Accurate Uncertainties for Deep Learning Using Calibrated Regression

Analysis of the paper by Kuleshov et al. (2018)

Project team: Piotr Pekala, Benjamin Yuen, Dmitry Vukolov

Source code available in the GitHub repository.

Outline

1. Problem Statement: Sources of Miscalibration
2. Existing Work
3. Contribution: The Calibration Algorithm
4. Technical Details
5. Experiments
6. Evaluation of the Claims
7. Future Work
8. References

Problem Statement

The Issue of Miscalibration

Problem statement: Proper quantification of uncertainty is crucial for applying statistical models to real-world situations. The Bayesian approach to modeling provides us with a principled way of obtaining such uncertainty estimates. Yet, due to various reasons, such estimates are often inaccurate. For example, a 95% posterior predictive interval does not contain the true outcome with 95% probability. Such a model is *miscalibrated*.

Context: Correct uncertainty estimates along with sharp predictions are especially important for mission-critical applications where the cost of error is high. For example, knowing that a model isn't sure about a particular outcome might prompt human involvement for difficult decision making. Additionally, measuring different types of uncertainty such as *epistemic* and *aleatoric* allows researchers to have a better understanding of the model's predictive capabilities and work on systematically improving it.

Sources of Miscalibration

Below we demonstrate that the problem of miscalibration exists and show why it exists for **Bayesian neural networks** (BNNs) in regression tasks. We focus on the following sources of miscalibration:

- The **prior** is wrong, e.g. too strong and overcertain
- The **likelihood function** is wrong. There is bias, i.e. the neural network is too simple and is unable to model the data.
- The **noise** specification in the likelihood is wrong.
- The **inference** is approximate or is performed incorrectly.

Our aim is to establish a causal link between each aspect of the model building process and a bad miscalibrated outcome.

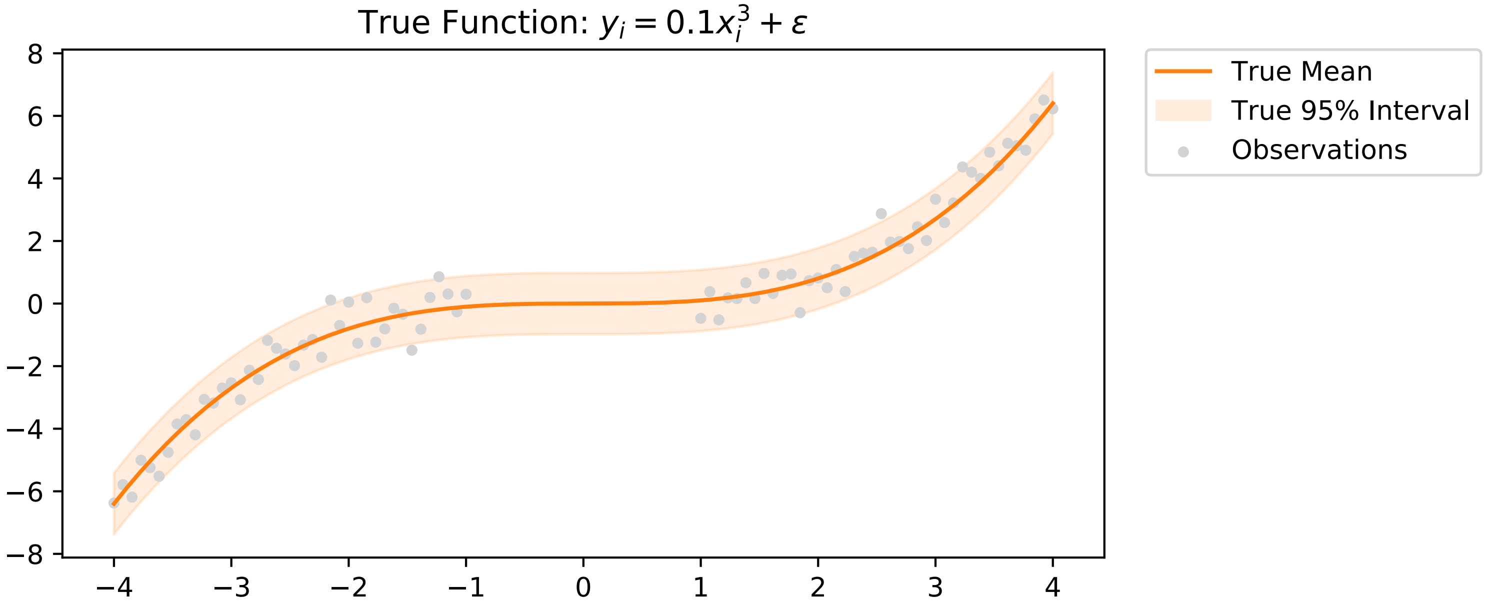
Methodology

1. **Data Generation:** We generate the data from a known true function with Gaussian noise. We then build multiple feedforward BNN models using:
    - different network architectures
    - several priors on the weights, depending on model complexity
    - different variance of the Gaussian noise in the likelihood function
2. **Inference**: We obtain the posterior of the model by:
    - sampling from it with the No-U-Turn algorithm
    - approximating the posterior using Variational Inference with reparametrization and isotropic Gaussians
3. **Diagnostics**: We check for convergence using trace plots, the effective sample size, and Gelman-Rubin tests. In the case of variational inference, we track the ELBO during optimization. The simulated posterior predictive is evaluated visually.

The probabilistic library NumPyro provides fast implementations of both algorithms, which we make use of in this research. Due to time constraints we do not perform multiple random restarts, so the results may be subject to randomness.

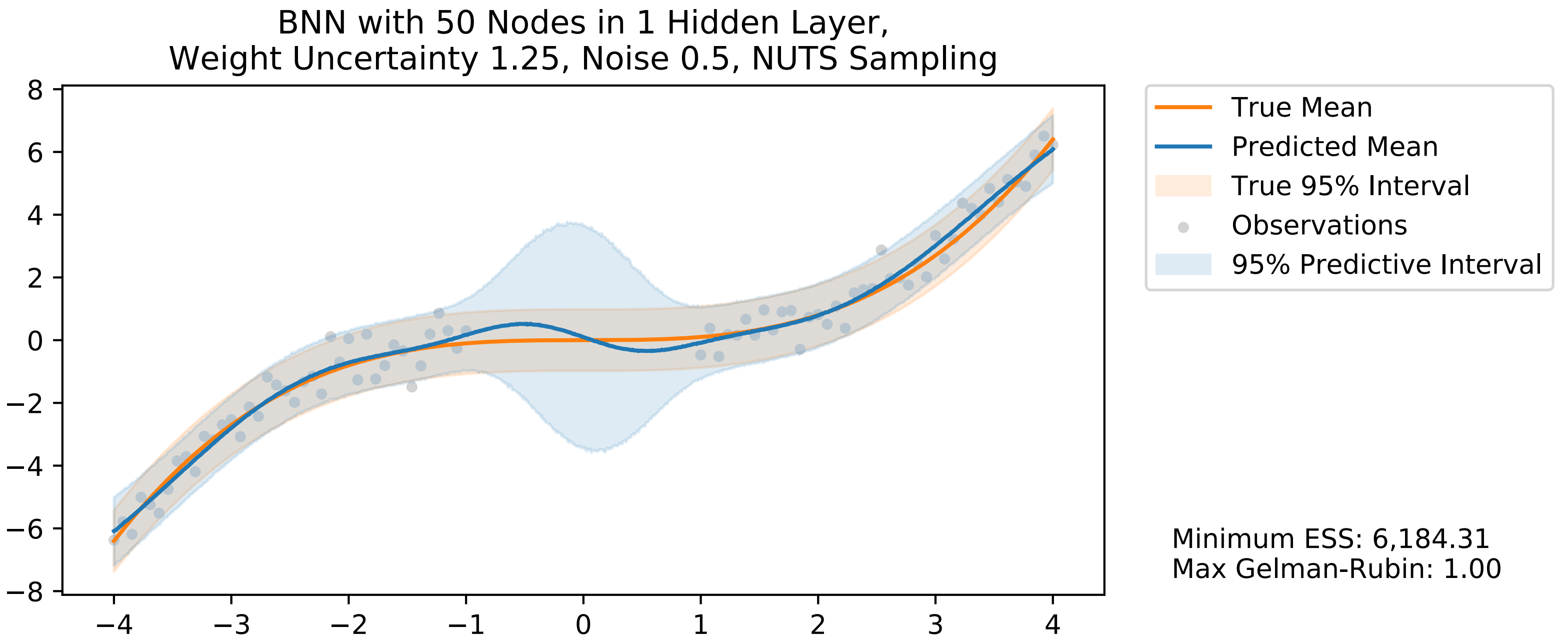
Example: a Toy Dataset

Using a simple data-generating function $y_i = 0.1x_i^3 + \varepsilon$, where $\varepsilon \sim \mathrm{N}(0, 0.5^2)$ and a series of BNN models we evaluate the impact of our design choices on the posterior predictive.



Proper Posterior Predictive

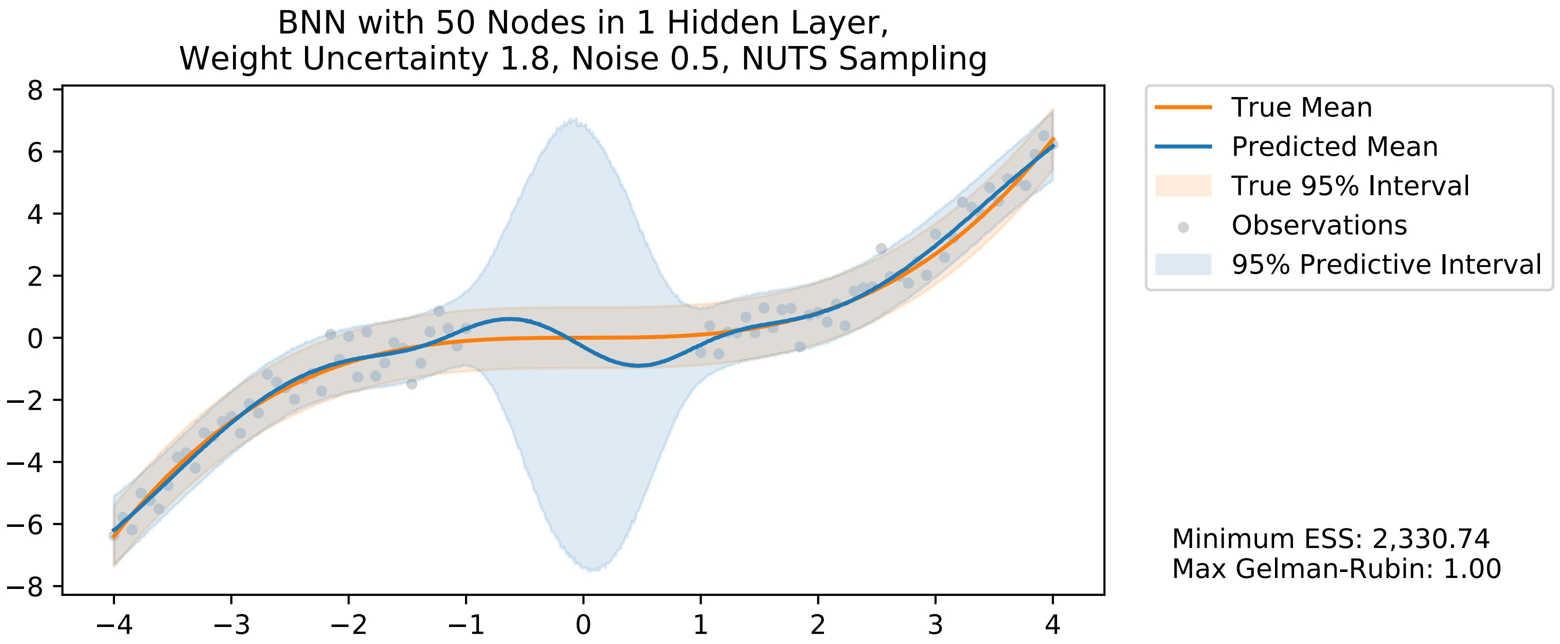
A neural network with 50 nodes in a single hidden layer, well-chosen prior and noise values, as well as correctly performed inference using sampling produce a posterior predictive that adequately reflects both epistemic and aleatoric uncertainty:



Naturally, our statements regarding the adequacy of epistemic uncertainty are subjective due to the absence of universal quantitative metrics. In our case, we know the true data-generating function, which impacts our expectations of the right width of the 95% predictive interval in out of distribution regions.

Wrong Prior: Too Wide

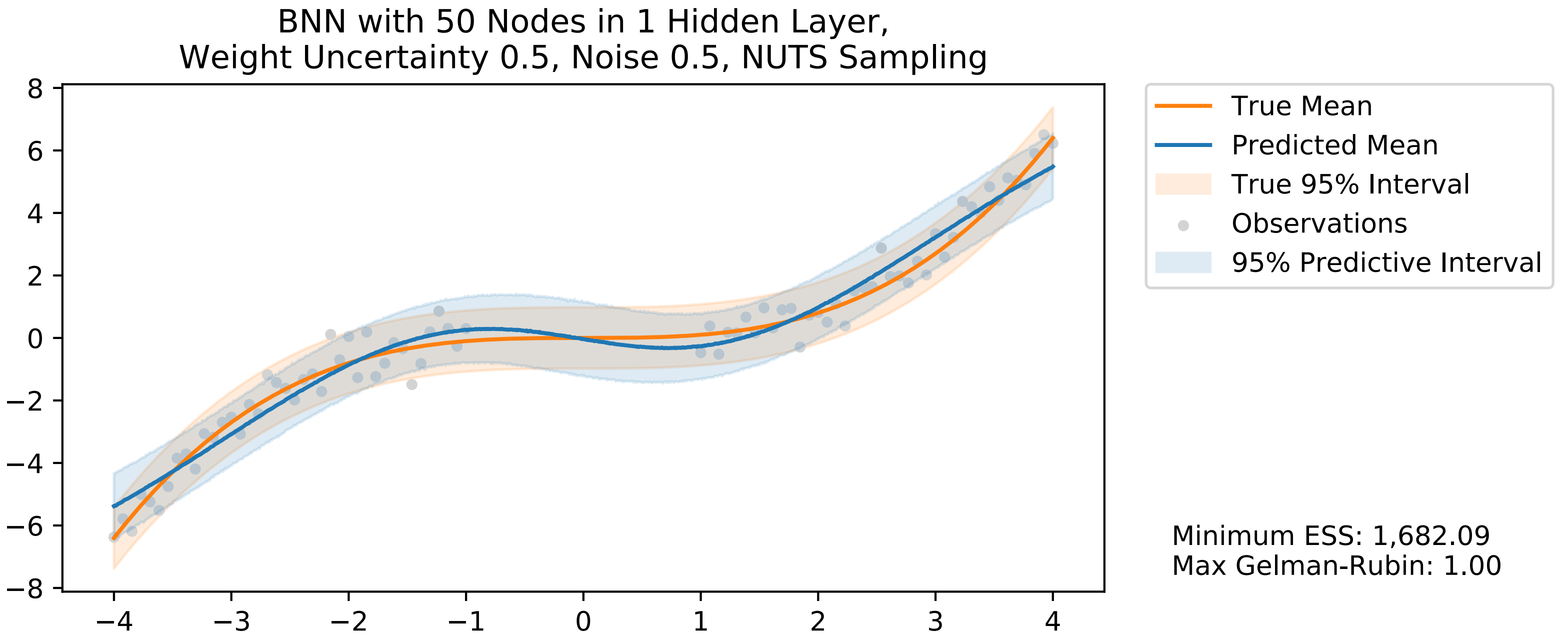
The prior on the network weights defines epistemic uncertainty. A higher than necessary variance of the prior results in a significantly larger and most likely unreasonable epistemic uncertainty:



In the more general case we wouldn't be able to formulate our expectations of the appropriate amount of epistemic uncertainty if **(1)** the true data-generating process was unknown and/or **(2)** the data was not possible to visualize, or **(3)** there were no quantitative metric that takes into account the downstream task.

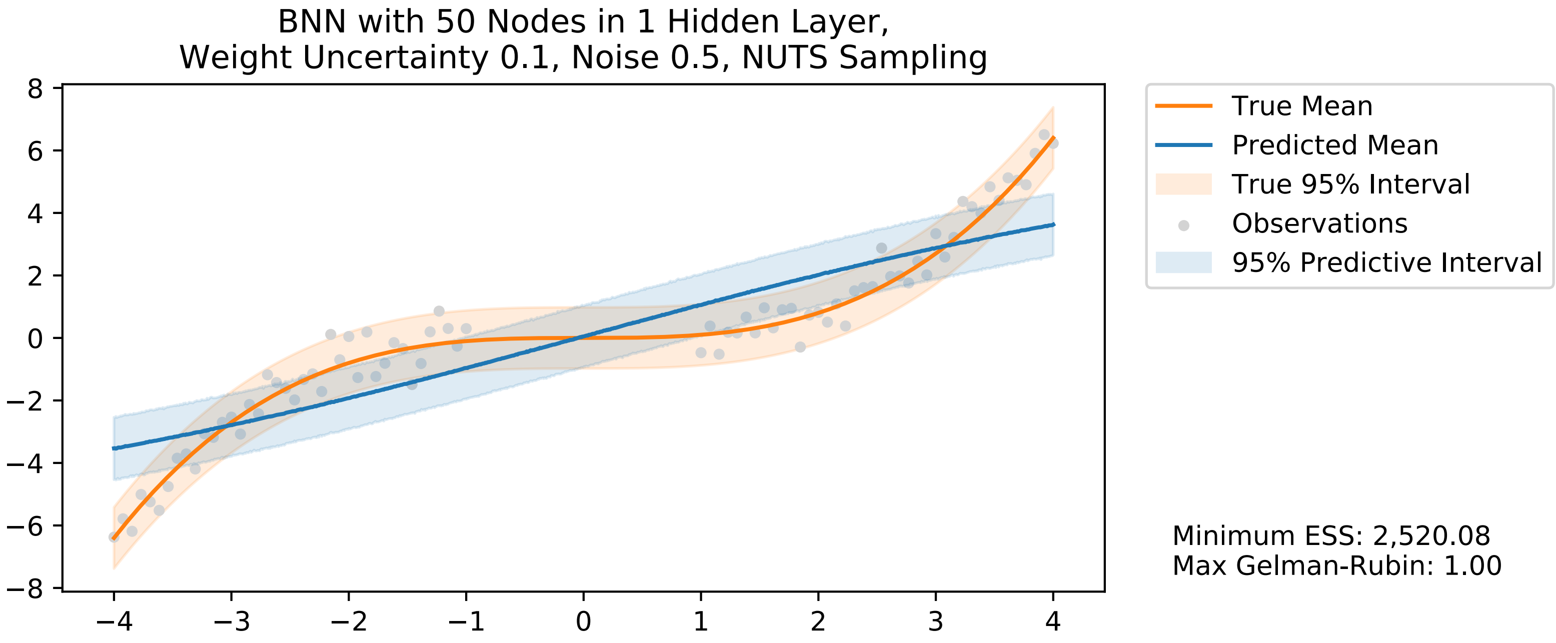
Wrong Prior: Too Narrow

A prior which is too restrictive prohibits the network from fitting the data and indicating the areas of higher epistemic uncertainty. It introduces bias: a neural network with 50 nodes in a single hidden layer (i.e. 151 weights) is unable to fit a cubic function. Ideally, we should fix that by selecting wider priors and allowing more flexibility in the model:



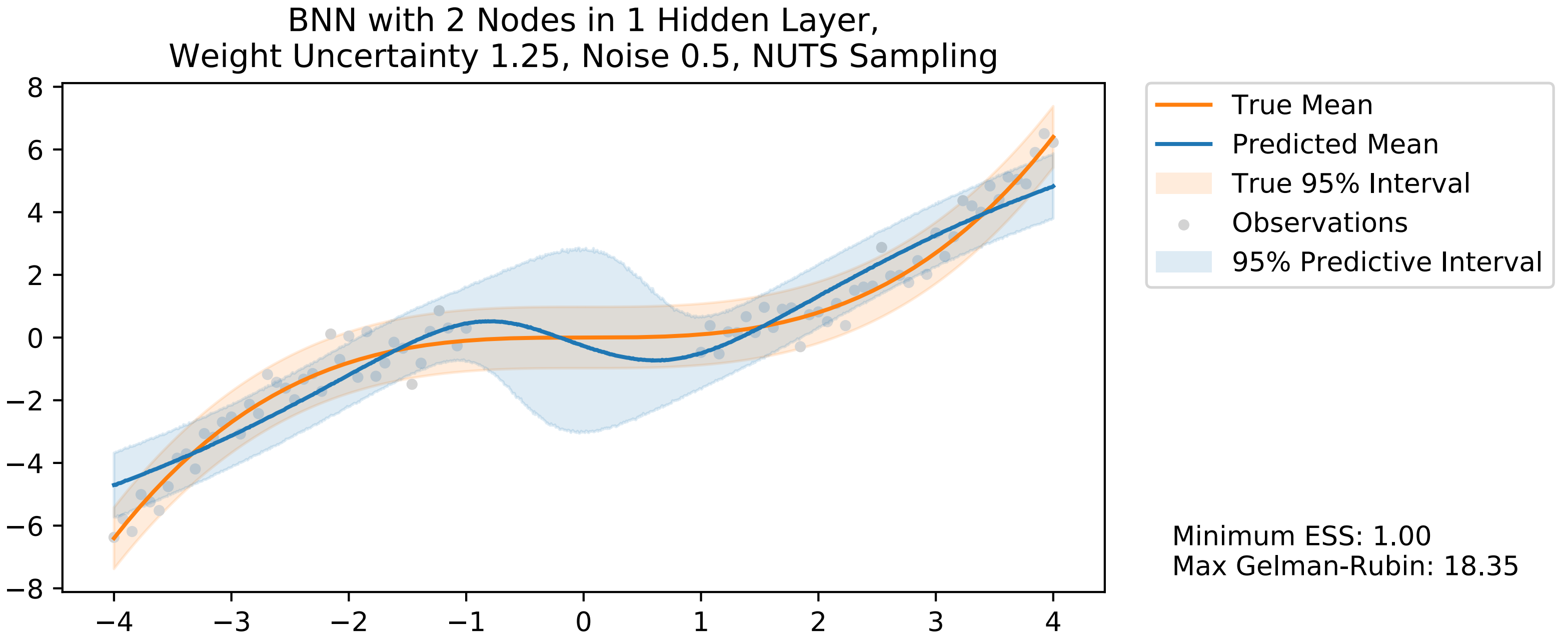
Wrong Prior: Extremely Restrictive

The bias becomes apparent with an even narrower prior on the weights. This is a major issue with the model that needs to be fixed. No other technique such as recalibration of the incorrect posterior predictive would be justifiable in this case.



Wrong Likelihood Function

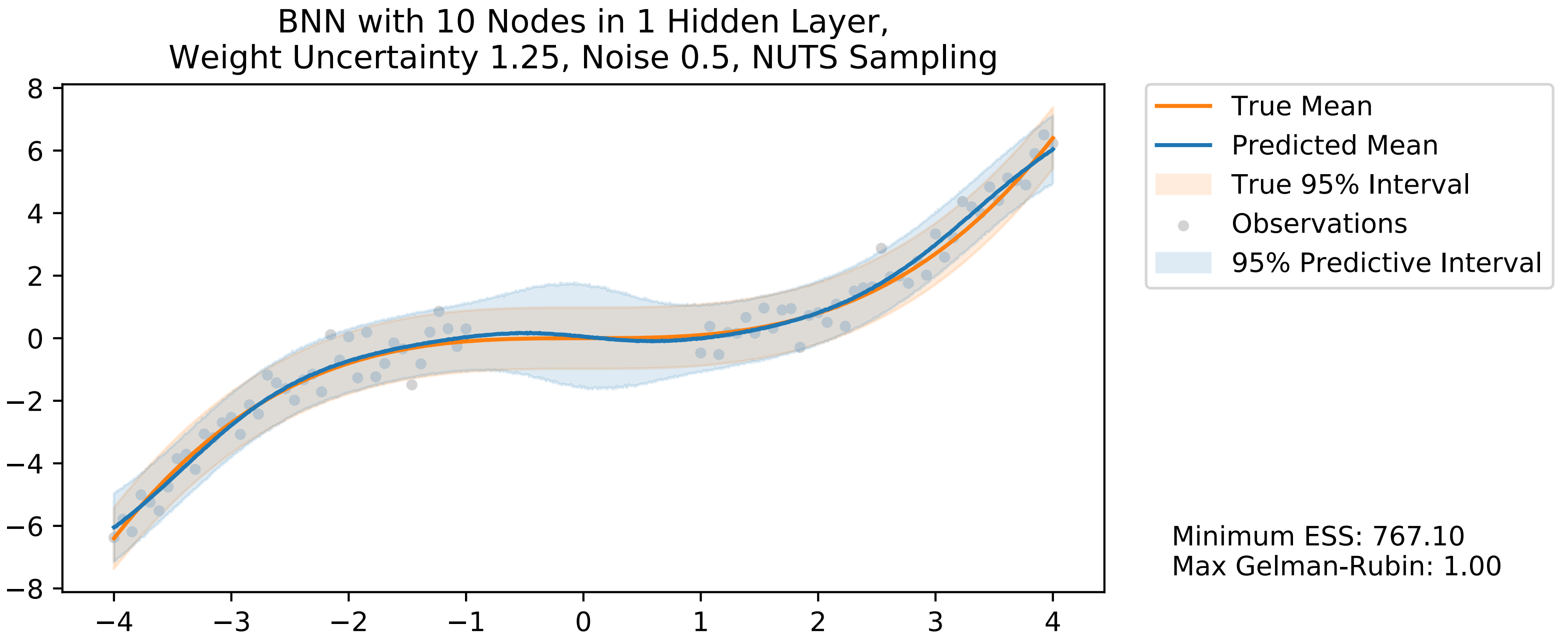
Similar to the previous example, a BNN may demonstrate bias by being too simple architecturally. That is difficult to demonstrate for a dataset generated by a cubic function, which can be described by just 4 points. Still, if we reduce the number of nodes in our network we can observe bias in the resulting posterior predictive. The sampler does not converge in this setup, which is fine since we are looking for examples of an improper model, rather than a correct one.



Minimum ESS: 1.00
Max Gelman-Rubin: 18.35

Link Between Prior and Network Architecture

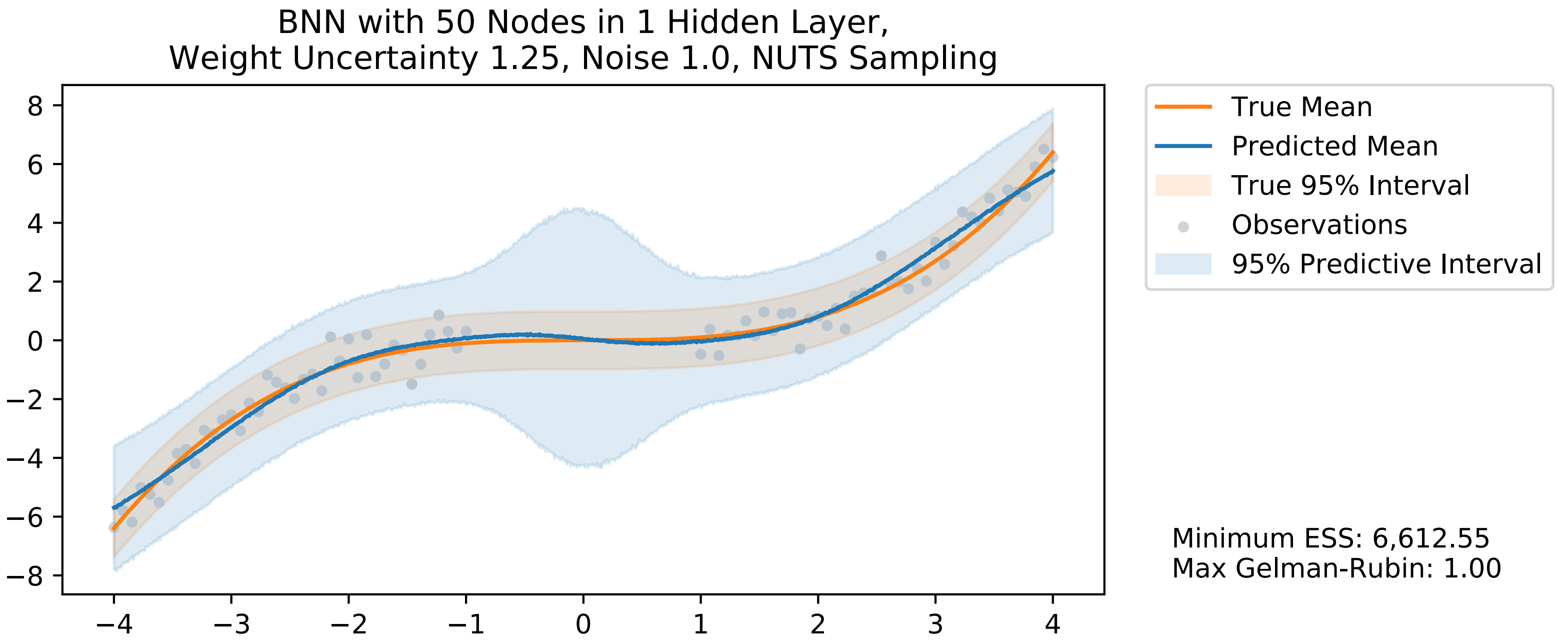
The appropriate level of the prior variance depends on the network complexity. For instance, a simpler network with 10 nodes and the same prior variance as our original benchmark model predicts much lower epistemic uncertainty. Therefore, the prior has to be selected for each particular network configuration.



Minimum ESS: 767.10
Max Gelman-Rubin: 1.00

Wrong Noise: Too High

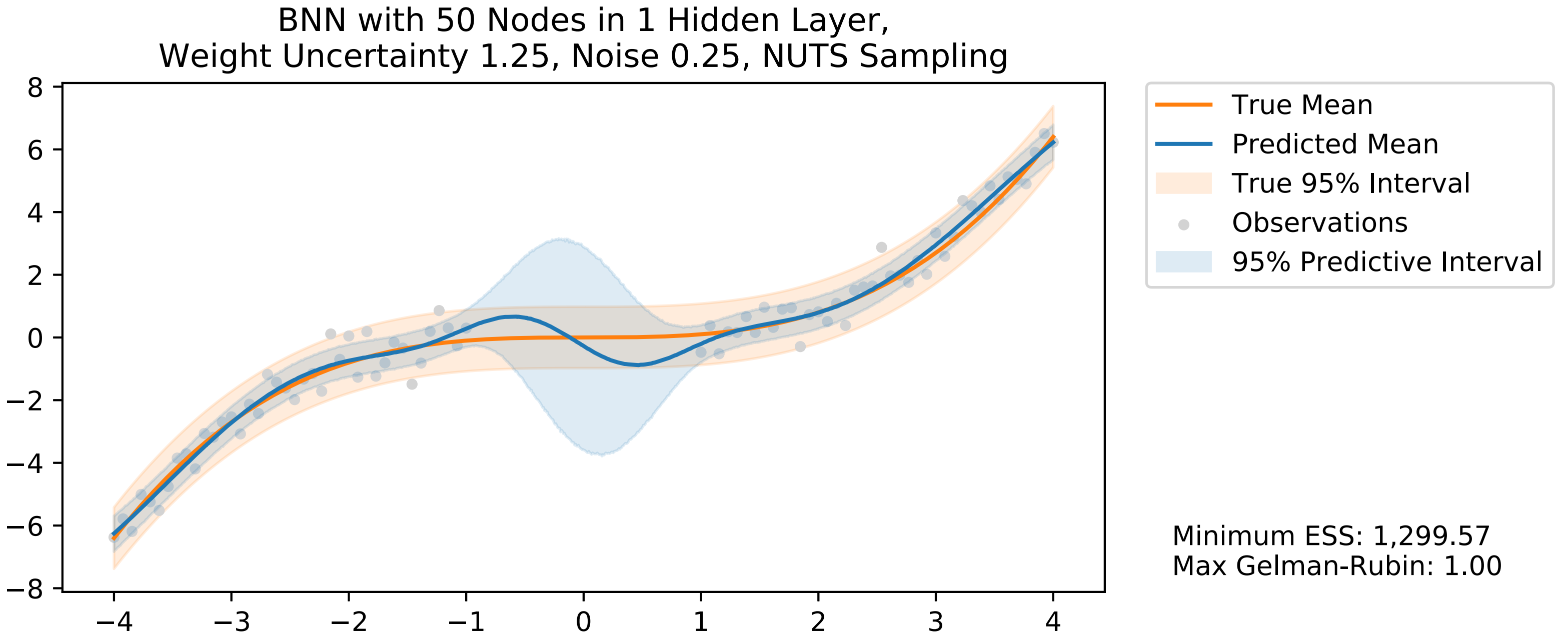
The noise in the likelihood function corresponds to aleatoric uncertainty. The effect of a wrong noise specification is that aleatoric uncertainty is captured incorrectly. In the model below the noise is still Gaussian, but has a higher variance than the true noise in the data:



This case might be a good candidate for later recalibration. Alternatively, one could find ways for the network to learn the noise from the data or put an additional prior on the variance of the noise.

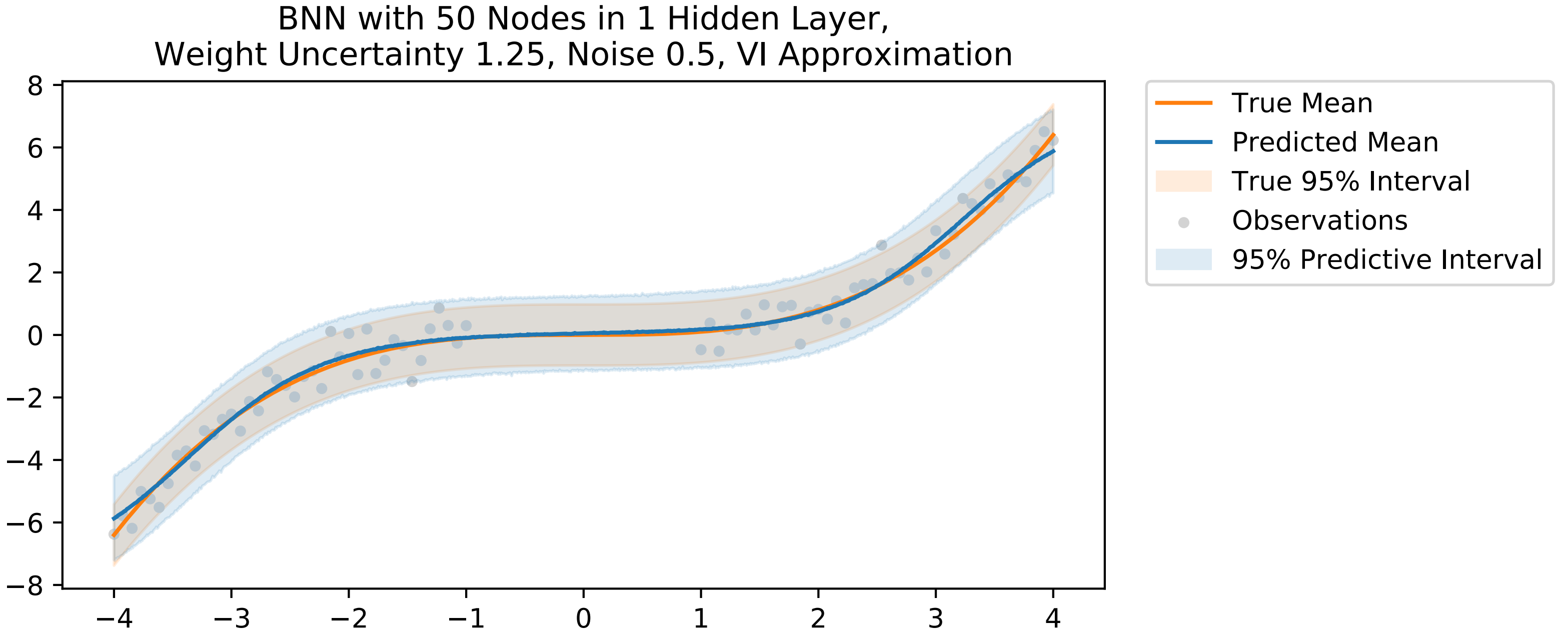
Wrong Noise: Too Small

Similarly, if the noise is too small, the resulting aleatoric uncertainty captured by the posterior predictive will be unrealistically low:



Approximate Inference

Using approximate methods of inference is also likely to lead to a miscalibrated posterior predictive. In the example below, Variational Inference with reparametrization on a network with 50 nodes produces too low epistemic uncertainty and slightly larger aleatoric uncertainty. Recalibration might turn out to be useful for correcting the latter.



Existing Work

Importance of Uncertainty Assessment

Scenarios: Correct uncertainty estimation is crucial in multiple machine learning and statistical modeling applications. [Tagasovska & Lopez-Paz, 2018] list numerous scenarios in which correct accounting for prediction uncertainty is paramount for the usefulness of a forecasting procedure. These include: dealing with anomalies (outliers, out-of-distribution test examples, adversarial examples), assessing when to delegate a prediction to a human or simply comparing and interpreting models.

Apart from the assessment of overall uncertainty, it is crucial in many settings to be able to distinguish between the reducible (**epistemic** or statistical) and irreducible (**aleatoric** or systematic) uncertainty [Hullermeier & Waegeman, 2019]. See also [Der Kiureghian & Ditlevsen, 2009] for a discussion of sources of uncertainty.

Calibration: The need for high-quality measurement of uncertainty in modeling naturally entails a question of assessing how suited different models are for representing the uncertainty. The ability of a model to properly capture uncertainty is referred to as model **calibration**, while **miscalibration** is the discrepancy between model (subjective) forecasts and (empirical) long-run frequencies in the frequentist paradigm [Lakshminarayanan et al., 2017]. Importantly, predictions may be accurate but still miscalibrated, i.e. the model might correctly label the test data, but produce wrong conclusions on how frequent, given the input data, a particular label should be. Indeed, despite the tremendous advances in prediction accuracy achieved with neural networks, many of the modern machine learning models turn out to be miscalibrated [Guo et al. 2017].

Improving Calibration of Neural Networks

Research directions: The Bayesian approach is believed to provide a general and principled framework for measuring uncertainty in machine learning [Gal, 2016]. By putting priors on weights of the network, Bayesian neural networks produce predictive posterior distributions allowing for assessment of uncertainty related to forecasts (see e.g. [McKay, 1992]; [Neal, 1995]). Unfortunately, Bayesian inference methods in deep learning are almost always approximate and computationally expensive. As a result, the research community focused on improving the efficiency of obtaining Bayesian solutions, approximating Bayesian results or bypassing the burden of Bayesian inference in general.

Methods: A variety of techniques to efficiently obtain correct uncertainty estimates for neural networks have been studied. These include Dropout [Gal & Ghahramani, 2016]; [Phan et al., 2019]; [Maeda, 2014], different types of ensembling [Tomczak et al., 2018], [Pearce et al., 2019], extending BNNs with latent variables [Depeweg et al., 2018], probabilistic backpropagation [Hernandez-Lobato & Adams, 2015], Laplace approximation [Foong et al., 2019], simultaneous quantile regression [Tagasovska & Lopez-Paz, 2019] or stochastic weight averaging [Maddox et al., 2019]. [Loquericio et al., 2019] presented a framework for uncertainty estimation of neural network predictions using a combination of Monte-Carlo sampling and Gaussian belief networks claiming that the framework meets three postulates: **(1)** it is independent of the network architecture, **(2)** it does not require changes in the optimization process and **(3)** it can be applied to already trained architectures.

Contribution: The Calibration Algorithm

Contribution of the Reviewed Paper

Proposition: [Kuleshov et al., 2018] propose a simple **calibration algorithm** for regression. The method is heavily inspired by Platt scaling [Platt, 1999], which consists of training an additional sigmoid function to map potentially non-probabilistic outputs of a classifier to empirical probabilities. Originally Platt scaling was proposed for calibration of support vector machines but subsequently extended to other classification algorithms.

Unique contribution: The study contributes to the subject literature by:

- extending the recalibration methods used so far for classification tasks (Platt scaling) to regression;
- proposing a procedure that is universally applicable to any regression model, be it Bayesian or frequentist and does not require modification of the model. Instead, the algorithm is applied to the output of any existing model in a postprocessing step.

Claim: The authors claim that the method outperforms other techniques by consistently producing well-calibrated forecasts, given enough i.i.d. data. Based on their experiments, the procedure also improves predictive performance in several tasks, such as time-series forecasting and reinforcement learning.

Technical Details

Recalibration of Regression Models

The algorithm has two main steps (from Algorithm 1 listing in the paper):

1. Construct a recalibration dataset D:

   $$\mathrm{D} = \left\{ \left(\left[H\left(x_t\right) \right]\left(y_t\right), \hat{P}\left(\left[H\left(x_t\right) \right]\left(y_t\right) \right) \right) \right\}_{t=1}^{T}$$

   where:

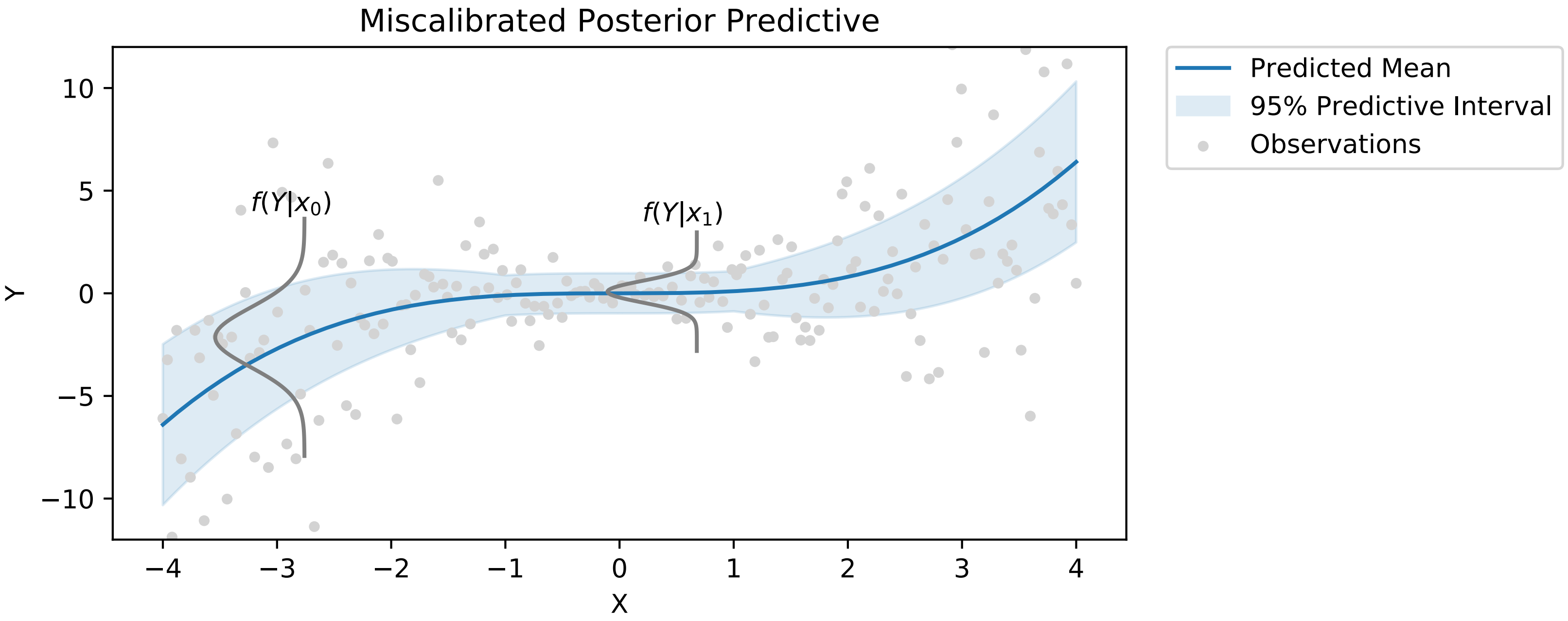
   - T is the number of observations
   - $H(x_t)$ is a CDF of the posterior predictive evaluated at x_t
   - $H(x_t)(y_t)$ is the predicted quantile of y_t
   - $\hat{P}(p)$ is the empirical quantile of y_t, computed as:

   $$\hat{P}(p) = \left| \left\{ y_t \mid \left[H\left(x_t\right) \right]\left(y_t\right) < p, t = 1 \ldots T \right\} \right| / T$$

2. Train a model R (e.g. isotonic regression) on D.

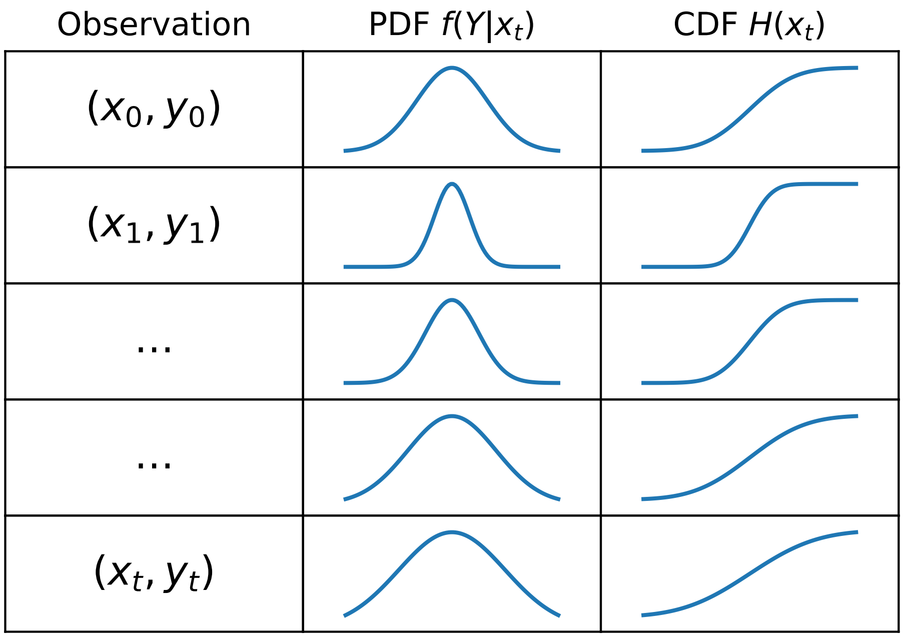
The Algorithm Step-by-Step

Suppose we have the following hypothetical posterior predictive, which is heteroscedastic and is underestimating uncertainty. For each value of the covariate X, the posterior predictive provides us with a conditional distribution $f(Y \mid X)$:



Step 1: Construct a Recalibration Dataset

The first step of the calibration algorithm is to obtain predictive conditional distributions for each X in the dataset. If no closed-form is available we simulate the posterior predictive based on the samples of the posterior:

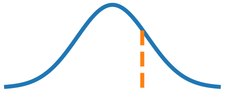
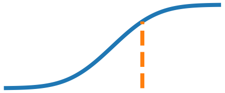
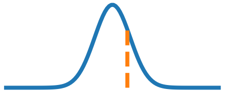
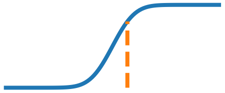
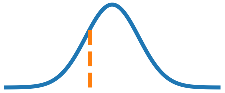
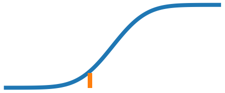
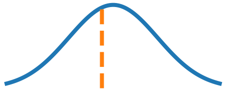
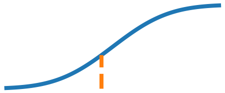
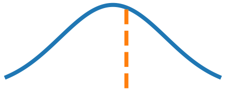
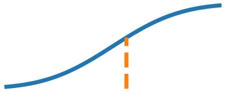


Observation	PDF $f(Y \mid x_t)$	CDF $H(x_t)$
(x_0, y_0)		
(x_1, y_1)		
...		
...		
(x_t, y_t)		

An alternative, more commonly used notation for $H(x_t)$ is F_t (a CDF).

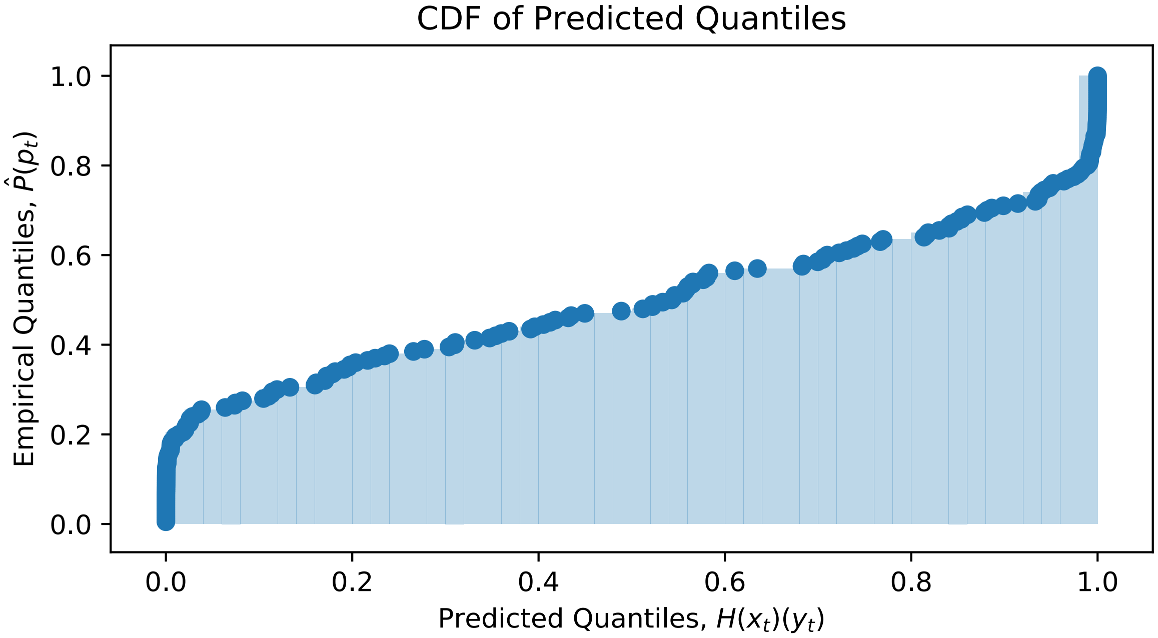
Step 1a: Compute the Predicted Quantiles

The observed Y of each data point (denoted by y_t) falls somewhere within those conditional distributions. We evaluate the conditional CDFs at each observed value of the response Y to obtain the predicted quantiles. In the absence of analytical form, we simply count the proportion of samples that are less than y_t. This gives us the estimated quantile of y_t at x_t in the posterior predictive distribution:

Observation	PDF $f(Y \mid x_t)$	CDF $H(x_t)$	$H(x_t)(y_t)$
(x_0, y_0)			0.8
(x_1, y_1)			0.8
...			0.2
...			0.4
(x_t, y_t)			0.6

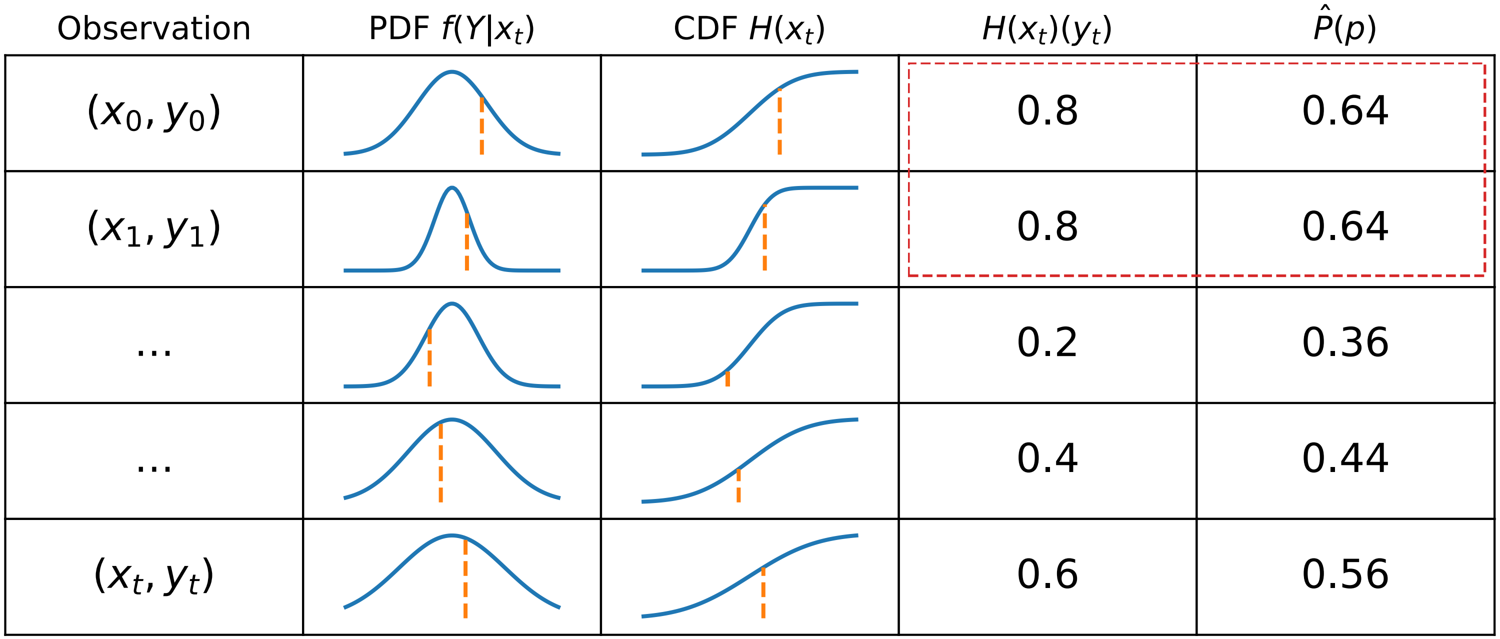
Step 1b: Estimate the Empirical Quantiles

We next find the empirical quantiles, which are defined as the proportion of observations that have lower quantile values than that of the current observation. This is equivalent to finding the empirical CDF of the predicted quantiles. The mapping of predicted quantiles to empirical quantiles will form a recalibration dataset:



Step 1c: Form a Recalibration Dataset

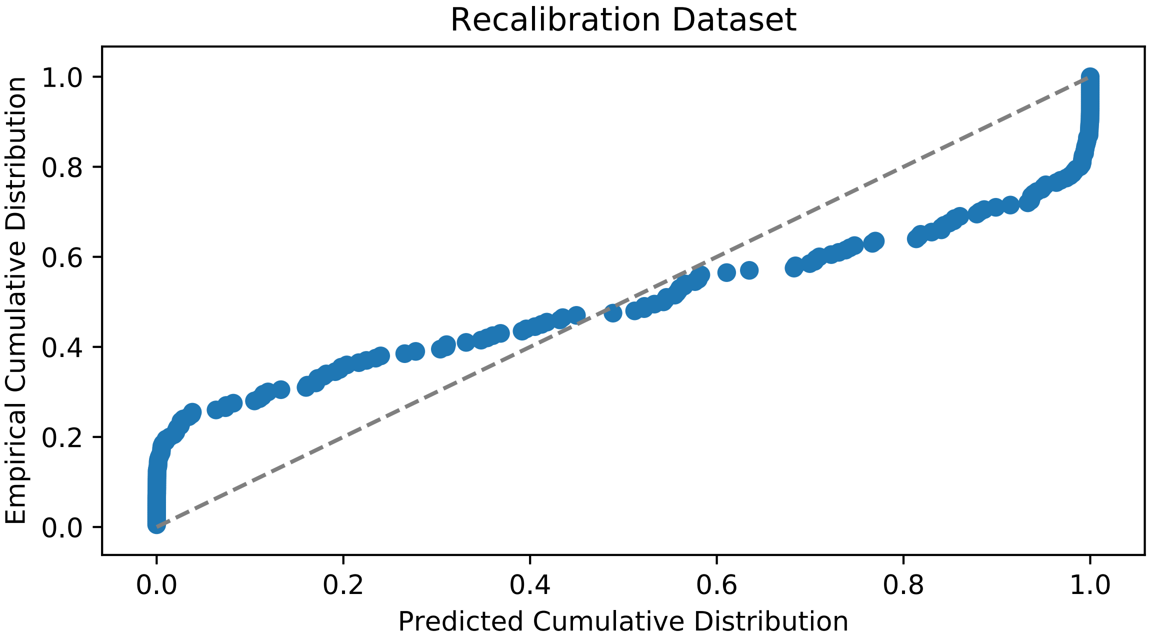
The mapping is obtained for all observations in the dataset. Note that in this example the first two observations have different conditional distributions, but the same values of the predicted and empirical quantiles. The calibration procedure doesn't distinguish between such cases:



Observation	PDF $f(Y\|x_t)$	CDF $H(x_t)$	$H(x_t)(y_t)$	$\hat{P}(p)$
(x_0, y_0)			0.8	0.64
(x_1, y_1)			0.8	0.64
…			0.2	0.36
…			0.4	0.44
(x_t, y_t)			0.6	0.56

Step 1c: Form a Recalibration Dataset

The inverse S-curve of the recalibration dataset in our illustration is characteristic of a posterior predictive that underestimates uncertainty. The diagonal line denotes perfect calibration:



Step 2: Train a Model

We then train a model (e.g. isotonic regression) on the recalibration dataset and use it to output the actual probability of any given quantile or interval. Here the 95% posterior predictive interval corresponds to a much narrower calibrated interval:

Predicted quantiles	Calibrated quantiles
0.025	0.224
0.5	0.477
0.975	0.776

Ideally, the model should be fit on a separate calibration set in order to reduce overfitting. Alternatively, multiple models can be trained in a way similar to cross-validation:

- use $K - 1$ folds for training
- use 1 fold for calibration
- at prediction time, the output is the average of K models

Detailed Steps (Part 1)

Concretely, for models without closed form posterior predictive CDF, the calibration algorithm is restated as:

1. Generate N samples from the posterior, $\theta = \left\{\theta_n, n = 1\ldots N\right\}$.
2. For each observation, $t \in 1\ldots T$
    - Generate N samples of posterior predictive, s_{t_n}, from θ and evaluated at x_t
    - Let p_t be the quantile of y_t. Estimate the quantile of y_t as

$$p_t = \left[\hat{H}(x_t)\right](y_t) = \left|\left\{s_{t_n} \mid s_{t_n} \le y_t, n = 1\ldots N\right\}\right|/N$$

3. For each t
    - calculate $\hat{P}\left(\left[\hat{H}\left(x_t\right)\right]\left(y_t\right)\right) = \hat{P}\left(p_t\right)$ as

$$\hat{P}\left(p_t\right) = \left|\left\{p_u \mid p_u < p_t, u = 1\ldots T\right\}\right|/T$$

That is, find the proportion of observations that have lower quantile values than that of the current observation.

Detailed Steps (Part 2)

4. Construct $\mathrm{D} = \left\{ \left(\left[\widehat{H(x_t)} \right] (y_t), \hat{P} \left(\left[\widehat{H(x_t)} \right] (y_t) \right) \right) \right\}_{t=1}^{T}$
5. Train calibration transformation using D via isotonic regression (or other models). Running prediction on the trained model results in a transformation R, $[0, 1] \rightarrow [0, 1]$. We can compose the calibrated model as $R \circ H(x_t)$.
6. To find the calibrated posterior predictive or confidence intervals, we need to remap the original upper and lower limits. For example, the upper limit $y_{t\ high}$ is mapped to the calibrated value $y'_{t\ high}$ as:

$$y'_{t\ high} = \left[H(x_t) \right]^{-1} \left(R^{-1} \left\{ \left[H(x_t) \right] (y_{t\ high}) \right\} \right)$$

Making Predictions with the Calibrated Model

In order to make predictions with the calibrated model, we need to construct its posterior predictive. This can be done by applying the equation in step 6 to all uncalibrated posterior predictive samples. The resulting set of samples reflects the calibrated posterior predictive distribution.

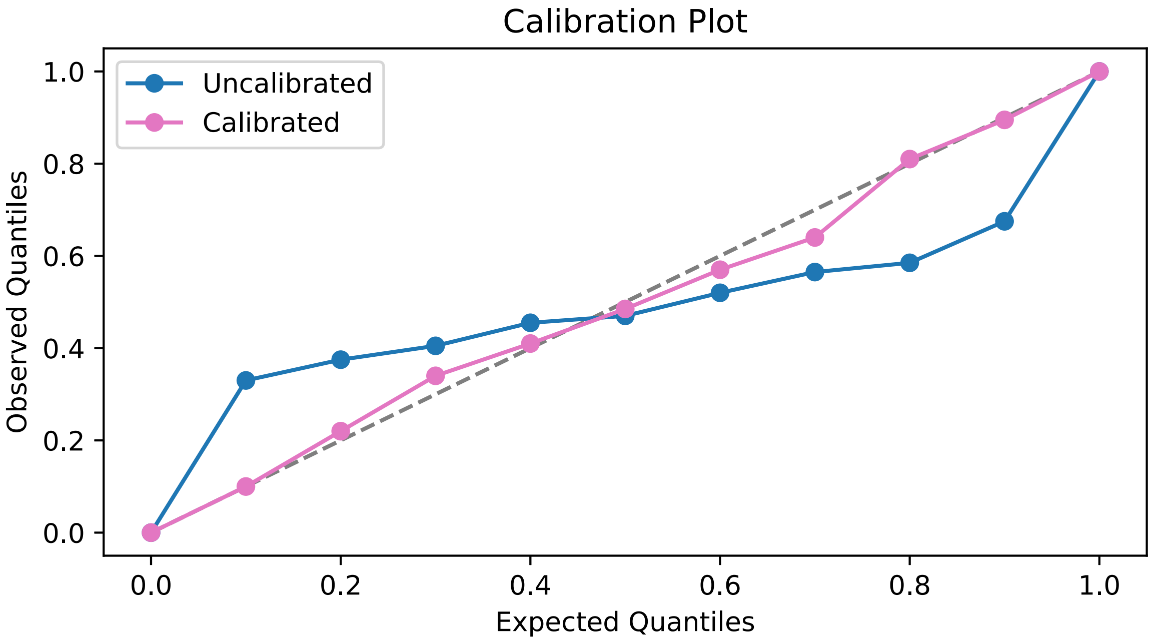
Point estimates, like the mean, can then be computed for the calibrated posterior predictive.

To directly sample from the calibrated posterior predictive, we can also use the inverse CDF method. Recall that the CDF is $R \circ H\left(x_t\right)$, and the inverse is $H\left(x_t\right)^{-1} \circ R^{-1}$. We can therefore sample from a uniform distribution and then apply the calibrated inverse empirical CDF to obtain further samples.

In our implementation, we obtain R^{-1} by training isotonic regression in reverse (swapping the calibration dataset inputs). We obtain $\left[H\left(x_t\right)\right]^{-1}$ by doing a quantile lookup from the uncalibrated posterior predictive samples with `numpy.quantile()`.

Diagnostics

As a visual diagnostic tool, the authors suggest using a calibration plot that shows the true frequency of points in each quantile compared to the predicted fraction of points in that interval. Well-calibrated models should be close to a diagonal line:



Quantitative Metrics

Several alternatives are available, each with specific advantages and disadvantages:

1. Calibration error

$$cal(F_1, y_1, \ldots, F_N, y_N) = \sum_{j=1}^{m} w_j \cdot (p_j - \hat{p}_j)^2$$

Provides a synthetic measure representing the overall *'distance'* of the points on the calibration curve from the $45\,°$ straight line. The weights (w_j) might be used to reduce the importance of intervals containing few observations. The value of 0 indicates perfect calibration. The metric is sensitive to binning.

2. Predictive RMSE

$$\sqrt{\frac{1}{N}\sum_{n=1}^{N} ||y_n - \mathrm{E}_{q(W)}[f(x_n, W)]||_2^2}$$

Measures the model *fit* to the observed data by normalizing the difference between the observations and the mean of the posterior predictive. Minimizing RMSE does not guarantee calibration of the model.

Quantitative Metrics (cont.)

3. Mean prediction interval width

$$\frac{1}{N}\sum_{n=1}^{N}\hat{y}_n^{high} - \hat{y}_n^{low},$$

where $\hat{y}_n^{high}$ and $\hat{y}_n^{low}$ are respectively the 97.5 and 2.5 percentiles of the predicted outputs for x_n. The average difference between the upper and lower bounds of predictive intervals evaluated for all the observations (different significance values might be used to define the predictive intervals). By itself provides information on the precision of the prediction (*confidence* with which a prediction is made) rather than calibration or miscalibration of the model. However, it may be used in conjunction with PICP.

4. Prediction interval coverage probability (PICP)

$$\frac{1}{N}\sum_{n=1}^{N}\mathbb{1}_{y_n \leq \hat{y}_n^{high}} \cdot \mathbb{1}_{y_n \geq \hat{y}_n^{low}}$$

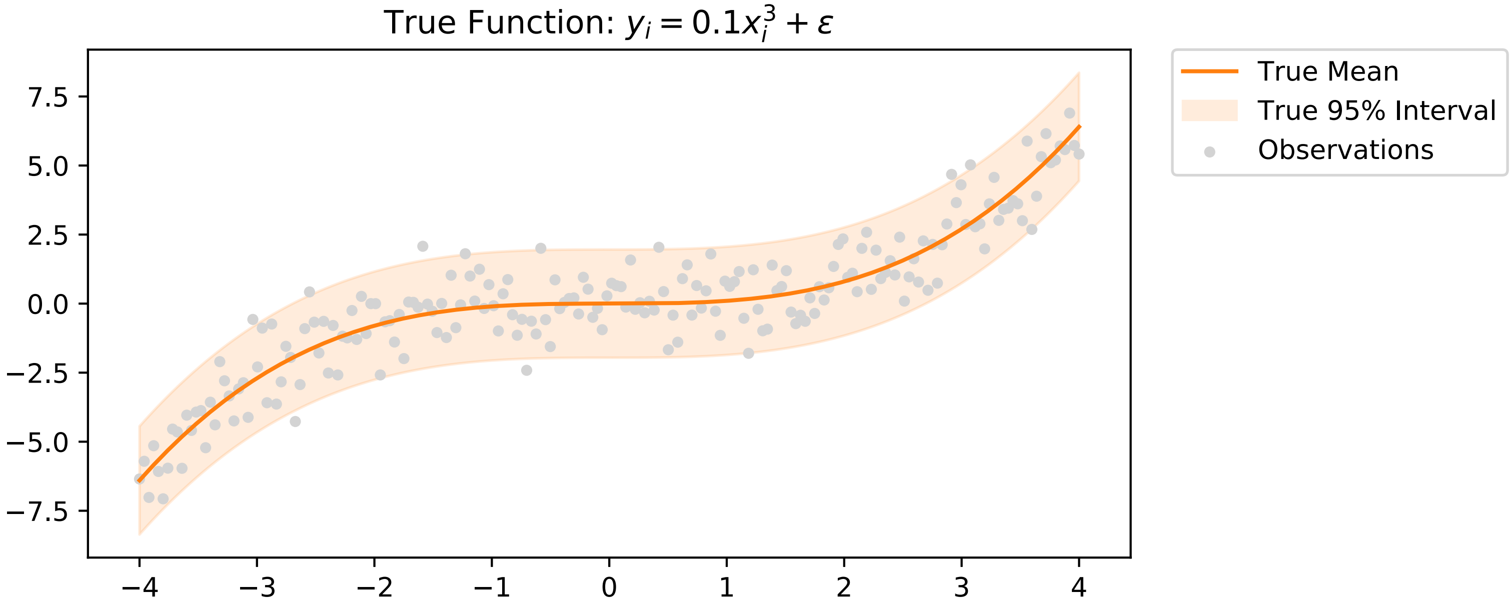
Calculates the share of observations covered by 95% (or any other selected) predictive intervals. Alignment of the PICP with the probability density assigned to the predictive interval generating it may misleadingly point to proper calibration if the true noise distribution belongs to a different family than the posterior predictive. Requires a large sample of observations.

Experiments

Homoscedastic Dataset

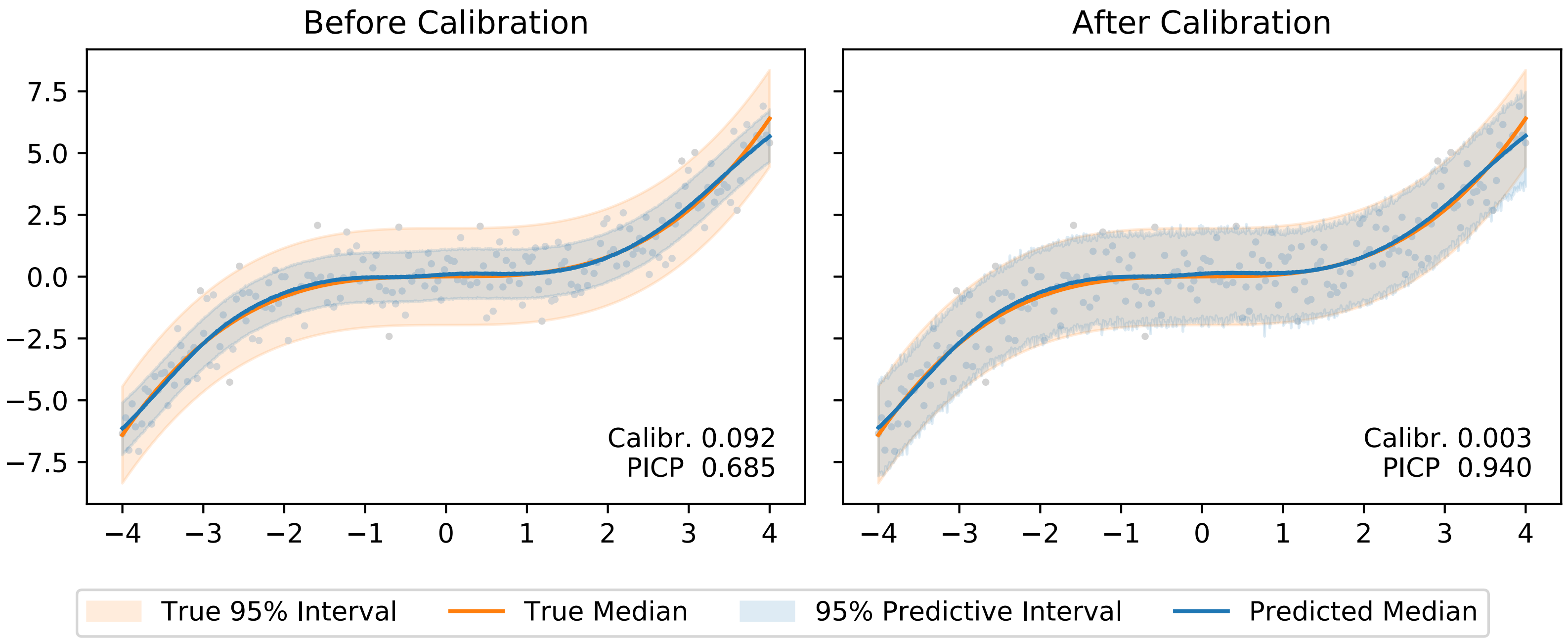
Rather than try to reproduce the experiments from the paper we chose to run the calibration algorithm on a series of synthetic datasets. This allows us to analyze the effects of the procedure on different purposefully miscalibrated posterior predictives.

The first dataset is a cubic polynomial with homoscedastic Gaussian noise:



Low Noise: Recalibration

Through sampling, we perform inference of a BNN that underestimates uncertainty due to low variance of the noise in the likelihood. The calibration model is trained on a separate hold-out dataset of the same size. After calibration, the posterior predictive aligns with the data really well:

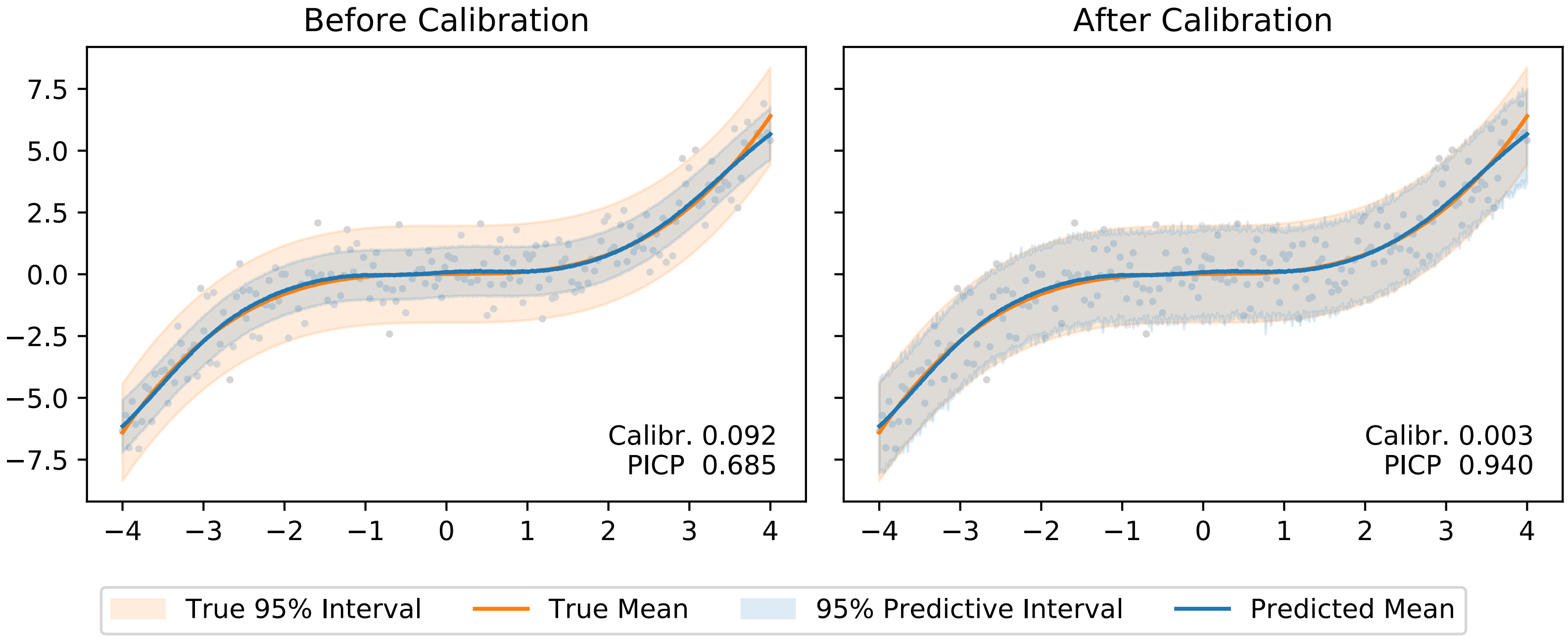


Both quantitative metrics show significant improvement. The absolute value of the calibration error depends on binning — here we use 10 equally spaced quantiles.

Point Estimates: the Median and the Mean

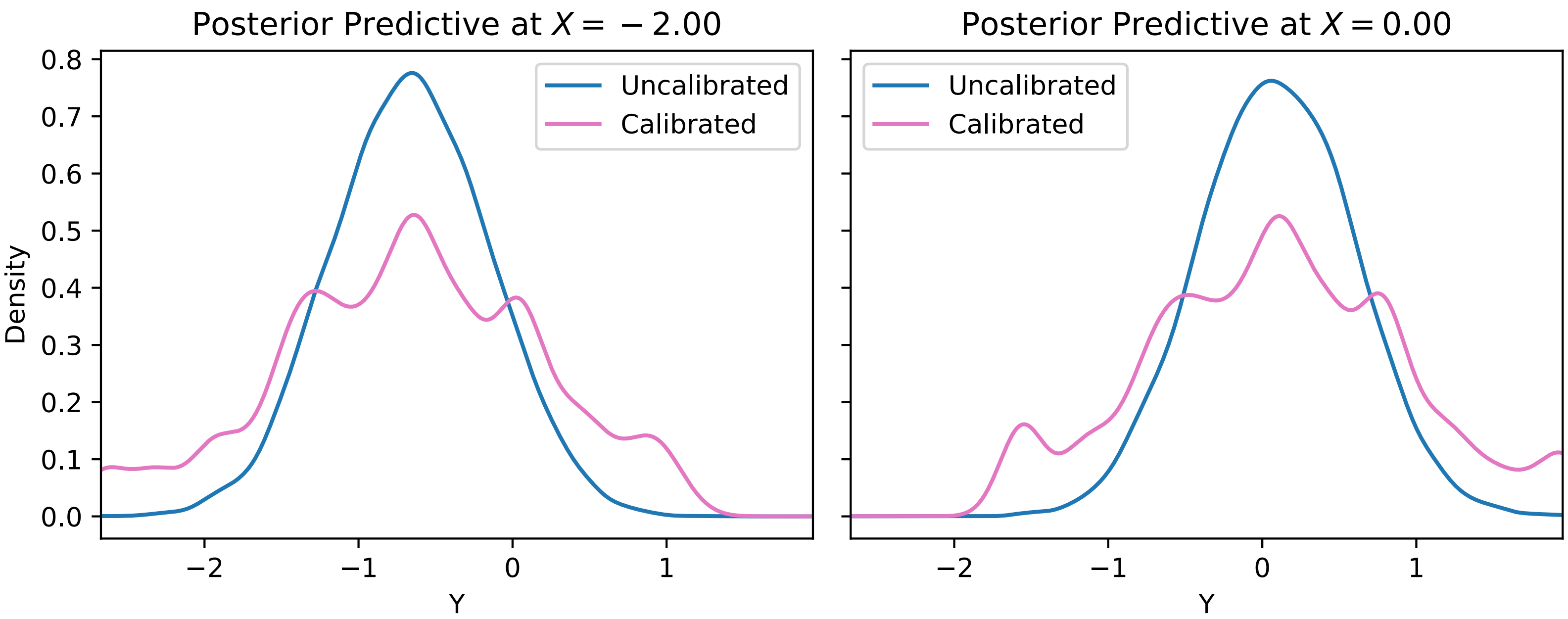
The charts below show the means of the calibrated and uncalibrated posterior predictives, together with the true mean.

The means coincide with the medians shown on the previous slide. This is expected as our data is generated with Gaussian noise, which has a symmetric distribution. For all subsequent experiments with Gaussian noise, we only show the median plots.



Conditional Distributions

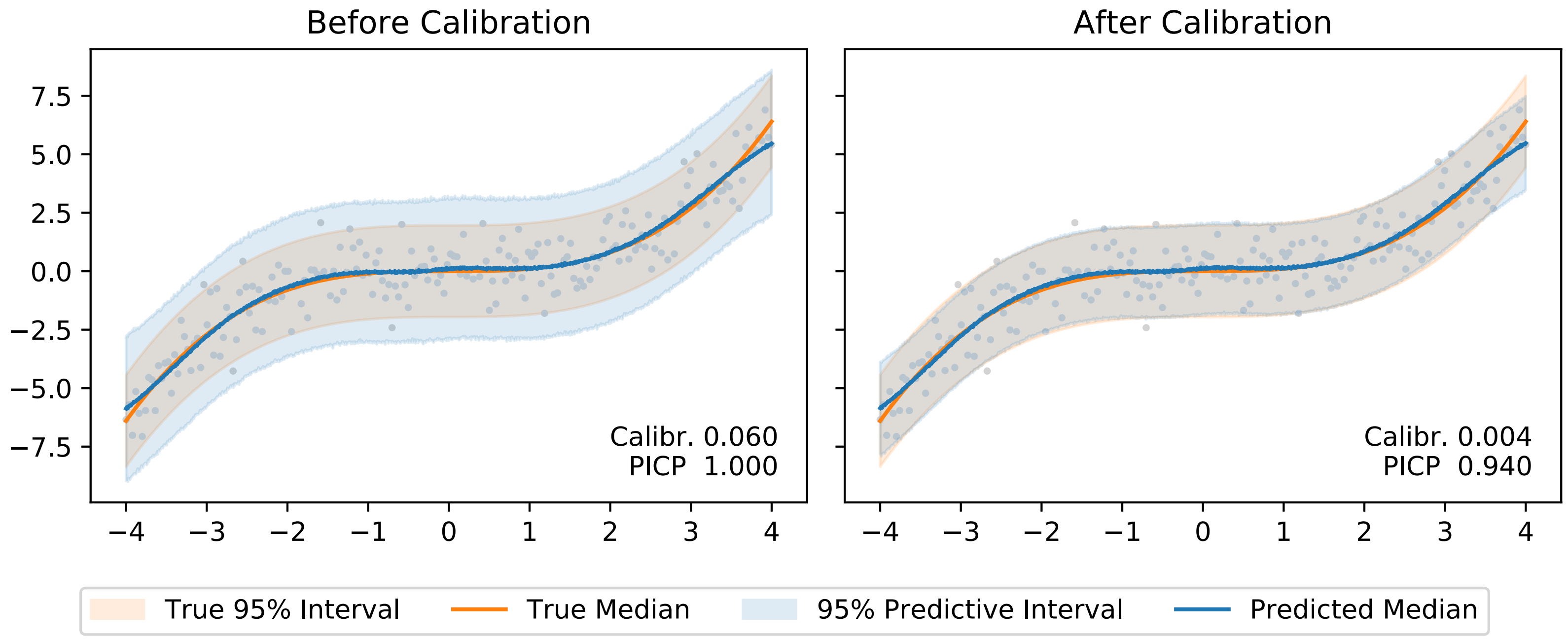
Each of the charts below corresponds to a cross-section at a specific value of X, showing the conditional posterior predictive at that point. We see that the calibrated posterior predictive in this experiment is more spread out, which agrees with the wider uncertainty bands:



The calibrated posterior predictive isn't smooth and unimodal like the uncalibrated one, which is an artifact of a small dataset. If we were to increase the number of observations, the calibrated densities produced by isotonic regression would become more smooth.

High Noise: Recalibration

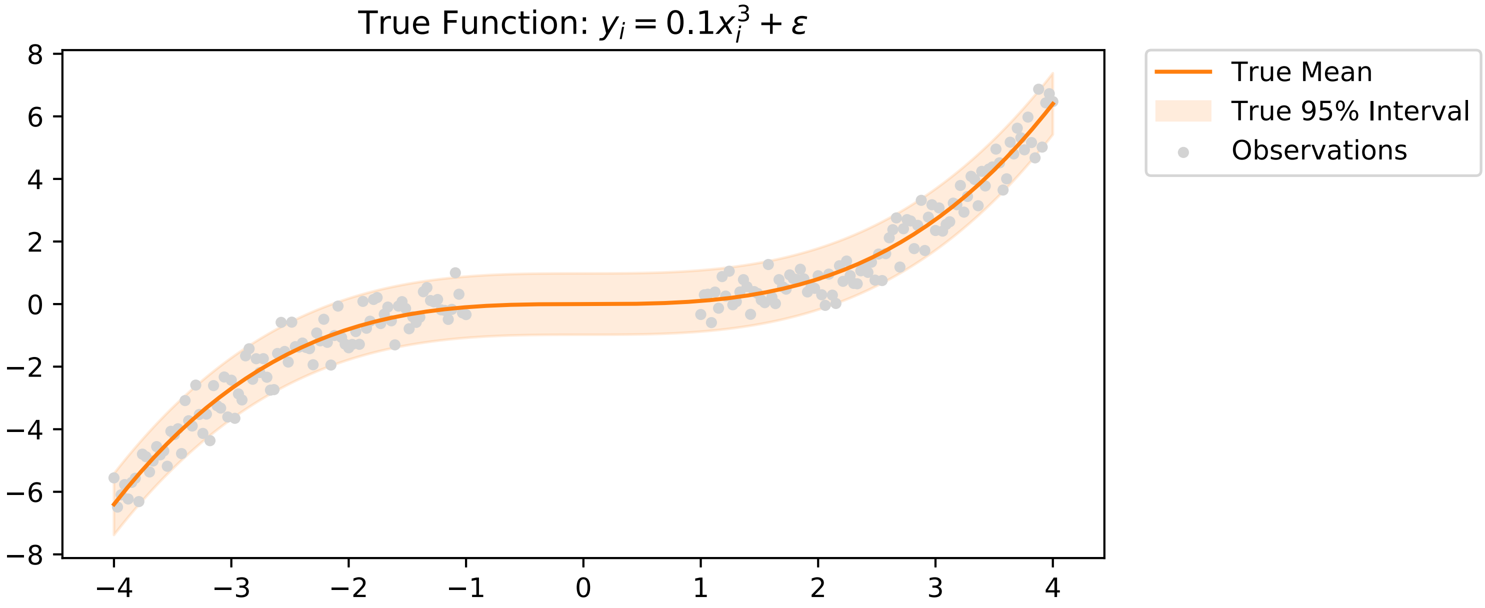
Similarly to the previous experiment, good results are obtained when we apply the calibration algorithm to a BNN that overestimates uncertainty due to the high variance of the noise in the Gaussian likelihood function. The resulting posterior predictive captures aleatoric uncertainty well:



The Predictive Interval Coverage Probability (PICP) is calculated for the 95% interval. It is improved after recalibration, i.e. 95% of the observations are covered by that interval.

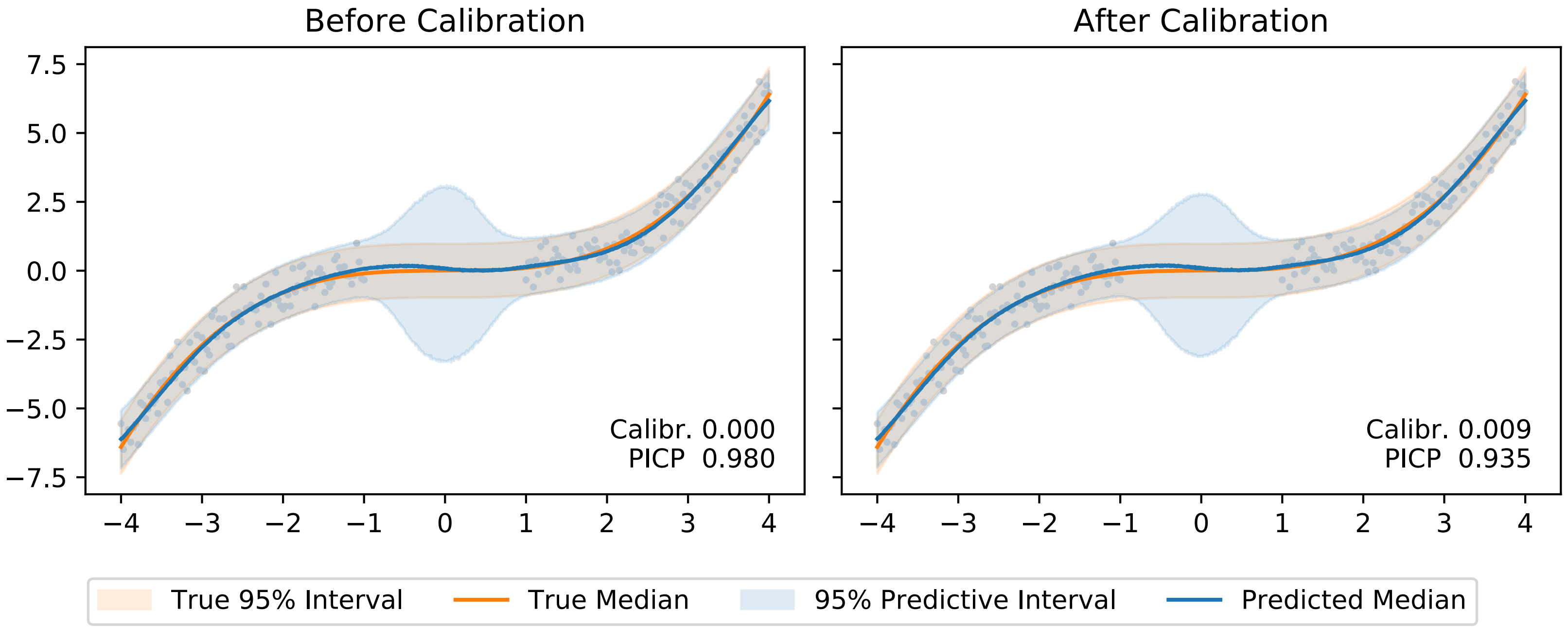
Missing or Insufficient Data

The next experimental dataset is the one we used previously in our miscalibration examples — a third-degree polynomial with a gap in the middle. This will allow us to evaluate the impact of the calibration algorithm on epistemic uncertainty for out-of-distribution examples.



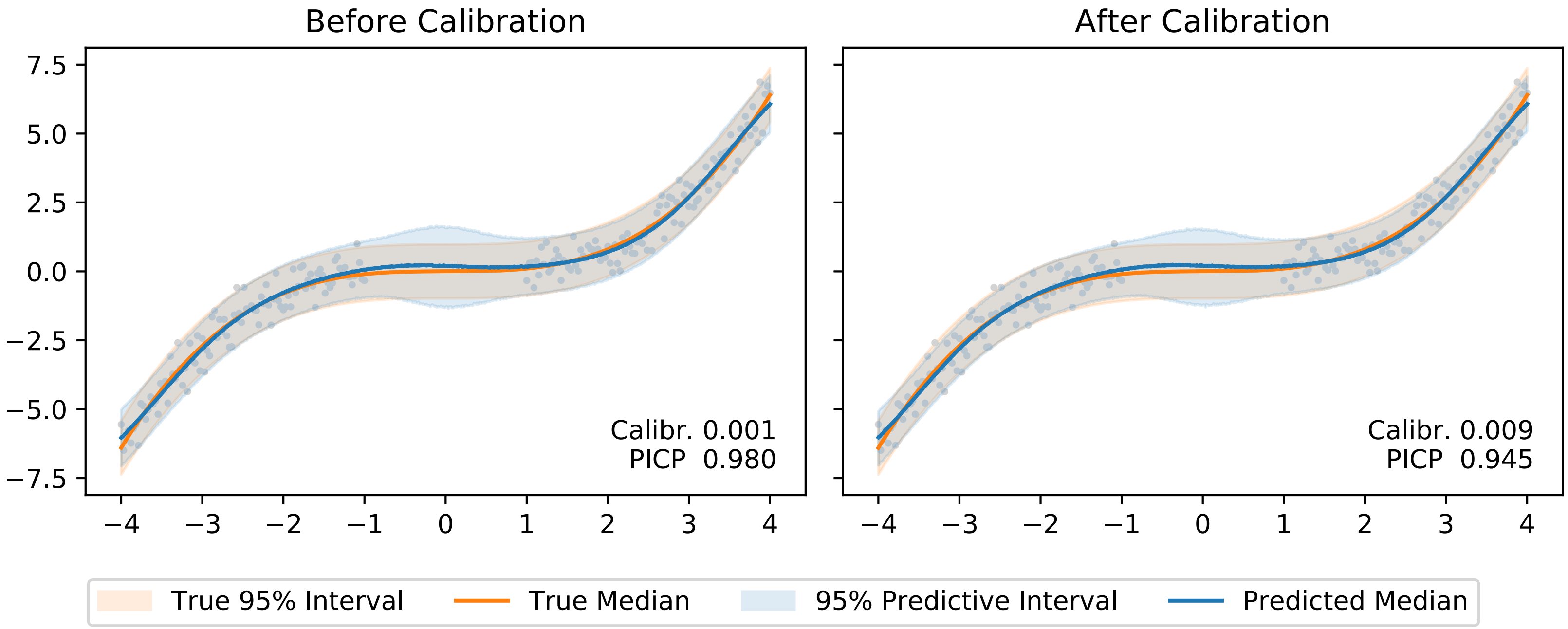
Proper Posterior: Recalibration

We first sample from a BNN that produces a reasonably good posterior predictive, both in terms of aleatoric and epistemic uncertainty. After calibration, epistemic uncertainty shrinks, but only slightly. Since our definition of "good" epistemic uncertainty is subjective, the algorithm doesn't seem to ruin a valid model:



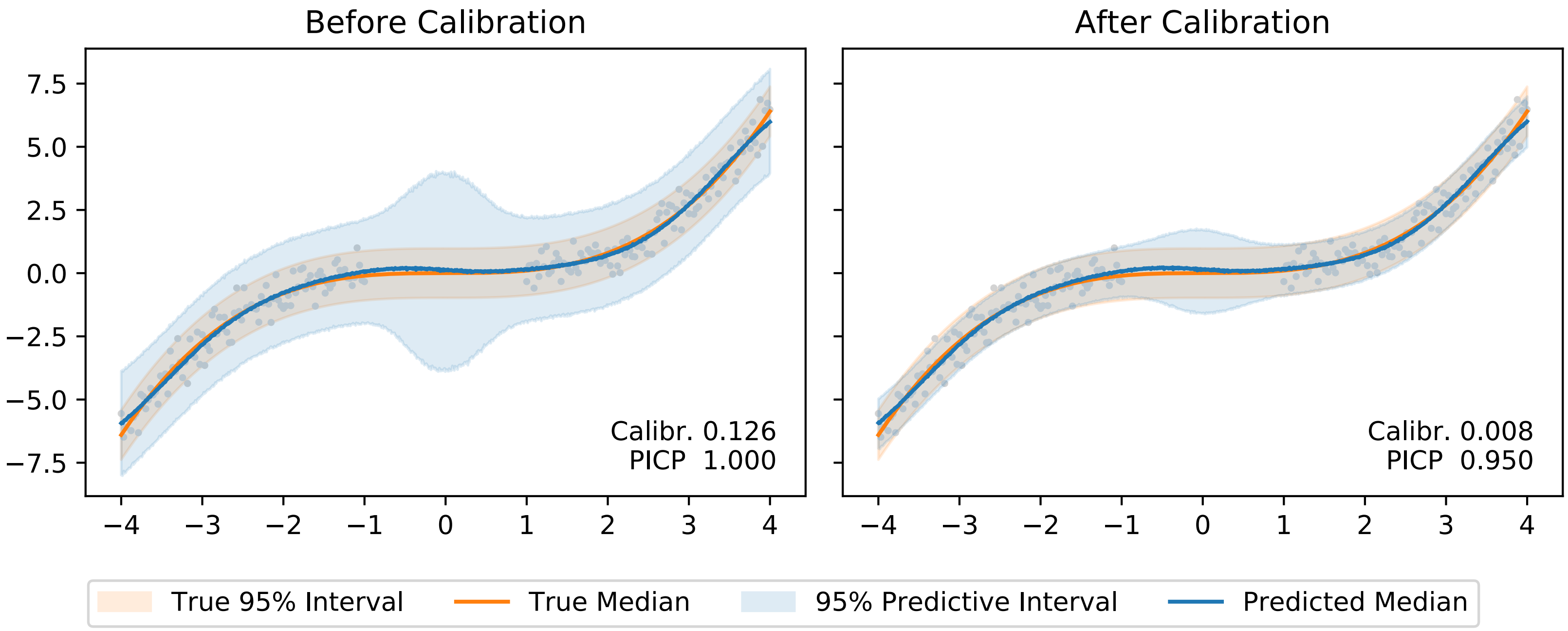
Wrong Prior: Recalibration

The same is true for the posterior predictives that either over- or underestimate uncertainty due to the wrong prior. The calibration algorithm has little effect on epistemic uncertainty:



Wrong Noise: Recalibration

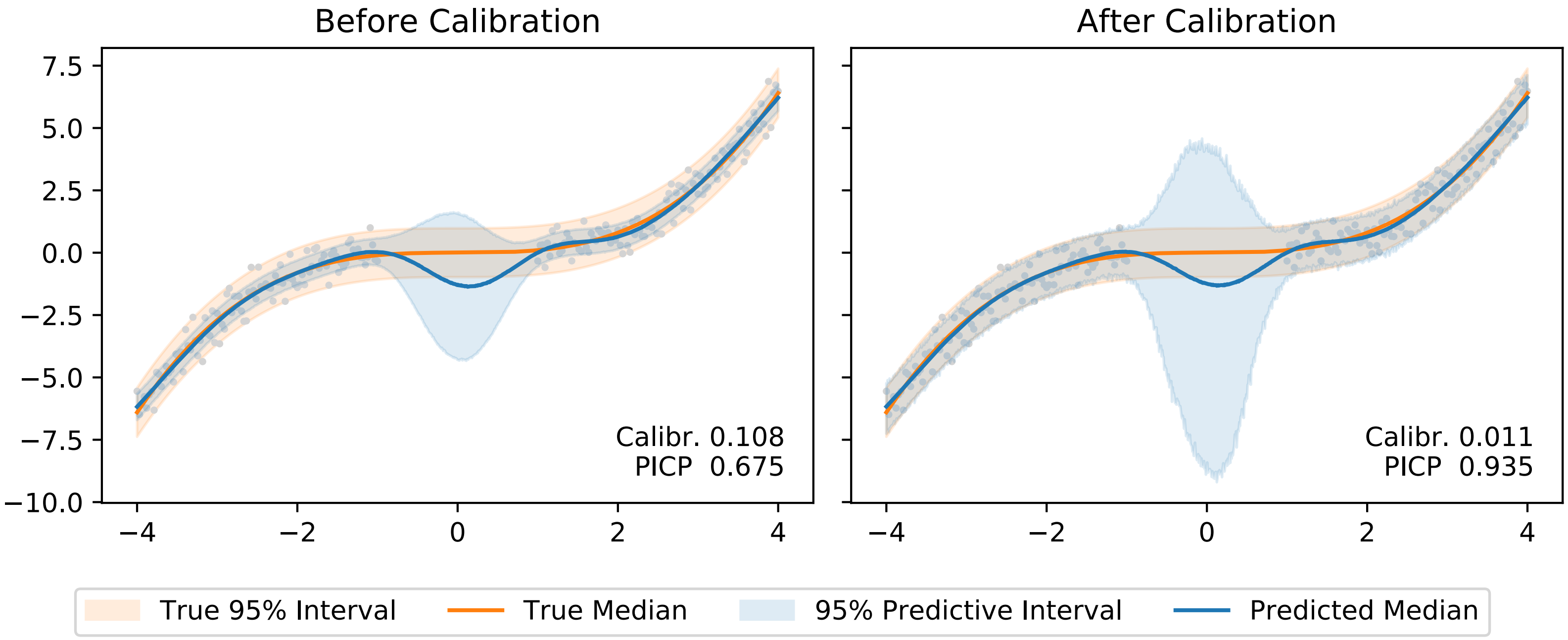
The situation changes when the noise is specified incorrectly *and* there is missing data. Since the algorithm maps predicted quantiles to empirical ones uniformly across all input space, this calibration method produces perfect aleatoric uncertainty, but reduces epistemic uncertainty drastically:



This fact is also not reflected in the metrics that we're using, which show improvement across the board.

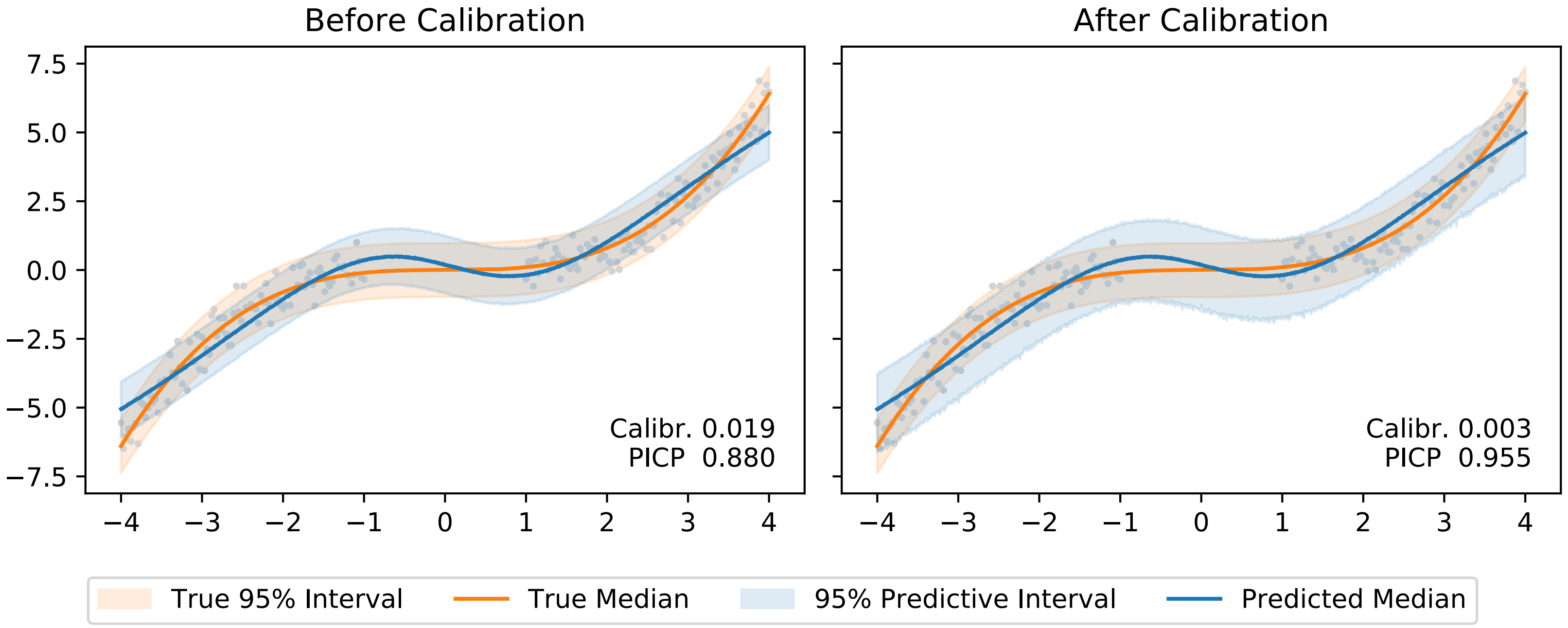
Wrong Noise: Recalibration

Analogously, if aleatoric uncertainty is underestimated by the model, after recalibration, it will be aligned with the data as much as possible, while epistemic uncertainty will be blown up. The authors of the method explicitly state that the suggested approach only works given enough i.i.d. data. Here we see one instance of how the method fails:



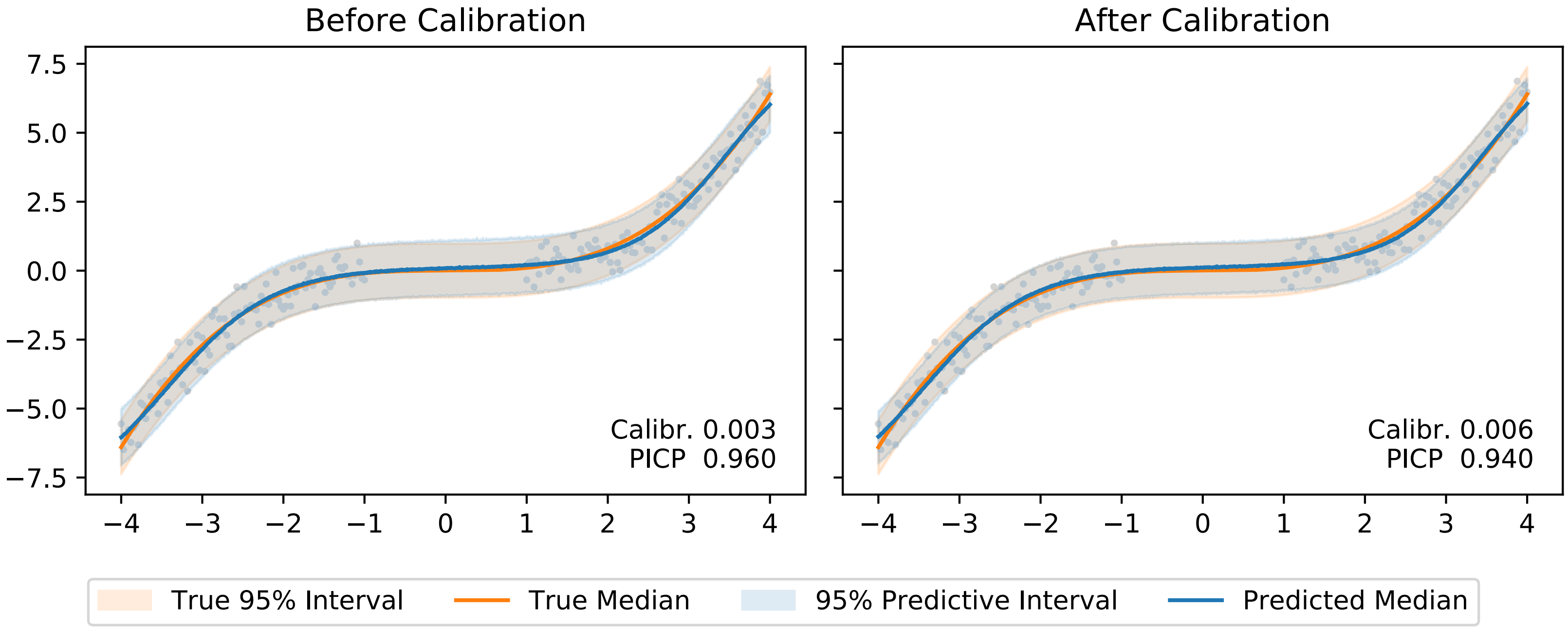
Wrong Likelihood: Recalibration

Another case of failure can be observed in the situation when there is bias, i.e. the network is not sufficiently expressive to describe the data due to a combination of the prior and the architecture. In an effort to fit the data the calibration algorithm increases uncertainty uniformly across the whole input space. A much better approach would be to change the bad model, rather than try to recalibrate it:



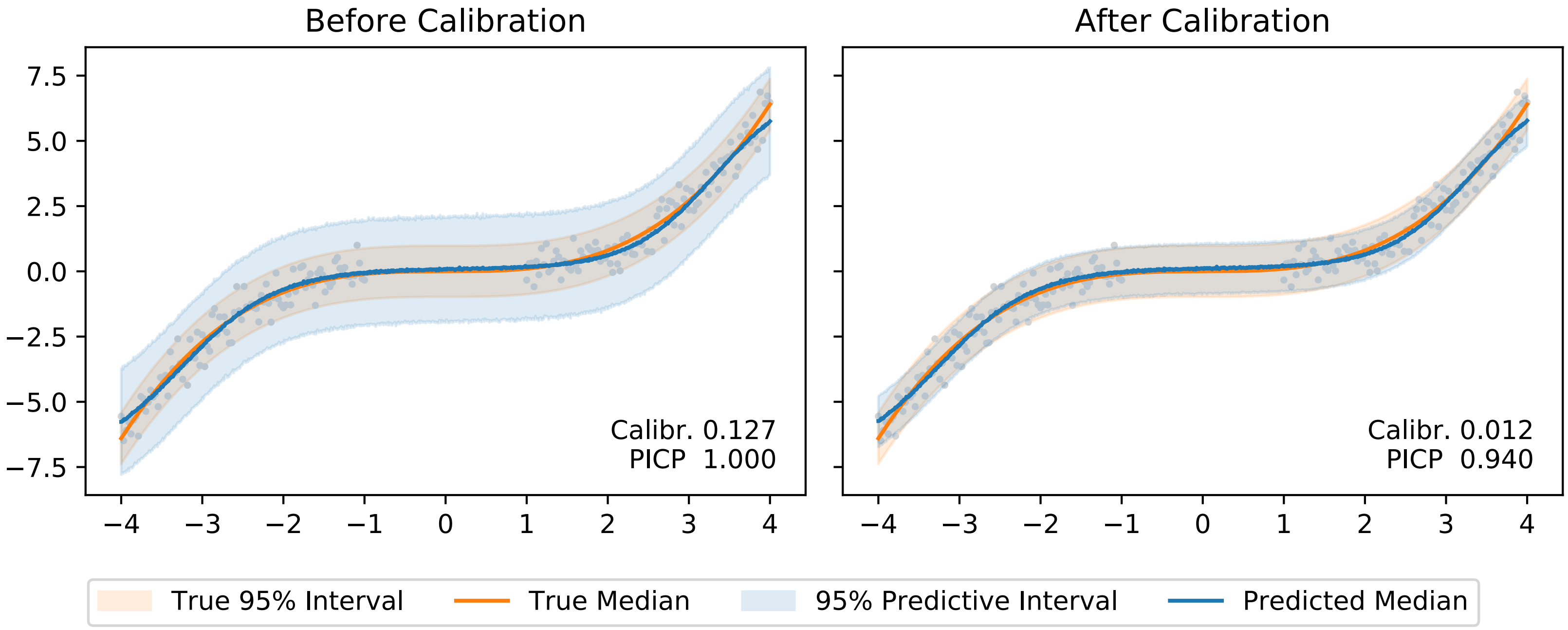
Approximate Inference: Recalibration

Correctly performed Variational Inference using isotropic Gaussians is often associated with underestimated epistemic uncertainty. The quantile-based calibration algorithm does little to such posterior predictives: both epistemic and aleatoric uncertainties mostly remain the same:



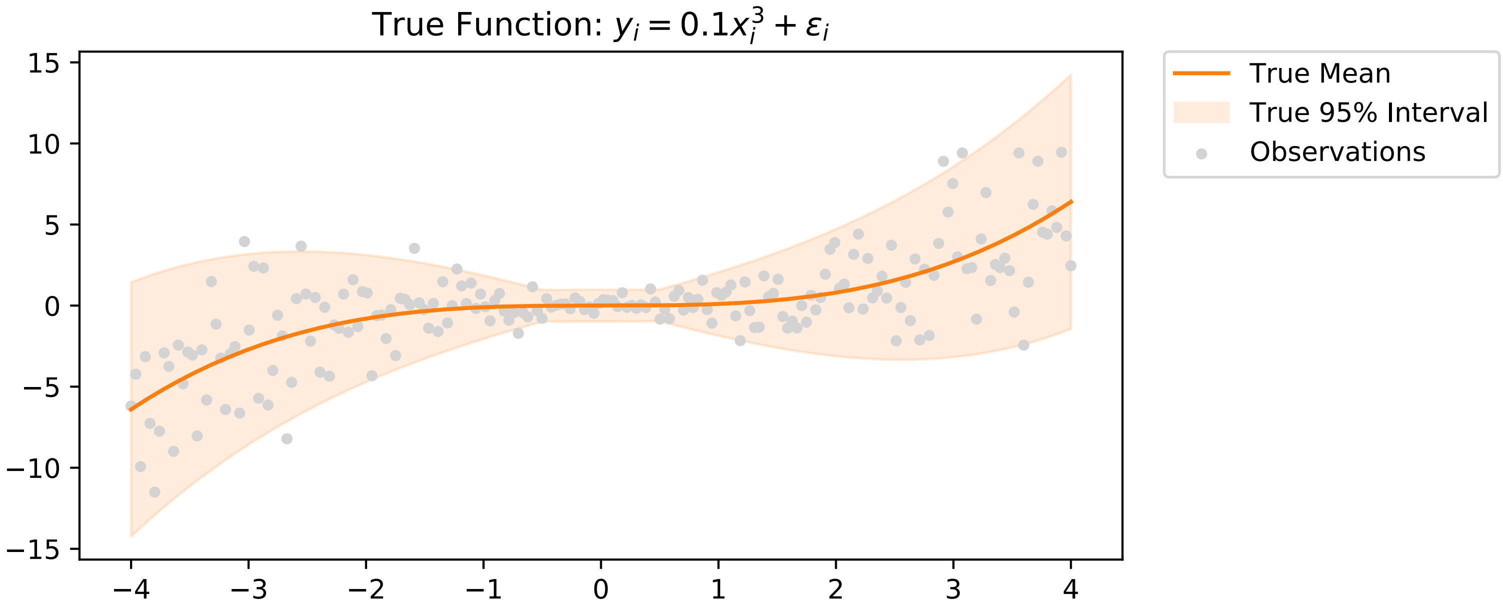
VI with Noise Misspecification: Recalibration

When the variance of the noise in the likelihood is specified incorrectly, the calibration method corrects that, aligning aleatoric uncertainty with the data. The resulting epistemic uncertainty, however, is again too low. The algorithm is unable to remedy the issues arising from variational approximation:



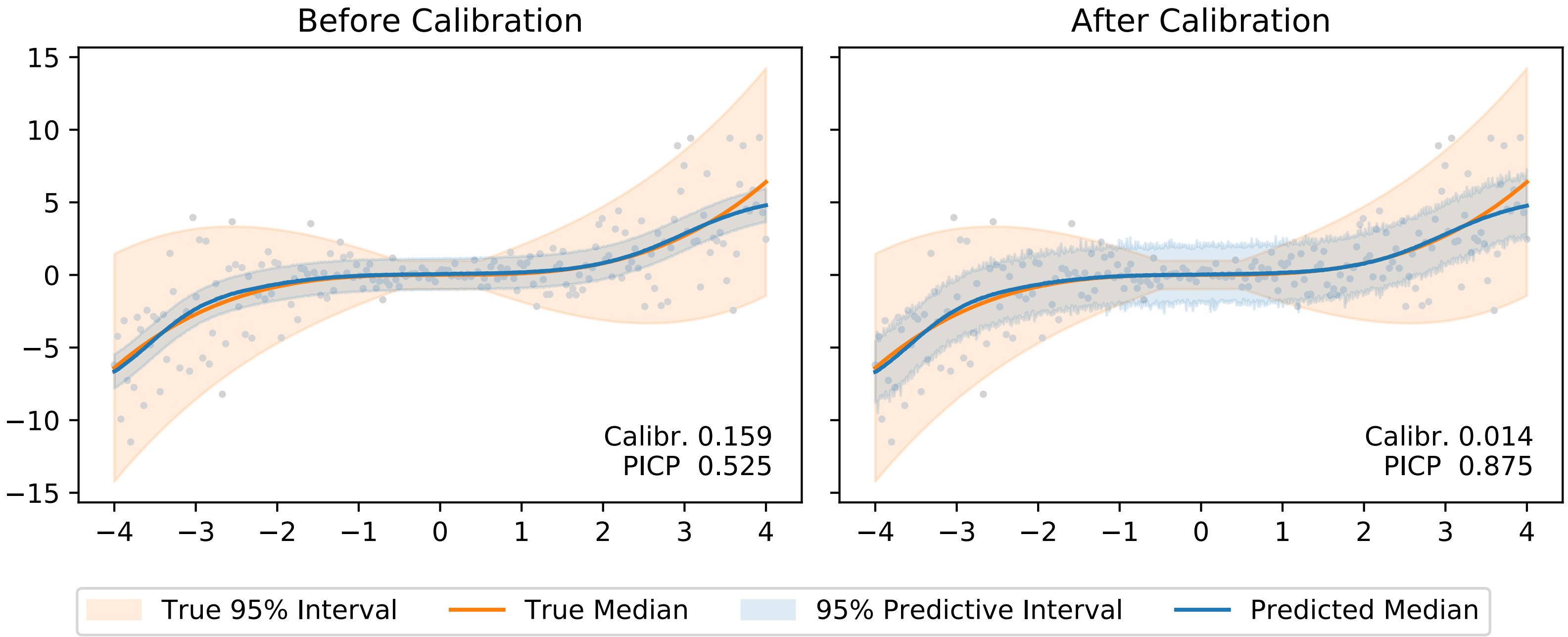
Heteroscedastic Dataset

The next dataset is generated by the same third-degree polynomial, but with heteroscedastic noise that depends on the value of the predictor X. We will model it using a BNN with a constant variance of the noise in the likelihood, and see if the calibration procedure can fix the resulting miscalibrated posterior predictive.



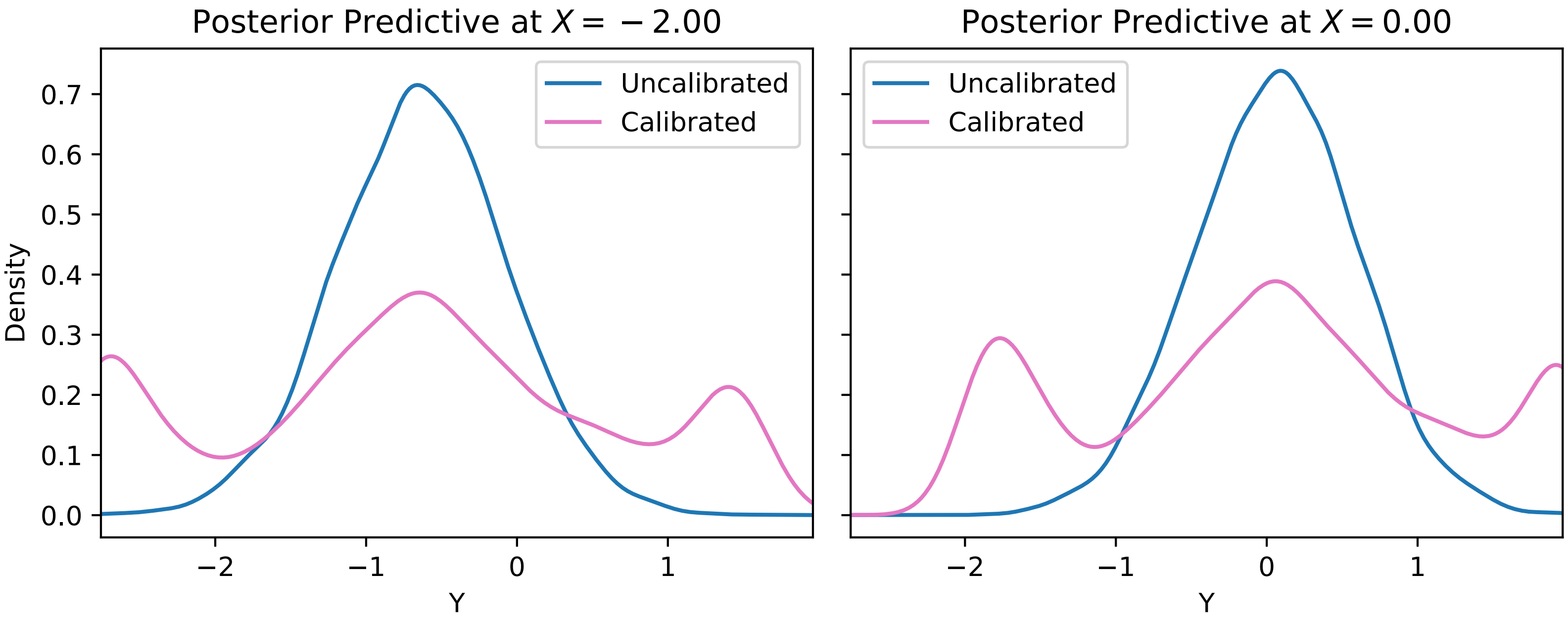
Heteroscedastic Noise: Recalibration

When a BNN is unable to capture heteroscedastic noise, the quantile calibration only makes the posterior predictive worse. The central region of aleatoric uncertainty that the original posterior predictive captured correctly is now inflated. The resulting model might be producing better uncertainty on average (which is reflected in the metrics), but is less precise in specific segments of the input space:



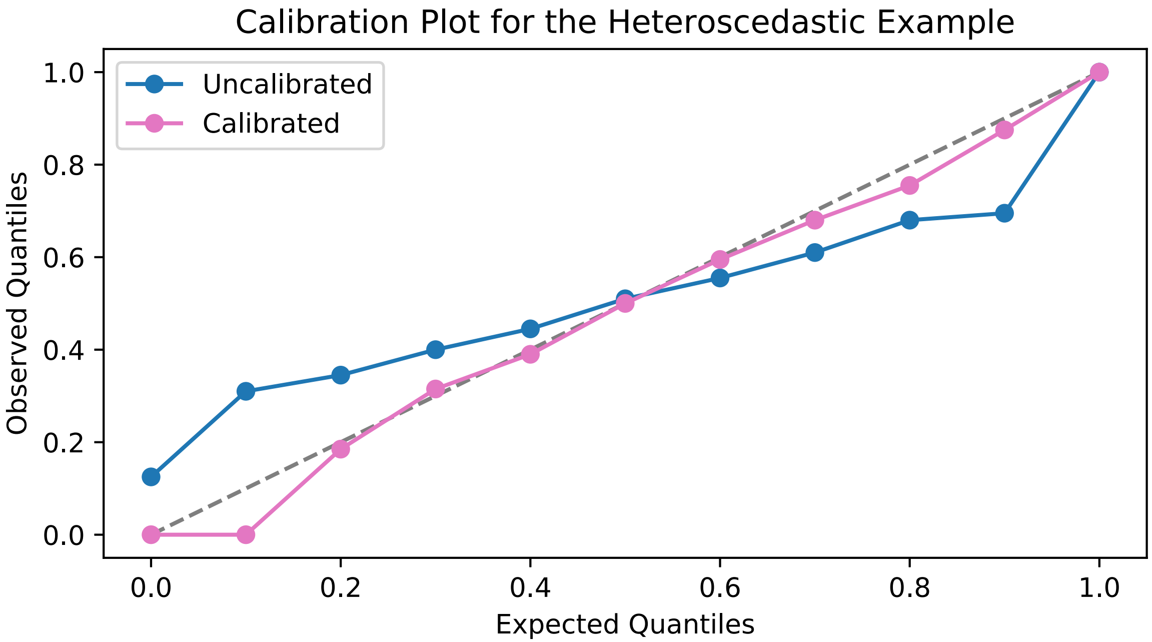
Conditional Distributions

From the plots below, we see how calibration transforms the posterior predictive. We see that at both $X = -2$ and $X = 0$, the transformation is the same, only shifted according to the value of Y. This agrees with the previous observation, that the uncertainty band widens uniformly across all values of X. The calibrated conditional posterior predictive is also multimodal:



Issue with the Definition of Quantile Calibration

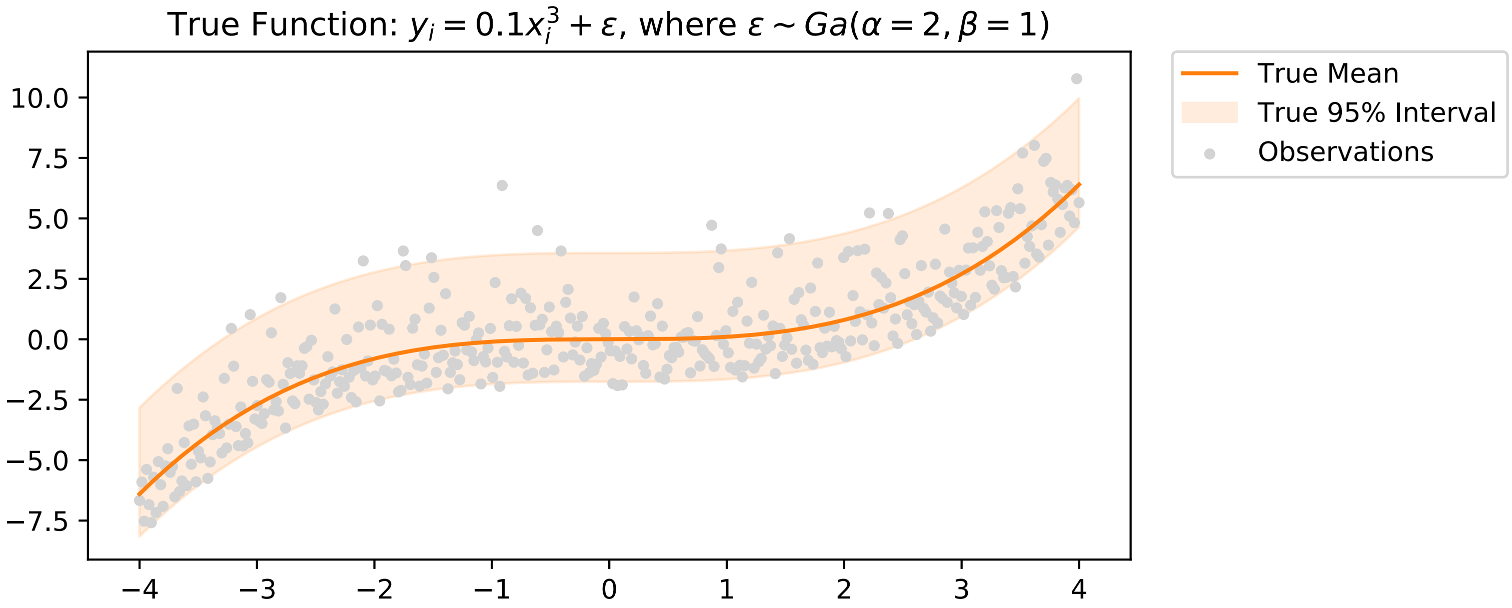
The resulting posterior predictive after recalibration does a bad job of capturing heteroscedastic noise in our latest example. Yet, from the perspective of the calibration error and the calibration plot, calibration has been significantly improved:



The issue lies in the definition of quantile calibration. The algorithm aims to match the predicted to empirical quantiles across the whole input space. That, however, does not necessarily produce posterior predictives that align with the data.

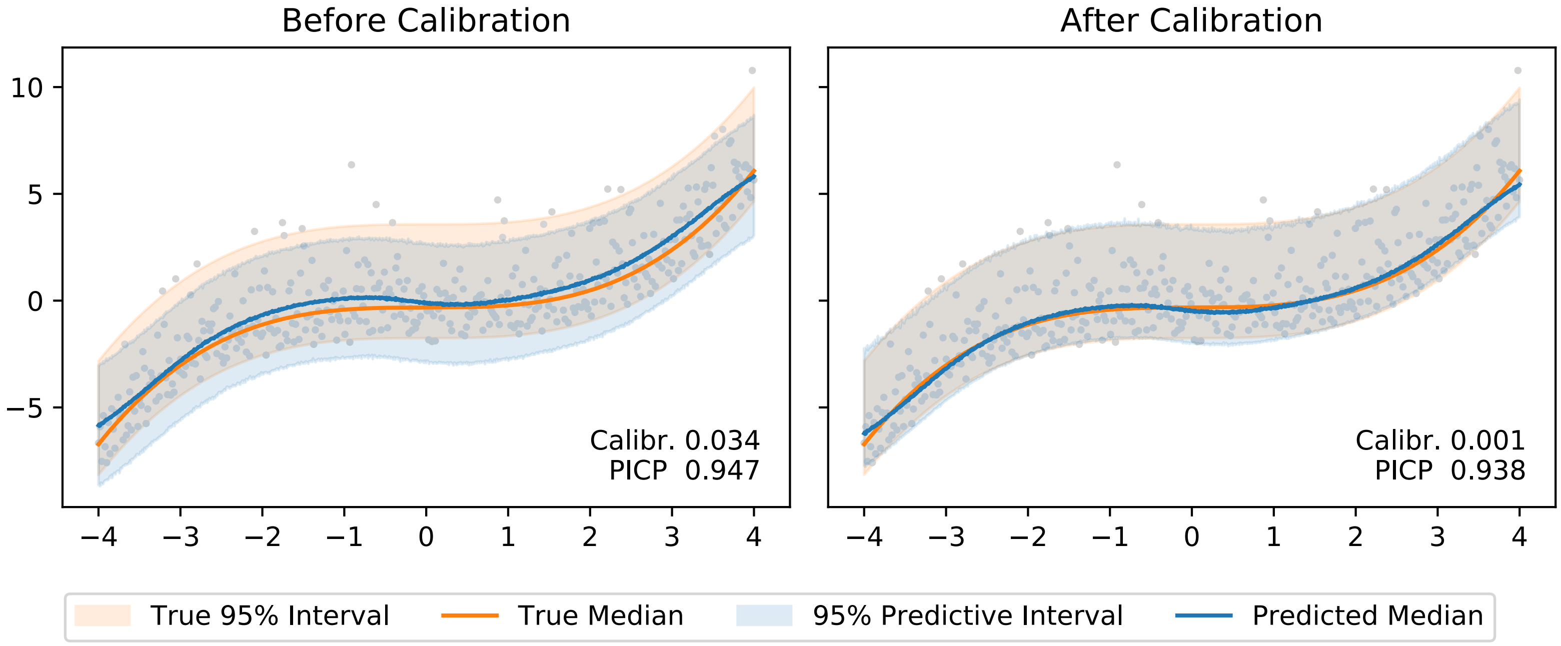
Non-Gaussian Data

The authors of the paper state that "*if the true data distribution $P(Y|X)$ is not Gaussian, uncertainty estimates derived from the Bayesian model will not be calibrated*". We will construct such a dataset by generating observations with Gamma noise, instead of Normal noise, fit an ordinary BNN to it and see how the proposed calibration algorithm performs:



Non-Gaussian Noise: Recalibration

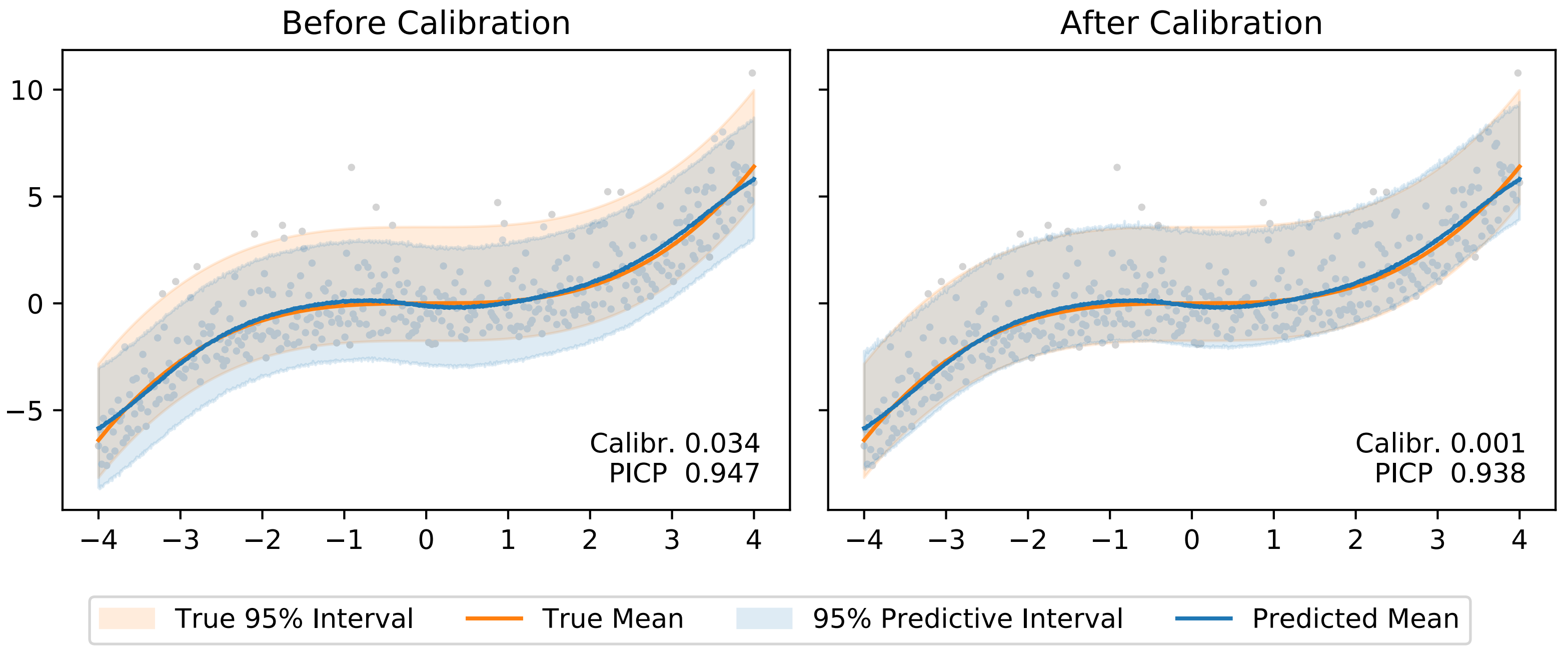
The simulated posterior predictive obtained from a BNN with a Normal likelihood turns out to be indeed miscalibrated. All of the quantiles including the median are off. After applying the calibration procedure, all of the quantiles are aligned with the data. The non-parametric isotonic regression that lies at the core of the proposed calibration method seems to excel in this setting:



Point Estimates: the Median and the Mean

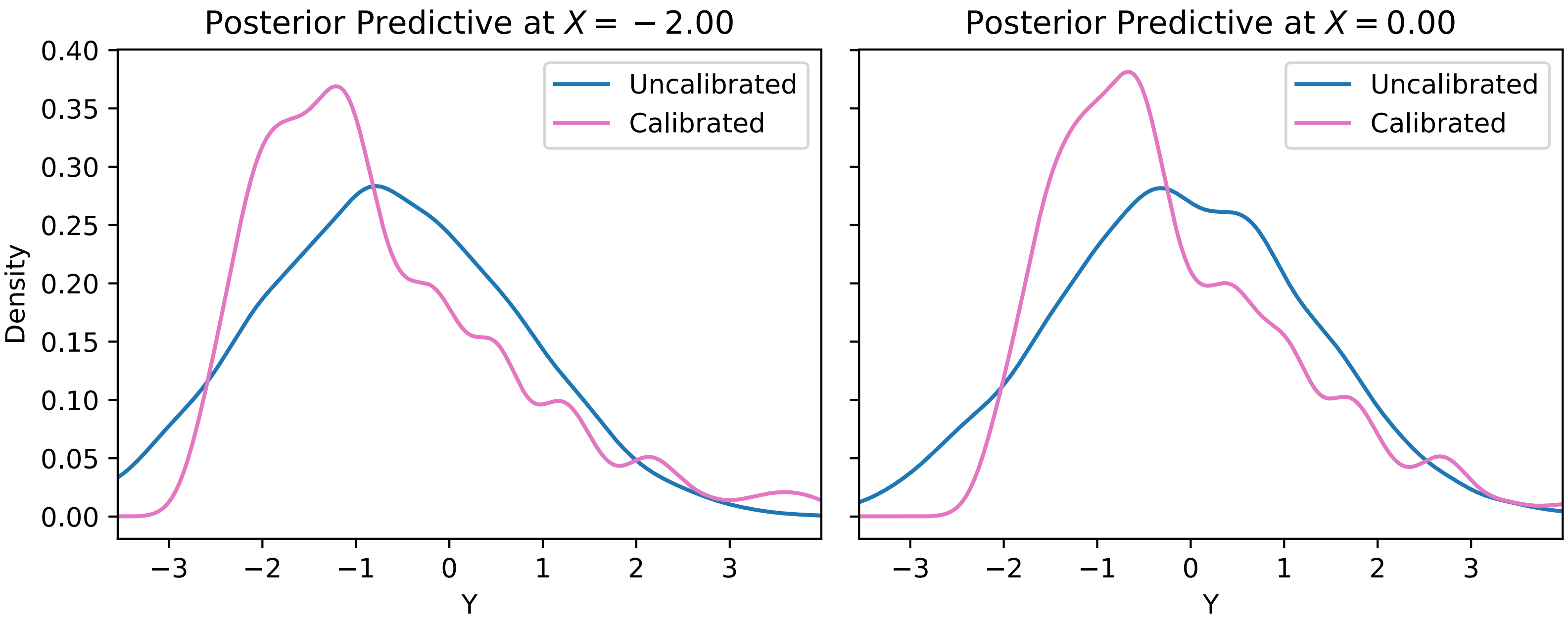
Due to asymmetric noise in our data generation process, the median and the mean of the uncalibrated posterior predictive differ. It appears that the median deviates further from the true median.

We also see that calibration improves the posterior predictive median while not affecting the mean:



Conditional Distributions

Although the data is generated with non-Gaussian noise, our BNN model uses a Gaussian likelihood. Therefore, the uncalibrated predictive has a symmetric distribution. We observe that in this case, the calibration algorithm is able to adjust the posterior predictive to become skewed to track the data:



Evaluation

Summary of the Findings

Let us first summarise the results from the conducted experiments, noting the effect that the proposed calibration algorithm has on the posterior predictives:

Dataset	**Miscalibration**	**Effect of Calibration**	**Aleatoric**	**Epistemic**
Homoscedastic, no gaps	Wrong noise	Aligns the posterior predictive with the data	OK	OK
Homoscedastic, with missing data	None	Doesn't ruin a good model	OK	OK
– – – " – – –	Wrong prior	Does not improve bad epistemic uncertainty	OK	Incorrect
– – – " – – –	Wrong noise	Aligns aleatoric uncertainty with the data. Unreasonably shrinks or inflates epistemic uncertainty.	OK	Incorrect
– – – " – – –	Bias	Uniformly changes the uncertainty band to fit the data	Incorrect	Incorrect
– – – " – – –	VI approximation	Does not improve bad epistemic uncertainty	OK	Incorrect
– – – " – – –	VI + wrong noise	Improves aleatoric uncertainty. Unable to remedy epistemic uncertainty.	OK	Incorrect
Heteroscedastic, no gaps	Wrong noise (constant)	Makes the posterior predictive worse in specific segments	Incorrect	Incorrect
Non-Gaussian data	Wrong noise (Gaussian)	Significantly improves the posterior predictive, aligning it with the data	OK	OK

Evaluation of the Claims

Overall, our understanding is that the claims made by the authors of the paper are valid. In strict accordance with the definition of *quantile-calibrated* regression output, their method produces well-calibrated uncertainty estimates, *given enough i.i.d. data*. The employed definition of a well-calibrated output involves matching the *marginal* probabilities of the response variable Y.

Advantages:

- **Sound & simple:** The algorithm is statistically sound, is very simple to implement and easy to apply to any regression model.
- **Model-agnostic:** The calibration algorithm is model-agnostic, which is both its strength (as it can be applied in a post-processing step to any black-box model) and its weakness (since it doesn't understand the output of the model and may perform arbitrary mapping of the quantiles).
- **Avoids overfitting:** The quantiles are not perfectly calibrated, which would pose an issue of overfitting, but are calibrated reasonably well to the empirical ones using a hold-out dataset or cross-validation.

Evaluation of the Claims (cont.)

Additional advantages of the algorithm:

- **Improves aleatoric uncertainty:** The calibration algorithm performs well in terms of aleatoric uncertainty on homoscedastic datasets with no missing data.
- **Excels on non-Gaussian data:** Due to the non-parametric nature of the algorithm, it also excels if the true noise of the data is not Gaussian.
- **Preserves or improves point estimates**
  - In all cases, both Gaussian and non-Gaussian, the algorithm preserved the mean
  - In all Gaussian noise cases, the algorithm preserved the median
  - In the non-Gaussian case, the uncalibrated median was off, and the calibration algorithm improved it

Evaluation of the Claims (cont.)

At the same time, the calibration algorithm has its limits, performs poorly or makes the posterior predictive worse in a number of scenarios.

Cases of failure:

- **Distorts epistemic uncertainty:** The quantile-based calibration doesn't know how to deal with epistemic uncertainty due to its reliance on data availability. In certain situations, this might destroy or distort originally reasonable epistemic uncertainty.
- **Can't fix a bad model:** The algorithm is unable to fix a bad model, e.g. the one with bias due to an incorrect combination of the prior and network architecture.
- **Should be used with care on heteroscedastic data**: The technique also cannot remedy bad posterior predictives obtained on heteroscedastic datasets, occasionally making them worse. The algorithm maps quantiles uniformly across the input space, which only makes sense if the model is capturing heteroscedastic noise.
- **Needs a lot of data:** Heavy dependence on sufficient (ideally infinite) i.i.d. data might pose a problem in practice, as the dimensionality of the problem grows.
- **May cause multimodality:** When the dataset is small, the calibrated posterior predictive is not smooth and in extreme cases may display multimodality, absent in the original uncalibrated distribution.

Future Work

Potential Directions of Research

Benjamin Yuen and Dmitry Vukolov are looking forward to continuing the research during the Spring semester. The quantile-calibration algorithm is simple and has many shortcomings. Therefore, among the possible directions of future research we see:

- Additional research and incremental tweaks of the quantile calibration algorithm
    - Investigation of the relationship between the log-likelihood and calibration, trying to understand if calibration improves the log-likelihood
    - Application of the quantile calibration algorithm to maximum-likelihood models
    - Making the calibration procedure dependent on X by binning or through other means
- Beyond the calibration method
    - Reproduction and comparative analysis of alternative algorithms to estimate aleatoric and epistemic uncertainty from data
    - Evaluation of existing techniques for modeling heteroscedastic noise with Bayesian neural networks
    - Research into quantitative metrics for epistemic uncertainty, that are useful for a downstream task

References

- Depeweg S., Hernandez-Lobato J. M., Doshi-Velez F., Udluft S., 2018, **Decomposition of Uncertainty in Bayesian Deep Learning for Efficient and Risk-sensitive Learning**
- Der Kiureghian A., Ditlevsen O., 2009, **Aleatory or epistemic? Does it matter?**
- Foong A. Y. K., Li Y., Hernandez-Lobato J. M., Turner R. E., 2019, **'In-Between' Uncertainty in Bayesian Neural Networks**
- Gal Y., 2016, **Uncertainty in Deep Learning**
- Gal Y., Ghahramani, Z., 2016, **Dropout as a Bayesian approximation: Representing model uncertainty in deep learning**
- Guo Ch., Pleiss G., Sun Y., Weinberger K. Q., 2017, **On Calibration of Modern Neural Networks**
- Hernandez-Lobato J. M., Adams R., 2015, **Probabilistic backpropagation for scalable learning of Bayesian neural networks**

References (cont.)

- Hullermeier E., Waegeman W., 2019, **Aleatoric and Epistemic Uncertainty in Machine Learning: A Tutorial Introduction**
- Kuleshov V., Fenner N., Ermon S., 2018, **Accurate Uncertainties for Deep Learning Using Calibrated Regression**
- Lakshminarayanan B., Pritzel A., Blundell Ch., 2017, **Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles**
- Loquericio A., Scaramuzza D., Segu M., 2019, **A General Framework for Uncertainty Estimation in Deep Learning**
- Maddox W. J., Garipov T., Izmailov P., Vetrov D., Wilson A. G., 2019, **A Simple Baseline for Bayesian Uncertainty in Deep Learning**
- Maeda S., 2014, **A Bayesian Encourages Dropout**
- McKay D., 1992, **Bayesian Methods for Adaptive Models**

References (cont.)

- Neal R., 1995, **Bayesian Learning for Neural Networks**
- Pearce T., Leibfried F., Brintrup A., Zaki M., Neely A., 2019, **Uncertainty in Neural Networks: Approximately Bayesian Ensembling**
- Phan B., Khan S., Salay R., Czarnecki K., 2019, **Bayesian Uncertainty Quantification with Synthetic Data**
- Platt J., 1999, **Probabilistic Output for Support Vector Machines and Comparisons to Regularized Likelihood Methods**
- Tagasovska N., Lopez-Paz D., 2018, **Frequentist uncertainty estimates for deep learning**
- Tagasovska N., Lopez-Paz D., 2019, **Single-Model Uncertainties for Deep Learning**
- Tomczak M. B., Swaroop S., Turner R. E., 2018, **Neural network ensembles and variational inference revisited**