 (0.001)
SQR_{sU} (logit)	0.047 (0.014)	1.258 (0.050)	1.112 (0.052)	0.918 (0.010)	0.043 (0.001)

Table 11: Results from the protein dataset. The average and standard deviation (in parentheses) of five seeded training runs. The best metric values are in bold. Sharpness is only made bold when ACE is within standard deviation (or better) of QR's value. SQR_{eU} and SQR_{sU} are ablation results of the τ-scaler approaches that sample $\tau \sim U(0,1)$.

Approach	ACE	$\text{Winkler}_{\text{avg}}$	Sharpness	PICP_{95}	Pinball
QR	0.025 (0.008)	2.257 (0.068)	2.087 (0.038)	0.928 (0.007)	**0.083 (0.001)**
SQR	0.052 (0.015)	2.658 (0.173)	1.655 (0.090)	0.892 (0.012)	0.085 (0.002)
SQR_{head}	0.070 (0.016)	3.112 (0.116)	1.383 (0.078)	0.854 (0.024)	0.088 (0.001)
SQR $\beta(0.5, 0.5)$	0.017 (0.003)	2.339 (0.066)	2.034 (0.042)	0.932 (0.005)	0.086 (0.001)
SQR_{wide}	0.062 (0.019)	2.918 (0.198)	1.582 (0.123)	0.858 (0.025)	0.090 (0.001)
SQR (8 layers)	0.083 (0.039)	2.730 (0.228)	1.494 (0.143)	0.854 (0.035)	0.084 (0.003)
SQR (loss reweight)	0.023 (0.004)	2.453 (0.074)	**1.961 (0.107)**	0.921 (0.008)	0.087 (0.002)
SQR cali. loss	0.063 (0.021)	2.769 (0.120)	1.846 (0.071)	0.871 (0.019)	0.092 (0.002)
SQR lr 1e-2, 5 epochs	0.044 (0.007)	2.798 (0.059)	1.705 (0.065)	0.885 (0.010)	0.090 (0.001)
SQR lr 1e-1, 1 epoch	0.086 (0.010)	3.095 (0.140)	1.843 (0.077)	0.830 (0.015)	0.103 (0.003)
SQR (SiLU)	0.049 (0.015)	2.798 (0.158)	1.701 (0.062)	0.882 (0.013)	0.089 (0.002)
SQR (constant τ-scale)	0.029 (0.007)	2.533 (0.096)	2.000 (0.074)	0.913 (0.006)	0.088 (0.001)
IQN	0.019 (0.008)	**2.222 (0.042)**	2.588 (0.087)	0.937 (0.013)	0.085 (0.001)
AQR	0.022 (0.005)	2.334 (0.042)	2.447 (0.150)	**0.944 (0.018)**	0.088 (0.001)
SQR_{e} (Φ^{-1})	0.019 (0.012)	2.279 (0.011)	2.252 (0.109)	0.937 (0.013)	0.085 (0.001)
SQR_{eU} (Φ^{-1})	0.021 (0.010)	2.378 (0.067)	2.131 (0.083)	0.924 (0.011)	0.086 (0.002)
SQR_{e} (logit)	**0.007 (0.003)**	2.253 (0.085)	2.352 (0.068)	0.941 (0.003)	0.085 (0.002)
SQR_{eU} (logit)	0.025 (0.016)	2.385 (0.091)	2.240 (0.095)	0.926 (0.013)	0.087 (0.002)
SQR_{s} (Φ^{-1})	0.015 (0.006)	2.278 (0.052)	2.350 (0.055)	0.938 (0.007)	0.085 (0.001)
SQR_{sU} (Φ^{-1})	0.026 (0.008)	2.407 (0.063)	2.230 (0.091)	0.919 (0.006)	0.087 (0.001)
SQR_{s} (logit)	0.023 (0.011)	2.301 (0.061)	2.427 (0.121)	0.933 (0.010)	0.086 (0.002)
SQR_{sU} (logit)	0.014 (0.011)	2.324 (0.099)	2.443 (0.070)	0.936 (0.010)	0.086 (0.002)

Table 12: Results from the wages dataset. The average and standard deviation (in parentheses) of five seeded training runs. The best metric values are in bold. Sharpness is only made bold when ACE is within standard deviation (or better) of QR's value. SQR_{eU} and SQR_{sU} are ablation results of the τ-scaler approaches that sample $\tau \sim U(0,1)$.

Approach	ACE	$\text{Winkler}_{\text{avg}}$	Sharpness	PICP_{95}	Pinball
QR	**0.005 (0.003)**	**3.187 (0.372)**	3.330 (0.083)	**0.950 (0.005)**	**0.106 (0.008)**
SQR	0.022 (0.005)	3.375 (0.394)	2.446 (0.109)	0.926 (0.014)	0.107 (0.008)
SQR_{head}	0.022 (0.003)	3.390 (0.406)	**2.356 (0.110)**	0.926 (0.007)	0.108 (0.008)
SQR $\beta(0.5, 0.5)$	0.018 (0.008)	3.325 (0.367)	2.637 (0.159)	0.930 (0.012)	0.107 (0.008)
SQR_{wide}	0.022 (0.003)	3.388 (0.380)	2.455 (0.159)	0.923 (0.012)	0.108 (0.008)
SQR (8 layers)	0.026 (0.008)	3.372 (0.426)	2.480 (0.098)	0.929 (0.015)	0.108 (0.008)
SQR (loss reweight)	0.019 (0.003)	3.313 (0.408)	2.582 (0.137)	0.938 (0.009)	0.107 (0.008)
SQR cali. loss	0.027 (0.010)	3.447 (0.425)	2.347 (0.075)	0.915 (0.009)	0.108 (0.009)
SQR lr 1e-2, 5 epochs	0.021 (0.004)	3.363 (0.393)	2.450 (0.099)	0.925 (0.010)	0.107 (0.008)
SQR lr 1e-1, 1 epoch	0.045 (0.008)	3.660 (0.432)	2.076 (0.157)	0.886 (0.008)	0.109 (0.009)
SQR (SiLU)	0.023 (0.004)	3.309 (0.374)	2.442 (0.119)	0.934 (0.009)	0.107 (0.008)
SQR (constant τ-scale)	0.017 (0.002)	3.292 (0.377)	2.602 (0.164)	0.941 (0.006)	0.107 (0.008)
IQN	0.015 (0.003)	3.203 (0.367)	3.498 (0.160)	0.947 (0.012)	**0.106 (0.008)**
AQR	0.016 (0.007)	3.261 (0.358)	3.370 (0.104)	**0.950 (0.012)**	0.107 (0.008)
SQR_{e} (Φ^{-1})	0.007 (0.003)	3.201 (0.376)	3.202 (0.179)	**0.950 (0.005)**	**0.106 (0.008)**
SQR_{eU} (Φ^{-1})	0.006 (0.003)	3.214 (0.359)	2.980 (0.225)	0.948 (0.002)	**0.106 (0.008)**
SQR_{e} (logit)	0.006 (0.003)	3.204 (0.374)	3.321 (0.170)	0.951 (0.008)	**0.106 (0.008)**
SQR_{eU} (logit)	0.009 (0.005)	3.203 (0.368)	3.068 (0.150)	0.948 (0.007)	**0.106 (0.008)**
SQR_{s} (Φ^{-1})	0.009 (0.006)	3.207 (0.354)	3.374 (0.214)	0.952 (0.008)	**0.106 (0.008)**
SQR_{sU} (Φ^{-1})	0.008 (0.006)	3.204 (0.371)	3.132 (0.200)	0.949 (0.007)	**0.106 (0.008)**
SQR_{s} (logit)	0.011 (0.006)	3.203 (0.373)	3.361 (0.107)	0.949 (0.010)	**0.106 (0.008)**
SQR_{sU} (logit)	0.010 (0.004)	3.202 (0.364)	3.282 (0.113)	0.951 (0.009)	**0.106 (0.008)**

Table 13: Results from the work hours dataset. The average and standard deviation (in parentheses) of five seeded training runs. The best metric values are in bold. Sharpness is only made bold when ACE is within standard deviation (or better) of QR's value. SQR_{eU} and SQR_{sU} are ablation results of the τ-scaler approaches that sample $\tau \sim U(0, 1)$.

Approach	ACE	$\text{Winkler}_{\text{avg}}$	Sharpness	PICP_{95}	Pinball
QR	0.021 (0.008)	2.582 (0.049)	2.778 (0.099)	0.943 (0.011)	**0.104 (0.002)**
SQR	0.076 (0.029)	3.064 (0.199)	1.935 (0.045)	0.870 (0.043)	0.108 (0.003)
SQR_{head}	0.040 (0.006)	3.015 (0.173)	2.055 (0.106)	0.912 (0.014)	0.108 (0.003)
SQR $\beta(0.5, 0.5)$	0.049 (0.011)	2.810 (0.091)	**2.246 (0.068)**	0.921 (0.035)	0.107 (0.002)
SQR_{wide}	0.061 (0.010)	3.124 (0.128)	1.912 (0.028)	0.881 (0.022)	0.109 (0.002)
SQR (8 layers)	0.064 (0.005)	3.003 (0.090)	1.967 (0.041)	0.907 (0.005)	0.107 (0.002)
SQR (loss reweight)	0.048 (0.014)	2.860 (0.164)	2.140 (0.109)	0.907 (0.023)	0.107 (0.003)
SQR cali. loss	0.077 (0.036)	3.338 (0.140)	1.953 (0.052)	0.855 (0.026)	0.111 (0.002)
SQR lr 1e-2, 5 epochs	0.053 (0.010)	2.995 (0.139)	1.951 (0.035)	0.888 (0.019)	0.106 (0.002)
SQR lr 1e-1, 1 epoch	0.173 (0.134)	4.370 (1.373)	1.669 (0.157)	0.728 (0.154)	0.121 (0.016)
SQR (SiLU)	0.059 (0.034)	2.964 (0.104)	1.998 (0.027)	0.875 (0.044)	0.107 (0.002)
SQR (constant τ-scale)	0.037 (0.011)	2.812 (0.142)	2.184 (0.029)	0.914 (0.019)	0.106 (0.002)
IQN	0.031 (0.016)	2.564 (0.044)	2.854 (0.073)	0.955 (0.009)	**0.104 (0.001)**
AQR	0.033 (0.008)	2.644 (0.074)	2.570 (0.151)	0.958 (0.012)	0.105 (0.001)
SQR_{e} (Φ^{-1})	0.018 (0.015)	2.584 (0.069)	2.837 (0.078)	0.953 (0.011)	0.105 (0.002)
SQR_{eU} (Φ^{-1})	0.019 (0.017)	2.646 (0.063)	2.724 (0.097)	0.936 (0.014)	0.105 (0.002)
SQR_{e} (logit)	0.018 (0.011)	2.569 (0.061)	2.857 (0.078)	0.943 (0.012)	0.105 (0.002)
SQR_{eU} (logit)	0.034 (0.016)	2.608 (0.070)	2.766 (0.046)	0.938 (0.015)	0.105 (0.002)
SQR_{s} (Φ^{-1})	0.025 (0.017)	2.555 (0.068)	2.757 (0.040)	0.943 (0.015)	**0.104 (0.002)**
SQR_{sU} (Φ^{-1})	0.023 (0.019)	2.584 (0.083)	2.653 (0.089)	0.943 (0.022)	**0.104 (0.002)**
SQR_{s} (logit)	0.019 (0.009)	**2.535 (0.079)**	2.808 (0.050)	**0.952 (0.008)**	**0.104 (0.002)**
SQR_{sU} (logit)	**0.014 (0.006)**	2.556 (0.074)	2.852 (0.101)	0.958 (0.010)	**0.104 (0.002)**