การประมาณแบบหลายวัตถุประสงค์ของช่วงการทำนายเพื่อการพยากรณ์เชิงความน่าจะเป็น: การประยุกต์ใช้กับการพยากรณ์กำลังไฟฟ้าแสงอาทิตย์

นายวรชิต อำนวยพงศา

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2567

MULTI-OBJECTIVE ESTIMATION OF PREDICTION INTERVALS FOR PROBABILISTIC FORECASTING: APPLICATION TO SOLAR POWER FORECASTING

Mr. Worachit Amnuaypongsa

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Electrical Engineering
Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2024

Thesis Title	MULTI-OBJECTIVE ESTIMATION OF PREDICTION INTERVALS FOR PROBABILISTIC FORECASTING: APPLICATION TO SOLAR POWER FORECASTING
By	Mr. Worachit Amnuaypongsa
Field of Study	Electrical Engineering
Thesis Advisor	Professor Jitkomut Songsiri, Ph.D.

วรชิต อำนวยพงศา: การประมาณแบบหลายวัตถุประสงค์ของช่วงการทำนายเพื่อการพยากรณ์เชิงความน่าจะเป็น: การประยุกต์ใช้กับการพยากรณ์กำลังไฟฟ้าแสงอาทิตย์. (MULTI-OBJECTIVE ESTIMATION OF PREDICTION INTERVALS FOR PROBABILISTIC FORECASTING: APPLICATION TO SOLAR POWER FORECASTING) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ศ. ดร. จิตโกมุท ส่งศิริ, 124 หน้า.

วิทยานิพนธ์นี้นำเสนอสองระเบียบวิธีสำหรับการพยากรณ์เชิงความน่าจะเป็นผ่านช่วงการทำนายซึ่งให้ค่าขอบเขตบนและล่างที่ระดับความเชื่อมั่นที่กำหนด คุณภาพของช่วงการทำนายถูกประเมินด้วยสองวัตถุประสงค์ที่ขัดแย้งกัน ได้แก่ ความน่าเชื่อถือ (ความน่าจะเป็นคุ้มร่วมของช่วงการทำนาย: PICP) และความคมชัด (ความกว้างของช่วงการทำนาย) ลักษณะของช่วงการทำนายที่ดีสามารถบ่งบอกด้วยค่า PICP ที่สูง และช่วงการทำนายที่แคบ ในทางปฏิบัติความกว้างของช่วงการทำนายมักใช้เตรียมทรัพยากรสำรองในการบริหารความเสี่ยง โดยกระบวนการตัดสินใจมักให้ความสำคัญกับกรณีที่เลวร้ายที่สุด ซึ่งมักจะเกี่ยวข้องกับช่วงการทำนายที่ใหญ่ภายใต้สภาวะสุดโต่ง เพื่อรับมือกับสภาวะดังกล่าว ระเบียบวิธีที่นำเสนอจึงรวมแง่มุมใหม่ของความกว้างของช่วงการทำนายในรูปแบบการหาค่าเหมาะสุด ระเบียบวิธีแรกนำเสนอการหาค่าเหมาะสุดแบบคอนเวกซ์สามแบบโดยใช้ฟังก์ชันจุดประสงค์เดียวกันคือฟังก์ชันพินบอล ระเบียบวิธีที่สองใช้หลักการนำ PICP และฟังก์ชันความกว้างของช่วงการทำนายรวมกันเป็นฟังก์ชันสูญเสียโดยตรง โดยประยุกต์ใช้ฟังก์ชันผลรวมของค่ามากที่สุด K ค่า เพื่อลงโทษช่วงการทำนายที่กว้างมาก ระเบียบวิธีนี้สามารถประยุกต์ใช้กับแบบจำลองที่ไม่เป็นเชิงเส้น เช่น โครงข่ายประสาทเทียมได้ ฟังก์ชันสูญเสียที่นำเสนอสามารถใช้อัลกอริธึมอิงเกรเดียนต์ในการฝึกสอนแบบจำลอง ทำให้สามารถประยุกต์ใช้ร่วมกับสถาปัตยกรรมโครงข่ายประสาทเทียมที่ซับซ้อน และแบบจำลองการเรียนรู้เชิงลึกได้ ผลการทดลองบ่งชี้ว่าระเบียบวิธีทั้งสองสามารถลดช่วงการทำนายที่กว้างมากได้อย่างมีประสิทธิภาพในขณะที่รักษาความน่าเชื่อถือไว้ได้โดยเฉพาะกรณีที่ข้อมูลมีความแปรปรวนสูง ในการพยากรณ์ความเข้มแสงอาทิตย์ วิธีการที่เสนอสามารถลดช่วงการทำนายที่กว้างมากได้อย่างมีนัยสำคัญในสภาวะที่มีเมฆมาก การวิเคราะห์ต้นทุนในการเตรียมกำลังไฟฟ้าสำรองจากพลังงานแสงอาทิตย์แสดงให้เห็นว่าวิธีที่นำเสนอช่วยลดการจัดสรรกำลังไฟฟ้าสำรองที่เกินความจำเป็น และลดต้นทุนรวมทั้งในส่วนของการจัดเตรียมและค่าปรับจากการขาดแคลนกำลังไฟฟ้าที่ผลิตในสภาวะที่มีความไม่แน่นอนสูงได้อย่างมีประสิทธิภาพ อันเป็นผลมาจากขอบล่างของช่วงการทำนายที่ครอบคลุมการผลิตพลังงานจริงได้ดียิ่งขึ้นส่งผลให้ค่าปรับจากการขาดโหลดลดลง อีกทั้งในปัญหาการจัดการพลังงานแบบทนทาน ช่วงต้นทุนพลังงานไฟฟ้าสุทธิที่ประเมินจากข้อมูลช่วงการทำนายมีความแปรปรวนที่แคบที่สุดเมื่อเทียบกับวิธีอื่น เนื่องจากสามารถลดความอนุรักษ์นิยมของความกว้างช่วงการทำนายของค่าพยากรณ์โหลดสุทธิ

ภาควิชา	วิศวกรรมไฟฟ้า	ลายมือชื่อนิสิต	....
สาขาวิชา	วิศวกรรมไฟฟ้า	ลายมือชื่ออ.ที่ปรึกษาหลัก	....
ปีการศึกษา	2567		

6670220021: MAJOR ELECTRICAL ENGINEERING

KEYWORDS: PROBABILISTIC FORECASTING, PREDICTION INTERVALS (PIS), UNCERTAINTY QUANTIFICATION, NEURAL NETWORKS, SOLAR IRRADIANCE FORECASTING

WORACHIT AMNUAYPONGSA : MULTI-OBJECTIVE ESTIMATION OF PREDICTION INTERVALS FOR PROBABILISTIC FORECASTING: APPLICATION TO SOLAR POWER FORECASTING. ADVISOR : Prof. Jitkomut Songsiri, Ph.D., 124 pp.

This thesis introduces two methodologies for probabilistic forecasting through prediction intervals (PIs), which provide upper and lower bounds at a specified confidence level. The quality of a PI is assessed based on reliability (PICP) and sharpness (PI width), two inherently conflicting objectives. A good PI is determined by a high PICP while maintaining a narrow PI width. In real-world applications, the PI width is generally used in risk management to prepare reserve resources, as decision-making processes often focus on the worst-case scenarios arising with large PI widths under extreme conditions. To address this, the proposed methodologies incorporate new aspects of PI width into optimization formulations. The first methodology uses three convex formulations that all share a common pinball loss objective. The second methodology, based on PICP with width control, introduces a loss function that directly integrates PICP while penalizing large PI widths through a sum-of-K-largest function. This approach incorporates a nonlinear model showcased by a neural network in this thesis. This formulation is compatible with gradient-based optimization, making it well-suited for deep learning models and complex neural network architectures. Experimental results demonstrate the effectiveness of both methodologies in reducing large PI widths while maintaining high reliability, particularly in high volatility. In solar irradiance forecasting, our methods significantly reduce the PI width in high-uncertainty conditions, which typically occur during cloudy conditions. From engineering system applications, a cost analysis of solar power reserve preparation indicates that the proposed approach leads to reduced over-allocation of reserves and lower total costs, including both provision and deficit penalties under high uncertainty. This is due to an improved PI's lower bound, which better captures actual generation, thereby decreasing lost load penalties. Moreover, in robust energy management, the net electricity cost range assessed using PI information exhibits the narrowest variation compared to benchmarked methods due to the conservatism reduction in PI widths of net load forecasts.

Department	:	Electrical Engineering	Student's Signature	....
Field of Study	:	Electrical Engineering	Advisor's Signature	....
Academic Year	:	2024		

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisor, Professor Jitkomut Songsiri, for her invaluable guidance, patience, and dedication throughout my master's degree. Her unwavering support and insightful mentorship have shaped my academic journey. Under her guidance, I have never encountered a problem that seemed insurmountable. Moreover, her teaching in mathematics and optimization at CUEE has been a great source of inspiration, motivating me to pursue my master's degree under her supervision. Beyond academics, she has imparted valuable life lessons and served as a role model through her diligence, vision, and unwavering commitment to excellence. This thesis would not have been possible without her support, and I am truly honored and proud to be her student.

I would like to sincerely thank my thesis committee, Professor David Banjerdpongchai, Professor Kitsuchart Pasupa, and Associate Professor Naebboon Hoonchareon, for their valuable time, insightful feedback, and constructive comments on my thesis. Their expertise and guidance have been instrumental in refining my work.

I am deeply grateful to Assistant Professor Pisitpol Chirapongsananurak for encouraging me to pursue my master's degree at CUEE. His support and mentorship have been invaluable, guiding me through stressful times. I also thank Associate Professor Naebboon Hoonchareon for believing in my abilities and providing instrumental support and opportunities during my studies. Additionally, I appreciate Dr. Suwichaya Suwanwimolkul for her deep learning teaching and the valuable experiences she shared from industry and academia, which enhanced my master's journey. Her conversations are always enjoyable, making research in the lab more engaging and lively.

I am grateful for the CUEE High-Performance Scholarship, which provided full financial support throughout my master's studies at CUEE.

I would like to express my gratitude to all my friends at Lohit Club Chula for their continuous support. They have always been there for me, whether at the Central Library, sharing meals, exercising, or traveling together. Their companionship has made graduate studies much easier than facing it alone.

Lastly, I want to express my sincere gratitude to my family for their unwavering support in every aspect of my life.

CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Page

Chapter I

INTRODUCTION

The importance of probabilistic forecast. A rise in global temperatures due to climate change is a critical issue that demands global attention, prompting many nations to endorse a net-zero carbon emissions policy (Davis et al., 2018). In accordance with the policy, many countries worldwide are promoting the generation of renewable energy, including solar and wind energy, as demonstrated in Solangi et al. (2011); Zhang et al. (2013); Byrnes et al. (2013); Kitzing et al. (2012). However, higher penetration of renewable energy sources into the power system can introduce significant uncertainty in power generation, which is highly dependent on natural factors and potentially impacts system reliability. Reliability refers to the risk of a blackout occurring when renewable generation suddenly drops below the actual load, especially given the high uncertainty associated with renewable energy sources. To tackle reliability challenges, forecasting plays a crucial role today in predicting the future value of clean energy. This forecast assists in shaping future generation preparation plans by utilizing the predicted values as inputs for unit commitment and economic dispatch problems Li and Zhang (2020). The traditional forecast, deterministic forecast, offers only a single value that does not account for the uncertainty of forecasting, leaving us unaware of its reliability. This resulted in the development of probabilistic forecasting, which offers insights into the uncertainties of forecasting to support more effective risk management planning.

Application of probabilistic forecast. The probabilistic forecast has gained much attention for its ability to provide the uncertainty information of the forecasted value (Gneiting and Katzfuss, 2014). The probabilistic forecast can help users in decision-making in various applications, especially in power applications such as solar (van der Meer et al., 2018; Li and Zhang, 2020), wind (Zhang et al., 2014), electrical load (Hong and Fan, 2016; Zhao et al., 2020), and electricity price (Khosravi et al., 2013; Nowotarski and Weron, 2018) forecasting. In power system operations, operators typically assess net load forecasting based on the predicted values of load and renewable generation, defined as $\textit{net load} = \textit{load} - \textit{renewable generation}$. Both load and renewable generation forecasts inherently involve uncertainty. Net load forecasting informs operators about how much conventional power should be prepared to meet the upcoming load demand. This process primarily involves reserve power management to address net load forecast uncertainty and ensure system reliability, typically in unit commitment and economic dispatch problems (optimization problems). Probabilistic forecasting with accurately predicting uncertainty leads to better decision-making, effective resource planning, and reduced costs and risks. For example, Apostolopoulou et al. (2018) addressed the uncertainty arising from the net load, assuming that all uncertainty originated from solar generation as part of the power balance constraint in the hydroelectric power system. The power balance constraint aimed to ensure that the power generated by the system equaled the net load. The uncertainty from solar generation was incorporated to calculate an optimal dispatch scheme for a hydroelectric system. In Cordova et al. (2018), unit commitment modeling was developed by combining it with the representation of uncertainty in renewable energy generation. The information of uncertainty determined the resources required for risk hedging in the unit commitment process. Probabilistic forecasting is also applied in various domains, such as landslide displacement prediction for risk management planning (Lian et al., 2016), crop yield prediction to develop strategies for uncertainty management (Morales and Sheppard, 2023), and construction project management to mitigate financial risks associated with budgets (Mir et al., 2021).

Uncertainty representation. The probabilistic forecast can represent the uncertainty in three forms: quantile, distribution function, and prediction interval (PI). Bremnes (2004) utilized quantiles for probabilistic wind power forecasting through a local quantile regression technique. In Jeon and Taylor (2012), the uncertainty of wind power due to stochastic behavior in relation to wind speed and direction is modeled in the form of a probability distribution function using conditional kernel density estimation. Among all forms of uncertainty representation, a PI is widely used for uncertainty quantification because it effectively illustrates possible outcomes by displaying the upper and lower bounds associated with the confidence level (Quan et al., 2020). For example, in Apostolopoulou et al. (2018) and Cordova et al. (2018), the uncertainty of renewable energy is represented through PI and incorporated into the optimization problems.

Prediction interval construction approach. There are indirect and direct approaches for PI construction. *The indirect method* requires two steps: first, train a point forecast model, and second, conduct a statistical analysis of the forecast error. For example, the authors of Lorenz et al. (2009) conducted a point forecast for solar irradiance using a physical model, and then the error is assumed to follow a normal distribution to determine the PI. In Li et al. (2022), XGBoost was utilized for point forecasting solar irradiance, followed by kernel density estimation to model the distribution of forecast errors for constructing PIs. For the neural network-based approach, methods to provide PI include the delta method, Bayesian method, and mean-variance estimation (MVE) method, as summarized in Khosravi et al. (2011). The delta and Bayesian methods can incur high computational costs due to the need for calculating the Jacobian and Hessian matrices, respectively (Quan et al., 2020). On the other hand, the MVE method tends to perform poorly when the target variable does not follow the Gaussian assumption. Thus, for the indirect method, the distribution of forecast error relates to the model, indicating that the performance of the PI depends on both the model and the statistical analysis method. Quantifying uncertainty in neural networks (NN) has been approached through various methodologies. MC dropout (Gal and Ghahramani, 2016) estimates uncertainty by performing multiple forward passes with dropout layers active during inference. Similarly, deep ensembles (Lakshminarayanan et al., 2017) train multiple NNs with different initializations. Both methods, when combined with a loss function predicting mean and variance, yield diverse predictions. Deep ensembles effectively capture model and data uncertainty, providing predicted means and total variance for PI calculation. However, these approaches have limitations. Deep ensembles incur significant computational costs, with training and inference times scaling with the number of models. MC Dropout significantly increases prediction times, and its uncertainty quality depends on dropout hyperparameters, often requiring separate tuning. Crucially, neither MC dropout nor deep ensembles offer guaranteed prediction interval coverage probability (PICP). This limitation motivated the development of conformalized quantile regression (CQR) (Romano et al., 2019). CQR combines classical quantile regression with conformal prediction to produce PIs with guaranteed PICP in finite samples. It constructs PIs by computing non-conformity scores, which quantify discrepancies between held-out calibration data and predicted labels, and then using the quantiles of these scores. While CQR ensures a desired PICP, its trade-offs include PI widths (sharpness), which depends on the quality of the base regression model and the calibration set. A commonality among these three frameworks is the post-hoc nature of PI construction. The base regression models are not inherently optimized to enhance the sharpness of the resulting prediction intervals. On the other hand, *the direct approach* offers PI from a model in one step, as the model is specifically trained to consider the uncertainty characteristics of the target variable (Quan et al., 2020). Quantile-based techniques, such as quantile regression (QR) (Koenker, 2005) and quantile regression forest (QRF) (Meinshausen, 2006), can be applied to construct PIs without requiring distribution assumptions. Among direct methods, the lower upper bound estimation (LUBE) technique is favorable as it directly outputs the upper and lower bounds from the NN, originally proposed in Khosravi et al. (2011). The quality of PI is evaluated based on reliability and sharpness, commonly measured through prediction interval coverage probability (PICP) and PI width, respectively. A high-quality PI is characterized by a high PICP and a narrow PI width. However, these two goals conflict, presenting a trade-off characteristic where enhancing PICP results in a wider PI width. To achieve a high PICP and a narrow PI width, a common approach is to formulate an optimization problem that effectively quantifies the quality of PIs in both aspects.

Prediction interval-based optimization problem. Many literature formulate the PI construction problem as an optimization problem in various ways. The PICP and PI width components can be combined into a scalar-valued objective function, which represents an unconstrained optimization problem. The LUBE approach introduces a coverage width-based criterion (CWC) as a loss function for training the NN, utilizing the multiplicative form for the PICP and PI width functions (Khosravi et al., 2011). Several alternative versions of CWC loss have been proposed within the LUBE framework, utilizing various models such as support vector machine (SVM) (Shrivastava et al., 2015), extreme learning machine (ELM) (Ni et al., 2017), and NN (Quan et al., 2014a; Ye et al., 2016), with NN being the most widely used. In Quan et al. (2014b), the authors defined the problem as minimizing the PI width, treating PICP as a constraint to achieve the desired probability. However, imposing the PICP as a hard constraint resulted in a larger PI width than necessary. Consequently, Zhang et al. (2015) introduced a deviation information-based criterion (DIC) as a new objective in an additive form that incorporated both PI width and the *pun* function, which penalizes the deviation of PI from the target variable. Schemes that treated the problem as a multi-objective optimization challenge, aiming to minimize the PI width and maximize the PICP as bi-objectives, were proposed in Li and Jin (2018); Galván et al. (2017). However, all of these problem formulations utilized heuristic optimization techniques such as simulated annealing and particle swarm optimization to find the optimal model parameters, which did not guarantee a local minimum of the objective function. Additionally, the loss function proposed with the NN model is incompatible with gradient-based algorithms due to its non-differentiability, while gradient-based methods are the standard for training state-of-the-art NN (Chen et al., 2024).

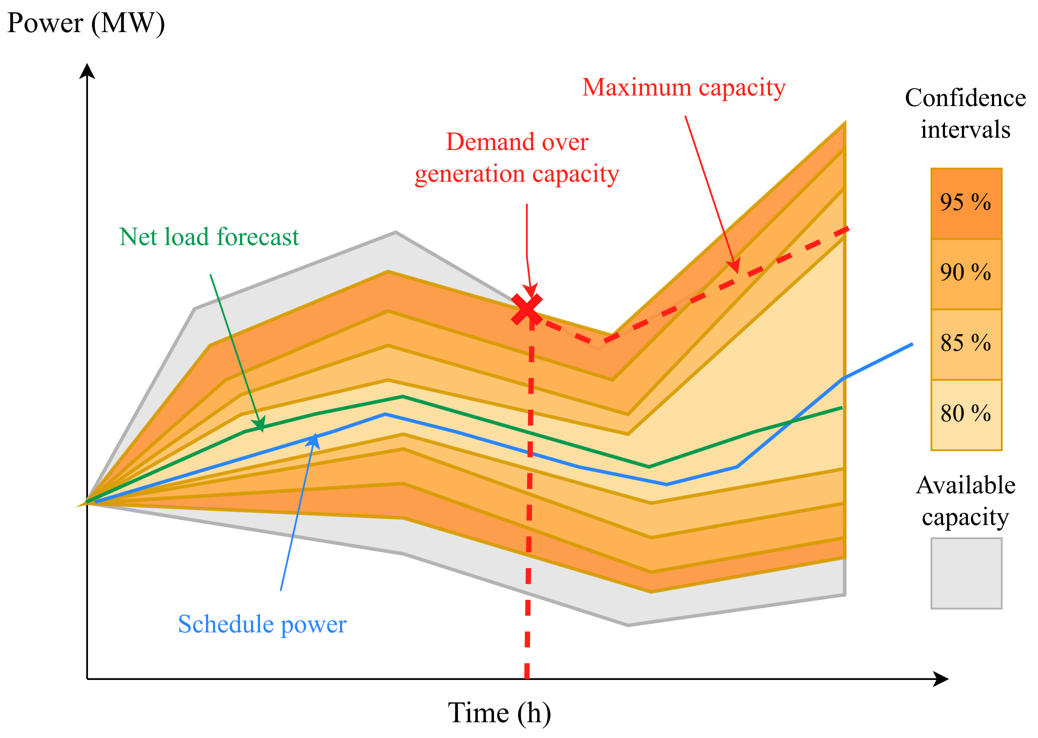


Figure 1.1: Concept of using PIs in power operation.

Cost of large PI widths. For grid integration, information about renewable energy forecasts is provided as point forecasts, and their uncertainty is represented by PI in the interval $[l, u]$. Next, the PI of renewable energy is used to calculate the PI of the net load. The PI width of the net load calculated by $u - l$ is utilized to prepare operational generation resources (Etingov et al., 2012), addressing potential uncertainties as shown in Figure 1.1. The upper bound of this PI with a specified confidence level indicates the reserve power required to ensure system reliability at the given confidence level, while the lower bound represents the minimum possible net load, which is used to reduce the risk of over-committing generation resources. A wider PI indicates greater uncertainty in the forecast, which may require a larger reserve margin to ensure reliability, resulting in higher operational costs due to an increased need for standby power resources. Especially in power system contexts, the PI of renewable forecasting is generally employed to quantify the uncertainty in renewable generation, directly assisting in the decision-making process. These intervals help power operators handle uncertainty, allowing them to plan reserve power generation to meet future electrical demand and increase the stability of the power system. In Zhao et al. (2021), the authors demonstrated that the deviation of point forecasts from the PIs in wind power applications was used to determine reserve power, impacting operational costs. According to Zhao et al. (2022), the PI of wind power was utilized to determine the amount of offered wind power within the PI, where the decision-making regarding the offered wind power was based on the worst-case cost. However, a large PI width, which may occur in some instances due to high uncertainty, can drastically affect the worst-case cost, impacting decision-making. In practice, unit commitment and economic dispatch often rely on robust optimization problems to address the uncertainty (Quan et al., 2020). The robust approach requires defining an uncertainty set to account for variability in renewable generation, emphasizing preparation for the worst-case scenario under extreme conditions, especially in cases with extremely large PI widths. As a result, the solution from robust optimization is generally considered conservative. The PI widths are generally a mix of small and large values, but a portion of a large PI width that occurs in some instances can increase reserve power preparation throughout the day in economic dispatch and unit commitment. Therefore, merely focusing on the average PI width may not be sufficient. It is also essential to monitor the group of large PI widths because planning under a significant number of large PI widths can raise operational costs. When the large PI width is reduced, the uncertainty set in the robust optimization scheme becomes smaller, which can be advantageous in reducing the conservatism of the unit commitment solution, as the worst-case scenario is less severe. A smaller uncertainty set can lead to reduced operating costs and enhanced efficiency while maintaining reliability since the robust optimization approach can focus on more realistic scenarios without the overallocation of reserve resources.

Research gap. As the large PI width significantly affects the operational cost. Most of the literature treats the PI width term as the average PI width incorporated in the optimization problem for generating PIs. However, few studies focus on reducing the large PI widths. Moreover, most of the proposed methods rely on heuristic optimization, which is heavily influenced by the randomness of the search direction and is highly sensitive to hyperparameter settings. When optimization problems involve multiple constraints, heuristic algorithms may struggle to find feasible solutions and may lack mathematical significance. In the case of NN-based loss functions, which are nonlinear and unconstrained, heuristic approaches face challenges when dealing with a large number of parameters, whereas gradient-based methods remain the standard for training NNs.

Research objectives. This thesis presents optimization formulations designed to address the challenges posed by large prediction interval (PI) widths. The proposed optimization problems tackle two conflicting objectives: maintaining the desired coverage probability

(PICP) while minimizing the PI width. The goal is to maintain PICP at the specified level while striving to reduce the PI width. Furthermore, we aim to apply exact optimization methods where optimality can be theoretically guaranteed for solving the proposed problem. To achieve this, we propose three convex optimization formulations and one nonlinear optimization formulation. The convex formulations are based on the pinball loss with a width control function, which regulates the PI width in various aspects, including average PI width, large PI widths, and maximum PI width. These convex formulations employ a linear additive model, which ensures that the formulation remains convex. The benefit of convexity is that a global minimum can be guaranteed. While the pinball-based formulation indirectly addresses both PICP and PI width, the nonlinear formulation directly targets these two objectives. Additionally, we incorporate the sum of the K largest PI widths into the loss function, encouraging the reduction in the large PI widths. This formulation is applicable to employing any nonlinear model to capture the nonlinear characteristics of the data. Moreover, a smooth approximation technique is applied to the PICP terms, allowing the use of gradient-based algorithms. The effectiveness of these formulations is demonstrated in the context of providing PI for solar irradiance forecasting, showcasing their practical application in real-world scenarios.

Chapters overview. This thesis is organized as follows. Chapter 2 presents the background knowledge relevant to this work. Chapter 3 introduces the proposed methodology, comprising two approaches: pinball-based formulation and PICP with width control formulation. Chapter 4 details the experimental design used to evaluate their performance in synthetic data and real-world data. Chapter 5 reports the experimental results for methodology 1, while Chapter 6 presents the results for methodology 2. Chapter 7 demonstrates the effectiveness of the methodology 2 in engineering system applications, such as lowering reserve preparation costs and enhancing robust energy management. Finally, Chapter 8 summarizes the contributions and discusses potential directions for future research.

1.1 Objectives

1. We aim to deliver a probabilistic forecast of solar power in the form of a prediction interval, assisting users in decision making for energy management.
2. We propose new optimization formulations to construct a prediction interval in the probabilistic forecast that encourages a trade-off characteristic between two objectives: high coverage and narrower PI width.

1.2 Scope of Work

1. A proposed probabilistic forecasting method is presented in the form of prediction intervals.
2. The solar data utilized for this research was gathered in Thailand.

1.3 Expected outcome

1. A methodology that generates quality-based prediction intervals for probabilistic forecasts, emphasizing high reliability and sharpness.
2. A software package that returns the prediction intervals corresponding to a given confidence level, which is provided in https://github.com/energyCUEE/PIestim_sumk.

Chapter II

BACKGROUND

This chapter presents the fundamental knowledge essential for this thesis. It is organized into five sections. The first section introduces the basic regression model. The second section covers quantile-based methods, including quantile estimation, quantile regression, and quantile regression forest. The third section discusses prediction interval (PI) estimation, encompassing fundamental concepts and PI evaluation metrics. The fourth section focuses on neural networks, detailing the model architecture, loss function, and training and validation procedures. Finally, the last section provides an overview of the lower-upper bound estimation (LUBE) method.

2.1 Regression model

Regression analysis is a statistical method used to examine the relationship between a target variable, $y \in \mathbf{R}^m$, and predictor variables $x \in \mathbf{R}^p$. The target variable is generated by a data-generating process (DGP) described as

$$y = f(x) + e, \tag{2.1}$$

where $f(x)$ is the ground truth function and e is an additive data noise. A regression task aims to predict the value of y given the information of predictors x, with $\hat{y}$ representing the predicted value of the target variable. A regression model, denoted as $\hat{f}(x;\theta)$, estimates the relationship between y and x, parameterized by the model parameter θ. Then, the prediction of y can be expressed as $\hat{y} = \hat{f}(x;\theta)$. The simplest form of regression analysis is linear regression, which assumes a linear relationship between $\hat{y}$ and θ. To estimate this relationship, regression analysis aims to minimize a regression loss (ℓ), which quantifies the difference between the predicted value ($\hat{y}$) and the actual value (y) of the target variable as

$$\mathcal{L}(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N} \ell\left(y_i - \hat{y}_i\right). \tag{2.2}$$

The regression loss can be in the form of statistical measures such as mean squared error (MSE) and mean absolute error (MAE). Minimizing MSE results in an optimal estimator that represents the conditional mean of the target variable, denoted as $\mathbf{E}[y|x]$. Conversely, minimizing MAE leads to the best estimator corresponding to the conditional median. The conditional mean and conditional median can differ depending on the distribution of $y|x$, particularly in cases where the distribution is non-symmetric, as illustrated in Figure 2.1.

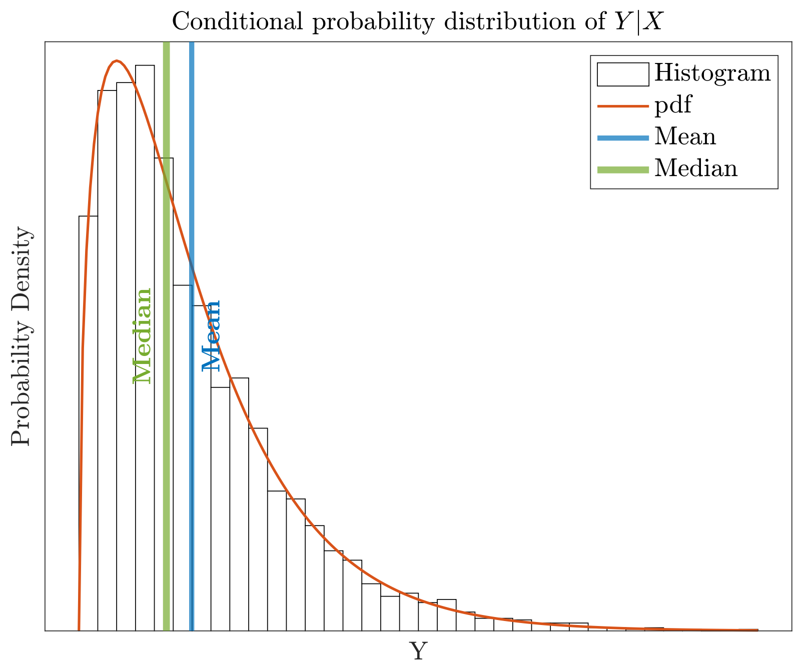


Figure 2.1: Conditional probability distribution of a target variable given predictors.

2.2 Quantile-based methods

This section covers the fundamentals of quantile estimation and the statistical techniques used to estimate conditional quantiles. It provides an overview of the optimization methods applied to estimate quantiles for one-dimensional data. Next, we detail quantile regression (QR) and quantile regression forest (QRF) as key statistical approaches for estimating the conditional quantiles of a target variable based on given predictors.

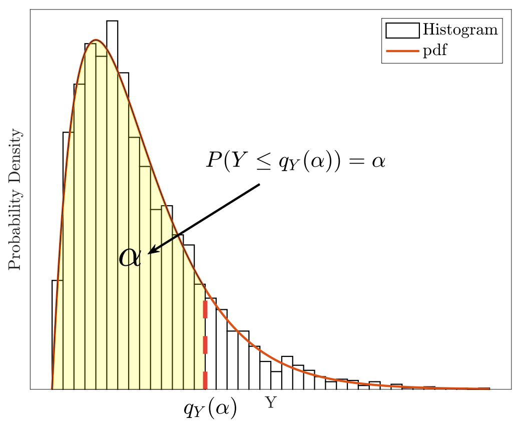
Quantile estimation

Let Y be a random variable with a probability density function (pdf) $f_Y(y)$ and a cumulative distribution function (CDF) given by $F_Y(y) = P(Y \leq y)$. The α^{th} quantile of Y, denoted as $q_\alpha(Y)$, is a statistical measure that partitions the pdf of Y so that a probability of α of the distribution falls below it, while proportion $1-\alpha$ falls above it as illustrated in Figure 2.2(a). The α-quantile of Y can be mathematically defined as

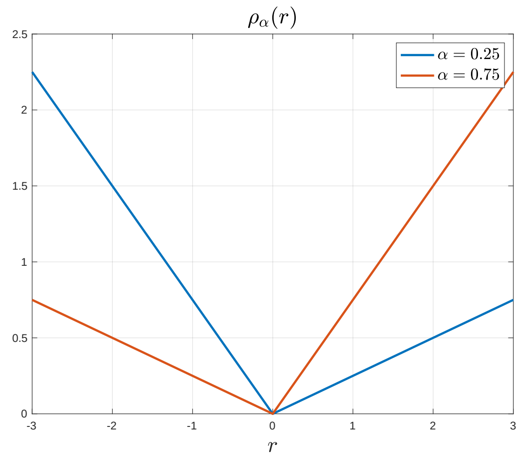
$$q_Y(\alpha) = F_Y^{-1}(\alpha) = \inf_y \{\ y \mid F(y) \geq \alpha\ \} \tag{2.3}$$

where α is the corresponding probability within the interval [0,1].

The $\alpha-$quantile of Y can be estimated by solving an optimization problem where the



(a) Histogram of sample quantiles.



(b) Pinball loss function.

Figure 2.2: Quantile estimation and pinball loss function.

objective is a pinball loss function (Koenker, 2005) defined as

$$\rho_\alpha(r) = \max(\alpha r, (\alpha - 1)r) = \begin{cases} \alpha r, & r \geq 0 \\ (\alpha - 1)r, & r < 0. \end{cases} \tag{2.4}$$

The structure of the pinball loss is a piecewise linear function with asymmetric slopes for positive and negative residue, as illustrated in Figure 2.2(b).

Theorem 2.2.1. *The quantile estimation can be obtained by minimizing the expectation of the* pinball loss *(Koenker, 2005)*

$$q_Y(\alpha) = \underset{\hat{y}}{\text{argmin}}\ \mathbf{E}[\rho_\alpha(Y - \hat{y})], \tag{2.5}$$

where $\hat{y}$ is the α^{th} quantile estimation of Y.

Proof. Given that y is drawn from random variable Y. The mean of pinball loss can be expressed as

$$\mathbf{E}[\rho_\alpha(Y - \hat{y})] = (\alpha - 1) \int_{-\infty}^{\hat{y}} (y - \hat{y}) f_Y(y) dy + \alpha \int_{\hat{y}}^{\infty} (y - \hat{y}) f_Y(y) dy.$$

Differentiating $\mathbf{E}[\rho_\alpha(Y - \hat{y})] = 0$ with respect to $\hat{y}$ by using Leibniz integral rule, we have

$$(1 - \alpha) \int_{-\infty}^{\hat{y}} f_Y(y) dy - \alpha \int_{\hat{y}}^{\infty} f_Y(y) dy = 0$$
$$(1 - \alpha) F_Y(\hat{y}) - \alpha[1 - F_Y(\hat{y})] = 0$$

which gives $F_Y(\hat{y}) = \alpha$ or equivalently $\hat{y} = F_Y^{-1}(\alpha)$ by the monotonically uniqueness property of CDF. Therefore, minimizing the mean of pinball loss yields $\hat{y}$, which is a quantile estimator where $\hat{y} = F_Y^{-1}(\alpha)$, represented by $q_Y(\alpha)$. ■

Corollary 2.2.1.1. *The pinball loss at the 0.5 quantile can be simplified as an absolute error. Minimizing this results in the estimation of the median equivalent at 0.5 quantiles.*

Suppose $y_1, y_2, ..., y_N$ are the samples of random variable Y. Sample quantile is traditionally determined by sorting the data and plotting a histogram, then calculating the sample quantile corresponding with given probability (α) as shown in Figure 2.2(a). In practice, the sample quantile can be estimated by minimizing the sample version of (2.5) as

$$\underset{z}{\text{minimize}} \quad \sum_{i=1}^{N} \rho_\alpha(y_i - z) \tag{2.6}$$

with $z \in \mathbf{R}$ is the estimated sample quantile corresponding to the α probability of the samples from Y.

To numerically solve (2.6), we can reformulate (2.6) into a linear programming (LP) problem by using epigraph form (Boyd and Vandenberghe, 2004). Estimating the quantile corresponding to any α can be achieved by solving a simple linear programming problem shown as

$$\begin{array}{ll} \underset{z,u,v}{\text{minimize}} & \alpha \mathbf{1}^T u + (1-\alpha)\mathbf{1}^T v \\ \text{subject to} & u \succeq y - \mathbf{1}z \\ & v \succeq \mathbf{1}z - y \\ & u, v \succeq 0, \end{array} \tag{2.7}$$

where $z \in \mathbf{R}$, $u \in \mathbf{R}^N$, $v \in \mathbf{R}^N$ are optimization variables, $y \in \mathbf{R}^N$ is the data, and $\mathbf{1}$ is a vector where all element is one with a dimension of N, $\mathbf{1} \triangleq \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T_{N\times 1}$. Linear programming algorithms, such as the simplex method, can be employed to solve this problem efficiently.

Quantile regression

Given that $X = \begin{bmatrix} x_1 & x_2 & \cdots & x_N \end{bmatrix}^T$ is a data matrix in which each $x_i \in \mathbf{R}^p$ for $i = 1, 2, \ldots, N$ represents the data of each sample with p predictors, and $y \in \mathbf{R}^N$ is a target variable. Quantile regression (QR) is a method used to estimate any quantiles of a target variable (y) given the data X (Koenker, 2005). This method is distribution-free and does not require an assumption about the error distribution. Compared with a traditional regression, in which the solution is obtained from solving the least-squares problem (minimizing MSE loss), a quantile regression solves the optimization problem defined as:

$$\underset{\theta}{\text{minimize}} \quad \sum_{i=1}^{N} \rho_\alpha(y_i - \hat{y}_i(x_i; \theta)) \tag{2.8}$$

where the solution, θ, is the regression parameter used for estimating the α^{th} samples quantile of the target variable.

Generally, the α^{th} conditional quantile function can be linear and non-linear in the parameters θ depending on the mathematical model. The type of optimization problem in (2.8) also depends on the model's linearity; therefore, different types of optimization problems require different numerical methods to solve. For a linear quantile regression model, the α^{th} sample quantile is modeled linear additive in θ shown as

$$\hat{y}_i = x_i^T\theta = \theta_1 x_{1,i} + \theta_2 x_{2,i} + \cdots + \theta_p x_{p,i} \tag{2.9}$$

where $\theta \in \mathbf{R}^p$ is the model parameter, and i is the sample index. The vectorized representation is given by $\hat{y} = X\theta$. Since the pinball function is convex, the optimization problem remains convex when $\hat{y}$ is linearly structured in the optimization variable θ. This holds because an affine transformation of a convex function preserves convexity (Boyd and Vandenberghe, 2004). Additionally, the linear quantile regression optimization problem can be simplified as an LP problem (Koenker, 2005) by substituting $z = X\theta$ in (2.7) as

$$\begin{array}{ll} \underset{\theta,u,v}{\text{minimize}} & \alpha\mathbf{1}^T u + (1-\alpha)\mathbf{1}^T v \\ \text{subject to} & u \succeq y - X\theta \\ & v \succeq X\theta - y \\ & u, v \succeq 0. \end{array} \tag{2.10}$$

For probabilistic forecasts, quantile regression can provide quantile, CDF, or prediction intervals (PIs). As this thesis focuses on generating PI, quantile regression is utilized to provide the upper and lower bounds corresponding with a confidence level $(1-\delta)$. The upper and lower bounds are obtained by solving two quantile regression problems that correspond to the two probabilities $\overline{\alpha}$ and $\underline{\alpha}$, respectively, where

$$\overline{\alpha} - \underline{\alpha} = 1 - \delta. \tag{2.11}$$

In the case of symmetric PIs, where the head and tail probabilities are equal, a pair of $\overline{\alpha}$ and $\underline{\alpha}$ corresponding to the confidence level can be defined as follows:

$$\underline{\alpha} = \frac{\delta}{2}, \quad \overline{\alpha} = 1 - \frac{\delta}{2}. \tag{2.12}$$

The example of two probabilities corresponding with confidence level can be shown in Table 2.1. Then, we can solve (2.10) separately to obtain the lower and upper bound of the target variable. For example, when $1 - \delta = 0.9$, we have two models with parameter $\overline{\theta}, \underline{\theta}$; each of which gives an estimation of the 0.95 quantile, and the 0.05 quantile, respectively.

Quantile regression forest

Observations of the data are given as $\{x_i, y_i\}_{i=1}^N$, where x_i represents a predictor variable, and y_i represents a target variable. Quantile regression forest (QRF) is an algorithm that estimates nonparametric conditional CDF proposed by Meinshausen (2006). The QRF

Table 2.1: Examples of lower and upper bound probabilities for symmetric prediction intervals corresponding to the nominal confidence level.

$1-\delta$	0.95	0.9	0.8	0.7	0.6	0.5
$\overline{\alpha}$	0.975	0.95	0.9	0.85	0.8	0.75
$\underline{\alpha}$	0.025	0.05	0.1	0.15	0.2	0.25

approach uses a random forest algorithm instead of minimizing the pinball loss function as in QR.

Random forest (RF) is a machine-learning algorithm that grows multiple decision trees proposed by Breiman (2001). The data in each tree is obtained through random sampling with replacement. At each split in the tree, a smaller subset of predictors is randomly chosen from the full set of predictors to reduce the correlation among all decision trees. In the regression task, the predicted value for new data is obtained by averaging the targets of samples in the same leaf as the new data across all trees. Given that θ represents the random parameter vector that describes the behavior of each tree, the corresponding tree is denoted as $T(\theta)$. For every leaf in the tree denoted as $\ell = 1, 2, \dots, L$, we denote the leaf in which the observation x falls as $\ell(x, \theta)$, and $R_{\ell(x,\theta)}$ represents a rectangular subspace in the predictor space. The structures of all trees are generated in the training process.

When making a prediction for a new observation $x^\star$, the weight vector of the tree is calculated $w_i(x^\star, \theta)$ which can be written as

$$w_i(x^\star, \theta) = \frac{\mathbf{1}(x_i \in R_{\ell(x^\star,\theta)})}{\sum_{\forall j} \mathbf{1}(x_j \in R_{\ell(x^\star,\theta)})}. \tag{2.13}$$

$\mathbf{1}(x_i \in R_{\ell(x^\star,\theta)})$ is a count function that equals to one when an observation x_i is in the same leaf as the new observation $x^\star$, and $\sum_{\forall j} \mathbf{1}(x_j \in R_{\ell(x^\star,\theta)})$ represents the number of observations falling in $R_{\ell(x^\star,\theta)}$. The prediction of a single tree given the new $x^\star$ is calculated by the average of target variables y_i for observations whose indices fall into the leaf for that observation written as

$$\hat{y}(x^\star, \theta) = \sum_{i=1}^{N} w_i(x^\star, \theta) y_i, \tag{2.14}$$

where $\hat{y}(x^\star, \theta)$ is the prediction from the leaf. An average of the prediction of m single trees where each constructed with $\theta_t, t = 1, 2, \dots, m$ is calculated to provide the final prediction. Given that w_i be the average of the weight vector of all trees, which is calculated by

$$w_i(x^\star) = \frac{1}{m} \sum_{t=1}^{m} w_i(x^\star, \theta_t). \tag{2.15}$$

The final prediction of random forest, which is the conditional mean of the target variable, is estimated by

$$\hat{y}(x^\star) = \sum_{i=1}^{N} w_i(x^\star) y_i = \frac{1}{m} \sum_{t=1}^{m} \hat{y}(x^\star, \theta_t). \tag{2.16}$$

Quantile regression forest expands the RF framework to provide the full conditional CDF instead of the conditional mean. The conditional CDF of Y given $X = x$ is expressed as $F(y|X = x) = P(Y \leq y|X = x)$. Using the same approach as the RF, the estimated CDF is calculated by computing the average of the indicator function that reflects the CDF instead of the target variable y as

$$\hat{F}(y|X = x) = \sum_{i=1}^{N} w_i(x) \mathbf{1}(y_i \leq y), \tag{2.17}$$

where $w_i(x)$ has the same meaning as (2.13) and $\mathbf{1}(y_i \leq y)$ equals one when $y_i \leq y$ and zero otherwise. Then, the conditional quantile can be estimated from QRF by replacing $\hat{F}(y|X = x)$ into (2.3) as $\hat{q}_{Y|X=x}(\alpha) = \inf_y \{\, y \mid \hat{F}(y|X = x) \geq \alpha \,\}$.

It seems that QRF provides a complete estimation of the whole conditional distribution. Regarding software usage, we can specifically select two quantiles representing lower and upper bound by providing $\overline{\alpha}, \underline{\alpha}$ to generate PI. The complexity of the tree structure can make the PI characteristic highly nonlinear in QRF, which may describe some nonlinear behavior of the data better than a linear model. However, it is possible that using a single QRF model may not effectively represent both the upper and lower bounds simultaneously. To overcome this limitation, we can use different QRF models to estimate the upper and lower bounds separately. These different models correspond to different hyperparameters of QRF, where each hyperparameter can be selected by searching for the best configuration using pinball loss as the evaluation criterion for both $\overline{\alpha}$ and $\underline{\alpha}$. However, this may not ensure that the crossing quantile does not occur. This thesis utilizes the QRF implementation available in scikit-learn-quantile documentation.

2.3 PI estimation

Given a dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^{N}$ with N samples where $x \in \mathbf{R}^p$ represented a p-dimensional predictor vector and $y \in \mathbf{R}^m$ denoted an m-dimensional target variable. According to the framework of regression analysis in (2.1), the overall uncertainty of y consists of model uncertainty and data noise. A prediction interval (PI) is a statistical tool that quantifies the overall uncertainty of y by providing an upper and lower bound of y at a confidence level of $(1 - \delta)$. In the direct PI construction approach, a model directly estimates the relationship between the PI and x using the model parameter θ, such that $\hat{f}(x;\theta) = \left(\hat{l}, \hat{u}\right)$, where $\hat{l} \triangleq \hat{l}(x;\theta)$ and $\hat{u} \triangleq \hat{u}(x;\theta)$ represent the estimated lower and upper bound of the PI respectively. A PI estimation method seeks to construct an interval such that the probability of covering y equals $1 - \delta$, shown as

$$P\left(\hat{l}(x;\theta) \leq y \leq \hat{u}(x;\theta)\right) = 1 - \delta. \tag{2.18}$$

As the confidence level $1-\delta$ increases, the PI inherently widens to capture more data points, ensuring higher coverage probability.

Evaluation metrics for PI

The performance of PI is assessed based on reliability and sharpness that can be quantified from a high prediction interval coverage probability (PICP) and narrow PI width, respectively. However, these two objectives exhibit a trade-off relationship, necessitating a complete evaluation of the PI performance considering both PICP and PI width. This section outlines the fundamental evaluation metrics for PI. We denote that the sample width of the PI is given by $w_i = \hat{u}_i - \hat{l}_i$ used in the study.

1. **Prediction interval coverage probability (PICP)** is the measure of reliability calculated by counting the proportion of the observers lying within the PIs. The PICP can be defined as
$$\text{PICP} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(\hat{l}_i \leq y_i \leq \hat{u}_i), \tag{2.19}$$
where $\mathbf{1}(A)$ is a count function that returns one if event A occurs and zero otherwise. The PICP is expected to reach the confidence level that is set for PI estimation.

2. **Prediction interval width (PI width)** measures the sharpness of the prediction interval that can be assessed in terms of averaged PI width across all samples called prediction interval average width expressed as:
$$\text{PIAW} = \frac{1}{N}\sum_{i=1}^{N} w_i. \tag{2.20}$$
Additionally, there is a normalized version of the PIAW called prediction interval normalized average width (PINAW), which eliminates the scale of the target variable shown as:
$$\text{PINAW} = \frac{1}{NR}\sum_{i=1}^{N} w_i, \tag{2.21}$$
where $R = y_{\text{max}} - y_{\text{min}}$ indicates the target variable's range, computed as the difference between the maximum and minimum values of the target variable, ensuring the PINAW scale is between 0 and 1.

3. **Winkler score** is a metric that evaluates the reliability and sharpness of the PIs corresponding to the $(1-\delta)$ coverage probability invented by Winkler (1972). The Winkler score is equivalent to a pinball loss when evaluating PIs with equal-tailed probability. A low Winkler score indicates that the lower and upper bounds of the PI match well with the quantiles of $\frac{\delta}{2}$ and $1-\frac{\delta}{2}$. The Winkler score is commonly found in the literature related to PI-based methods, while the pinball loss is predominantly found in quantile-based approaches. The Winkler for each sample is defined as

$$\text{Winkler}_i = \begin{cases} |\hat{u}_i - \hat{l}_i| + \frac{2}{\delta}(\hat{l}_i - y_i), & y_i < \hat{l}_i \\ |\hat{u}_i - \hat{l}_i|, & \hat{l}_i \leq y_i \leq \hat{u}_i \\ |\hat{u}_i - \hat{l}_i| + \frac{2}{\delta}(y_i - \hat{u}_i), & y_i > \hat{u}_i \end{cases}$$
$$= |\hat{u}_i - \hat{l}_i| + \frac{2}{\delta}\left[(\hat{l}_i - y_i)\mathbf{1}(y_i < \hat{l}_i) + (y_i - \hat{u}_i)\mathbf{1}(y_i > \hat{u}_i)\right], \tag{2.22}$$

where the function can be illustrated in Figure 2.3.

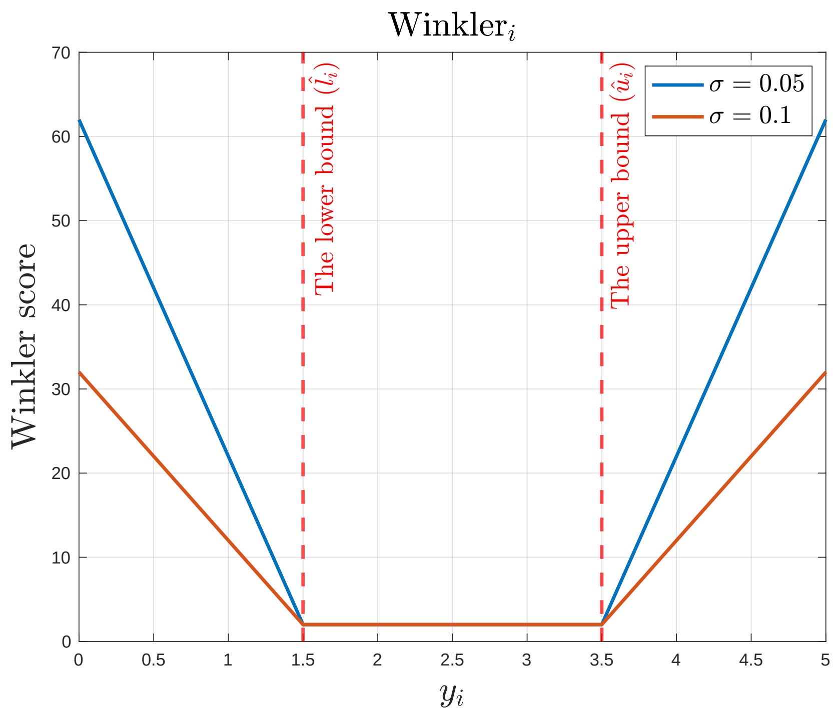


Figure 2.3: The illustration of the Winkler score evaluated in each sample.

Then, the Winkler score, with a normalization factor, is obtained by averaging the Winkler values for each sample, expressed as:

$$\text{Winkler} = \frac{1}{NR}\sum_{i=1}^{N} \text{Winkler}_i, \tag{2.23}$$

where the normalization factor R is used to eliminate data scaling effects in the Winkler score. This metric reflects how PI differs from the equal-tailed two quantiles.

Proposition 1. *The Winkler score at a confidence level $(1-\delta)$ is equivalent to the sum of two pinball losses corresponding to $\overline{\alpha} = 1 - \frac{\delta}{2}$ and $\underline{\alpha} = \frac{\delta}{2}$, respectively.*

Proof. Given that the pinball loss is written alternatively from (2.4) as

$$\begin{aligned}\rho_\alpha(r) &= \begin{cases}\alpha r, & r \geq 0\\ \alpha r - r, & r < 0\end{cases}\\ &= \alpha r - r\mathbf{1}(r < 0),\end{aligned} \tag{2.24}$$

where $\mathbf{1}(r < 0)$ equals one when $r < 0$. We utilize the pinball loss in (2.24) to demonstrate the proof, as it shares the same representation as the Winkler score.

$$\begin{aligned}&\rho_{\frac{\delta}{2}}(y_i - \hat{l}_i) + \rho_{1-\frac{\delta}{2}}(y_i - \hat{u}_i)\\ &= \left[\frac{\delta}{2}(y_i - \hat{l}_i) - (y_i - \hat{l}_i)\mathbf{1}(y_i < \hat{l}_i)\right] + \left[(1 - \frac{\delta}{2})(y_i - \hat{u}_i) - (y_i - \hat{u}_i)\mathbf{1}(y_i < \hat{u}_i)\right]\\ &= \frac{\delta}{2}(\hat{u}_i - \hat{l}_i) + (y_i - \hat{u}_i) + \left[(\hat{l}_i - y_i)\mathbf{1}(y_i < \hat{l}_i) + (\hat{u}_i - y_i)\mathbf{1}(y_i < \hat{u}_i)\right]\\ &= \frac{\delta}{2}(\hat{u}_i - \hat{l}_i) + \left[(\hat{l}_i - y_i)\mathbf{1}(y_i < \hat{l}_i) + (y_i - \hat{u}_i)\left[1 - \mathbf{1}(y_i < \hat{u}_i)\right]\right]\\ &= \frac{\delta}{2}(\hat{u}_i - \hat{l}_i) + \left[(\hat{l}_i - y_i)\mathbf{1}(y_i < \hat{l}_i) + (y_i - \hat{u}_i)\mathbf{1}(y_i > \hat{u}_i)\right]\\ &= \frac{\delta}{2}\left[(\hat{u}_i - \hat{l}_i) + \frac{2}{\delta}\left[(\hat{l}_i - y_i)\mathbf{1}(y_i < \hat{l}_i) + (y_i - \hat{u}_i)\mathbf{1}(y_i > \hat{u}_i)\right]\right]\\ &= \frac{\delta}{2}\text{Winkler}_i.\end{aligned}$$

Therefore, the sum of the pinball losses corresponding to $1 - \frac{\delta}{2}$ and $\frac{\delta}{2}$ probabilities is equivalent to the Winkler score at a confidence level of $1 - \delta$. ■

2.4 Neural network

A neural network (NN) is a computational model inspired by the human brain, consisting of interconnected neurons that process information through a structured network. Mathematically, an NN represents a complex function designed to approximate a target function for a given task. When the model is sufficiently large, NNs can capture intricate, nonlinear patterns in data.

The components defined for developing NN consist of three key components: the model, the loss function, and the training procedure. Given an input x and its corresponding target y, the model learns a mapping function $\hat{y} = f(x; \theta)$, where θ represents the model parameters, including weights and biases. These parameters are assigned to each neuron and are updated during training. The nonlinearity of NNs arises from activation functions, enabling deep networks with multiple layers to capture complex relationships.

The loss function quantifies how well the model approximates the desired output by measuring the difference between $\hat{y}$ and y. The training goal is to optimize θ such that the loss function is minimized, ensuring better model performance. This can be considered an optimization problem where the optimization variable is the model parameter. Different tasks require different loss functions, which will be discussed in this section.

Finally, the training procedure involves numerical optimization techniques to adjust θ and minimize the loss. This process iteratively updates parameters using algorithms such as gradient descent, refining the network to achieve optimal performance for the given task (Goodfellow et al., 2016).

Model

A NN model is a parametric model where the output $\hat{y}$ can be expressed as a mathematical function of the input x parametrized by the model parameters θ as

$$\hat{y} = \hat{f}(x; \theta). \tag{2.25}$$

Model parameters determine a model's ability to learn patterns in data. A greater number of model parameters corresponds to a higher degree of freedom, indicating a more complex model with an increased capacity to capture intricate patterns.

For a linear model, the output can be expressed as a linear function of θ where this model benefits from its simplicity and interpretation of the mapping between $\hat{y}$ and x. Moreover, using a simple model can be beneficial for optimization problems, particularly when leveraging efficient solving algorithms or when the solution's properties ensure global optimality, as in the case of convex optimization problems (Boyd and Vandenberghe, 2004). Examples of very popular machine learning methods utilizing linear models are linear regression, linear logistic regression, and support vector machine. However, linear models are not complex enough to capture nonlinear and intricate patterns across various applications. To utilize a linear model for mapping nonlinearity between input and output, some feature engineering is necessary to transform the original x into a nonlinear feature space. In real-world applications, these transformations can be challenging since the ground truth function relating output to input is typically unknown.

An NN is a nonlinear model parameterized nonlinearly in θ. The architecture of an NN typically refers to a deep NN, as it generally consists of multiple layers. Neural network parameters, including weights and biases, perform an affine transformation of the input from the previous layer. Nonlinearity is introduced by applying an activation function, which is a nonlinear operation, after each affine transformation. This allows the model to learn complex patterns in the data. The choice of activation function can be selected based on user preference, as the example shown in Table 2.2. Since deep NNs involve multiple layers of these operations, they can be described as a composition of numerous nonlinear functions combined with affine transformations. Neural networks typically have a large number of parameters with a deep architecture, which allows them to capture highly complex patterns, making them effective for solving intricate tasks. However, this complexity comes at a cost of interpretability, making it difficult to directly map inputs to outputs in an explainable manner. Additionally, a high number of parameters significantly raises computational requirements and necessitates large datasets for effective training, unlike simpler models. Despite these challenges, neural networks have gained significant attention due to their exceptional performance across various tasks.

Table 2.2: Common activation functions used in neural networks.

Activation Function	**Mathematical Expression**
Linear	x
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$
Hyperbolic Tangent (tanh)	$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
Gaussian	e^{-x^2}
Rectified Linear Unit (ReLU)	$\max(0, x)$
Leaky ReLU	$\max(0.01x, x)$
Softplus	$\log(1 + e^x)$
Exponential Linear Unit (ELU)	$\begin{cases} \alpha(e^x - 1), & \text{if } x \le 0 \\ x, & \text{if } x > 0 \end{cases}$

The NNs also come in diverse architectures tailored for different applications. These include feed-forward neural networks (ANNs), convolutional neural networks (CNNs) for spatial data, and sequential models such as recurrent neural networks (RNNs), long short-term memory networks (LSTMs), and transformers, which represent the state-of-the-art in modern deep learning. A specific NN architecture utilized in this thesis is detailed below.

Feed-forward neural network (ANN) model

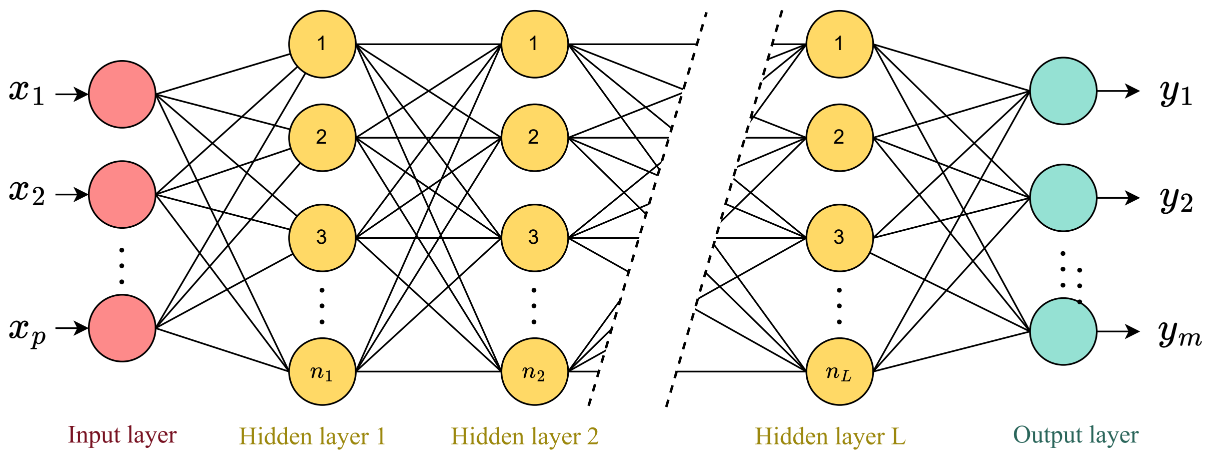
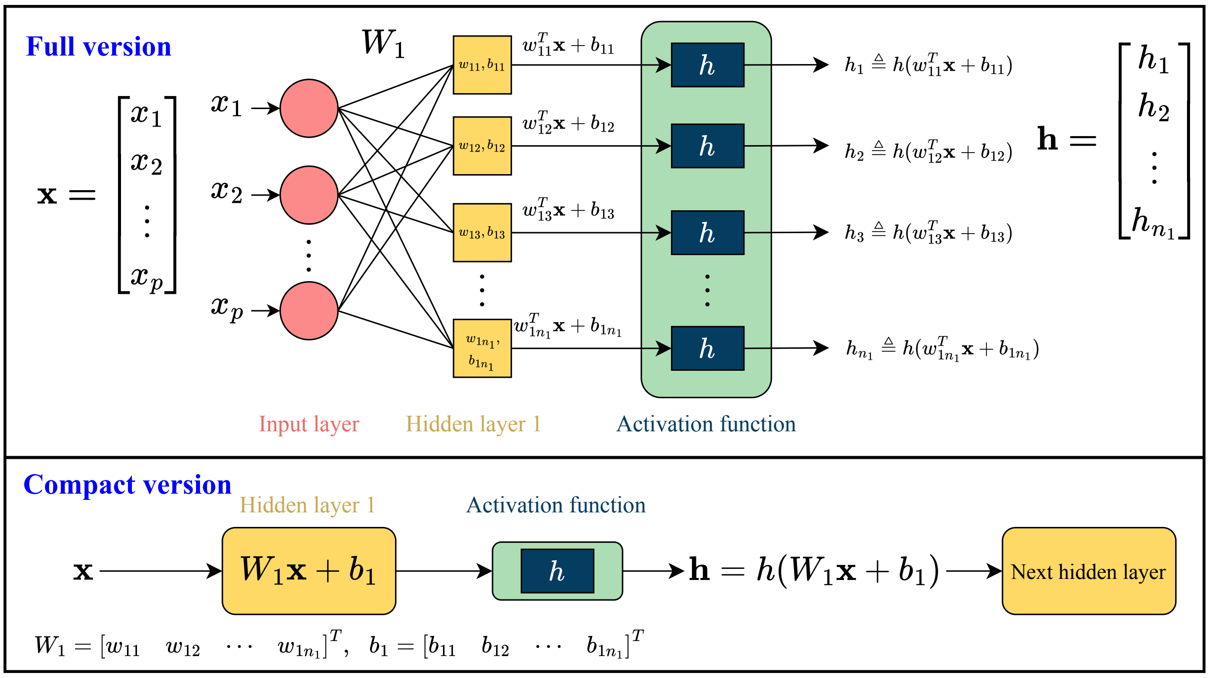


Figure 2.4: A feedforward neural network (ANN) architecture.

An ANN is the simplest neural network architecture, where the input is fed into the network, and the output is produced in a single direction without any feedback. The architecture of ANN is shown in Figure 2.4 where $x \in \mathbf{R}^p$ is the p-dimensional inputs, and $y \in \mathbf{R}^m$ is the m-dimensional outputs. Each circle in Figure 2.4 represents a neuron, and each connection between neurons indicates a corresponding weight parameter. The ANN includes an input layer, a hidden layer, and an output layer. The input layer receives x where the number of input neurons is set according to the number of features, p in this case. The hidden layer consists of many layers placed between the input and output layers. Each hidden layer is typically followed by a nonlinear activation function. In model selection, the number of hidden layers determines the model's depth, with deeper models exhibiting greater nonlinearity. The number of neurons in each layer is chosen based on the desired model capacity, including trainable parameters such as weights and biases. The output layer generates results, with the number of neurons depending on the specific learning task. For regression, a single output neuron provides the prediction. In PI estimation, two output neurons represent the upper and lower bounds. For classification tasks, the number of output neurons corresponds to the number of classes.



Full version
W_1
$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix}$
x_1
x_2
x_p
w_{11}, b_{11}
w_{12}, b_{12}
w_{13}, b_{13}
w_{1n_1}, b_{1n_1}
$w_{11}^T\mathbf{x} + b_{11}$
$w_{12}^T\mathbf{x} + b_{12}$
$w_{13}^T\mathbf{x} + b_{13}$
$w_{1n_1}^T\mathbf{x} + b_{1n_1}$
h
$h_1 \triangleq h(w_{11}^T\mathbf{x} + b_{11})$
$h_2 \triangleq h(w_{12}^T\mathbf{x} + b_{12})$
$h_3 \triangleq h(w_{13}^T\mathbf{x} + b_{13})$
$h_{n_1} \triangleq h(w_{1n_1}^T\mathbf{x} + b_{1n_1})$
$\mathbf{h} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_{n_1} \end{bmatrix}$
Input layer
Hidden layer 1
Activation function
Compact version
Hidden layer 1
Activation function
$\mathbf{x}$
$W_1\mathbf{x} + b_1$
h
$\mathbf{h} = h(W_1\mathbf{x} + b_1)$
Next hidden layer
$W_1 = [w_{11} \quad w_{12} \quad \cdots \quad w_{1n_1}]^T, \quad b_1 = [b_{11} \quad b_{12} \quad \cdots \quad b_{1n_1}]^T$

Figure 2.5: The mathematical expression of an ANN passing inputs to the first hidden layer layer.

The mathematical expression of the ANN is illustrated in Figure 2.5, illustrating the computation involved in passing inputs to the first hidden layer. Suppose the first hidden layer consists of n_1 neurons. The parameters for each mapping are represented as $w_{1i} \in \mathbf{R}^p$, denoting the weights connecting the input layer to the i^{th} neuron in the first hidden layer, and $b_{1i} \in \mathbf{R}$, representing the corresponding bias. Since each of the p input features is mapped to all hidden nodes with distinct weights, this structure is referred to as a fully connected layer.

At each neuron, an affine transformation is applied: $w_{1i}^T x + b_{1i}$, resulting in an output vector of dimension n_1, corresponding to the number of hidden neurons. Following this, a nonlinear activation function is applied elementwise, producing the hidden output, which also has a dimension of n_1. The hidden output serves as the input for subsequent hidden layers, where similar affine transformations and activation functions are performed. In deeper networks, these operations create a sequence of composite functions consisting of affine transformations followed by nonlinear activations. As the number of hidden layers increases, the number of parameters grows accordingly, introducing greater nonlinearity. Consequently, deep neural networks are often regarded as highly nonlinear models, even without explicitly analyzing their mathematical expressions.

Finally, in the output layer, another affine transformation is performed. The activation function at this stage is optional and depends on the application. For example, in two-class classification tasks where outputs represent probabilities, a sigmoid function may be applied in the final layer to force the output values between 0 and 1.

Long short-term memory (LSTM) model

A Long short-term memory (LSTM) model belongs to the class of recurrent neural networks (RNNs). RNNs are designed with a feedback mechanism, making them well-suited for capturing dependencies in sequential data, such as time series and natural language processing tasks. In an RNN, the hidden state $h(t)$ serves as a memory that retains information from previous time steps. This recurrent nature allows the model to take the hidden state from the previous time step, $h(t-1)$, as an input to compute the current hidden state while sharing the same model parameters across all time steps. This weight-sharing property enables RNNs to effectively model temporal dependencies in sequential data.

However, during training, the backpropagation process in RNNs relies on Backpropagation through time to compute gradients. This involves recursively applying the chain rule across multiple time steps, leading to repeated multiplication of gradients. As a result, RNNs often suffer from two major issues (Bengio et al., 1994; Kanai et al., 2017):

1. **Vanishing gradient:** When gradients decrease exponentially over time and approach zero, it stops weight updates in the earlier time steps. This limits the model's ability to learn long-term dependencies.

2. **Exploding gradient:** When gradients increase exponentially, they can become excessively large, which causes instability and leads to diverging weight updates.

To address these gradient issues, Long Short-Term Memory (LSTM) networks were introduced in Hochreiter and Schmidhuber (1997). LSTMs incorporate a gated mechanism, including forget $f(t)$, input $i(t)$, and output gates $o(t)$, as shown in Figure 2.6, which control the flow of information and prevent gradients from vanishing or exploding. Each gate in the LSTM is equipped with a sigmoid activation function, which outputs values between 0 and

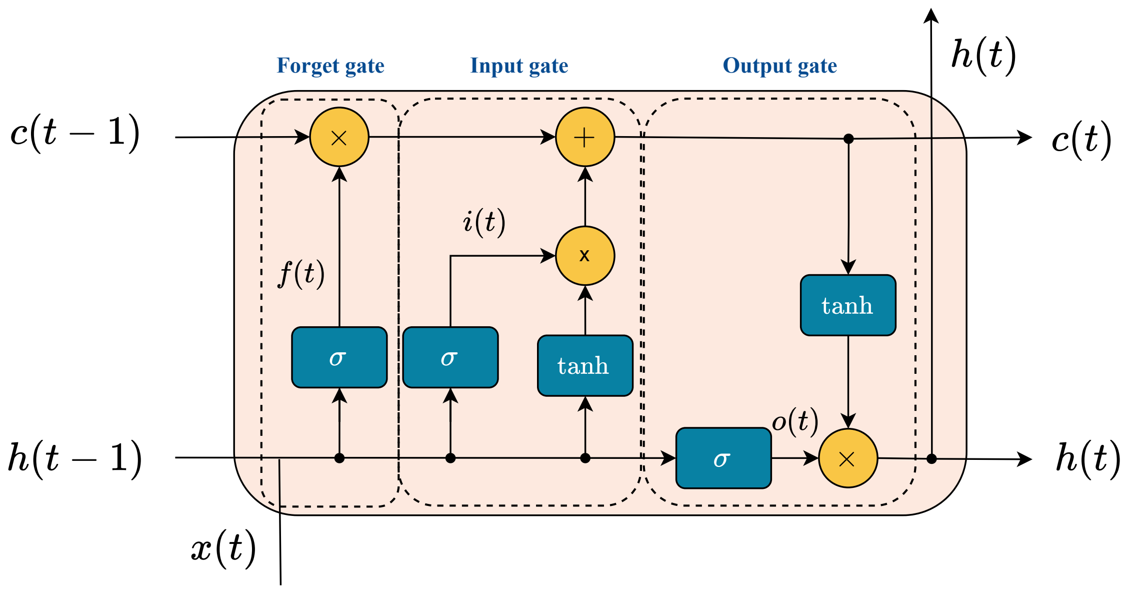


Figure 2.6: The long short-term memory (LSTM) cell architecture.

1, acting as a switch. A value of 1 allows the flow of data, while a value of 0 discards it. The forget gate determines which information from the previous cell state should be retained or discarded. The input gate decides which information should be incorporated into the current cell state, considering the previous hidden state. The output gate controls what information should be passed as the output (the next hidden state) based on the current cell state and the previous hidden state. This architecture enables LSTMs to effectively learn long-term dependencies and stabilize training. Additionally, the LSTM model contains hidden units in each layer, with users determining the number of hidden layers.

Given the input $x(t) \in \mathbf{R}^p$ where $c(t)$ is the cell state of LSTM, and d is the number of hidden units, the computations within the LSTM cell can be expressed as follows:

$$\begin{aligned}
f(t) &= \sigma\left(W_f x(t) + U_f h(t-1) + b_f\right) \\
i(t) &= \sigma\left(W_i x(t) + U_i h(t-1) + b_i\right) \\
o(t) &= \sigma\left(W_o x(t) + U_o h(t-1) + b_o\right) \\
c(t) &= f(t) \odot c(t-1) + i(t) \odot \tanh\left(W_c x(t) + U_c h(t-1) + b_c\right) \\
h(t) &= o(t) \odot \tanh\left(c(t)\right),
\end{aligned}$$

where $f(t), i(t), o(t), c(t), h(t) \in \mathbf{R}^d$, $W \in \mathbf{R}^{p\times d}, U \in \mathbf{R}^{d\times d}$ are the model weights, with subscripts corresponding to each gate, and $b \in \mathbf{R}^d$ is the bias vector associated with each gate (Hochreiter and Schmidhuber, 1997; Gers et al., 1999). These weights and biases for all gates are treated as trainable parameters for LSTM. These gating mechanisms prevent the model from suffering from the vanishing gradient problem, while the hyperbolic tangent (tanh) function scales the values within the range of -1 to 1, helping to stabilize model training and eliminate the risk of exploding gradients.

Architecture Building

The NNs include various architectures, such as ANN, CNN, and RNN, each providing advantages suited for different tasks. In real-world applications, NN architectures are designed to align with specific problem requirements. Each NN type can be viewed as a building block for extracting meaningful features from input data. These blocks can be combined to form larger networks, enabling the model to capture complex patterns more effectively for specific tasks. The interconnection of these computational blocks can be structured sequentially or in parallel. An example of the architecture design for multi-step prediction interval solar forecasting is shown in Figure 6.4 within Section 6.2. Although increasing network complexity leads to a higher number of parameters and computations, the fundamental training procedure remains unchanged. Training involves backpropagation and gradient-based optimization techniques, where gradients propagate from the output layer to the input layers, iteratively updating model parameters.

Loss function

A loss function, denoted as $\mathcal{L}(\theta)$, is a mathematical function that quantifies the discrepancy between predicted and target values. It is typically designed such that lower values indicate better performance, making the objective to minimize the loss function. The loss function defines the learning objective of an NN model. During training, the NN seeks to optimize its parameters by solving the optimization problem:

$$\underset{\theta}{\text{minimize}} \;\; \mathcal{L}(\theta)$$

where $\mathcal{L}(\theta)$ is based on a specific task. The different tasks have distinct objectives and errors characteristics. The choice of loss function directly influences how the model learns and optimizes its parameters. Even within the same task, different mathematical formulations can lead to varying solution behaviors. A loss function that imposes stronger penalties on certain terms will cause the model to prioritize minimizing those terms over others. Furthermore, the selection of the loss function affects the training process, as some loss functions possess favorable characteristics in gradient calculation, like being smooth functions, which demonstrate better convergence, whereas more complicated functions may pose challenges during training. We provide the example of loss function in the regression and classification task below.

For regression tasks, the model aims to find θ that minimizes the deviation of $\hat{y}(x;\theta)$ from the continuously valued target variable. The commonly used loss functions include mean square error (MSE), mean absolute error (MAE), and Huber loss, as shown in Table 2.3. These three losses differ in terms of robustness. The MSE has a squares term that penalizes large errors more heavily, making it more sensitive to outliers, while the others are more robust to outliers.

For classification tasks, the model aims to estimate the probability distribution over the possible classes and assign the input to the class with the highest predicted probability. The

Table 2.3: Common regression loss functions.

Loss	**Mathematical Expression**
MSE	$\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i(x;\theta)\right)^2$
MAE	$\frac{1}{N}\sum_{i=1}^{N}\lvert y_i - \hat{y}_i(x;\theta)\rvert$
Huber	$\frac{1}{N}\sum_{i=1}^{N} h\left(y_i - \hat{y}_i(x;\theta)\right), \quad h(x,\delta) = \begin{cases} \frac{1}{2}x^2, & \lvert x\rvert \le \delta \\ \delta\lvert x\rvert - \frac{1}{2}\delta^2, & \lvert x\rvert > \delta \end{cases}$

loss function in classification quantifies the discrepancy between the predicted probabilities and the true class labels, penalizing incorrect classifications. Commonly used classification loss functions are presented in Table 2.4.

Table 2.4: Common classification loss functions where $\hat{y}_i = \hat{y}_i(x;\theta)$.

Loss	**Mathematical Expression**
Binary cross-entropy	$-\sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i), \quad y_i \in \{0,1\}, \hat{y}_i \in [0,1]$
Hinge loss	$\sum_{i=1}^{N} \max(0, 1 - y_i \cdot \hat{y}_i), \quad y_i \in \{-1,1\}, \hat{y}_i \in (-1,1)$
K-classes cross-entropy	$-\sum_{i=1}^{N}\sum_{j}^{K} y_{ij}\log(\hat{y}_{ij}), \quad y_{ij} \in \{0,1\}, \hat{y}_{ij} \in [0,1]$

Training and validation

Once the loss function and model architecture are defined, the model seeks to optimize its parameters to minimize the loss. This process, known as model training, involves solving an optimization problem. Since NNs represent highly nonlinear functions, their optimization can typically be cast as an unconstrained nonlinear program. In many NN tasks, the loss function is typically smooth (continuously differentiable), enabling gradient-based algorithms to be utilized, with gradient computation carried out by backpropagation. Additionally, NNs contain a vast number of parameters, treated as optimization variables, making gradient-based methods well-suited for navigating the high-dimensional loss landscape efficiently.

Gradient descent

The simplest gradient-based optimization algorithm is gradient descent. It requires a smooth loss function that is continuously differentiable as it relies on computing the gradient of the loss with respect to the model parameters. Algorithm 1 outlines the algorithm for updating model parameters. The updates are performed iteratively in the direction that reduces the loss function, particularly along the negative gradient. The algorithm includes a hyperparameter known as the learning rate (step size), which is typically chosen based on the properties of the loss function. A larger learning rate can accelerate convergence but may also lead to divergence or instability in training, whereas a very small learning rate can result in slow convergence. The model parameters are updated until the stopping criterion is satisfied, which will be explained in more detail later.

Algorithm 1 Gradient Descent Algorithm

```
Require: Learning rate $\epsilon$
Require: Loss function $\mathcal{L}(\theta)$
Require: Initial model parameters $\theta_0$
1: while stopping criterion not met do
2:     Compute gradient: $\nabla\mathcal{L}(\theta)$
3:     Update model parameters: $\theta \leftarrow \theta - \epsilon\nabla\mathcal{L}(\theta)$
4: end while
5: return $\theta$
```

Mini-batch optimization

In NN-based loss functions, the loss is typically computed as a sum over the training samples. However, for many NN tasks, the dataset can be vast, making the evaluation of the loss function over the entire training set computationally expensive. Moreover, using only a smaller random subset of the dataset can often provide sufficient information to determine the update direction for the model parameters. To address this, mini-batch optimization is introduced, which computes the gradient step based on a random batch of data. In mini-batch optimization, the dataset is split into batches of a size determined by the user. Each batch is used to update the model parameters until all batches have been processed. Algorithm 2 illustrates the application of mini-batch optimization in the gradient descent algorithm. In the context of NN training, an *epoch* refers to one complete pass through an entire dataset, consisting of many batches, which depends on the batch size and the number of training samples.

A primary benefit of mini-batch optimization is that it significantly accelerates convergence compared to using the entire dataset (Goodfellow et al., 2016). Additionally, it reduces computational memory requirements by loading only smaller portions of the dataset into memory. Furthermore, mini-batch optimization introduces a stochastic element into the process through randomized sampling, which can help improve generalization. A special case of mini-batch optimization is when the batch size is set to one, known as *stochastic gradient descent (SGD)*, where the model parameters are updated after each individual training sample.

The selection of batch size is crucial. Smaller batch sizes can lead to greater fluctuations in the gradient estimates. While this fluctuation can potentially help the model generalize better, it may also cause instability (fluctuation) in the loss. On the other hand, larger batch sizes provide more accurate gradient estimates with less noise, resulting in more stable convergence. However, they come at the cost of requiring more memory and computational time per step.

Algorithm 2 Mini-Batch Gradient Descent

```
Require: Learning rate $\epsilon$, batch size $B$, dataset $\mathcal{D}$ of size $N$
Require: Loss function $\mathcal{L}(\theta)$
Require: Initial model parameters $\theta_0$
1: while stopping criterion not met do
2:     Shuffle dataset $\mathcal{D}$
3:     Partition $\mathcal{D}$ into $\lceil N/B \rceil$ mini-batches
4:     for each mini-batch $\mathcal{B} \subset \mathcal{D}$ do
5:         Compute mini-batch gradient: $\nabla \mathcal{L}_{\mathcal{B}}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla \mathcal{L}_i(\theta)$
6:         Update parameters: $\theta \leftarrow \theta - \epsilon \nabla \mathcal{L}_{\mathcal{B}}(\theta)$
7:     end for
8: end while
9: return $\theta$
```

Momentum-based algorithm

Momentum. Machine learning problems often involve optimizing a large number of parameters over extensive datasets. In such cases, traditional gradient descent can exhibit zig-zagging behavior, where the solution oscillates around the optimal point without efficiently converging. This issue arises due to inconsistent gradient directions, particularly in regions with high curvature. To accelerate convergence and reduce oscillations, the momentum-based algorithm is introduced. It smooths updates by incorporating an exponentially weighted moving average of past gradients. This introduces an auxiliary variable called velocity v, inspired by physics, which accumulates past gradients and acts as a driving force for parameter updates. The momentum update rule is given by:

$$\begin{aligned} v &\leftarrow \beta v - \epsilon \nabla \mathcal{L}(\theta) \\ \theta &\leftarrow \theta + v \end{aligned}$$

where $\beta \in [0, 1)$ is the momentum coefficient, controlling the influence of past gradients. A higher β leads to smoother updates but may cause overshooting near the optimum. Momentum-based methods help accelerate convergence, particularly in high-curvature regions, by reducing oscillations and reducing sensitivity to noisy gradients. A commonly recommended value is $\beta = 0.9$, which balances stability and speed. Moreover, momentum can help escape shallow local minima, as past gradients contribute to continued movement.

Normalized gradient descent. In high-dimensional optimization problems, flat regions exist in which the loss function remains nearly constant along multiple parameter directions. In such cases, the gradient magnitude can vanish significantly, leading to a stop in parameter updates. To mitigate this issue, normalized gradient descent is introduced to control the step size by normalizing the gradient. There are two approaches to this normalization. The first method involves normalizing the step size with the gradient magnitude, typically $\|\nabla \mathcal{L}(\theta)\|_2$. This ensures that when the gradient magnitude is small, the step size increases, preventing the updating of parameters. However, this approach can be problematic in cases where gradients vanish along specific directions, leading to the failure of parameter updates in those dimensions. The second approach involved normalizing the gradient component-wise for each parameter, a technique commonly referred to as the *adaptive learning rate*. By assigning a separate learning rate to each parameter and dynamically adjusting it based on the gradient, this method improves the solution's trajectory across a complex loss surface. Adaptive learning rates change the search direction from a gradient descent, improving convergence stability and efficiency. Several well-known optimization algorithms, such as AdaGrad (Duchi et al., 2011a), RMSProp (Tieleman and Hinton, 2012), and Adam (Kingma and Ba, 2017), incorporate this technique to enhance learning performance, particularly in high-dimensional spaces.

Adaptive moment estimation (Adam). Adam is one of the most widely used optimizers in NN tasks due to its computational efficiency and lower memory requirements. Adam combines the advantages of both momentum and adaptive learning rate techniques. For momentum, Adam incorporates the exponential moving average of gradients (first-moment estimate or the mean) and the squared gradients (second-moment estimate or the uncentered variance), where the decay rates are controlled by $\beta_1, \beta_2 \in [0, 1)$ respectively. Adam also applies a bias correction term in both moments, which converges to one, to address the bias in zero initialization. Additionally, it employs an adaptive learning rate by normalizing the first-moment estimate with the second-moment estimate, effectively adjusting the step size for each parameter. The entire procedure of Adam is outlined in Algorithm 3.

Algorithm 3 Adam Optimization Algorithm (Kingma and Ba, 2017), The original paper suggests setting $\epsilon = 0.001$, $\beta_1 = 0.9, \beta_2 = 0.999$, and $\delta = 10^{-8}$. All operations are element-wise. $[\nabla\mathcal{L}(\theta)]^2$ refers to elementwise square $\nabla\mathcal{L}(\theta) \odot \nabla\mathcal{L}(\theta)$.

```
Require: Learning rate $\epsilon$, exponential decay rates $\beta_1, \beta_2 \in [0, 1)$, small constant $\delta$
Require: Loss function $\mathcal{L}(\theta)$
Require: Initial parameters $\theta_0$
 1: Initialize $1^{\text{st}}$ and $2^{\text{nd}}$ moment vector $m_0 \leftarrow 0$, $v_0 \leftarrow 0$
 2: Initialize time step $t \leftarrow 0$
 3: while stopping criterion not met do
 4:     $t \leftarrow t + 1$
 5:     Compute gradient: $\nabla\mathcal{L}(\theta)$
 6:     Update biased first moment estimate: $m \leftarrow \beta_1 m + (1 - \beta_1)\nabla\mathcal{L}(\theta)$
 7:     Update biased second moment estimate: $v \leftarrow \beta_2 v + (1 - \beta_2)\,[\nabla\mathcal{L}(\theta)]^2$
 8:     Compute bias-corrected first moment estimate: $\hat{m} \leftarrow \frac{m}{1-\beta_1^t}$
 9:     Compute bias-corrected second moment estimate: $\hat{v} \leftarrow \frac{v_t}{1-\beta_2^t}$
10:     Update parameters: $\theta \leftarrow \theta - \epsilon\frac{\hat{m}}{\sqrt{\hat{v}}+\delta}$        ▷ Operation applied element-wise.
11: end while
12: return $\theta$
```

Training algorithm and early stopping for NN

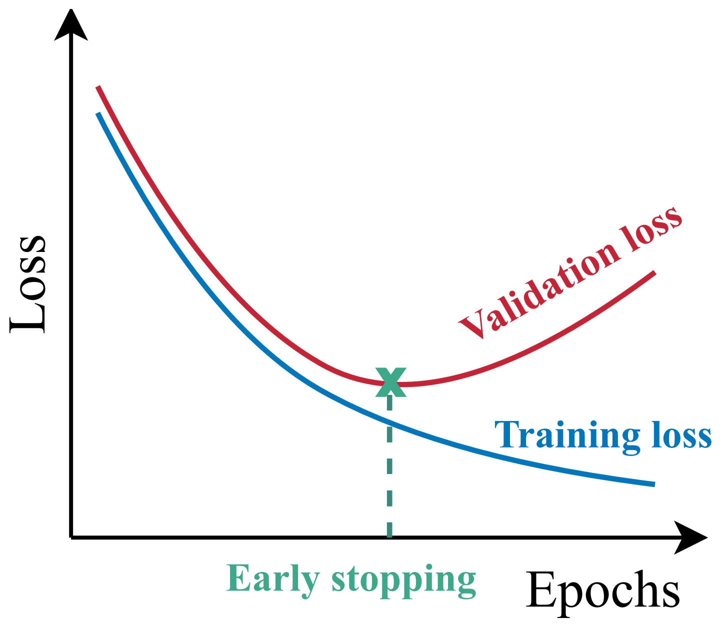


Figure 2.7: The training and validation loss for minimizing loss function in a neural network.

Training NNs differs from traditional optimization due to their complexity, nonlinearity, and high-dimensional parameter space. The optimization process typically begins with randomly initializing model parameters using a Gaussian or uniform distribution, which is automatically assigned when constructing the model architecture. The loss function is evaluated from the model's outputs and dataset, and the optimizer updates the parameters to minimize the training loss. However, this process risks overfitting, meaning the model may not generalize well to unseen data.

Gradient-based optimization methods theoretically ensure that the training loss decreases over time, which could lead to overfitting. To address this, training is typically controlled using a maximum number of epochs as a stopping criterion. Additionally, early stopping is commonly used to prevent overfitting by terminating training once the model's generalization performance stops improving. Implementing early stopping requires splitting the dataset into training and validation sets. The training set is used to update model parameters, while the validation set assesses model generalization after each epoch. Initially, both training and validation losses tend to decrease. However, after several epochs, the validation loss may start increasing, indicating reduced generalization as shown in Figure 2.7. To apply early stopping, training is terminated when the validation loss begins to rise, ensuring that the model parameters correspond to the lowest validation loss. The `patience` parameter is introduced to further refine early stopping, allowing training to continue for a specified number of epochs without improvement in validation loss before termination. Algorithm 4 illustrates the training procedure incorporating early stopping, mini-batch optimization, and gradient descent.

Algorithm 4 Early stopping during the training of a NN with a mini-batch optimization

```
Require: Training dataset $\mathcal{D}_{\text{train}}$ of size $N$, validation dataset $\mathcal{D}_{\text{validate}}$
Require: Learning rate $\epsilon$, batch size $B$
Require: Loss function $\mathcal{L}(\theta)$, Initial model parameters $\theta_0$
Require: max_epochs, patience
 1: Initialize epoch counter $t \leftarrow 0$, patience_counter $\leftarrow 0$
 2: Initialize best_validation_loss $\leftarrow \infty$, $\theta^\star \leftarrow \theta_0$
 3: while $t <$ max_epochs do
 4:     $t \leftarrow t + 1$
 5:     Shuffle dataset $\mathcal{D}_{\text{train}}$
 6:     Partition $\mathcal{D}_{\text{train}}$ into $\lceil N/B \rceil$ mini-batches
 7:     for each mini-batch $\mathcal{B} \subset \mathcal{D}_{\text{train}}$ do
 8:         Compute mini-batch gradient: $\nabla \mathcal{L}_{\mathcal{B}}(\theta)$
 9:         Update parameters $\theta$: $\theta \leftarrow \theta - \epsilon \nabla \mathcal{L}_{\mathcal{B}}(\theta)$   ▷ Formulation depends on algorithm.
10:     end for
11:     Evaluate training and validation loss: $\mathcal{L}_{\mathcal{D}_{\text{train}}}(\theta)$, $\mathcal{L}_{\mathcal{D}_{\text{validate}}}(\theta)$   ▷ For loss monitoring.
12:     if $\mathcal{L}_{\mathcal{D}_{\text{validate}}}(\theta) <$ best_validation_loss then
13:         best_validation_loss $\leftarrow \mathcal{L}_{\mathcal{D}_{\text{validate}}}(\theta)$
14:         $\theta^\star \leftarrow \theta$                                   ▷ Save the best $\theta$.
15:         patience_counter $\leftarrow 0$                          ▷ Reset the counter.
16:     else
17:         patience_counter $\leftarrow$ patience_counter $+ 1$
18:         if patience_counter $\geq$ patience then
19:             break         ▷ Early stopping when no improvement in validation loss.
20:         end if
21:     end if
22: end while
23: return $\theta^\star$
```

2.5 Lower upper bound estimation (LUBE) method

A lower upper bound estimation (LUBE) is a technique that provides a PI directly from a model without a distribution assumption for the target variable. The model directly maps the input features x to the upper and lower bounds to capture an uncertainty of y. The literature presents various models that establish PIs, including SVM (Shrivastava et al., 2015), ELM (Zhang et al., 2015; Zhao et al., 2020, 2022), and neural networks (NN), where the latter has been the most widely used option. Typically, the NN-based model is implemented with two output neurons representing the upper and lower bounds, as illustrated in Figure 2.8. The PI with the $1-\delta$ confidence level is obtained by minimizing the PI-based loss function. The PI-based loss function has two objectives: PICP and PI width, which can be combined either multiplicatively or additively into a single loss function. The PI-based loss function was originally introduced as the coverage-width criterion (CWC)

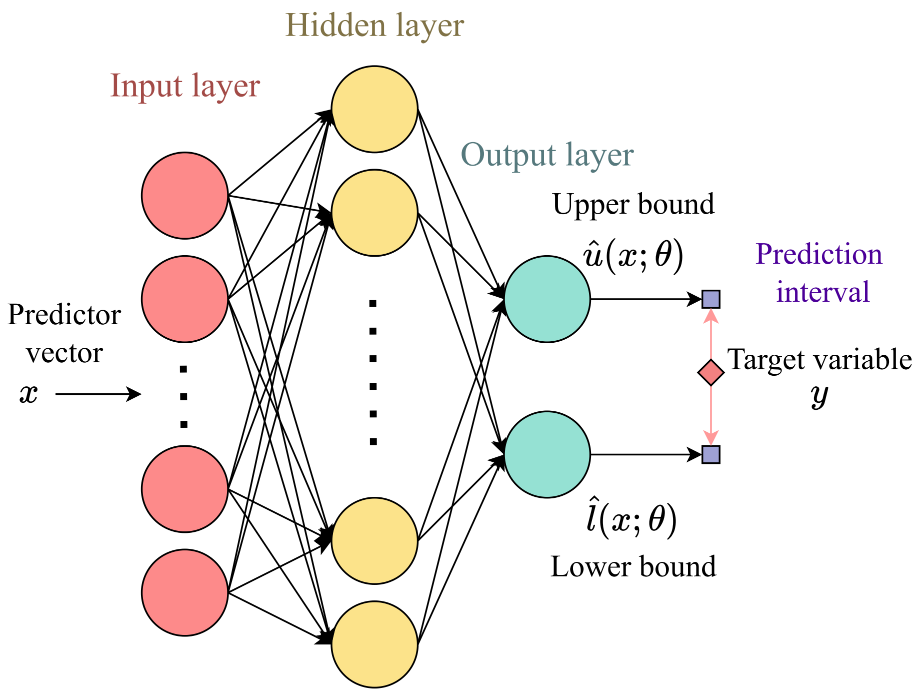


Figure 2.8: The NN model with two outputs utilized in the LUBE framework.

loss in Khosravi et al. (2011). The CWC function integrates both objectives using the following multiplicative form:

$$\mathsf{CWC}_{\mathsf{ori}}(\theta) = \mathsf{PINAW}(\theta)\left(1 + e^{\gamma \max(0,(1-\delta)-\mathsf{PICP}(\theta))}\right), \tag{2.26}$$

where γ balances the trade-off between the two objectives. The alternative versions of CWC loss are also represented in the multiplicative form as noted in Quan et al. (2014a); Ye et al. (2016). However, Shrivastava et al. (2015); Pearce et al. (2018) mention that the global minimum for (2.26) occurs when the PINAW has a PI width of zero, which is commonly an undesirable solution. Following this, additive versions of CWC are alternatively suggested to address the multiplicative drawback identified in the studies by Shrivastava et al. (2015); Shi et al. (2018).

The PICP term requires computing a count function in (2.19), which results in a non-differentiable loss. As a result, a population-based algorithm, which is a heuristic approach, is usually applied to identify the optimal model parameter. Recently, the log-likelihood-based loss in an additive form called a quality-driven (QD) loss was proposed in Pearce et al. (2018). The PI width term in the QD loss is calculated by averaging only the samples with y_i captured by the PI as:

$$\mathsf{PIAW}_{\mathsf{capt.}} = \frac{1}{N_{\mathsf{capt.}}} \sum_{i=1}^{N} w_i \mathbf{1}(\hat{l}_i \le y_i \le \hat{u}_i) \tag{2.27}$$

where $N_{\text{capt.}} = \sum_{i=1}^{N} \mathbf{1}(\hat{l}_i \le y_i \le \hat{u}_i)$ is the number of samples captured by the PI. This guarantees that the model does not gain advantages from the reduced PI width in the sample that is not covered by the PI. For the PICP term, with i.i.d. assumption for all samples, a value of the count function $\mathbf{1}(\hat{l}_i \le y_i \le \hat{u}_i)$ can be treated as a Bernoulli random variable with probability $1 - \delta$. Consequently, the number of covered samples $N_{\text{capt.}}$ is modeled as Binomial$(N, (1-\delta))$ shown as

$$\mathcal{L}_{\text{PICP}} = \binom{N}{N_{\text{capt.}}} (1-\delta)^{N_{\text{capt.}}} \delta^{N - N_{\text{capt.}}} \tag{2.28}$$

where $\mathcal{L}_{\text{PICP}}$ is the likelihood function of $N_{\text{capt.}}$.

The central limit theorem allows for approximating the probability mass function of the binomial distribution as a normal distribution when N is sufficiently large:

$$\begin{aligned} \text{Binomial}(N, (1-\delta)) &\approx \mathcal{N}\left(N(1-\delta), N\delta(1-\delta)\right) \\ &= \frac{1}{\sqrt{2\pi N\delta(1-\alpha)}} \exp - \frac{(N_{\text{capt.}} - N(1-\alpha))^2}{2N\delta(1-\delta)}. \end{aligned}$$

Hence, the negative log-likelihood of $N_{\text{capt.}}$ can be approximated as:

$$-\log \mathcal{L}_{\text{PICP}} \propto \frac{N}{\delta(1-\delta)} ((1-\delta) - \text{PICP})^2, \tag{2.29}$$

where $\text{PICP} = \frac{N_{\text{capt.}}}{N}$. Finally, the $\max(0, \cdot)$ is equipped to penalize only when $\text{PICP} < 1 - \delta$, resulting in the QD loss expressed as:

$$\text{Loss}_{\text{QD}}(\theta) = \text{PIAW}_{\text{capt.}}(\theta) + \gamma \frac{N}{\delta(1-\delta)} \max(0, (1-\delta) - \text{PICP}(\theta))^2, \tag{2.30}$$

where γ controls a trade-off level between two objectives. Moreover, in Pearce et al. (2018), a smooth approximation of the count function is introduced in the PICP term to ensure that the QD loss is differentiable and that gradient-based algorithms can be applied for optimization.

Chapter III

METHODOLOGY

This section outlines the proposed methodologies for providing prediction intervals (PI). Better PI is characterized by a higher coverage probability (PICP) and a narrower PI width, which are conflicting objectives. We aim to formulate optimization problems that deliver PI with the desired PICP and narrow PI width. This thesis proposes two schemes for these optimization formulations: a pinball-based formulation and a PICP with width control formulation. Both approaches comprise two components: one that controls PICP and the other that manages the PI width. For the pinball-based formulation, which is the first methodology, we present three optimization formulations that control PI width in different ways, whereas the PICP with width control formulation, the second methodology, consists of a single optimization formulation.

For a dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$, we define $\mathcal{I}$ as the index set for data samples, $i \in \mathcal{I}$, encompassing N samples where $x \in \mathbf{R}^p$ represent a p-dimensional predictor vector and $y \in \mathbf{R}$ denotes a target variable. The estimated PI is represented by the upper and lower bounds denoted as $\hat{u}(x;\theta)$ and $\hat{l}(x;\theta)$, respectively, with a confidence level of $(1-\delta)$, where θ refers to a model parameter.

3.1 Pinball-based formulation

The pinball-based formulation incorporates a pinball loss as part of the objective, which aims to estimate two quantiles corresponding to a specified confidence level. This methodology offers an alternative approach to achieving the desired PICP without using nonlinear functions in PICP due to the count function. The pinball function is utilized instead of the PICP function to maintain simplicity in the formulation and facilitate a simple numerical method for addressing these problems. The pinball loss exhibits favorable optimization properties because of its convexity. For the PI width term, we introduce three different width control functions: average width, sum of the K-largest widths, and maximum PI width, all of which are convex functions. We utilize a linear additive form, which is linearly parameterized by the model parameters. This facilitates preserving the problem's convexity because the composition of a convex function with an affine mapping remains convex. This enables us to formulate three pinball-based formulations as convex programs.

Methodology

Figure 3.1 shows the methodology of the pinball-based approach. The methodology can be divided into three parts: model, optimization formulation, and optimization technique. The model receives x as an input and provides the upper and lower bound as an output. The

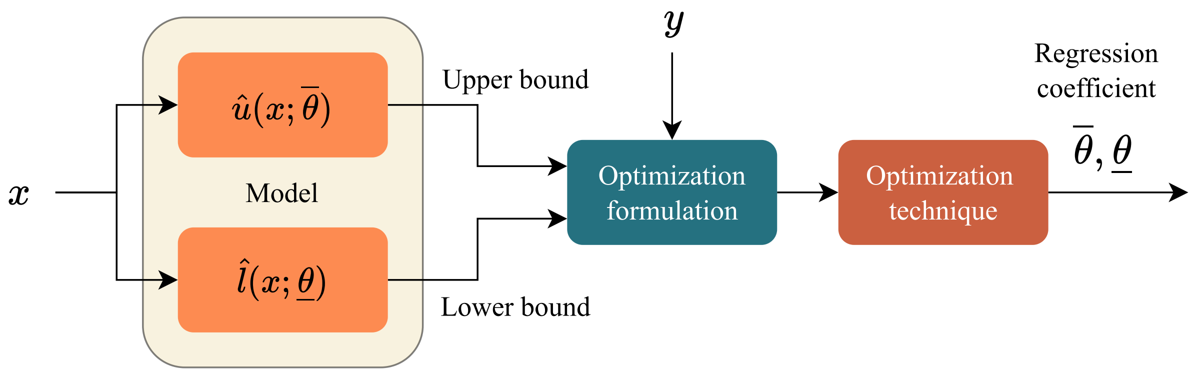


Figure 3.1: The methodology for the PI construction of pinball-based approach.

upper and lower bounds are estimated using different model parameters: $\overline{\theta}$ for an upper bound and $\underline{\theta}$ for a lower bound. In this approach, the linear additive model is applied, so the upper and lower bound can be written as:

$$\begin{aligned} \hat{u}(x;\overline{\theta}) &= \overline{\theta}_0 + \overline{\theta}_1 g_1(x) + \overline{\theta}_2 g_2(x) + \cdots + \overline{\theta}_p g_p(x) \\ \hat{l}(x;\underline{\theta}) &= \underline{\theta}_0 + \underline{\theta}_1 g_1(x) + \underline{\theta}_2 g_2(x) + \cdots + \underline{\theta}_p g_p(x) \end{aligned} \tag{3.1}$$

where $g_i(x)$ for $i = 1, \dots, p$ are basis functions that explains the characteristics of y. Next, the upper and lower bounds are incorporated into the optimization formulation with y. The optimization formulation outlines the objective of designing the PI to achieve the desired characteristics for controlling PI width, where the mathematical detail will be described later. Then, the optimization technique is a numerical method used to solve the proposed optimization formulations to obtain the optimal model parameters for the estimated upper and lower bounds.

Mathematical formulation

The three proposed optimization formulations utilize the same optimization framework described in (3.2).

$$\begin{aligned} \underset{\underline{\theta},\overline{\theta}}{\text{minimize}} \quad & \sum_{i\in\mathcal{I}} \rho_{\overline{\alpha}}(y_i - \hat{u}_i(\overline{\theta})) + \rho_{\underline{\alpha}}(y_i - \hat{l}_i(\underline{\theta})) \\ \text{subject to} \quad & \hat{l}_i(\underline{\theta}) \leq \hat{u}_i(\overline{\theta}), \quad \forall i \in \mathcal{I} \\ & \mathcal{W}(\overline{\theta},\underline{\theta}) \leq \gamma \cdot \text{sample width}, \quad \forall i \in \mathcal{I}, \end{aligned} \tag{3.2}$$

The objective function is the sum of the pinball loss function corresponding with two quantiles $\overline{\alpha}$ and $\underline{\alpha}$ for the upper and lower bound, respectively. Minimizing the objective function without constraints forces the upper and lower bounds to estimate the conditional quantile of y given x, corresponding to $\overline{\alpha}$ and $\underline{\alpha}$ probabilities. To provide the PI with a $1-\delta$ confidence level, the $\overline{\alpha}$ is set to $1-\delta/2$ while the $\underline{\alpha}$ is set to $\delta/2$. This configuration ensures a symmetric PI with equal probability in both tails. The first constraint ensures that the upper bound is greater than or equal to the lower bound, thus preventing any misbehavior from the PI. Given that $\mathcal{W}(\overline{\theta}, \underline{\theta})$ is a width function, the second constraint regulates the width function, which is presented in three different formulations, each aimed at addressing various aspects of the PI width. Next, the *sample width* of the target variable (y) is calculated as

$$\text{sample width} = q_y(\overline{\alpha}) - q_y(\underline{\alpha}), \tag{3.3}$$

where $q_y(\overline{\alpha})$ and $q_y(\underline{\alpha})$ are the sample quantiles of y corresponding to the $\overline{\alpha}$ and $\underline{\alpha}$ probabilities. The *sample width* calculated from the training data serves as the problem parameter. The proposed formulations aim to reduce the PI width from the sample width using a multiplicative factor, $\gamma \in (0, 1)$. The γ serves as a predefined formulation hyperparameter that regulates the level of penalization in the width function. When γ is decreased, the inequality becomes more stringent, so the PI width function tends to decrease. The stronger inequality constraint increases the objective function; therefore, an increase in pinball loss indicates the greater deviation of the upper and lower bounds from the quantiles corresponding to the probabilities $\overline{\alpha}$ and $\underline{\alpha}$. This leads to reduced PICP from the desired coverage probability. This scheme involves the trade-off characteristics between PICP and the width function by varying γ. The expected trade-off characteristics are desired to slightly decrease PI width while maintaining the PICP as the desired coverage probability. Equivalently, the width function constraint can also be interpreted as a regularization term within the objective function. Consequently, this modifies the objective function into a multi-objective problem that seeks to minimize both the pinball loss and the width function, incorporating a penalization parameter to regulate the penalization level in the PI width component. The increase in the penalty for PI width decreases the PI width term and increases the pinball loss term, causing the upper and lower bounds to mismatch with the quantiles corresponding with $\overline{\alpha}, \underline{\alpha}$ probabilities. This also leads to a reduction in PICP from the desired coverage probability, which is the same result as considering the width function as a constraint. We propose three formulations with different width functions, referred to as formulations P1, P2, and P3, described below.

Formulation P1 (pinball loss with controlled average width)

Formulation P1 follows (3.2) where the width function is the average width shown as

$$\begin{aligned}
\underset{\underline{\theta}, \overline{\theta}}{\text{minimize}} \quad & \sum_{i \in \mathcal{I}} \rho_{\overline{\alpha}}(y_i - \hat{u}_i(\overline{\theta})) + \rho_{\underline{\alpha}}(y_i - \hat{l}_i(\underline{\theta})) \\
\text{subject to} \quad & \hat{l}_i(\underline{\theta}) \leq \hat{u}_i(\overline{\theta}), \quad \forall i \in \mathcal{I} \\
& \frac{1}{N} \sum_{i \in \mathcal{I}} (\hat{u}_i(\overline{\theta}) - \hat{l}_i(\underline{\theta})) \leq \gamma \cdot \text{sample width}.
\end{aligned} \tag{3.4}$$

This formulation aims to minimize the pinball loss with a reduction in the average PI width simultaneously. The formulation includes one predefined formulation hyperparameter, γ, to regulate the level of penalization in the average PI width. Due to the first inequality that forces the upper bound to be greater than the lower bound, the sample PI width $\hat{u}_i(\overline{\theta})-\hat{l}_i(\underline{\theta})$ is always non-negative. So, the summation of the sample PI width can be considered as an ℓ_1-norm on the PI width. As a result, the number of inequality constraints is $N+1$, where N comes from the first constraint, and one comes from the average PI width control constraint.

Formulation P2 (pinball loss with controlled large widths)

Given that the sample width is written as $w_i = \hat{u}_i(\overline{\theta})-\hat{l}_i(\underline{\theta})$, we denote $w_{[i]}$ as the i^{th} largest PI width element where $w_{[1]} \geq w_{[2]} \geq \cdots \geq w_{[N]}$. Formulation P2 follows (3.2), which aims to control large PI widths by using the average of the K-largest elements of the PI width as a width function written as

$$\begin{array}{ll} \underset{\underline{\theta},\overline{\theta}}{\text{minimize}} & \sum_{i\in\mathcal{I}} \rho_{\overline{\alpha}}(y_i-\hat{u}_i(\overline{\theta}))+\rho_{\underline{\alpha}}(y_i-\hat{l}_i(\underline{\theta})) \\ \text{subject to} & \hat{l}_i(\underline{\theta}) \leq \hat{u}_i(\overline{\theta}), \quad \forall i \in \mathcal{I} \\ & \dfrac{1}{K}\sum_{i=1}^{K} w_{[i]} \leq \gamma \cdot \text{sample width}, \quad \forall i \in \mathcal{I}. \end{array} \tag{3.5}$$

This formulation involves two predefined formulation hyperparameters, γ and K. The γ controls the penalization of the large PI width term similarly to P1. Lowering γ typically results in a reduction of the large PI width because of a stricter inequality. The hyperparameter K defines the number of samples considered as large PI widths. For example, if $K=\lfloor 0.1N \rfloor$, the average PI width from the 90^{th} to 100^{th} percentiles must be reduced, while the other PI widths are not involved in the formulation. As a result, the number of inequality constraints is $N+1$, where N comes from the first constraint, and one comes from the average PI width control constraint.

Formulation P3 (pinball loss with controlled maximum width)

Formulation P3 follows (3.2) by using the maximum PI width as a function to control the maximum PI width assessed across all PI width samples, as shown in

$$\begin{array}{ll} \underset{\underline{\theta},\overline{\theta}}{\text{minimize}} & \sum_{i\in\mathcal{I}} \rho_{\overline{\alpha}}(y_i-\hat{u}_i(\overline{\theta}))+\rho_{\underline{\alpha}}(y_i-\hat{l}_i(\underline{\theta})) \\ \text{subject to} & \hat{l}_i(\underline{\theta}) \leq \hat{u}_i(\overline{\theta}), \quad \forall i \in \mathcal{I} \\ & \max_{i\in\mathcal{I}}[\hat{u}_i(\overline{\theta})-\hat{l}_i(\underline{\theta}))] \leq \gamma \cdot \text{sample width}. \end{array} \tag{3.6}$$

This formulation aims to minimize the pinball loss while simultaneously reducing the maximum PI width. The first inequality forces the PI width to be non-negative, so the maximum

PI width function can also be written as ℓ_∞-norm. Controlling the maximum PI width as the inequality constraint is also equivalent to controlling all PI width samples, shown as

$$\begin{aligned}
\underset{\underline{\theta},\overline{\theta}}{\text{minimize}} \quad & \sum_{i\in\mathcal{I}} \rho_{\overline{\alpha}}(y_i - \hat{u}_i(\overline{\theta})) + \rho_{\underline{\alpha}}(y_i - \hat{l}_i(\underline{\theta})) \\
\text{subject to} \quad & \hat{l}_i(\underline{\theta}) \le \hat{u}_i(\overline{\theta}), \quad \forall i \in \mathcal{I} \\
& |\hat{u}_i(\overline{\theta}) - \hat{l}_i(\underline{\theta}))| \le \gamma \cdot \text{sample width}, \quad \forall i \in \mathcal{I}.
\end{aligned} \tag{3.7}$$

This formulation involves one predefined formulation hyperparameter, γ, which prevents all PI widths from exceeding the sample width multiplied with γ. As a result, one maximum inequality constraint now transforms into N inequality constraints, resulting in a total of $2N$ inequality constraints for P3.

Numerical method

The numerical method is the optimization technique employed to solve the three formulations outlined in this approach. The choice of optimization technique depends on the type of optimization problem. A convex optimization problem is defined by having a convex objective function, convex inequality constraints, and affine equality constraints (Boyd and Vandenberghe, 2004, §4.2.1). The objective function, pinball loss, is a convex function (Koenker, 2005). Considering the convexity of the width functions, formulations P1 and P3 involve the ℓ_1, ℓ_∞ norms, which are convex functions due to the properties of norms (Boyd and Vandenberghe, 2004, §3.1.5). For P2, the sum of K-largest elements is also a convex function (Boyd and Vandenberghe, 2004, §3.2.3). So, the width functions from all formulations are convex functions. In this approach, we utilize a linear additive model (3.1), wherein both the upper and lower bounds can be considered as affine functions of the optimization variables $\overline{\theta}, \underline{\theta}$, respectively. From the property in (Boyd and Vandenberghe, 2004, §3.2.2), the composition of a convex function with an affine mapping is also a convex function. So, the objective function and the width function are convex. Moreover, the first inequality of (3.2) can be simplified into a linear inequality, which is obviously convex due to the linear additive model. These characteristics allow all formulations to be categorized as convex programs, which can be efficiently solved using convex solving algorithms like the interior-point methods.

Especially for P1 and P3, the width functions in the inequalities in (3.4) and (3.7) can be expressed linearly in the optimization variables. As mentioned in Section 2.2, minimizing the pinball loss can be cast as a linear programming (LP). For these reasons, formulations P1 and P3 can also be simplified as an LP and can be solved by any efficient LP algorithms such as simplex or interior-point methods.

The convexity of the problem arises from the linear additive form. When the model is extended to become more complex, it typically involves nonlinear relationships among the model parameters. The optimization problem now becomes nonlinear programming, where the problem structure depends on the model's parameterization.

Discussion

From the proposed formulations, the advantage of our formulation primarily stems from the convexity of these formulations. The convex formulations ensure that the solution represents a global minimum of the problem. Furthermore, they facilitate the application of efficient and straightforward numerical methods to solve these formulations. The favorable characteristics in optimization arise from the assumption of a linear additive form, which excels in simplicity and interpretability. When the data exhibit a nonlinear relationship between the predictor and target variable, transformation techniques can be used to convert predictors into nonlinear features, which are then incorporated into a model with a linear additive form. This process requires feature engineering. However, for complex models that are nonlinear in their parameters, such as neural networks, solving the proposed formulation numerically may be significantly more challenging. Additionally, the pinball loss serves as an indirect measure to represent the PICP term. The lowest pinball loss indicates the best match to the two quantiles, which is different from achieving the desired PICP. Having the best match with two quantiles also does not reflect the narrowest PI width with the desired PICP. Therefore, this indirectly addresses the trade-off between the PICP and PI width while instead aiming to achieve favorable characteristics in the numerical approach.

3.2 PICP with width control formulation

For PICP and width control formulation, we propose a loss function that directly incorporates the PICP into the objective function. We also introduce a new width control function by utilizing the sum of the K-largest function, referring to our proposed formulation as **Sum-k loss**. The full details of the proposed formulation will be explained below.

Motivation

According to this thesis, we aim to address two conflicting objectives: a high PICP and a narrow PI width. While the pinball-based formulation indirectly addresses the PICP to achieve the desired coverage probability, we seek to design a new approach that directly incorporates PICP into the objective function while also focusing on reducing the large PI width. Additionally, we aim to extend the linear model from a pinball-based approach to a nonlinear model that effectively captures the nonlinear relationships within the data, including the ANN and LSTM models. Since the PICP function is non-differentiable and incompatible with gradient-based algorithms, standard techniques for training state-of-the-art neural networks, we also introduce a new smooth approximation of the count function in the PICP term, enabling the application of these algorithms.

Literatures of formulation using a gradient-based algorithm

There are many studies that propose the PI-based loss function utilizing a gradient-based algorithm. Pearce et al. (2018) introduced the quality-based (QD) loss function, which incorporates the PICP and PI width terms based on statistical principles from the likelihood framework. The authors implemented a smooth approximation of the count function in PICP to ensure differentiability. Additionally, there are several enhanced versions of QD loss. For example, S. Salem et al. (2020) refined the QD loss by adding a mean squared error (MSE) term and a penalty function to provide point forecasts and ensure the integrity of the point forecasts alongside the PI from ensemble NN models. Lai et al. (2022) proposed a hybrid loss function that consists of two terms: point estimation with uncertainty and PICP loss. The first term includes MSE and PI width, derived from the Gaussian likelihood function, while the second term penalizes when the PICP drops below the desired probability. The point forecast in MSE is represented by the average of the upper and lower bounds. Saeed et al. (2024) presented an improved QD version by introducing a calibration function that imposes greater penalties on uncovered PIs to enhance multi-horizon predictive capabilities. This improved QD approach was utilized with a gated multi-scale convolutional sequence model. A multi-objective framework was available for the gradient-based approach, as detailed in Chen et al. (2024), which used the multi-gradient descent algorithm (MGDA) (Désidéri, 2009) along with the QD loss to identify the optimal descent direction for two objectives. Therefore, the loss compatible with the gradient-based algorithm leads to further implementation of more complex state-of-the-art NN models such as LSTM and the Transformers.

Methodology

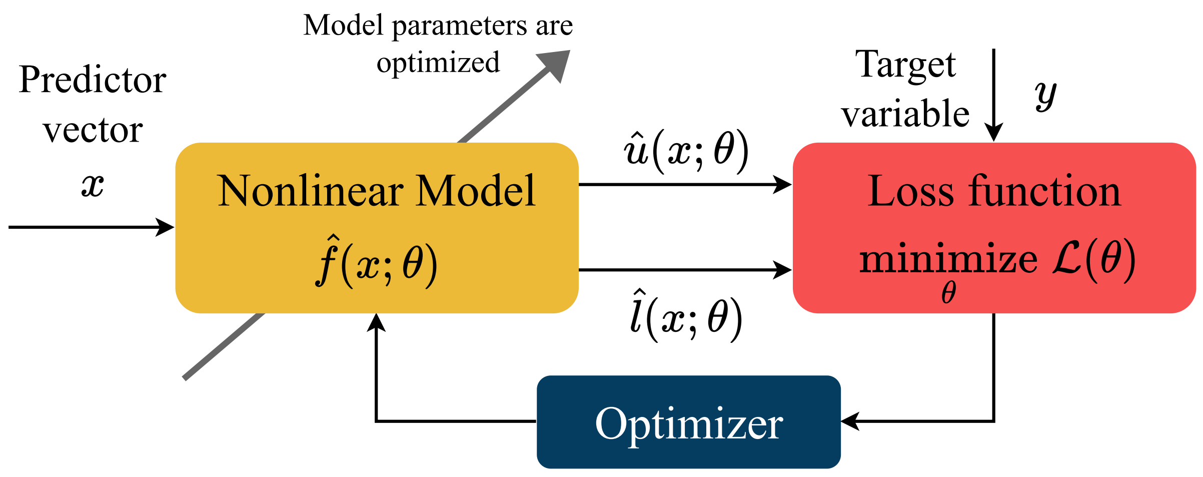

Model parameters are optimized
Predictor vector x
Nonlinear Model $\hat{f}(x;\theta)$
$\hat{u}(x;\theta)$
$\hat{l}(x;\theta)$
Target variable
y
Loss function
$\underset{\theta}{\text{minimize}}\ \mathcal{L}(\theta)$
Optimizer

Figure 3.2: The methodology for the training mechanism of the PI construction of PICP with width control approach.

Figure 3.2 illustrates the training procedures for PI construction in PICP with a width control approach. There are three key components: a model, a loss function, and an optimizer. First, the model refers to any nonlinear model that takes x as inputs and generates the PI as outputs. The model generates both upper and lower bounds using a single model with a common trainable parameter, denoted as θ, which corresponds to $\hat{u}(x;\theta)$ and $\hat{l}(x;\theta)$ respectively. Second, a loss function represents the quality of the PI, which is designed to be minimized. The loss function takes the PI $(\hat{u}, \hat{l})$ and y to evaluate the loss value. A lower loss indicates better PI performance. Various loss functions reflect good PI in different aspects related to their characteristics. Third, an optimizer refers to a numerical method that employs a gradient-based algorithm to solve the proposed loss function and obtain the optimal model parameters. The closed loop in the Figure 3.2 illustrates the iterative process of minimizing the loss function by updating the model parameters until the stopping criterion is satisfied. Once the training process is complete, the model with optimized parameters can generate PIs from an unseen data sample that aligns the PICP with the confidence level and the narrow PI width. In this approach, we focus on introducing a novel PI-based loss function, the sum of the K largest component loss, which incorporates a new aspect of PI width penalization. To illustrate the idea in this approach, we utilize a feedforward neural network (ANN) model with two output nodes shown as Figure 2.8 as a nonlinear model component in Figure 3.2.

Mathematical formulation

Large PI widths typically arise when data are corrupted by heteroskedastic noise dependent on x, resulting in higher uncertainty in certain regions of x. We aim to develop a loss function in this approach such that minimizing it will lead to a reduction in the *large* PI width for PI while ensuring that the PICP meets the desired coverage probability $(1-\delta)$. To address the issue of large widths, the loss function is specifically designed to impose a high penalty on large PI widths relative to narrow PI widths. So, the proposed loss function, **Sum-k loss function**, consists of two components: the PICP function and the PI width function, presented in an additive form as

$$\mathcal{L}_{\mathsf{Sum}-k}(\theta) = \max(0, (1-\delta) - \mathsf{PICP}(\theta)) + \gamma \mathcal{W}(\theta), \tag{3.8}$$

where $\gamma > 0$ controls the trade-off between the PICP and the PI width. Increasing γ places a greater penalty on the PI width function within the loss function, leading to a narrower PI width while losing the PICP.

The PICP term in the loss function focuses on minimizing the deviation between PICP and $1-\delta$, applying penalties only when PICP is below $1-\delta$. So, the $\max(0,\cdot)$ function in the PICP term is used according to Pearce et al. (2018). Additionally, we adjust the quadratic function in the PICP term of the QD loss to a linear function, as demonstrated in (3.8). This adjustment enforces a stronger penalty for PICP deviation compared to QD loss when the PICP is close to the desired probability. This is because a linear function penalizes more heavily than a quadratic function within a small range where $x > x^2$ for $x \in (0,1)$. The PICP can be calculated according to the definition in (2.19).

Considering the PI width term, we propose a new PI width function, $\mathcal{W}(\theta)$, employing the sum of the K-largest function to impose a strong penalty on large PI widths, referred to in our loss function as the *Sum-k* loss ($\mathcal{L}_{\mathsf{Sum}-k}$), expressed in (3.8). The PI width function includes two terms: the average of the K-largest PI widths and the average of the remaining

PI widths shown as

$$\mathcal{W}(\theta) := \mathcal{W}(\theta|K,\lambda) = \frac{1}{R_{\text{quantile}}}\left[\frac{1}{K}\sum_{i=1}^{K}|w(\theta)|_{[i]} + \lambda\cdot\frac{1}{N-K}\sum_{i=K+1}^{N}|w(\theta)|_{[i]}\right], \tag{3.9}$$

where $|w(\theta)|_{[i]}$ is the i^{th} largest width's absolute value, with $|w|_{[1]} \geq |w|_{[2]} \geq \cdots \geq |w|_{[N]}$. While the sign of w is neglected in the width function, the negative widths (crossing PI) is unlikely to occur because it would introduce a penalty in PICP. In (3.9), K is the number of samples treated as large PI widths, while the others are considered narrow PI widths. In practice, we introduce a tuned hyperparameter $k \in (0,1)$ representing a portion of the data treated as large PI widths, where K is set to $\lfloor kN \rfloor$. When $K = 1$, $\mathcal{W}(\theta)$ reduces to **Chebyshev norm** (or ℓ_∞-norm) of the widths which penalizes the maximum PI width (the most extreme case). A higher value of k indicates a larger number of PI width samples classified as large widths during the training process. A hyperparameter, $\lambda > 0$, represents a relative weight of the averaged narrow PI width to the averaged large PI width. When $\lambda < 1$, the average of the narrow widths receives less penalty than the large PI widths, resulting in the loss function prioritizing the reduction of the large PI widths.

To combine the PI width function into the loss function, the PI width must be normalized to eliminate the effect of the data scale. The normalization factor typically uses $R = y_{\text{max}} - y_{\text{min}}$, following the PINAW. However, when the data contain outliers, this range can be very high, leading to an underestimation of the PI width term in the loss function. Therefore, we introduce a new normalization factor, the quantile range, to remove the effect of outliers and the scale of the PI width in (3.9), expressed as

$$R_{\text{quantile}} = q_y(0.95) - q_y(0.05), \tag{3.10}$$

where $q_y(0.95)$ and $q_y(0.05)$ are the corresponding quantiles of y at 0.95 and 0.05 probability of y. The proposed PI width function is a special case of the ordered weighted ℓ_1 norm, which is a convex function and classified as a norm in the studies by Bogdan et al. (2013); Zeng and Figueiredo (2014, 2015); Figueiredo and Nowak (2016). We present the proof of convexity and the norm properties of the ordered weighted ℓ_1 norm as follows Zeng and Figueiredo (2014).

Let $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ be a weight vector with components that form a non-increasing sequence of nonnegative values,

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0, \tag{3.11}$$

with $\lambda_1 > 0$ corresponding to the largest magnitude element of x, the ordered weighted ℓ_1 norm (OWL1) of a vector $x \in \mathbf{R}^n$ is defined as

$$\Omega_\lambda(x) = \lambda_1|x|_{[1]} + \lambda_2|x|_{[2]} + \cdots + \lambda_n|x|_{[n]}, \tag{3.12}$$

where $|x|_{[1]} \geq |x|_{[2]} \geq \cdots \geq |x|_{[n]}$ represents the components of x arranged in non-increasing order of magnitude.

Let $\tilde{x} \in \mathbf{R}^n$ be the vector obtained by sorting x in non-increasing order of magnitude. We can express $\tilde{x}$ in terms of of x as

$$\tilde{x} = P(x)x, \tag{3.13}$$

where $P(x)$ is a permutation matrix that sorts x into $\tilde{x}$, so we can write (3.12) in a vector form as

$$\Omega_\lambda(x) = \left\| \begin{bmatrix} \lambda_1 x_{[1]} \\ \lambda_2 x_{[2]} \\ \vdots \\ \lambda_n x_{[n]} \end{bmatrix} \right\|_1 = \left\| \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \lambda_n \end{bmatrix} \begin{bmatrix} x_{[1]} \\ x_{[2]} \\ \vdots \\ x_{[n]} \end{bmatrix} \right\|_1 = \|\,\mathbf{diag}(\lambda)\tilde{x}\|_1 = \|\,\mathbf{diag}(\lambda)P(x)x\|_1 \tag{3.14}$$

Lemma 1. *$\Omega_\lambda : \mathbf{R}^n \to \mathbf{R}$ is a convex function.*

Proof. Let $x, y \in \mathbf{R}^n, \theta \in [0,1]$, and let $z = \theta x + (1-\theta)y$ be a convex combination of x, y, then:

$$\begin{aligned} \Omega_\lambda(z) &= \|\,\mathbf{diag}(\lambda)P(z)z\|_1 = \|\,\mathbf{diag}(\lambda)P(z)(\theta x + (1-\theta)y)\|_1 \\ &= \|\theta\,\mathbf{diag}(\lambda)P(z)x + (1-\theta)\,\mathbf{diag}(\lambda)P(z)y)\|_1 \end{aligned}$$

From the triangle inequality, and the homogeneity property of ℓ_1-norm, we obtain:

$$\Omega_\lambda(z) \leq \theta\|\,\mathbf{diag}(\lambda)P(z)x\|_1 + (1-\theta)\|\,\mathbf{diag}(\lambda)P(z)y)\|_1 \tag{3.15}$$

Since the component of λ form a non-increasing non-negative sequence, it follows that $\|\,\mathbf{diag}(\lambda)P(z)x\|_1 \leq \|\,\mathbf{diag}(\lambda)P(x)x\|_1$ for any x, z. This result holds because $P(z)x$ does not necessarily arrange x in the non-increasing order, causing the larger elements of λ to not align with the larger magnitude elements of x. The same inequality holds for y as well. Therefore, we have:

$$\begin{aligned} \Omega_\lambda(z) &\leq \theta\|\,\mathbf{diag}(\lambda)P(x)x\|_1 + (1-\theta)\|\,\mathbf{diag}(\lambda)P(y)y\|_1 \\ &= \theta\Omega_\lambda(x) + (1-\theta)\Omega_\lambda(y). \end{aligned}$$

Thus, Ω_λ is a convex function. ■

Lemma 2. *If $\lambda_1 > 0$, then $\Omega_\lambda : \mathbf{R}^n \to \mathbf{R}$ is a norm.*

Proof. The properties of homogeneity and triangle inequality have already been established in (3.15) using the properties of ℓ_1-norm. We now proceed to prove the positive definiteness property that $\Omega_\lambda(x) \geq 0,\ \Omega_\lambda(x) = 0 \leftrightarrow x = 0$. From the definition of $\Omega_\lambda(x)$ in (3.12), and given that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$ with $\lambda_1 > 0$ and $|x|_{[1]} \geq |x|_{[2]} \geq \cdots \geq |x|_{[n]} \geq 0$, it follows that $\Omega_\lambda(x) = 0$ if and only if $x = 0$. This is evident because of $\lambda_1 > 0$, $|x|_{[1]}$ must be zero to satisfy the positive definiteness property. Since the largest magnitude element of x is zero, all other elements of x are also forced to be zero.

Thus, $\Omega_\lambda(x)$ is a norm. ■

The proposed PI width formulation in (3.9) can be viewed as a special case of the OWL, where $\lambda_1 = \lambda_2 = \cdots = \lambda_K = \frac{1}{K}$ and $\lambda_{K+1} = \lambda_{K+2} = \cdots = \lambda_N = \frac{\lambda}{N-K}$. To satisfy (3.11), the inequality $\lambda \leq \frac{N}{K} - 1$ is necessary for it to be classified as a convex function and norm. However, this approach uses a nonlinear model for the upper and lower bounds, making the PI width nonlinear in the model parameter and resulting in a nonconvex function. Nevertheless, due to the favorable properties of the structure of the PI width function, we can achieve good convergence during the optimization process.

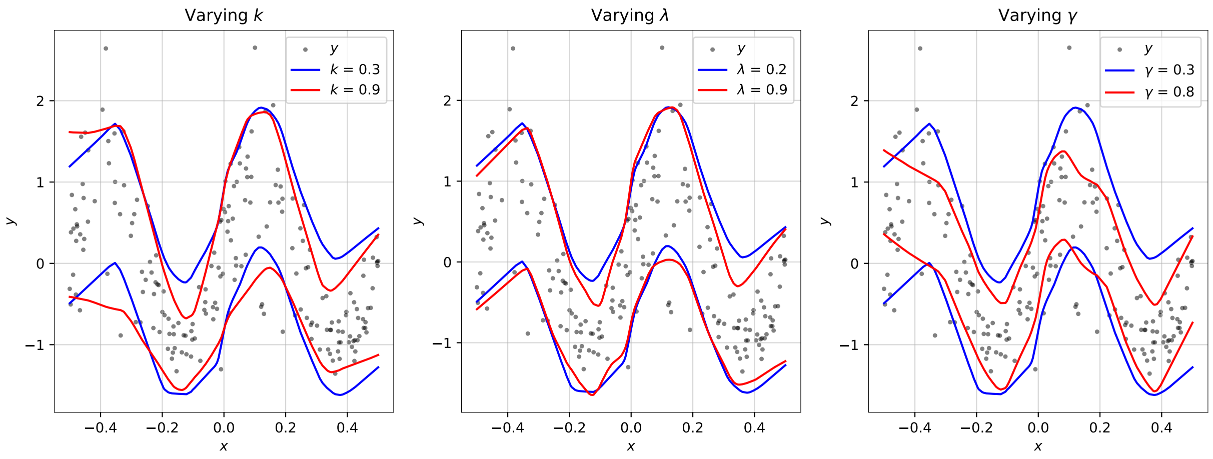


Figure 3.3: The effect of the hyperparameter in the sum-k formulation: (left) Reducing k decreases large PI widths. (central) Lowering λ reduces large PI widths while increasing narrow widths. (right) Increasing γ reduces the overall PI width but loses the PICP.

Therefore, the Sum-k formulation has three hyperparameters that need to be tuned: k, λ, γ. Figure 3.3 illustrates the effect of modifying each hyperparameter. For k, when it approaches one, the proportion of PI widths categorized as the large PI widths increases. Setting lower k highlights the difference penalization between the large and narrow PI width, causing a heavy reduction in the large PI width. The value of k should be determined by examining the proportion of high-variance samples in the dataset. As a start, users can set $k = 0.3$ in practice. The hyperparameter λ controls the relative penalization level of narrow PI widths compared to large PI widths. Decreasing λ places greater emphasis on large PI widths. During the training process, reducing large PI widths results in a more significant reduction in the loss function than reducing narrow PI widths at the degree of λ. For instance, $\lambda = 0.1$ indicates that for the same 1-unit reduction in both the averaged large and narrow PI widths, the loss function decreases ten times more from the reduction in large PI widths. Consequently, the optimizer tends to update the model parameters in a direction of reducing large PI widths rather than narrow ones. As a result, lowering λ decreases large PI widths while causing narrow PI widths to become slightly wider. The value of λ is recommended based on the user's preferences regarding the level of large PI width to be reduced while permitting an increase in narrow PI widths. For γ, different PICP levels are achievable by adjusting γ, where a higher γ reduces the PI width but decreases PICP. The value of γ consistently remains within the tenth decimal scale, independent of sample sizes and data scale, because the loss function is already normalized with N and R_{quantile}. To choose an appropriate γ, we recommend tuning γ based on cross-validated PICP. Start by splitting the data into training and validation sets in the training process. Then, vary γ and choose the value that achieves the desired PICP at the confidence level on the validation set. This approach ensures that the model's performance is evaluated on an unseen dataset, leading to better generalization. In summary, k and λ are required to be set according to user preferences, while γ should be adjusted using the validation set.

Smooth approximation

The calculation of PICP as defined in (2.19) in the Sum-k formulation involves a non-differentiable count function, making it unsuitable for gradient-based algorithms. To address this issue, in Pearce et al. (2018); Lai et al. (2022); Chen et al. (2023); Saeed et al. (2024), a smooth approximation of the count function is presented as a product of sigmoid functions.

$$\mathbf{1}_{\text{sigmoid}}(\hat{l}_i(\theta) \leq y_i \leq \hat{u}_i(\theta)) = \sigma(s(y_i - \hat{l}_i(\theta))) \cdot \sigma(s(\hat{u}_i(\theta) - y_i)), \tag{3.16}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ and s is the softening factor. A larger s results in a steeper sigmoid function, making it more similar to a step function. In this study, we introduce an alternative smooth approximation using the sum of the hyperbolic tangent functions, called the tanh-smooth approximation, which is simpler than (3.16). To show the simplicity of the tanh-smooth approximation, Firstly, we define a tanh-count function ξ calculated as

$$\xi_i = \frac{1}{2}\left(\text{tanh}(s(y_i - \hat{l}_i(\theta))) + \text{tanh}(s(\hat{u}_i(\theta) - y_i))\right), \tag{3.17}$$

where $\xi_i \to 1$ when $y_i \in \left(\hat{l}_i, \hat{u}_i\right)$. From $\sigma(x) = \frac{1+\text{tanh}\left(\frac{x}{2}\right)}{2}$, we can show that

$$\begin{aligned}\sigma(s(y_i - \hat{l}_i(\theta))) \cdot \sigma(s(\hat{u}_i(\theta) - y_i)) &= \frac{1}{4}\left(1 + \text{tanh}\left(\frac{s}{2}(y_i - \hat{l}_i(\theta))\right)\right)\left(1 + \text{tanh}\left(\frac{s}{2}(\hat{u}_i(\theta) - y_i)\right)\right) \\ &= \frac{1}{4}\left[1 + \text{tanh}\left(\frac{s}{2}(y_i - \hat{l}_i(\theta))\right)\text{tanh}\left(\frac{s}{2}(\hat{u}_i(\theta) - y_i)\right) + 2\xi_i\right].\end{aligned}$$

It can be seen that ξ is part of the sigmoid approximation, showing a simpler mathematical expression of the tanh-smooth approximation. Moreover, to obtain a similar curve from the two functions, it is recommended to set the softening factor s of the tanh function to be half that of the sigmoid function. However, ξ can be negative since the tanh function ranges from $(-1, 1)$. The negative case arises when $\hat{u}_i < y_i$ and $\hat{l}_i > y_i$, which is undesirable. To prevent misbehavior of the negative counting function, the operation $\max(0, \cdot)$ is incorporated into ξ to set the counting function to zero in this scenario. Therefore, the proposed tanh-smooth approximation of a count function can be expressed as

$$\mathbf{1}_{\text{tanh}}(\hat{l}_i(\theta) \leq y_i \leq \hat{u}_i(\theta)) = \frac{1}{2}\max\left(0, \text{tanh}(s(y_i - \hat{l}_i(\theta))) + \text{tanh}(s(\hat{u}_i(\theta) - y_i))\right). \tag{3.18}$$

The tanh-smooth approximation utilizes the $\max(0, x)$ function, which is non-smooth due to its non-differentiability at $x = 0$. Nevertheless, most NN software implementations provide one of the derivatives on either side of zero instead of generating an error (Goodfellow et al., 2016). Moreover, the $\max(0, x)$ function, commonly known as a ReLU function, is widely used as an activation function in the NN model. This ensures its compatibility with gradient-based optimization algorithms (Pearce et al., 2018). In the case of the tanh-smooth approximation, the case that ξ is exactly zero is rarely encountered numerically, allowing us to consider it as a differentiable function.

Employing any choice of smooth approximations leads to a numerical problem when $\hat{u}_i, \hat{l}_i, y_i$ are exactly equal. As indicated in (3.16) and (3.17), the smooth approximations do not yield the value of one in such cases, meaning that this sample is not qualified as a PI-covered sample. To be considered a PI-covered sample, a small margin between $\hat{u}_i, \hat{l}_i, y_i$ is necessary. Suppose $\hat{u}_i = y_i + \epsilon$ and $\hat{l}_i = y_i - \epsilon$, the sigmoid and tanh-smooth approximation now can be simplified to $\sigma^2(s\epsilon), \max(0, \tanh(s\epsilon))$ respectively.

Figure 3.4 shows the differences in the choices of smooth approximation (left) and the effect of margin (ϵ) on the smooth approximation (right). The left panel in Figure 3.4 shows that the tanh-smooth approximation closely resembles the sigmoid approximation when the softening factor is set to 50 and 100, respectively. The right panel in Figure 3.4 shows the smooth approximation function varying with margin ϵ, indicating a minimum margin of approximately 0.1 for a PI-covered sample. The minimum margin decreases with a higher softening factor, but excessive softening can cause a divergence in the model training process due to backpropagation. In this study, the PICP within the loss function employs the tanh-smooth approximation function shown as:

$$\text{PICP}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_{\tanh}(\hat{l}_i(\theta) \leq y_i \leq \hat{u}_i(\theta)), \tag{3.19}$$

where s is set to 50, demonstrating good convergence results for the gradient-based algorithm.

Numerical method

A numerical problem in this study is to minimize the proposed loss function to obtain optimal model parameters. Since the Sum-k loss function consists of a sum of nonlinear smooth approximations of count functions in the PICP term, the overall loss function is highly nonlinear. Thus, the optimization problem can be cast as an unconstrained nonlinear optimization problem. In addition, the Sum-k loss is continuous and differentiable, allowing the use of a gradient-based algorithm to solve the problem. The benefit of using a gradient-based optimization algorithm is that it enables the application of various techniques, such as minibatch optimization, adaptive learning rates for faster convergence, momentum-based updates to help escape from flat regions, and improved stability overall. For simple linear models with a low number of model parameters (but the loss function is still nonlinear in parameters), stochastic gradient descent and AdaGrad (Duchi et al., 2011b) can be implemented to minimize the Sum-k loss. For large models, often found in NN, the number of model parameters is usually very high. Several widely used optimization tools are available

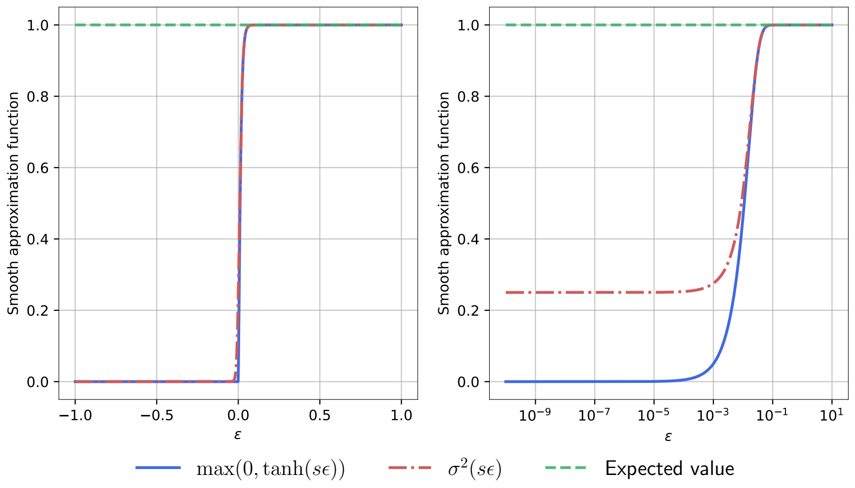


Figure 3.4: The margin effect of the smooth approximation: (left) Comparison of smooth approximation function values. (right) Impact of the margin on the smooth approximation displayed on a log scale.

for training NNs, including RMSProp (Tieleman and Hinton, 2012), Adam (Kingma and Ba, 2017), and Nadam (Dozat, 2016). These optimizers enhance the training of large NN by using adaptive learning rates and a momentum mechanism, resulting in faster and more stable convergence on nonlinear and high-dimensional loss surfaces.

Discussion

The advantage of this approach can be divided into three parts, following the diagram in Figure 3.2. First, the proposed loss function of this approach directly addresses the PICP term calculated using the definition of PICP with a smooth approximation. Thus, minimizing the loss function results in the PIs that satisfy the PICP and also reflect the narrowest PI width function at the operating point, given γ. Moreover, the trade-off characteristics can also be controlled by adjusting γ, allowing users to select the level of trade-off according to their specifications. Second, this method utilizes a nonlinear model, allowing it to effectively capture the nonlinear and complex characteristics of the data compared to a linear model in the pinball-based approach. Third, since the formulation is differentiable, it is suitable for gradient-based algorithms. A wide range of optimization tools and frameworks offer built-in support for these formulations, enhancing both implementation and computational efficiency. However, the nonlinearity of the loss function, combined with the complexity of the model, leads to a high-dimensional, non-convex loss surface. As a result, gradient-based algorithms may struggle with local minima, making optimization more challenging. Additionally, complex nonlinear models, such as neural networks, often incur high computational costs and sacrifice interpretability, making it difficult to understand the relationships between predictors and the target variable.

Chapter IV

EXPERIMENTAL DESIGN

This chapter describes the experimental design utilized to validate our proposed methodology. It also includes the evaluation metrics used in the experiments, the benchmarked methods, and details for solar forecasting, along with a description of the solar dataset. This thesis includes two methodologies: a pinball-based formulation and a PICP with width control formulation, referred to as methodology 1 and methodology 2, respectively. Figure 4.1 provides an overview of how each experiment is linked to the corresponding methodology, with full details presented in Section 4.3.

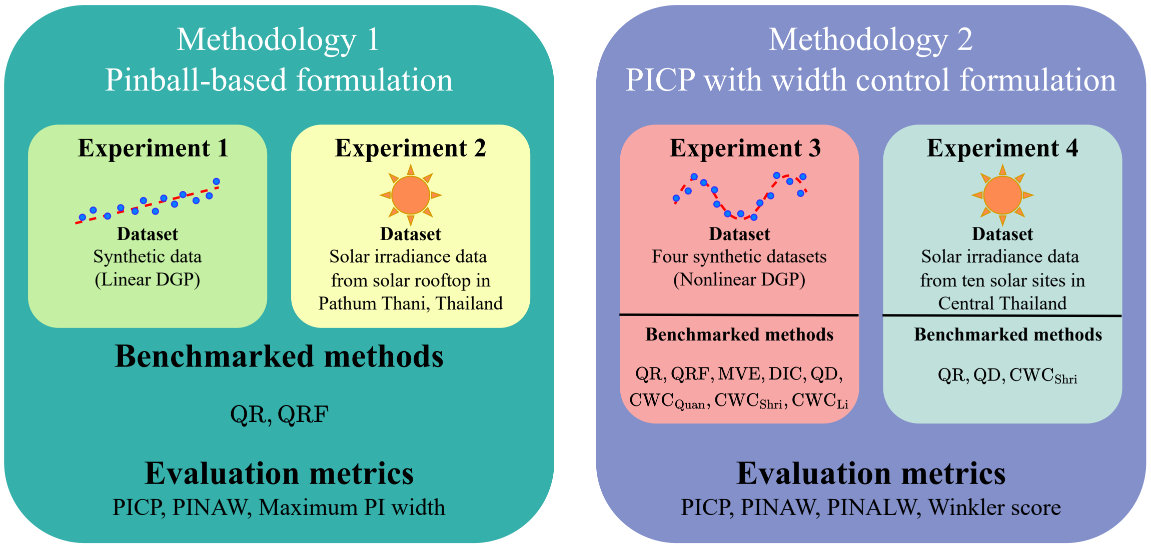


Figure 4.1: The overview of all experiments performed in this thesis.

4.1 Evaluation metrics used in the experiments

This section outlines the evaluation metrics utilized to assess the performance of PI across all experiments. The reliability of PI is quantified by the PICP calculated as (2.19). The sharpness of PI is represented by the average PI width (PINAW) (2.21). Additionally, the Winkler score (2.23) is evaluated to demonstrate the correspondence of the PI derived from each method with the equal tail quantile. In this thesis, we introduce a new normalization factor called the quantile range $R_{\text{quantile}} = q_y(0.95) - q_y(0.05)$, calculated from the quantiles of the target variable at 0.05 and 0.95 probabilities. So, the R_{quantile} is used instead of R in PINAW (2.21) and Winkler score (2.23) throughout the entire study to eliminate the influence of outliers on the target variable. Since this study aims to reduce the large PI width, we present two new evaluation metrics to assess this large PI width as follows.

- **Maximum prediction interval width** measures the maximum element of the PI widths expressed as
$$\text{Maximum width} = \max_{i=1,2,\ldots,N} w_i. \tag{4.1}$$
Additionally, a normalized version, referred to as the *normalized maximum width*, will be used in the experiments, where it is normalized by R_{quantile}.

- **Prediction interval normalized average large width (PINALW)** is the average of normalized *large* PI width calculated from the samples that have a PI width more than the p-quantile of PI width shown as:
$$\text{PINALW}(p) = \frac{1}{KR_{\text{quantile}}} \sum_{i=1}^{K} w_{[i]}, \tag{4.2}$$
where $w_{[i]}$ represents the i^{th} largest element of the PI width such that $w_{[1]} \geq w_{[2]} \geq \cdots \geq w_{[N]}$, and $K = \lfloor (1-p)N \rfloor$ is the number of samples classified as having large PI widths corresponding to the p-quantile. In this study, we select the PINALW with $p = 0.5$ to assess the average width of the top half of the PI width, which refers to the large PI width.

4.2 Benchmarked methods

This section provides an overview of all the benchmarked methods used throughout the experiments. In the experiments for Methodology 1, we apply our proposed formulations using a linear additive model and benchmark performance against simpler models, including linear quantile regression (QR) and QRF. In the experiments for methodology 2, we employ a neural network (NN) model and compare its performance with other NN-based methods, including the tree-based QRF. Most of the selected methods in these experiments are formulated as loss functions and implemented within an NN model to minimize the loss during training. We denote θ as the model parameter for both linear and nonlinear models. Additionally, formulations that combine PICP and PI width follow the same structure as our approach, where the penalty parameter γ is applied to the PI width penalty function.

1. **Quantile regression (QR)** is a method that estimates the conditional quantile of the target variable (y) given the predictor (x) (Koenker, 2005). The quantile value is determined by minimizing the pinball loss for a specified quantile. To construct a PI with a confidence level of $(1-\delta) \times 100\%$ from QR, we define the lower and upper bounds based on the quantiles $\frac{\delta}{2}$ and $1 - \frac{\delta}{2}$, respectively. So, the aggregated pinball loss to construct PI is shown as
$$\text{Loss} = \frac{1}{N} \left[\sum_{i=1}^{N} \rho_{\frac{\delta}{2}}(y_i - \hat{l}) + \rho_{1-\frac{\delta}{2}}(y_i - \hat{u}) \right], \tag{4.3}$$

where $\rho_\alpha(r) = \max(\alpha r, (\alpha - 1)r)$ is a pinball function. For linear QR, the upper and lower bounds are linearly parameterized by different model parameters, $\overline{\theta}$ and $\underline{\theta}$, as follows:

$$\hat{u} = \hat{u}(x; \overline{\theta}) = \overline{\theta}_0 + \overline{\theta}_1 x_1 + \ldots + \overline{\theta}_p x_p, \quad \hat{l} = \hat{l}(x; \overline{\theta}) = \underline{\theta}_0 + \underline{\theta}_1 x_1 + \ldots + \underline{\theta}_p x_p$$

Using different model parameters, the pinball loss function in (4.3) is separable, meaning that the upper and lower quantiles can be estimated independently. This allows the use of standard quantile regression tools, such as those available in `sklearn` for Python, to estimate each quantile separately. For NN-based QR used to benchmark methodology 2, we implement a single model to release both upper and lower bounds using an NN with two output nodes representing two quantiles. So, the upper and lower bound, denoted as $\hat{u} = \hat{u}(x; \theta)$ and $\hat{l}(x; \theta)$, share the same model parameter θ and are trained using the pinball loss as outlined in (4.3).

2. **Quantile regression forest (QRF)** is a tree-based method using the random forest framework to provide the full conditional CDF (Meinshausen, 2006). The estimated CDF is determined by calculating the average of the indicator function of the target variable across all decision trees. To determine a PI, two quantiles are chosen to define the upper and lower bounds at the specified confidence level, similar to QR.

3. **Mean-variance estimation (MVE)** estimates the mean and variance of the target variable, relying on the Gaussian assumption (Nix and Weigend, 1994). Our study applies the MVE to the NN with two output neurons: mean ($\hat{\mu}(x_i; \theta)$) and variance ($\hat{\sigma}^2(x_i; \theta)$). The loss function of MVE is based on the log-likelihood of the Gaussian distribution as:

$$\text{Loss}_{\text{MVE}}(\theta) = \frac{1}{2} \sum_{i=1}^{N} \left(\log(\hat{\sigma}^2(x_i; \theta)) + \frac{(y_i - \hat{\mu}(x_i; \theta))^2}{\hat{\sigma}^2(x_i; \theta)} \right). \tag{4.4}$$

After estimating $\hat{\mu}(x_i; \theta)$ and $\hat{\sigma}^2(x_i; \theta)$, the PI is constructed based on the given confidence level as

$$\hat{u}(x_i; \theta) = \hat{\mu}(x_i; \theta) + z_{1-\frac{\delta}{2}} \hat{\sigma}(x_i, \theta), \quad \hat{l}(x_i; \theta) = \hat{\mu}(x_i; \theta) - z_{1-\frac{\delta}{2}} \hat{\sigma}(x_i, \theta) \tag{4.5}$$

where $z_{1-\frac{\delta}{2}}$ is the z-score corresponding with a $(1 - \delta) \times 100\%$ confidence level. However, if the NN directly realeases the variance ($\hat{\sigma}^2(x_i; \theta)$), it may initially produce negative values. This can lead to numerical issues, as taking the logarithm of a negative value in (4.4) results in an error. To address this issue, we modify the model to produce $\log(\hat{\sigma}^2(x_i; \theta))$ instead, while the loss function is still implemented based on the mathematical formulation in (4.4).

4. **Coverage-width-based criterion (CWC)** is a PI-based loss function that directly combines the PICP and PI width term as a single loss function. The CWC was first proposed in Khosravi et al. (2011) as a loss function that combines PINAW and PICP

for training NN of a lower upper bound estimation (LUBE) method. The LUBE approaches have many alternative loss functions. In Quan et al. (2014a), the PINAW in the CWC loss is replaced by PINRW shown as

$$\text{CWC}_{\text{Quan}}(\theta) = \text{PINRW}(1 + \mathbf{1}(\text{PICP} < 1-\delta) \cdot e^{-\gamma(\text{PICP}-(1-\delta))}). \tag{4.6}$$

The PINRW utilized the ℓ_2-norm concept for the PI width term, which penalizes more on the large width calculated as $\text{PINRW} = \frac{1}{R_{\text{quantile}}}\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{u}(x_i;\theta) - \hat{l}(x_i;\theta))^2}$. In this study, we adjust (4.6) to its equivalence, (4.7), to ensure continuity, allowing it to be solved with a gradient-based algorithm as the benchmark method.

$$\text{CWC}_{\text{Quan-eq}}(\theta) = \text{PINRW}(1 + e^{\gamma \max(0,(1-\delta)-\text{PICP})}). \tag{4.7}$$

However, the multiplicative term of PI width in (4.7) can lead to abnormal characteristics of PI width where the PINRW reaches zero, as derived from the global minimum of the loss function. Subsequently, the authors of Shrivastava et al. (2015) modified the CWC loss function to be an additive form as demonstrated in (4.8).

$$\text{CWC}_{\text{Shri}}(\theta) = \text{PINAW} + \mathbf{1}(\text{PICP} < 1-\delta) \cdot e^{-\gamma(\text{PICP}-(1-\delta))}. \tag{4.8}$$

The version presented in (4.9) is implemented in our experiment to guarantee continuity and compatibility with gradient-based algorithms.

$$\text{CWC}_{\text{Shri-eq}}(\theta) = \text{PINAW} + e^{\gamma \max(0,(1-\delta)-\text{PICP})}. \tag{4.9}$$

In Li et al. (2020), the authors presented a CWC formulation (4.10) to evaluate PI more efficiently when the PICP is below the desired coverage. They apply an affine transformation to the PI width term of the CWC function. Because the original CWC is notably sensitive to the PICP term, the CWC_{Li} is proposed to address the issue that the original CWC fails to accurately evaluate the variation of the PINAW when the PICP is less than $1-\delta$.

$$\text{CWC}_{\text{Li}}(\theta) = \begin{cases} \beta\text{PINAW}, & \text{PICP} \geq 1-\delta \\ (\alpha + \beta\text{PINAW})(1 + e^{-\gamma(\text{PICP}-(1-\delta))}), & \text{PICP} < 1-\delta \end{cases} \tag{4.10}$$

However, the CWC_{Li} is not continuous in the PICP argument. Therefore, we also modify it to (4.11) as a continuous version to be applicable to a gradient-based algorithm.

$$\text{CWC}_{\text{Li-eq}}(\theta) = \frac{\beta}{2}\text{PINAW} + \left(\alpha + \frac{\beta}{2}\text{PINAW}\right) e^{\gamma \max(0,(1-\delta)-\text{PICP})}. \tag{4.11}$$

Therefore, we utilized CWC_{Quan}, CWC_{Shri}, and CWC_{Li} as the benchmarked losses implemented as (4.7), (4.9), and (4.11) respectively.

5. **Deviation information-based criterion (DIC)** is proposed in Zhang et al. (2015) to account for the PIs deviation information in the loss function shown in (4.12). The exponential term in the CWC is substituted with the function pun, as defined in (4.13), within the DIC loss framework. This adjustment is made to evaluate the deviation of the target variable from the PI based on samples lying outside the PI.

$$\text{DIC}(\theta) = \text{PINAW} + \mathbf{1}(\text{PICP} < 1-\delta) \cdot \text{pun} \tag{4.12}$$

where

$$\text{pun} = \gamma \left[\sum_{i=1}^{N_L} (\hat{l}(x_i;\theta) - y_i) + \sum_{i=1}^{N_U} (y_i - \hat{u}(x_i;\theta)) \right], \tag{4.13}$$

where N_L and N_U represent the number of observations located below $\hat{l}(x_i,\theta)$ or above $\hat{u}(x_i,\theta)$, respectively. The penalty parameter γ is set as $1/\delta$ according to Zhang et al. (2015).

6. **Quality driven loss function (QD)** is proposed based on the high-quality principle of obtaining narrow PI with achieving a desired PICP (Pearce et al., 2018). The QD loss consists of two components: PI width and coverage probability, combined in the additive form as presented in (2.30).

   The PI width term is proposed as the captured PI width measured from only the PI that captures the data point. The coverage term is derived based on the likelihood principle. The count function is approximated using a smooth version, and then the loss function is proposed as applicable to the gradient-based algorithm. In this study, we modify the original QD loss by adjusting the trade-off parameter γ to focus on penalizing the PI width term. The influences of N and δ in (2.30) are removed, and instead, the PINAW is employed to enable γ to manage the trade-off regardless of the number of samples, the desired probability, and the data range. As a result, the updated version used in this study, presented in (4.14), is still equivalent to (2.30).

$$\text{Loss}_{\text{QD-eq}}(\theta) = \max(0, (1-\delta) - \text{PICP})^2 + \gamma \text{PINAW}_{\text{capt.}}. \tag{4.14}$$

4.3 Experiment design

The experiments are split into two groups to validate two methodologies based on Figure 4.1. Each group includes two experiments that demonstrate the performance of each methodology using synthetic data and real-world data. The first methodology is the pinball-based formulation, which utilizes a linear additive model. The second methodology is PICP with width control formulation, which employs an NN (nonlinear model).

The synthetic data experiments for both methodologies mainly focus on analyzing the properties of PI based on the proposed formulations. This includes examining the trade-off curve between PICP and PI width, the distribution of PI widths, and the characteristics of the PIs. The real-world data experiments illustrate the application of the proposed methods in solar forecasting, which involves a high level of uncertainty, to demonstrate how the large PI widths are reduced from each formulation.

Since PI performance is evaluated based on both PICP and PI width, directly determining the best formulation is challenging. A method with a higher PICP and wider PI width (Method A) cannot be directly compared to another method (Method B) with a lower PICP but a narrower PI width. As many methods have formulation hyperparameters to set the operating point while varying this hyperparameter can cause the PICP and PI width to vary, there are many schemes to benchmark the performance for each formulation that can be found in the literature. The complete literature on benchmarked experiments can be found in the Appendix A. In this thesis, we design a benchmarking scheme for PIs by adjusting the formulation's hyperparameter to achieve a consistent PICP at the desired probability. Once PICP is controlled, we compare the PI width across different methods. The full details for each experiment are provided below.

Experiments for methodology 1

Two experiments, synthetic and real-world dataset, were designed to test the performance of formulations P1, P2, and P3. These formulations control various PI width functions, including the average PI width, the large PI widths, and the maximum PI width. We assess the performance by analyzing PICP, PINAW, and maximum PI width. As we utilize a linear additive model, we compare its performance with a linear QR and QRF, which are also simple models. Experiment 1 carries out all formulations using a synthetic dataset generated from a linear data-generating process with 100 noise trials. The goal of Experiment 1 is to observe the trade-off characteristics between PICP and PI width across all methods by varying γ for each formulation. The PI width in the trade-off curve also assesses both the average PI width and the maximum PI width. Additionally, this experiment examines the distribution of PI width and its characteristics for each formulation. Finally, the experiment demonstrates an example of applying formulations with a quadratic function to the data while the model remains linear in its parameters.

Experiment 2 applies formulation P3 due to its superior performance from the result of the first experiment. This experiment aims to provide the PI of the solar irradiance with a 30-minute lead time at a confidence level of 0.9. Due to the varying characteristics of solar irradiance throughout the day, different time periods require different predictors. Therefore, we divide the linear model into three separate models for morning, noon, and evening, each incorporating distinct features. We then analyze the trade-off curve and PI characteristics in a time series plot to evaluate performance and compare PI widths while keeping a controlled PICP.

Experiments for methodology 2

Experiments 3 and 4 demonstrate the effectiveness of reducing the large PI width from the Sum-k formulation using an NN model with synthetic and real-world datasets, respectively. The performance is evaluated by PICP, PINAW, PINALW, and Winkler score in both experiments. Experiment 3 employs four nonlinear synthetic datasets with 100 noise trials to generalize the results. Experiment 3 investigates the trade-off between PICP and PI width, including PINAW and PINALW, across various benchmarked methods with formulation hyperparameters, including $\text{Sum} - k$, QD, CWC_{Quan}, CWC_{Shri}, and CWC_{Li}. Then, the characteristics of PIs from each formulation are compared among these methods. Additionally, this experiment compares the distribution of PI widths and evaluates performance indices across all methods, adding $\text{QR}, \text{QRF}, \text{MVE}$, and DIC, while ensuring a controlled PICP in the validation set.

Experiment 4 demonstrates the performance of the proposed formulations in a solar forecasting application, as it serves as an appropriate example of data characterized by heteroskedastic noise and high volatility. This experiment aims to provide a one-hour-ahead solar irradiance forecast, PI, with a confidence level of 0.9 at a 15-minute resolution, corresponding to four lead times. We also use the Sum-k formulation with ANN and LSTM to evaluate performance against a more complex model, which is then benchmarked with QR, QD, and CWC_{Shri}. The results of this experiment are reflected in the comparison of evaluation metrics, characteristics of PI in the solar irradiance time series plot, and a four-step real-time PI forecast.

4.4 Real-world data description: Solar irradiance forecasting

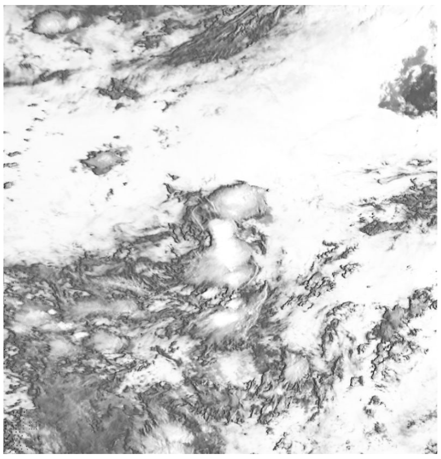
In this thesis, we show the application of our methodologies for solar irradiance forecasting. Solar irradiance carries significant uncertainty due to fluctuating weather conditions, especially in Thailand, which is located in an equatorial region. The techniques and factors influencing solar forecasting significantly depend on the forecasting horizons. In the literature, the forecasting horizon is primarily divided into three categories: intra-hour, intra-day, and day-ahead (Ahmed et al., 2020). The intra-hour forecast encompasses horizons from a few seconds to one hour, utilized in demand response. The intra-day forecast horizons extend from 1 to 6 hours, serving economic dispatch purposes. The day-ahead forecast horizon ranges from 6 to 48 hours, applied in unit commitment. Forecasts longer than 2 days do exist, but they are rarely found in the literature.

Different applications of solar irradiance forecasts require different forecasting specifications, comprising three key components: forecast time horizon, forecast time resolution, and forecast lead time. The forecast time horizon indicates the future time frame for which predictions will be made, usually reported as one hour ahead or one day ahead. The forecast time resolution indicates how often a prediction value is generated, typically reported in intervals like a 15-minute resolution. The forecast lead time specifies the number of time steps into the future for which predictions are generated within the forecast horizon. For example, when the forecast horizon is 1 hour with a resolution of 15 minutes, the forecast lead time includes 4 time steps, each corresponding to a 15-minute interval. This thesis focuses on intra-hour solar forecasting, with experiments 2 and 4 conducted 30 minutes and one hour ahead, respectively. In this context, the uncertainty in solar irradiance arises mainly from cloud cover, which is influenced by the unpredictable nature of cloud behavior

(Antonanzas et al., 2016). We provide key predictors for the intra-hour solar irradiance used in this thesis based on the available data, as follows.

1. **Measurement data of solar irradiance** is historical data of solar irradiance measured from sensors or meters. The data resolution depends on the specifications of the measurement tool, which may be in seconds or minutes. Different forecasting specifications require different data resolutions; in this thesis, we utilize a 15-minute resolution of historical data to provide forecasts for solar irradiance 30 minutes and 1 hour ahead. The resampling method is necessary to down-sample the measured data for use as a predictor aligned with forecasting specifications. To use it as a predictor, solar measurement data is treated as a lagged regressor, where past observations are utilized to forecast future values.

2. **Cloud data** is obtained from the Himawari-8 satellite image that covers Thailand with a spatial resolution of 2×2 km^2. The cloud images are available from 06:00 to 19:50 at 10-minute intervals and have been resampled to meet specifications. The cloud images consist of two types: cloud mask (grayscale) and overall RGB (color) images shown in Figure 4.2. From the preliminary analysis in Figure 4.3, both the cloud mask and the overall R channel show strong anti-correlations with the clear-sky index, defined as $k(t) = I(t)/I_{\text{clr}}(t)$. To use cloud images as predictors, we extract the pixel intensity of cloud images in the pixel closest to the site's location based on latitude and longitude. Next, the cloud index (CI) is calculated using the formula $\text{CI} = \frac{X - \text{LB}}{\text{UB} - \text{LB}}$, where X represents the pixel intensity, and LB and UB denote the lower and upper bounds set at 0 and 255, respectively. In this thesis, we employ the cloud mask and the R-channel of the RGB image to compute the cloud index, based on a strong correlation observed in the pre-analysis. Then, the cloud index is considered a lagged regressor because it is extracted from the actual measurement of image data.


(a) Cloud mask channel.

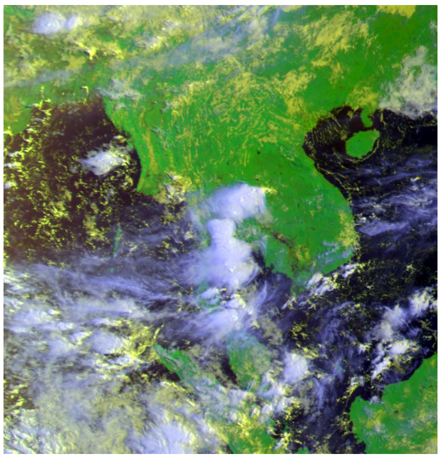

(b) Overall channel.

Figure 4.2: Cloud images from the Himawari satellite received at Chulalongkorn station.

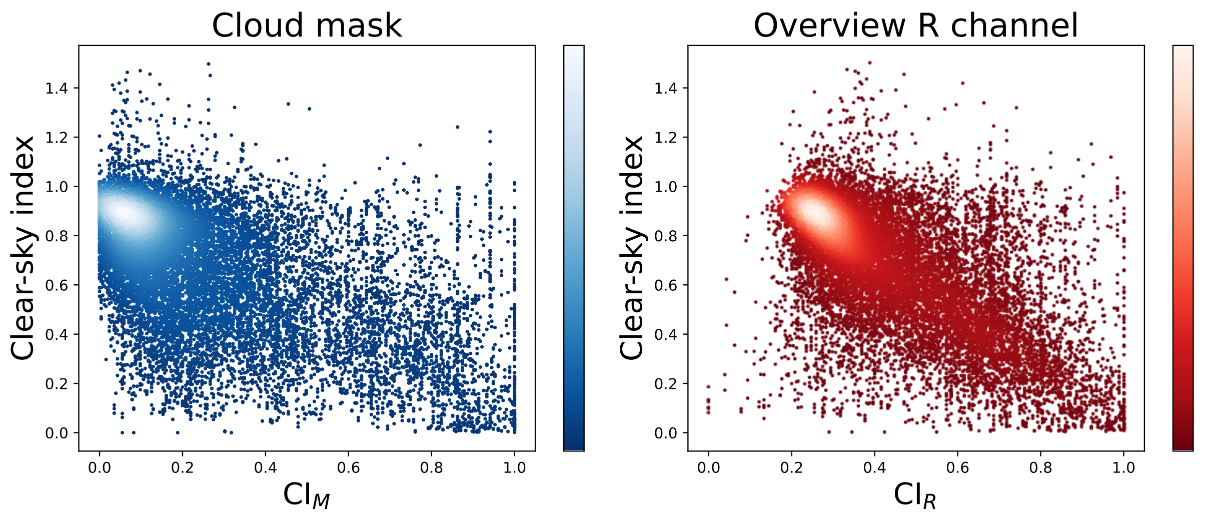


Figure 4.3: The scatter plot between cloud index and clear-sky index. (left): Cloud mask. (right): R-channel.

3. **Clear sky irradiance** refers to the solar irradiance value under the assumption of no clouds at a specific time and location. The clear sky irradiance (I_{clr}) can be calculated using the clear sky model, expressed in the mathematical formulation. The clear sky model requires two types of inputs: solar geometry inputs, which determine the sun's position, and atmospheric parameter inputs, used to estimate the atmospheric attenuation conditions at the specified location (Antonanzas-Torres et al., 2019). There are various clear-sky models that require different atmospheric parameters (Antonanzas-Torres et al., 2019). In this thesis, we generate clear-sky irradiance from the Ineichen clear-sky model (Ineichen and Perez, 2002). This model uses date and time, along with site location, including latitude and longitude, as inputs for solar geometry. These inputs are utilized to calculate the zenith angles θ, determining the sun's position. For the atmospheric parameter, this model requires only Linke turbidity (T_L), which is simpler than alternative models. The Ineichen clear-sky model can be expressed as:

$$I_{\text{clr}}(t) = a_1 I_0 \cos\theta(t) e^{a_2 \text{AM}(t)(f_{h_1}+f_{h_2}(T_L-1))} \tag{4.15}$$

where $I_0 = 1366.1$ W/m^2 is the extraterrestrial irradiance constant, and θ is in degree. The airmass coefficient (AM) can be calculated following Paulescu et al. (2013):

$$\text{AM}(t) = \frac{1}{\cos\theta(t) + 0.50572(96.07995 - \theta(t))^{-1.6364}}. \tag{4.16}$$

The constants in (4.15) can be calculated as:

$$a_1 = 5.09 \times 10^{-5}h + 0.868,\ a_2 = 3.92 \times 10^{-5}h + 0.0387,\ f_{h_1} = e^{-\frac{h}{8000}},\ f_{h_2} = e^{-\frac{h}{1250}}, \tag{4.17}$$

where h is the altitude in meters of the site location. Our metadata does not include the site's altitude, so we retrieve API data from the open access source https://www.opentopodata.org/datasets/srtm/ according to Suwanwimolkul et al. (2025). The Linke turbidity indicates the amount of solar radiation absorption and scattering in the atmosphere under clear skies, driven by water vapor and aerosols (Paulescu et al., 2013). In this thesis, we assess the monthly Linke turbidity using open-source software PVlib (Anderson et al., 2023) by inputting the site location and datetime. As a result, the clear-sky irradiance can be generated for any given datetime and site location, and it is considered a future regressor in forecasting.

4. **Numerical weather prediction (NWP) data** is the weather prediction method that is based on the mathematical model. The mathematical model involves the partial derivative equation describing the physical process in the atmosphere, such as thermodynamics and fluid motion. The model uses the current weather condition as the initial condition in solving the partial derivative equation. These require massive computational costs and very powerful computers to solve for weather prediction. This thesis utilizes reanalyzed forecast data from the MERRA-2 (Modern-Era Retrospective Analysis for Research and Applications) model from service in https://www.soda-pro.com. Users can select the data resolution from 1 minute to 1 month, and we have chosen 15 minutes for this thesis. The data are available from January 1980 to one month ago from the current date. The spatial resolution is 2×2 km^2. The NWP data is considered a future regressor in solar forecasting.

Additionally, we include the hour index, which indicates the hour of the day, as a future regressor in this study because of the nonstationary and seasonal characteristics of solar irradiance. In this thesis, we utilize the notation defined in Table 4.1 for real-world physical variables. As experiments 2 and 4 are performed on solar irradiance forecasting applications, we utilize different sources of datasets explained below.

Table 4.1: Notation used in solar application

Notation	**Definition**	**Unit**
t	Time	minute
I	Solar irradiance	W/m^2
CI$_M$	Cloud index from the cloud mask image	-
CI$_R$	Cloud index from R channel of overall RGB image	-
I_{clr}	Clear-sky irradiance	W/m^2
I_{nwp}	Forecasted solar irradiance from NWP	W/m^2
T	Temperature	K
P	Pressure	hPa
WS	Wind speed	m/s
WD	Wind direction	degree ($^\circ$)
RF	Rainfall	kg/m^2
RH	Relative humidity	%
HI	Hour index	hour

Solar data in experiment 2

This section provides the data description used in experiment 2. The measurement data was collected from a solar rooftop (Site 001) with an installed capacity of 0.98 MW located in Pathum Thani province in the central region of Thailand. The dataset includes measurements of generated solar power, solar irradiance, ambient temperature, and photovoltaic module temperature, collected every 15 minutes from April 2022 to July 2023. In this dataset, we use only the solar irradiance data measured by the SMP 6 pyranometer model, which complies with the ISO 9060 spectrally flat class B standard. The actual measurement of solar irradiance was pre-processed to enhance data quality. First, the data was filtered from 07:00 to 17:00 to capture only the daytime irradiance. Then, the irradiance values were constrained to the range of 0 to 1400 W/m^2 to align with their physical significance. Next, the average daytime solar irradiance was calculated for each date, and dates with unusually low values were removed to avoid issues related to insufficient solar radiation or measurement inaccuracies. To address missing data for some timestamps, if any date has consecutive missing data periods exceeding six hours, the solar irradiance data for that date will be discarded. For shorter missing intervals, we employed linear interpolation. For cloud data, the cloud index was resampled from 10 minutes to 15 minutes to align with actual solar irradiance data. We generate clear-sky irradiance that corresponds with datetime from solar irradiance data at a 15-minute resolution. Finally, we combine solar irradiance data, cloud data, clear-sky irradiance, and NWP forecast data into an aggregated data frame and drop any missing data. As a result, the dataset contains 15,888 samples with comprehensive details on selecting significant predictors to be included in the experimental results.

Solar data in experiment 4

This section provides the data description used in experiment 4. Solar irradiance data was collected from ten solar stations in Central Thailand from January to December 2023, with a resolution of milliseconds. The measurements were taken using a CMP11 pyranometer, which complies with the ISO 9060 Class A standard. The solar irradiance data was pre-processed to enhance data quality. First, we checked the range of possible solar irradiance values to prevent sensor overflow. Then, we eliminated samples with negative values. Next, we resampled the raw data to a 1-minute resolution using a rolling mean. Finally, we downsampled it to a 15-minute resolution.

The cloud index was extracted differently depending on the site location and was resampled to a 15-minute resolution. In this experiment, we utilized only the R-channel cloud index due to its strong correlation identified in the pre-analysis. The clear-sky irradiance was generated corresponding to each site location and date time, aligned with the measurement of solar data. For NWP forecast data, we utilize only short-wave irradiance due to a strong correlation. Finally, the solar irradiance data was merged with the R-channel cloud index, clear-sky irradiance, and short-wave irradiance from NWP, and any missing data was dropped. As a result, the dataset spanned from 06:45 to 17:00 with a 15-minute resolution aligned with the forecasting specifications, totaling 113,793 samples.

Chapter V

EXPERIMENTAL RESULTS OF METHODOLOGY 1

This chapter presents the experimental results of experiments 1 (synthetic data) and 2 (real-world data), which were utilized to validate the concept of the first methodology, including formulations P1, P2, and P3. This chapter consists of two sections: experiments 1 and 2. Each section includes the experiment's objective, dataset generation for synthetic data, feature selection for real-world data, the experimental setting, and the results with discussion. The code for two experiments in this chapter can be found at `https://github.com/energyCUEE/probforecast`.

5.1 Experiment 1 - Simulation of P1, P2, P3 on synthetic data

Objective

This experiment aims to observe the performance of pinball-based formulations P1, P2, and P3 on the synthetic dataset (linear DGP) compared with QR and QRF. The performance is shown by how each method constructs PI while reducing the width of PI across various aspects by observing the trade-off curve between PICP and PI width, the characteristics of PI, and the distribution of PI widths. The trade-off curve between PICP and PI width (average PI width and maximum PI width) illustrates the trade-off mechanism between these two objectives when varying γ for pinball-based formulations or varying confidence level for QR and QRF. Each point on the trade-off curve corresponds to an operating point chosen by the user. When comparing the trade-off curves, better results are indicated by a significant reduction in PI width while maintaining PICP at the desired confidence level. This outcome highlights the impact of γ and suggests how to choose the operating point in practical applications. Each formulation controls a different PI width function, so we also observe PI characteristics (shape of PI) with the same PICP. The histogram of PI widths is used to verify the change in the distribution of PI widths when varying γ for each formulation. The histogram illustrates how each formulation attempts to reduce the amount of PI width in different aspects. Finally, we provide an example demonstrating the application of the proposed formulation to a nonlinear function, specifically a quadratic function with heteroskedastic noise.

Dataset

The data-generating process (DGP) was generated as

$$y = \beta_0 + \beta_1 x + e,$$

where x is an independent variable, y is a target variable, and e is a corrupted noise. Ground truth value of β_0 and β_1 was one-time generated from $\mathcal{N}(0,1)$. The variable x was generated from $\mathcal{N}(0,1)$ with a total of 2,000 samples, and then the values are sorted from low to high. The noise exhibits non-uniform variance, indicating heteroskedastic characteristics. The corrupted noise was generated as $e \sim \mathcal{N}\left(0, 4\,\text{max}(0, \text{sign}(x-0.3))+1\right)$, meaning that for $x > 0.3$, the noise was drawn from $\mathcal{N}(0,5)$ (high variance region), whereas for the others, the noise was generated from $\mathcal{N}(0,1)$ (low variance region). The noise was generated for 100 trials to ensure result generalization, while x, β_0, and β_1 were one-time generated.

As an example of a quadratic function, we generated DGP in the form of

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + e,$$

where $\beta_0, \beta_1, \beta_2$ were generated from $\mathcal{N}(0,1)$ with values of 1.26, -0.92, 0.44 respectively. The variable x was generated separately for the training and test sets, each sampled from $\mathcal{N}(0,1)$ with 2,000 samples, sorted in ascending order. The noise was generated independently for both the train and test sets, with two scenarios: high variance and low variance noise. For high-variance cases, the data was divided into three portions, each with the same number of samples based on the value of x. The first portion is corrupted with noise generated from $\mathcal{N}(0,5)$, while the other portion is corrupted with noise from $\mathcal{N}(0,1)$. In a low variance case, the noise was consistently generated from $\mathcal{N}(0,0.2)$ for all samples.

Experiment setting

We compared the results of our formulations with QR and QRF, where all methods use only x as a predictor. For our formulations, we set the model to align with the DGP where $\hat{u} = \overline{\theta}_0 + \overline{\theta}_1 x$, $\hat{l} = \underline{\theta}_0 + \underline{\theta}_1 x$. The confidence level was set to 0.9, where the range of γ for P1, P2, and P3 was varied from 0.5 to 1.0 with a spacing of 0.05. For P2, we set K to $\lfloor 0.3N \rfloor$ to consider the top 30% of PI width as large PI widths. Three formulations were solved for each γ over 100 noise trials. We varied the confidence level from 0.1 to 0.9 with a spacing of 0.05 for QR and QRF. For QR, we configured the model with a fitted intercept for both upper and lower bounds. For QRF, we fitted two models corresponding to the upper and lower bounds using two quantiles for each confidence level. The optimal model hyperparameters were selected using `RandomizedSearchCV` based on the lowest pinball score corresponding with 0.95 and 0.05 quantiles for the upper and lower bound. As a result, the optimal model hyperparameters are shown in Table 5.1.

Table 5.1: The formulation model hyperparameters for QRF used in experiment 1.

Model specification	**Upper bound**	**Lower bound**
`max_depth`	20	10
`min_samples_leaf`	30	4
`min_samples_split`	30	20
`n_estimators`	250	100

Result and discussion

Trade-off characteristics. Figure 5.1 shows trade-off characteristics between PICP and PI width, including average and maximum PI widths. The trade-off curve is obtained by averaging the results from all trials for generalization. A better result is indicated by the curve positioned in the lower right, which exhibits high PICP and a narrow PI width. Both the average and maximum PI width indicate that reducing γ in our formulations also causes the PI width to be narrower while also losing the PICP. This result aligns with the proposed optimization problem, where decreasing γ leads to a more stringent inequality of the PI width, equivalent to a larger penalization in the PI width term. The results also indicate that by setting the confidence level to 0.9, the trade-off curve of our formulation can attain any value of PICP by varying γ. So, we can select any operating point that satisfies the requirements of PICP while narrowing the PI width to the user's preference. Therefore, the trend showing that PI width decreases with a trade-off with PICP is clearly supported by this result. Additionally, for QR and QRF, lowering the confidence level results in a narrower PI width, while the PICP remains consistent at the specified confidence level. Based on the average PI width, all curves are positioned almost identically, indicating that there is no significant difference in the average PI width. Regarding maximum PI width, the trade-off curve from QR with varying confidence level overlaps with P1 with varying γ, which is also shown in the average PI width trade-off curve. So, the trade-off characteristics of QR and P1 are the same. The QRF exhibits the widest maximum PI width on the trade-off curve. This maximum PI width occurs in the region of high uncertainty within the data. This result is due to the structure of the tree-based model, which possesses greater complexity than the linear model, allowing it to capture the data's uncertainty more effectively. The superiority in reducing the maximum PI width is illustrated in P3. The maximum PI width of P3 is the smallest among all methods. The PI width decreases slightly while the PICP slowly falls below the confidence level of 0.9, as shown in Table 5.2. This result aligns with the meaning of formulation P3, which aims to control the maximum PI width, ensuring the best performance in that aspect. The performance of formulation P2 in the trade-off curve of maximum PI width lies between that of P1 and P3. This result is due to the formulation framework, which aims to control the large PI width, including the maximum PI width.

Characteristics of PI. Figure 5.2 compares the PI characteristics of each method using two operating points of γ with a confidence level of 0.9, based on a DGP selected from 100 trials. When γ is 0.8, the PI from QR and P1 is similar, supporting the trade-off curve result. The P1 aims to reduce the average PI width, encompassing all PI width samples and allowing us to observe narrow PI widths in low-variance regions and wide PI widths in high-variance regions. The slope of PI from P1 indicates a divergence to capture samples in the high variance zone. For QRF, the PI characteristics confirm the resulting trade-off characteristics, where the PI from QRF exhibits high nonlinearity in x due to its complex tree structure. This structure attempts to capture samples in high-variance regions, leading to a greater maximum PI width in the trade-off curve result. For P3, the slope of PI from P3 exhibits the lowest divergence level. The PI width is narrowest in the high-variance region and widest in the low-variance region. This outcome results from the formulation of P3, which aims to limit the maximum PI width without regard for other PI widths. This constraint is more stringent than that of P1 and P2. Furthermore, the slope of P2 is observed to be between P1 and P3 because the average of the K-largest PI width function places the penalty level between the average and maximum PI width. When γ is 0.5, the PI from P1 is narrower than QR while also losing PICP. For P3, the PI width of all samples is equal, which is the same as P2. This result occurs when the critical value of γ is reached, and the PI width constraint of P3 becomes active for all samples. This is also supported by the trade-off curve in Figure 5.1, which shows that when γ is lower than 0.7, the maximum PI width is consistently reduced refer to reduction of PI width for all samples. When comparing the feasible sets of constraints for P1 and P3, the optimal value of the objective function, pinball loss, for P1 tends to be lower. This occurs because the feasible set obtained from the average PI width constraint is larger than the maximum PI width constraint due to the equivalence of the ℓ_1 norm and the ℓ_∞ norm. Thus, the P1 solution is closer to the QR solution. This result emphasizes the differences between P1 and P3 in comparison to the QR solution.

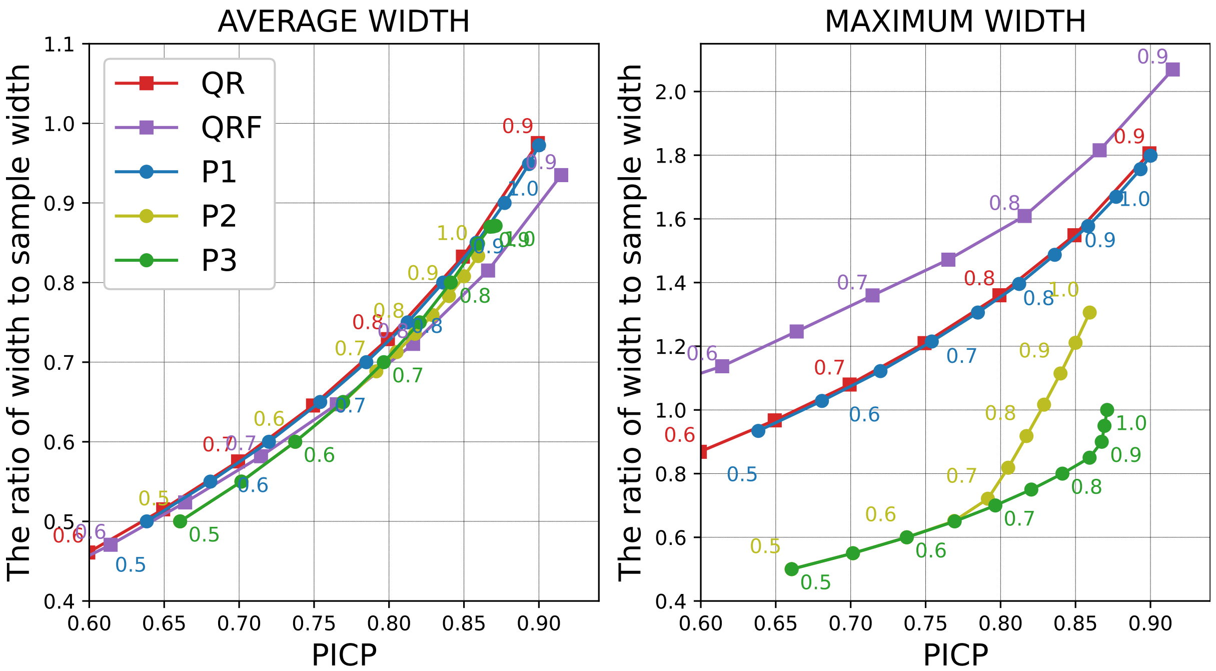


Figure 5.1: Experiment 1 result: Comparison of trade-off characteristics from each method by varying γ for P1, P2, and P3 and varying confidence level for QR, QRF. (left): Between PICP and PINAW. (right): Between PICP and normalized maximum PI width.

Distribution of PI width. Figure 5.3 illustrates the comparison of PI width characteristics aggregated from all trials using two values of γ. The histogram illustrates the comparison of the PI width distribution aggregated from all trials across all methods. Even though the average PI width does not show significant differences, there is a clear variation in

Table 5.2: Average PICP over 100 trials.

Width factor	1	0.95	0.9	0.85	0.8	0.75	0.7
P1	0.900	0.893	0.877	0.858	0.836	0.812	0.785
P2	0.859	0.850	0.840	0.829	0.817	0.805	0.790
P3	0.871	0.869	0.867	0.856	0.839	0.820	0.796

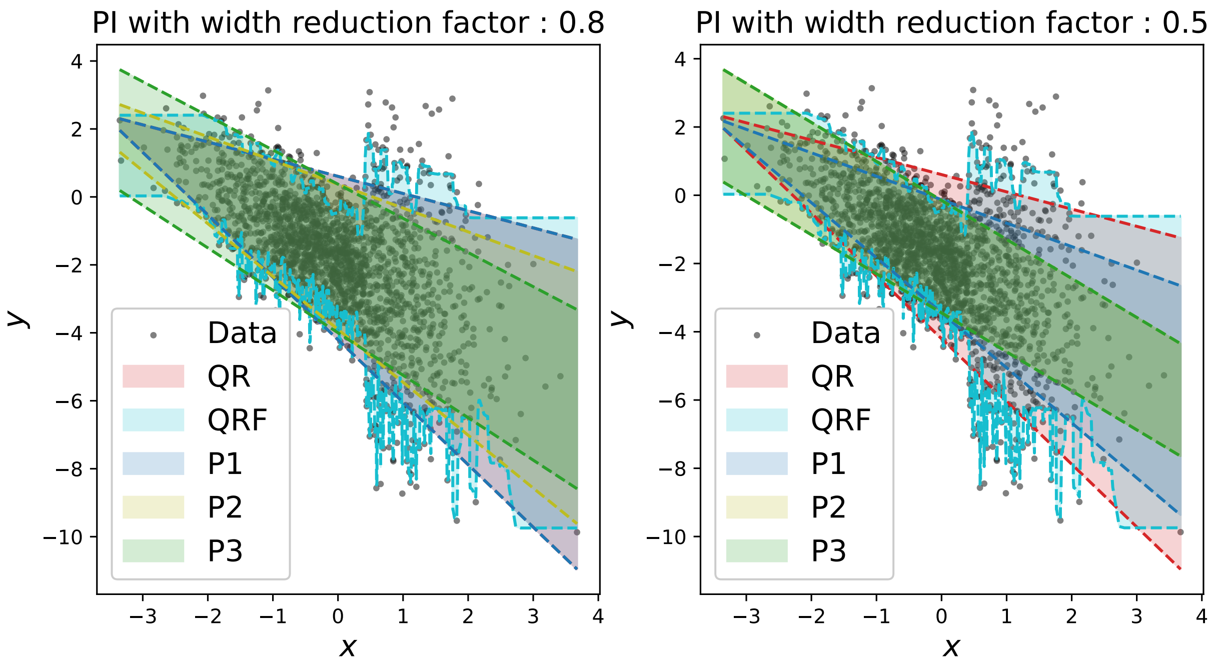


Figure 5.2: Experiment 1 result: Comparison of PI characteristics from each method in the synthetic dataset (linear DGP) with a confidence level 0.9.

the PI width distribution. The histogram of PI width distribution from QRF shows two peaks resulting from the DGP, which contain regions of high and low variance. When γ is 0.9, the histogram shows the lowest variation in the distribution of PI width from P3. Each sample PI width does not exceed 0.9, corresponding to the constraint in P3. The maximum PI width from P3 shows the lowest results, indicating that the narrow PI width has also been widened compared to other methods. When γ is 0.7, all PI widths from P3 are equal because the constraint controls almost all PI widths. This result confirms that with a critical γ, all PI widths from P3 are equal. Reducing γ below the critical point yields a constant PI width, which fails to account for the uncertainty of the data and results in a loss of interpretation of PI. Therefore, selecting γ requires careful consideration to ensure an appropriate value based on the requirements of each application.

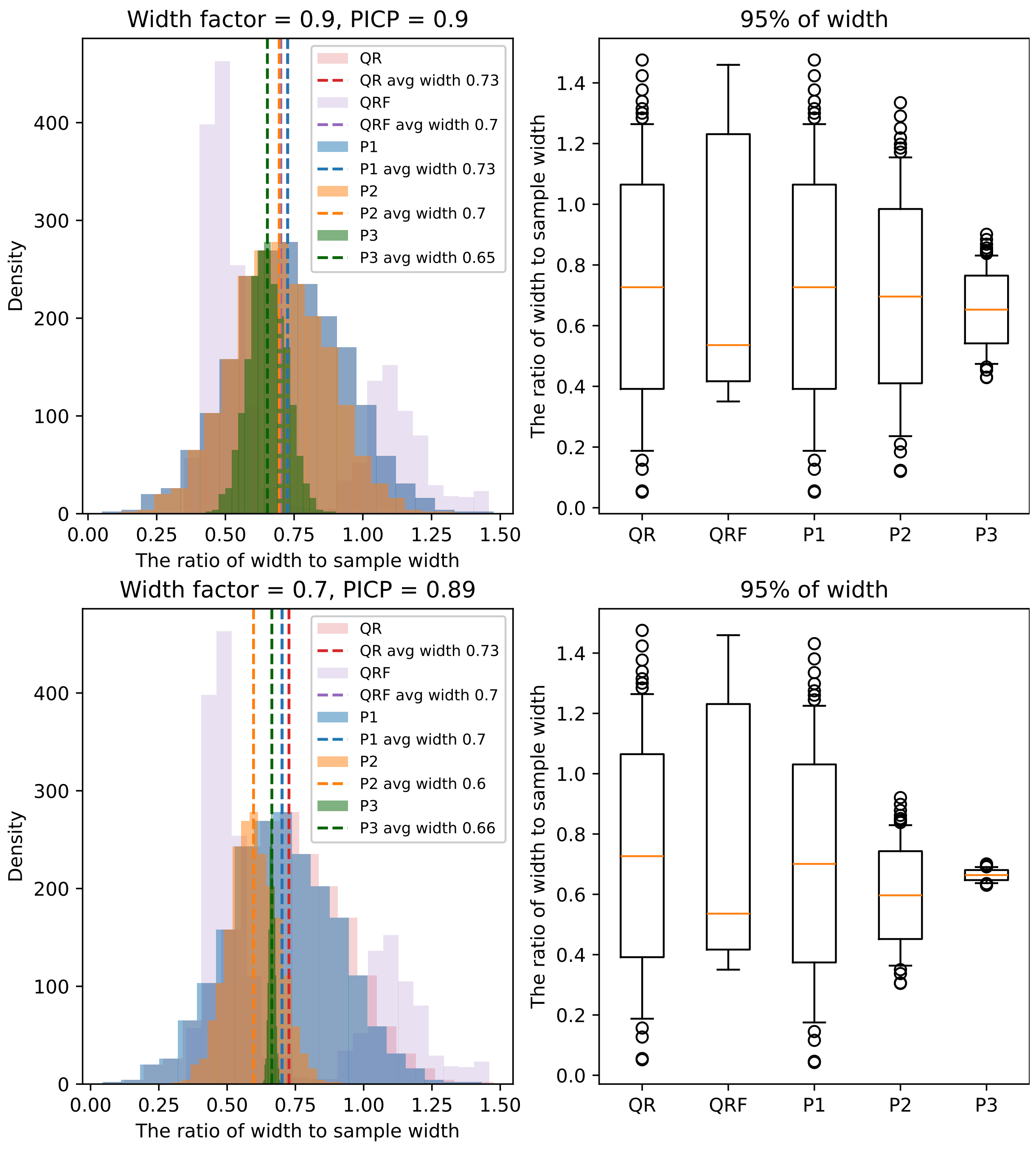


Figure 5.3: Experiment 1 result: Comparison of the PI width characteristics from each method. (left): Histogram of PI widths. (right): Boxplot of PI widths, where inside the box represents 95 % of all samples.

Simulation with quadratic DGP. The proposed formulation is also demonstrated to apply with a nonlinear function in x while remaining linear in the model parameter. Figure 5.4 shows the PI result from all methods in quadratic DGP, including high and low variance cases. All methods are trained on the training set and validated with the test set. In a low variance scenario, there is no significant difference in performance among all methods except QRF, which displays a different PI shape due to its complex model. In a high variance scenario, formulations P2 and P3 demonstrate improved results in reducing PI width in high volatility regions, with P3 delivering the best performance. In this case, QRF exhibits highly fluctuating PI characteristics, failing to accurately capture the behavior of the quadratic function. In conclusion, the proposed formulation, P3, shows superior performance in reducing PI width when dealing with data containing heteroskedastic noise involving highly volatile noise. Formulation P3 handles PI width in a more robust way, ensuring that it does not become excessively wide because of outliers. Furthermore, while the proposed formulation is limited to linear models, this result demonstrates that incorporating feature engineering with nonlinear transformations of predictors can enhance its applicability.

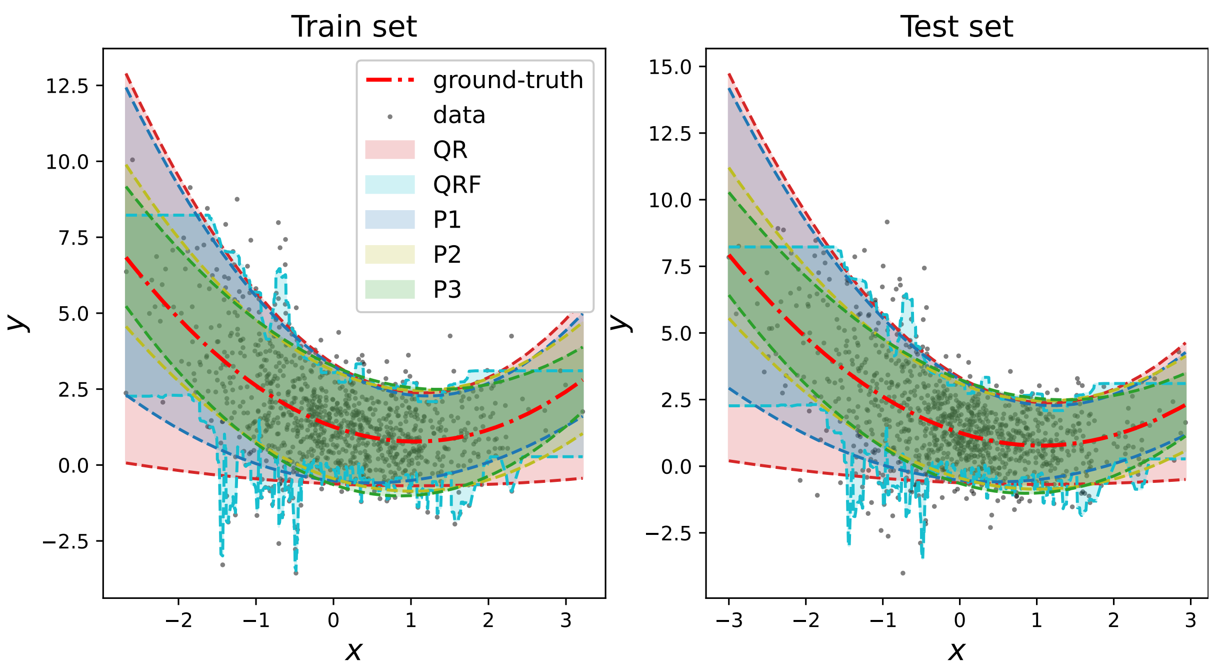
5.2 Experiment 2 - Performing P3 on solar irradiance forecasting

Objective

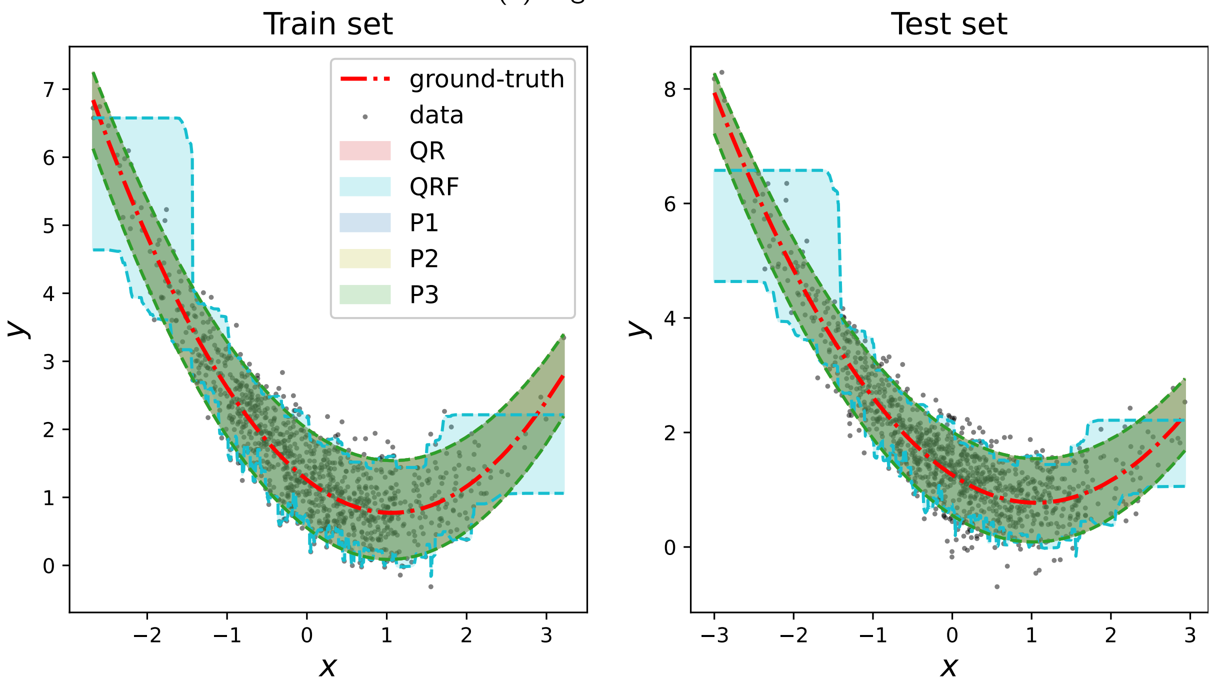
This experiment aims to apply P3 for a solar irradiance forecasting application, based on the superior performance observed in experiment 1 compared to QR and QRF. We aim to provide a one-step PI 30 minutes ahead of solar irradiance $I(t+30)$, with a confidence level of 0.9, from 07:00 to 17:00. The data resolution is 15 minutes, based on the measurement data of solar irradiance. The data is split into training and test sets. We aim to examine the trade-off characteristics between PICP and PI width, including both the average and maximum PI width in real-world applications on a training dataset. Next, we compare the PI characteristics by presenting the time series plot of the PI obtained from each method on the training dataset, illustrating how each method addresses the uncertainty in solar irradiance. Finally, we also examine the reliability diagram on a test dataset to assess the consistency of PICP in our formulation with a specified γ as the confidence level varies, ensuring the alignment of PICP with the expected confidence level.

Dataset

We used the measurement data collected from Pathum Thani, where a complete description of the dataset can be found in Section 4.4. We aggregated all predictors, including measured solar irradiance, cloud index from both the cloud mask and R-channel, clear-sky irradiance, all variables from NWP data, and hour index as candidate predictors to forecast solar irradiance. To use as a feature, actual solar irradiance and cloud index must be used as lag regressors where the past observations are used to predict future value. For I, we selected two lags as regressors: $I(t)$ and $I(t-15)$, based on the availability of the data and correlation analysis. For CI, we use only one lag for both channels: $\text{CI}_M(t)$ and $\text{CI}_R(t)$



(a) High variance.



(b) Low variance.

Figure 5.4: Experiment 1 result: Comparison of PI characteristics from each method in the synthetic dataset (quadratic DGP) at a confidence level 0.9.

as determined by correlation analysis. The clear-sky irradiance, NWP variables, and hour index are considered as future regressors.

Since the characteristics of solar irradiance vary significantly with time, the impact of each predictor on solar irradiance can differ at different times. For instance, the impact of clouds can be critical for solar irradiance during the noon period, whereas its effect is minimal during the morning period. Feature engineering is required as the linear model is utilized in this experiment. So, we divided the dataset into three time periods, each corresponding to a distinct model: morning (07:00 - 09:45), noon (10:00 - 15:00), and evening (15:15 - 17:00). Each model utilizes different predictors to describe I as the target variable. To determine the predictors for each model, we incorporated all predictors mentioned in the first paragraph and introduced quadratic interaction terms to enhance nonlinearity in the feature set as candidate predictors. Next, feature selection for each model was performed using stepwise regression on the candidate predictors, utilizing the Bayesian Information Criterion (BIC) (James et al., 2023) as the model selection score. As a result, three models consist of 29, 27, and 18 predictors corresponding to morning, noon, and evening, respectively. The chosen predictors are listed in Table 5.3, arranged by the significance level of the features from highest to lowest. Finally, the data was split into a training set for 2022 and a test set for 2023. The summary of sample sizes for each dataset and period is presented in Table 5.4.

Table 5.3: Selected predictors from stepwise regression for different models used in experiment 2 where blue text refers to quadratic interaction term.

Model	Predictors
Morning (07:00 - 09:45)	$I(t)^2, \text{CI}_R(t), T(t+30) : \text{RH}(t+30), \text{RH}(t+30), I_{\text{clr}}(t+30) : I(t), I_{\text{clr}}(t+30) : \text{CI}_M(t), T(t+30), \text{RH}(t+30) : I(t), I(t), \text{RF}(t+30), I_{\text{nwp}}(t+30), \text{WD}(t+30) : \text{RF}(t+30), T(t+30) : \text{WD}(t+30), \text{WD}(t+30), \text{CI}_M(t) : \text{CI}_R(t), T(t+30) : I_{\text{clr}}(t+30), I_{\text{clr}}(t+30), T(t+30) : I(t), \text{WS}(t+30) : I_{\text{clr}}(t+30), \text{RH}(t+30) : \text{RF}(t+30), \text{RH}(t+30) : I_{\text{nwp}}(t+30), T(t+30) : \text{WS}(t+30), I(t) : \text{CI}_M(t), \text{WS}(t+30), I_{\text{nwp}}(t+30) : I_{\text{clr}}(t), P(t+30) : \text{WS}(t+30), P(t+30)$, and $\text{CI}_M(t)$
Noon (10:00 - 15:00)	$I(t)^2, I(t) : \text{CI}_M(t), \text{HI}, \text{CI}_M(t) : \text{CI}_R(t), I_{\text{clr}}(t+30) : \text{CI}_M(t), I(t) : I(t-15), \text{WD}(t+30), T(t+30) : \text{WD}(t+30), I(t-15), \text{WD}(t+30)^2, \text{CI}_R(t), \text{RF}(t+30) : I_{\text{clr}}(t+30), \text{WS}(t+30) : \text{RF}(t+30), I_{\text{clr}}(t+30), I(t) : \text{CI}_R(t), I_{\text{clr}}(t+30) : I(t), \text{RH}(t+30) : I(t), I(t), \text{RF}(t+30), \text{RH}(t+30) : \text{WS}(t+30), \text{WD}(t+30), \text{HI}, T(t+30) : I(t), \text{WS}(t+30), T(t+30), \text{RH}(t+30)$, and $\text{CI}_M(t)$
Evening (15:15 - 17:00)	$I(t), I(t)^2, I_{\text{clr}}(t+30) : I(t), I_{\text{clr}}(t+30), I(t-15)^2, \text{CI}_R(t), P(t+30) : \text{CI}_R(t), I(t) : \text{CI}_R(t), I(t) : \text{CI}_M(t), P(t+30), I(t) : \text{CI}_M(t), I(t-15), \text{RF}(t+30), \text{RF}(t+30) : I_{\text{clr}}(t+30), \text{RH}(t+30)^2, \text{CI}_M(t)$, and $\text{RH}(t+30)$

Experiment setting

We compared the results of P3 with QR and QRF, with all methods utilizing the same predictors based on Table 5.3. For QR, we set up the model with a fitted intercept for both the upper and lower bounds. For QRF, the two models were fitted based on the upper

Table 5.4: The number of samples for the training and test sets in the experiment 2.

Period	Training set	Test set
Morning	2,294	1,476
Noon	5,345	3,417
Evening	2,047	1,309
Total	9,686	6,202

and lower bounds, utilizing two quantiles for each confidence level across three periods, resulting in six models in total. The optimal model hyperparameters were selected using `RandomizedSearchCV` with a pinball score corresponding to the 0.95 and 0.05 quantiles for the upper and lower bounds, respectively. The optimal QRF model hyperparameters are shown in Table 5.5. To determine the trade-off curve, we varied the confidence level from 0.1 to 0.9 with a spacing of 0.05 for QR and QRF.

Table 5.5: The model hyperparameters for QRF used in experiment 2.

Period	Model specification	Upper bound	Lower bound
Morning	`max_depth`	15	10
	`max_features`	`sqrt`	`sqrt`
	`min_samples_leaf`	8	4
	`min_samples_split`	5	20
	`n_estimators`	100	100
Noon	`max_depth`	25	25
	`max_features`	`sqrt`	`sqrt`
	`min_samples_leaf`	10	10
	`min_samples_split`	10	10
	`n_estimators`	200	200
Evening	`max_depth`	25	20
	`max_features`	`sqrt`	`sqrt`
	`min_samples_leaf`	10	30
	`min_samples_split`	10	30
	`n_estimators`	200	250

For P3, a linear additive model was used, incorporating a constant term, with the confidence level set at 0.9. Then, the hyperparameter γ was varied from 0 to 1.0 with a spacing of 0.02. Since solar irradiance cannot be negative, we modify the formulation P3 by adding a constraint to ensure the lower bound does not become negative, as shown by:

$$\begin{aligned} \underset{\underline{\theta},\overline{\theta}}{\text{minimize}} \quad & \sum_{i\in\mathcal{I}} \rho_{\overline{\alpha}}(y_i - \hat{u}_i(\overline{\theta})) + \rho_{\underline{\alpha}}(y_i - \hat{l}_i(\underline{\theta})) \\ \text{subject to} \quad & 0 \leq \hat{l}_i(\underline{\theta}) \leq \hat{u}_i(\overline{\theta}), \quad \forall i \in \mathcal{I} \\ & \max_{i\in\mathcal{I}}[\hat{u}_i(\overline{\theta}) - \hat{l}_i(\underline{\theta}))] \leq \gamma \cdot \text{sample width.} \end{aligned} \tag{5.1}$$

Result and discussion

Trade-off characteristics. Figure 5.5 illustrates the trade-off characteristics between PICP and PI width, including the average and maximum PI widths over different time periods. QRF outperforms QR in reducing both average and maximum PI width. For the average PI width, the QRF achieves the best results across all time periods because of its more complex model. There is no significant difference in performance between QR and P3. Considering the maximum PI width, the trade-off curve from P3 is located in the lowest right region, which shows the best performance across all time periods. The maximum PI width of P3 decreased rapidly over each time period, while the PICP slightly decreased. Especially at noon, the maximum PI width from P3 with $\gamma = 1$ decreases by 33% compared to QR while maintaining the same PICP. Therefore, based on the trade-off curve of P3, we select $\gamma = 0.5$ as the operating point for comparing PI characteristics, as it significantly reduces the maximum PI width while maintaining an acceptable PICP.

Characteristics of PI. Figure 5.6 presents a time series comparison of PI characteristics among QR, QRF, and P3 of solar irradiance from the training dataset. The time series plot displays data for four days, from August 20 to August 23, 2022. The data from August 20 to 22 shows significant fluctuations in solar irradiance due to clouds, while August 23 exhibits a trend similar to clear-sky irradiance, which has low fluctuations. The result of PI indicates that our formulation has the narrowest PI width to date, with high uncertainty, particularly from August 20 to 22. This result demonstrates the superior performance of P3 when the data is corrupted with a high level of noise. However, on August 23, P3 has an unnecessarily wide PI width when the data involves a low level of noise. The QRF demonstrates the narrowest PI width in a low-noise day. In conclusion, P3 exhibits strong performance, demonstrating a narrow PI width in the data involves high uncertainty. This is because it try to reduce the maximum PI width, which typically arises in high uncertainty data. The result also aligns with the performance observed in synthetic data.

Reliability diagram. Figure 5.7 shows a reliability diagram and PI width performed on the test set. The reliability diagram is obtained by varying the confidence levels of QR, QRF, and P3 with $\gamma = 0.5$ and calculating PICP. The confidence levels vary from 0.1 to 0.9 with a spacing of 0.05. The good performance in the diagram should have the line aligned with the diagonal to demonstrate the ability to achieve the PICP at the specified confidence level. The QRF displays a line over the diagonal that indicates the PICP in the test set is greater than the confidence level, but this can lead to an excessively wide PI width. The diagram shows that P3 and QR can consistently achieve PICP at the desired confidence level, even when the confidence level varies. The bar graph compares the average and maximum PI width selected from the point with a PICP of 0.9 for each method. QRF has the lowest average PI width, while QR and P3 exhibit similar performance. The maximum PI width of P3 is significantly lower than that of other methods. Especially at noon, the difference between P3 and QR is about 700 W/m^2. Therefore, the maximum PI width can be greatly reduced when using P3. This result leads to a significant decrease in the uncertainty of solar irradiance, which leads to lower solar power uncertainty.

Application in solar power conversion. Solar power is widely recognized to have a linear proportional relationship with solar irradiance. Figure 5.8 presents an uncertainty analysis in predicted solar power based on the conversion formula $\hat{P} = \hat{a}\hat{I}$ where $\hat{a}$ is an estimated plant factor. The estimated plant factor can be calculated using any statistical method, such as linear regression, which provides a confidence interval for the estimator, denoted $[\hat{a}_{\text{lower}}, \hat{a}_{\text{upper}}]$, as shown in Figure 5.8. When the PI of solar irradiance decreases from blue bar to green bar, the PI of solar power also narrows, as illustrated by green bar in Figure 5.8. For P3 with $\gamma = 0.5$, Figure 5.5 illustrates that its maximum PI width is approximately 200 W/m^2 lower than that of QRF at noon. This led to a reduction of 0.2 per unit (p.u.) in the solar power PI width. In conclusion, applying P3 to solar irradiance reduces the PI width in solar power, lowering uncertainty in estimated solar output. This enhances planning efficiency and helps reduce operating costs in the power system.

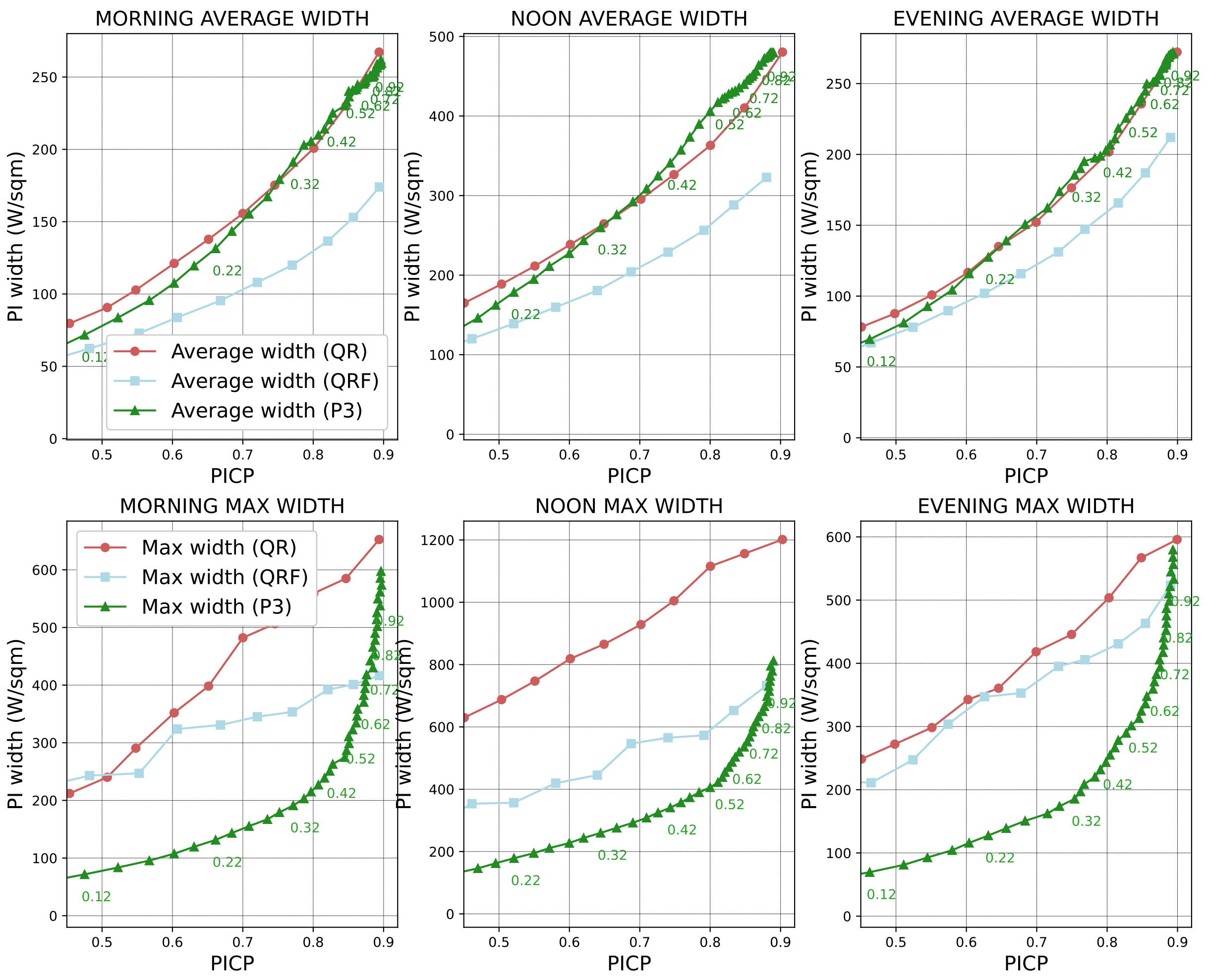


Figure 5.5: Experiment 2 result: Comparison of trade-off characteristics between PICP and PI width for QR, QRF, and P3 in solar application across different periods. (top): Between PICP and average PI width. (bottom): Between PICP and maximum PI width.

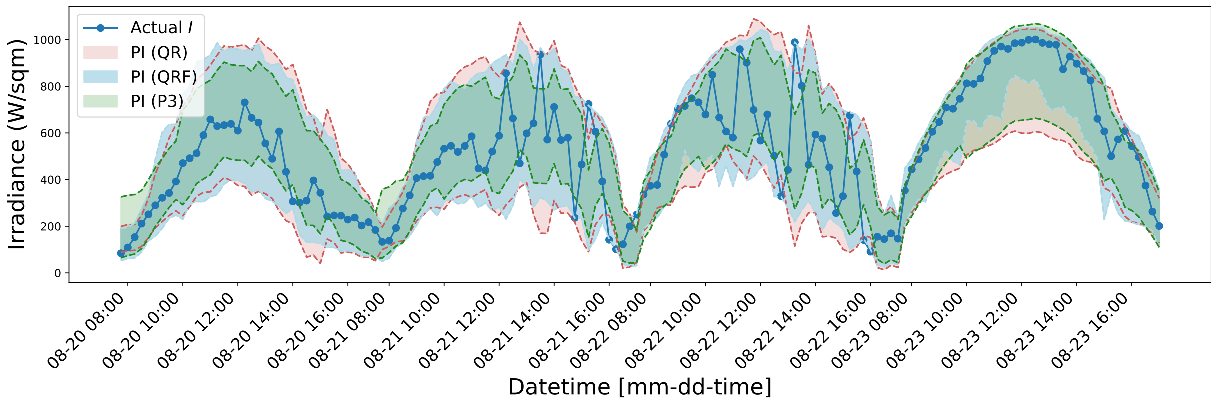


Figure 5.6: Experiment 2 result: Comparison of the 30-minute ahead PI forecast of solar irradiance from QR, QRF, and P3 with $\gamma = 0.5$.

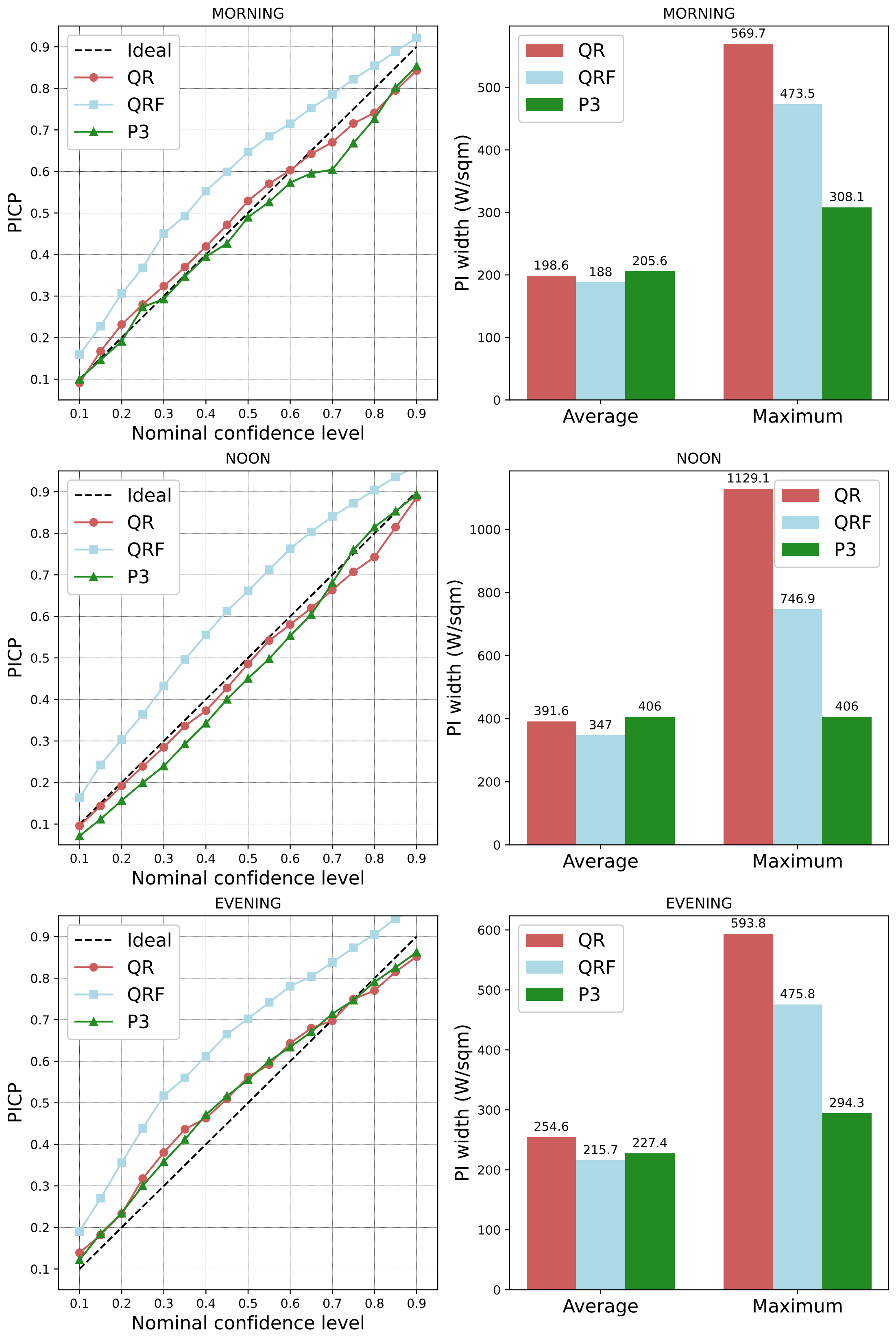


Figure 5.7: Experiment 2 result: Reliability diagram and PI widths result on test dataset obtained from varying confidence level of QR, QRF, and P3 with $\gamma = 0.5$.

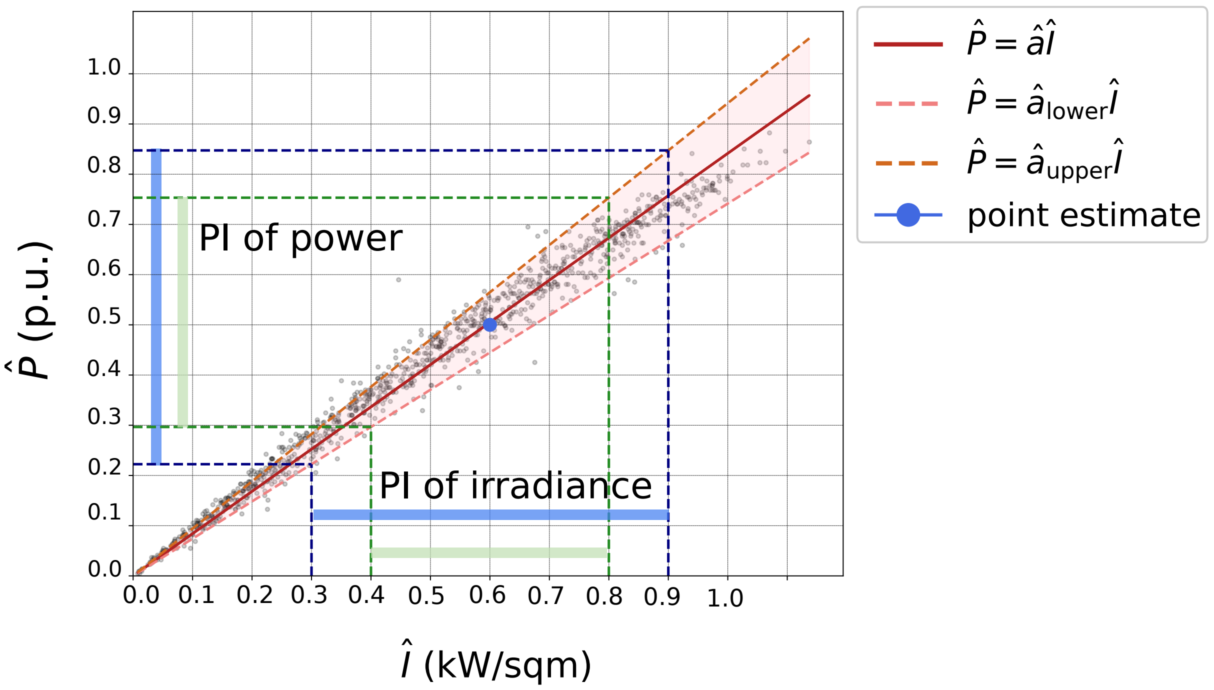


Figure 5.8: Experiment 2 result: The effect of reducing uncertainty in solar irradiance on the uncertainty in solar power.

Chapter VI

EXPERIMENTAL RESULTS OF METHODOLOGY 2

This chapter includes the experimental results that validate the concept of the second methodology, Sum-k loss, in reducing large PI widths. The second methodology involves the proposed loss function applied to a nonlinear model referred to as NN in this study. The chapter consists of two sections: experiment 3 conducted on synthetic data and experiment 4 conducted on real-world data. Each section includes the objectives, datasets or feature selection, model descriptions, experimental settings, and results with discussions. The code for experiment 3 is available at https://github.com/energyCUEE/PIestim_sumk, and the code for experiment 4 can be accessed at https://github.com/energyCUEE/PIestim_solar.

6.1 Experiment 3 - The performance of Sum-k formulation on synthetic data

This experiment aims to observe the performance of the Sum-k formulation on four synthetic datasets. The experiment can be divided into two sub-experiments: experiment 3.1 and experiment 3.2. Experiment 3.1 aims to observe the trade-off characteristics between PICP and PI width, including PINAW and PINALW, and compare them with methods with formulation hyperparameters. The trade-off curve is obtained by varying these formulation hyperparameters. Next, the characteristics of PIs among these methods are also compared with the selected operating point. Experiment 3.2 aims to compare the distribution of PI width while controlling for equal PICP as 0.9 in the validation set by utilizing the results from the previous experiment and incorporating methods without any formulation hyperparameters. This is to observe how the distribution of PI width changes due to the increased penalties on large PI widths. Next, we aim to compare the performance metrics of all methods across all datasets, including PICP, PINAW, PINALW, and the Winkler score, to provide a comprehensive evaluation of their overall performance. Two sub-experiments used the same four synthetic datasets, each containing 100 noise trials. These two sub-experiments also employed the same architecture of the feed-forward NN (ANN) model. The same optimization process settings are also utilized in both sub-experiments. A description of synthetic data, the model structure, and the optimization procedures is provided below.

Dataset

We generated four synthetic datasets using different mathematical functions. Given x as predictors, y as the target variable, and e as corrupted noise, the DGP of all datasets can be described as follows.

- **Sum of Gaussian function:** The DGP is shown as:

$$y = \beta_0 + \sum_{i=1}^{4} \beta_i \exp - \frac{(x-\mu_i)^2}{2} + e,$$

  where x is generated from uniform distribution ($\mathcal{U}$) in $[-4, 4]$ with 2,000 sorted samples. $\beta_0, \ldots, \beta_4$ are one-time generated from $\mathcal{N}(1, 1)$, and $\mu_0, \ldots, \mu_4$ is fixed as -2.4, -0.8, 0.8, and 2.4. with equal spacing. The noise e is generated from $\mathcal{N}\left(0, (\sqrt{2}\max(0, \text{sign}(|x| - 1.5)) + 0.2)^2\right)$, which means that the standard deviation equal 0.2 when $|x| \leq 1.5$, while equal $\sqrt{2} + 0.2$ otherwise. This noise causes the DGP to encompass both high and low-volatility noise regions. .

- **Polynomial function:** The DGP is inspired by the studies in Hernandez-Lobato and Adams (2015); Lin et al. (2021), which express DGP as follows:

$$y = x^3 + e$$

  where x is drawn from $\mathcal{U}(-4, 4)$ with 1,000 sorted samples. We apply heteroskedastic noise rather than uniform variance noise in the references, so e is generated from $\mathcal{N}(0, (2|x| + \exp(x))^2)$.

- **Sinusoid function:** This DGP is implemented based on Zhou et al. (2021). Given x is generated from a linearly spaced sequence between -0.5 to 0.5 with sorted 1,000 samples, y is generated from a distribution defined as:

$$y \sim \mathcal{N}(\sin(4\pi x), (0.5 + 0.3\sin(4\pi x))^2).$$

- **Multivariate function:** This DGP is inspired by Papadopoulos et al. (2001) to demonstrate the case with multiple variables of x. The DGP is defined as:

$$y = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + e$$

  where x_i for $i = 1, \ldots, 5$ are generated from $\mathcal{U}(0, 1)$ with 1,000 samples, sorted based on x_1. Then, each noise sample e is generated from $\mathcal{N}(0, 9\|x\|_2^2)$.

In the experiment, we generated 100 noise trials using the same ground truth to generalize the results. The data was randomly divided into 80% for training and 20% for validation. The training dataset was utilized to estimate the model parameters, whereas the validation set was employed to evaluate performance.

Model description and optimization scheme

In this experiment, all methods using the ANN model were controlled to have the same architecture. The model structure is shown in Table 6.1. We set a fixed random seed to control the initialization of the model parameters. For the optimizer, we selected Adam to solve an optimization problem to obtain optimal model parameters that minimize the associated loss function. Adam is well known for its high computational efficiency and its ability to avoid getting stuck in local minima (Kingma and Ba, 2017). The optimizer's learning rate was chosen based on the loss characteristics, ensuring fast convergence by monitoring the loss behavior. We used minibatch optimization to obtain fast convergence, selecting the batch size based on the number of samples. To terminate the training process, we set the maximum epochs as 2,000. A patience parameter, defined as the maximum number of epochs without improvement in loss, was set to 100 and serves as an early stopping criterion.

Table 6.1: The model architecture of the ANN used in experiment 3.

Model specification	**Setting**
Hidden layers	3
Neurons per layer	no. of input features, 100, 100, 100, 2
Activation function	ReLU
Batch Normalization	Added after hidden layers
Total number of trainable parameters	21,102 + no. of input features $\times$ 100

Experiment 3.1 - The trade-off characteristics between PICP and PI width

Objective

This sub-experiment aims to observe the trade-off characteristics between $(1-\delta)-$PICP and PI width, including PINAW and PINALW. We compare the trade-off curve among CWC$_{\text{Quan}}$, CWC$_{\text{Shri}}$, CWC$_{\text{Li}}$, QD, and Sum-k, all of which involve the formulation hyperparameter γ. To generate the trade-off curves, γ is varied for each method, and PICP and PI width are evaluated for each trial on the validation set. Then, the results from all trials for each dataset are averaged, enabling the trade-off curve to be plotted. This sub-experiment also compares the characteristics of the PI for each method across all datasets. The operating point for each method is selected to compare PI with an equal PICP of 0.9 across all methods. The PI illustrates how our formulation effectively reduces large PI widths.

Experiment setting

The desired coverage probability $(1-\delta)$ was set as 0.9 across all methods. For the formulation that includes the PICP term, the tanh smooth approximation with a softening factor of $s = 50$ was used. To plot trade-off, we varied γ with linear spacing from γ_{max} to γ_{min} for ten values to vary PICP from 0.85 to 0.9 for each dataset. For CWC$_{\text{Li}}$, we set the formulation hyperparameters $\alpha = 0.1$ and $\beta = 6$ according to the original paper. For the Sum-k, we set $k = 0.3$ to consider the widest 30 % of PI widths as large PI widths while using $\lambda = 0.1$ to emphasize the different penalties between narrow and large PI widths.

Result and discussion

Trade-off characteristics. The trade-off characteristics in all datasets are shown in Figure 6.2. The better trade-off curve is represented by the line located in the lower left region, indicating high PICP and narrow PI width. All formulations can achieve any level of PICP by varying γ across all datasets. Considering the sum of a Gaussian and a sinusoidal function, the trade-off characteristics are similar in PINAW and PINALW. The trade-off between PINAW and PICP in these datasets indicates that the curves from the CWC family are aligned and show comparable performance, while the Sum-k has the largest average PI width, as demonstrated in the trade-off curve. The benchmarked methods perform well in reducing PINAW because it is part of their loss function. For the trade-off between PINALW and PICP, the Sum-k formulation performs best in reducing the large PI width, as its loss function penalizes larger PI widths more heavily. Among benchmarked methods, the CWC$_{\text{Quan}}$ also effectively reduces the large PI width because it utilizes ℓ_2-norm in the PI width term, which heavily penalizes the large PI width. For the polynomial dataset, the trade-off curve between PINAW and PICP also reveals the same as that of previous datasets. However, in PINALW, the performance of our formulation does not show significantly better results than other methods. This result may be due to the polynomial DGP involving a lower portion of the highly volatile noise shown in Figure 6.1. For multivariate datasets, the Sum-k formulation yields the best results in both PIANW and PINALW trade-off curves. We can observe reversal characteristics where the trade-off curve has an abnormal elbow in CWC$_{\text{Shri}}$ and CWC$_{\text{Li}}$. These results are from instability in performance, as the performance on 100 trials is inconsistent, causing the average value to behave improperly. As observed in all datasets, using linearly spaced γ, the trade-off curve in the CWC family displays many closely positioned operating points. The reason comes from the mathematical formulation of the CWC family, where γ is located inside the exponential term, causing a strong penalization for deviation from PICP. Consequently, balancing the two objectives and choosing a suitable operating point can be challenging. For the formulations with an additive form of two objectives, including QD and Sum-k, we can see clear trade-off curves where increasing γ directly decreases the PI width, which is losing the PICP. On the other hand, increasing γ in the CWC family does not ensure an improvement in PICP. Additionally, the trade-off curve from CWC$_{\text{Quan}}$ exhibits nonsmooth behavior, even after averaging 100 trials, making its trade-off trend unclear.

Characteristics of PI. The comparison of PI characteristics across all datasets is shown in Figure 6.1 with equal PICP 0.9. The sum of the Gaussian and sinusoidal datasets contains both high and low volatile noise. Most methods involve a large PI width in high-volatility regions, while a narrow PI width is observed in low-volatility regions. In highly volatile regions, the Sum-k formulation shows the smallest PI width compared to benchmarked methods, while in low-volatile regions, it displays a wider PI width. The results align with the objective of the Sum-k loss, which imposes greater penalization on large PI widths that typically occur in high-variance regions while allowing the PI width in low-volatility regions to be slightly wider than usual. In the polynomial dataset, the high volatile noise portion is less than in the other datasets, where it shows in $x > 2$. The Sum-k formulation tries to reduce the large PI width in highly volatile regions, while the low volatility causes an unnecessarily wide PI width. For multivariate data, we present the value of y in the first 20 samples in Figure 6.1. The PI of Sum-k in this dataset occurs more narrowly than others, demonstrating good performance when handling multivariate functions with more than one predictor. Therefore, the Sum-k formulation alters the distribution of PI width by reducing larger PI widths while allowing smaller ones to increase. To illustrate the impact of each formulation on PI width, we compare the distribution of PI width across all methods in the next sub-experiment.

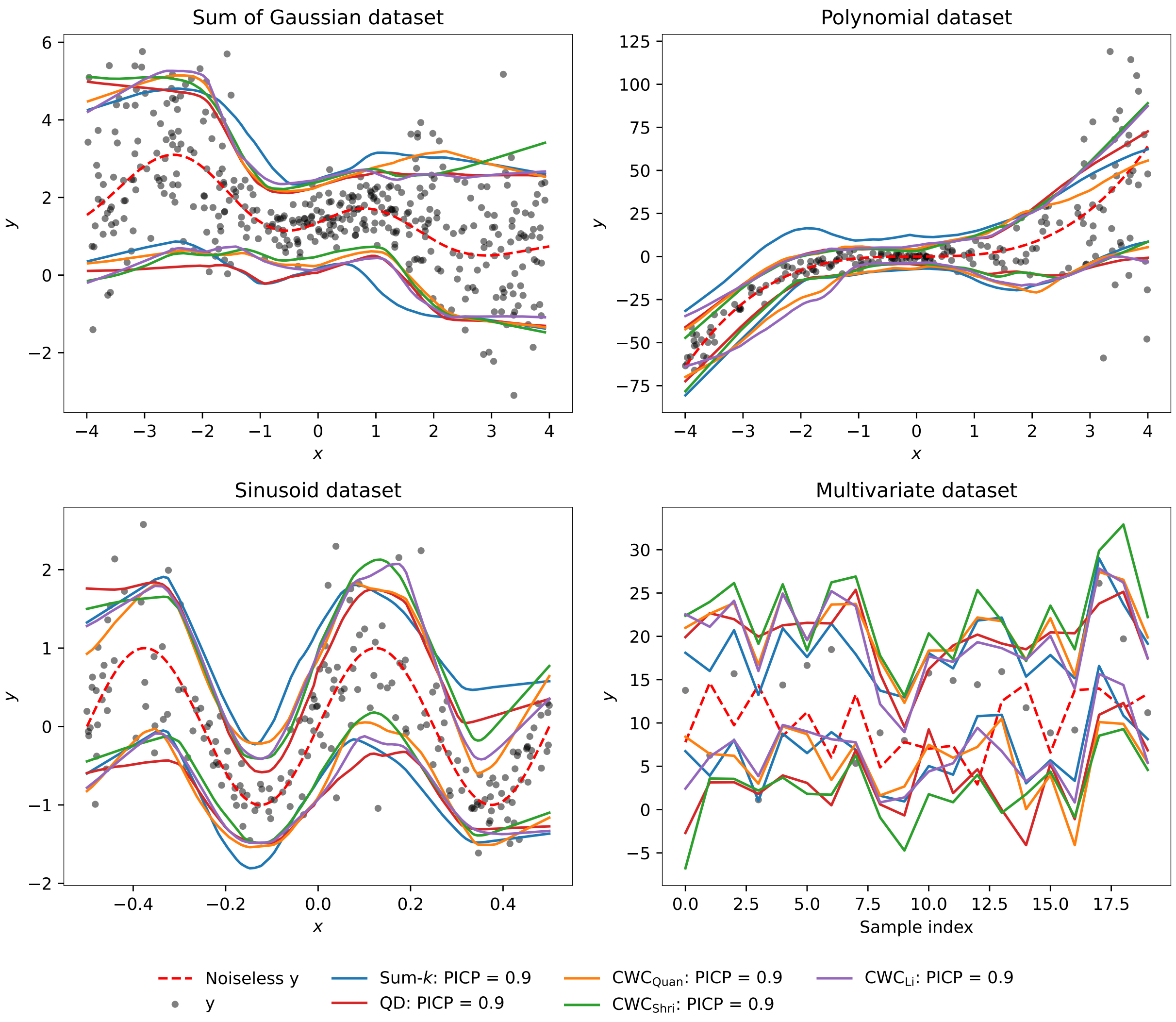


Figure 6.1: Experiment 3 result: Comparison of PI characteristics from each formulation in the sum of Gaussian dataset.

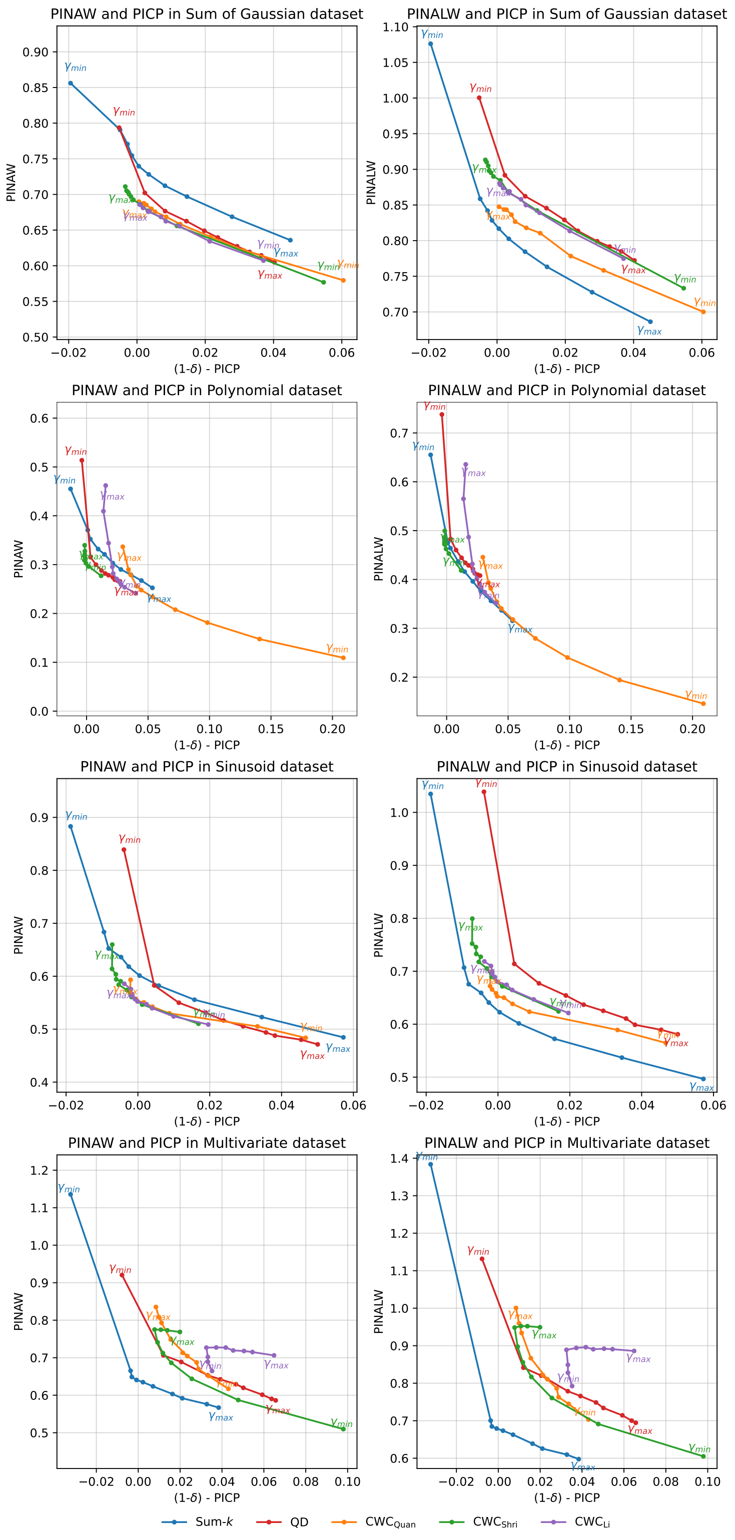


Figure 6.2: Experiment 3 result: Comparison of the trade-off curve by varying the trade-off parameter for each formulation in all datasets. (left): Between $(1-\delta)$ - PICP and PIANW. (right): Between $(1-\delta)$ - PICP and PIALW.

Experiment 3.2 - The PI width characteristics and performance metrics with control PICP in the validation set

Objective

This sub-experiment aims to compare the PI width distribution and performance metrics, maintaining a PICP control level of 0.9 across all methods, with evaluation conducted on the validation set. We benchmark our method with QR, QRF, MVE, DIC, CWC_{Quan}, CWC_{Shri}, CWC_{Li}, and QD. The performance metrics are represented by the mean and standard deviation, reflecting the results from 100 trials of noise. The distribution of PI widths for each method is illustrated using histogram plots, aggregated from 100 data trials, comparing our formulation against other methods in a one-on-one manner.

Experiment setting

For CWC_{Quan}, CWC_{Shri}, CWC_{Li}, QD, and Sum-k, we used the same setting as experiment 3.1. The operating point γ for each method was determined by analyzing the trade-off curve and selecting the γ that causes PICP to be closest to PICP 0.9. For MVE, QR, QRF, and DIC, these methods require only the confidence level as a hyperparameter, which we set to 0.9. For QR and QRF, the upper and lower bound quantiles were set at 0.95 and 0.05, respectively. The optimal hyperparameters for the QRF model were determined using the same procedure as in experiments 1 and 2. The optimal QRF models for each dataset are shown in Table 6.2.

Table 6.2: The model hyperparameters for QRF used in experiment 3.

Dataset	Model specification	Upper bound	Lower bound
Sum of Gaussian	`max_depth`	20	20
	`min_samples_leaf`	30	30
	`n_estimators`	100	100
Polynomial	`max_depth`	25	25
	`min_samples_leaf`	20	20
	`n_estimators`	100	100
Sinusoid	`max_depth`	5	25
	`min_samples_leaf`	10	20
	`n_estimators`	200	100
Multivariate	`max_depth`	25	40
	`max_features`	`sqrt`	`sqrt`
	`min_samples_leaf`	4	8
	`n_estimators`	50	50

Result and discussion

Distribution of PI width. Figure 6.3 presents a histogram of PI width obtained using the Sum-k formulation, compared to other methods, using the sum of Gaussian data as a representative example. The histogram displays two peaks that correspond to the data characteristics, which include both high and low volatile noise. The tail of the distribution from Sum-k is the shortest, demonstrating the effectiveness of our formulation in reducing the large PI width. On the other hand, the narrow PI width from Sum-k tends to be larger. Consequently, the PI width distribution from Sum-k shows the least variation with a two-sided inward shift. This result arises from setting $\lambda = 0.1$, which penalizes the mean of large PI widths relatively more than the mean of narrow PI widths by a factor of ten. Increasing λ results in a widening of the PI width distribution, aligning with the outcome observed when using PINAW as part of the loss function, which penalizes PI width equally. Therefore, the hyperparameter λ should be chosen based on the user's preference for controlling the variation in PI width.

Performance metrics. The overall performance, averaged over 100 trials, is reported in Table 6.3. Most methods can achieve the PICP at the specified confidence level in most case datasets. However, some methods may not reach a PICP of 0.9 on some datasets such as CWC_{Quan} in polynomial function. It achieves the lowest PINALW, but the PICP falls below 0.9 while the Sum-k can maintain PICP. For MVE, PICP usually falls below 0.9, especially in multivariate datasets, because the noise does not follow the Gaussian assumption required by MVE. As observed in the results, when the number of features rises, the performance of MVE is affected by the limited sample size in the log-likelihood formulation. The QR and QRF techniques indicate that the PICP corresponds with the specified confidence level. However, for the multivariate dataset, the PICP derived from QR decreases to 0.82 because it fails to align with the particular 0.05 and 0.95 quantiles.

Observing PINAW, PINALW, and Winkler scores demonstrates that the Sum-k formulation achieves the lowest PINALW across most datasets when controlling for PICP. However, the PINAW from the Sum-k increases. To improve PINAW from the Sum-k, we can increase λ, although this may lead to a decline in PINALW where selecting this hyperparameter depends on the specified requirements of each case. Consider the Winkler score; QR-based methods have the lowest Winkler score, indicating the best performance. This is because the Winkler score relates to the pinball loss, which serves as the QR objective. A lower Winkler score signifies a better alignment with the upper and lower quantiles, 0.95 and 0.05, in this experiment. Comparing QR and QRF, the QR demonstrates a better Winkler score due to the model's complexity. The QR employs a neural network model with three hidden layers, making it more complex than the QRF, which uses a tree-based model. However, the lowest Winkler score does not guarantee the lowest PI width, as shown in Table 6.3. Therefore, this result verifies that PIs from the PI-based loss with the lowest PINAW do not consistently match a fixed quantile, as discussed in Chen et al. (2024).

Upon examining the training convergence with multivariate data, we found that the maximum epochs, combined with a patience parameter, pose challenges for the convergence of CWCQuan, CWCLi, and DIC. If patience parameters are low and maximum epochs are limited, the PI width from these methods is excessively wide. The convergence of these methods is particularly slow, requiring thousands of iterations, whereas alternative methods can achieve convergence in just a few hundred iterations. In addition, a high learning rate can lead to divergence in loss and instability. To handle the divergence problem, we increase the maximum epochs and patience in this experiment to support convergence, with the performance reported in Table 6.3. A slow convergence from CWCQuan and CWCLi occurs from the mathematical formulation, where the PICP term and PI width are in multiplicative form. This leads to zero PI width as a global minimum, which serves an undesired property of PI (Pearce et al., 2018). Thus, using the additive form of the loss function leads to better compatibility with gradient-based algorithms, as illustrated by the convergence observed in CWC_{Shri}. However, DIC, also in additive form, encounters challenges in optimization due to discontinuities. This leads to a significantly high loss value when the PICP falls from 0.9 and drops to decimal values when the PICP exceeds 0.9. Then, it results in a longer time to achieve a low PI width that meets PICP requirements due to the function discontinuity. However, the entire CWC family was presented with heuristic optimization to determine optimal model parameters due to the original non-differentiable loss from the count function. In this experiment, we used a smooth approximation of the counting function in this CWC family to ensure compatibility with gradient-based algorithms, but it still struggles with poor convergence. Therefore, we have decided to exclude CWC_{Quan}, CWC_{Li}, and DIC from the benchmark candidates in the real-world dataset.

Table 6.3: Comparison of PIs performance indices on a validation set of synthesis datasets, maintaining a controlled PICP of 0.9 (mean±one standard deviation). **The bold value** indicates the best performance, with an acceptable PICP that does not fall below 0.89.

Data	Method	PICP	PINAW	PINALW	Winkler
Sum of Gaussian function	QR	0.8990±0.0134	0.7163±0.0282	0.9751±0.0427	**0.9051±0.0495**
	QRF	0.8833±0.0164	0.7192±0.0164	0.9746±0.0254	0.9547±0.0442
	MVE	0.8616±0.0229	0.6482±0.0206	0.8671±0.0312	0.9551±0.0695
	DIC	0.9028±0.0039	0.7211±0.0296	0.9497±0.0390	0.9361±0.0520
	QD	0.8977±0.0054	0.7022±0.0249	0.8917±0.0375	0.9841±0.0600
	CWC_{Quan}	0.8994±0.0067	0.6899±0.0296	0.8475±0.0399	0.9830±0.0603
	CWC_{Shri}	0.8990±0.0054	**0.6859±0.0308**	0.8845±0.0489	0.9681±0.0604
	CWC_{Li}	0.8992±0.0056	0.6861±0.0252	0.8803±0.0380	0.9681±0.0561
	Sum-k	0.8995±0.0056	0.7396±0.0269	**0.8168±0.0328**	1.0359±0.0621
Polynomial function	QR	0.8967±0.0233	0.4077±0.0274	0.7230±0.0517	0.5016±0.0507
	QRF	0.8938±0.0251	0.3516±0.0194	0.6062±0.0361	**0.4576±0.0507**
	MVE	0.8680±0.0591	0.3573±0.0309	0.6131±0.0481	0.5869±0.1685
	DIC	0.9037±0.0036	0.3692±0.0289	0.6379±0.0529	0.5497±0.0674
	QD	0.8968±0.0061	0.3153±0.0478	0.4827±0.0665	0.7361±0.1126
	CWC_{Quan}	0.8705±0.0191	0.3366±0.1229	0.4456±0.1381	1.0408±0.2040
	CWC_{Shri}	0.9005±0.0048	**0.3028±0.0280**	**0.4625±0.0516**	0.7645±0.1090
	CWC_{Li}	0.8863±0.0141	0.4094±0.1826	0.5650±0.2087	0.9640±0.2248
	Sum-k	0.8990±0.0060	0.3707±0.0344	0.4753±0.0474	0.8813±0.1049
Sinusoid function	QR	0.9041±0.0201	0.5737±0.0242	0.7806±0.0352	**0.7081±0.0443**
	QRF	0.8832±0.0221	0.6071±0.0198	0.8171±0.0342	0.8220±0.0556
	MVE	0.8722±0.0290	0.5263±0.0195	0.7074±0.0290	0.7356±0.0571
	DIC	0.9038±0.0054	0.6167±0.0312	0.8080±0.0437	0.7878±0.0518
	QD	0.8954±0.0091	0.5827±0.0252	0.7140±0.0335	0.8208±0.0689
	CWC_{Quan}	0.9002±0.0079	0.5532±0.0265	0.6526±0.0332	0.7984±0.0653
	CWC_{Shri}	0.8988±0.0077	**0.5463±0.0215**	0.6715±0.0285	0.7803±0.0589
	CWC_{Li}	0.9001±0.0091	0.5523±0.0237	0.6802±0.0333	0.7873±0.0593
	Sum-k	0.8995±0.0096	0.6012±0.0235	**0.6226±0.0240**	0.8737±0.0674
Multivariate function	QR	0.8160±0.0295	0.5517±0.0268	0.6722±0.0350	0.8990±0.0746
	QRF	0.9123±0.0220	0.7534±0.0184	0.8471±0.0216	0.9350±0.0616
	MVE	0.4705±0.0401	0.2981±0.0183	0.3669±0.0264	1.8691±0.1623
	DIC	0.9050±0.0056	0.6800±0.0435	0.8249±0.0524	**0.8784±0.0629**
	QD	0.8880±0.0073	0.7066±0.0878	0.8414±0.1156	0.9829±0.1065
	CWC_{Quan}	0.8915±0.0111	0.8353±0.1185	1.0006±0.1784	1.1242±0.1415
	CWC_{Shri}	0.8922±0.0121	0.7751±0.0711	0.9484±0.0982	1.0500±0.0988
	CWC_{Li}	0.8674±0.0255	0.7269±0.0555	0.8890±0.0744	1.1013±0.1041
	Sum-k	0.9008±0.0055	**0.6407±0.0395**	**0.6792±0.0486**	0.8798±0.0656

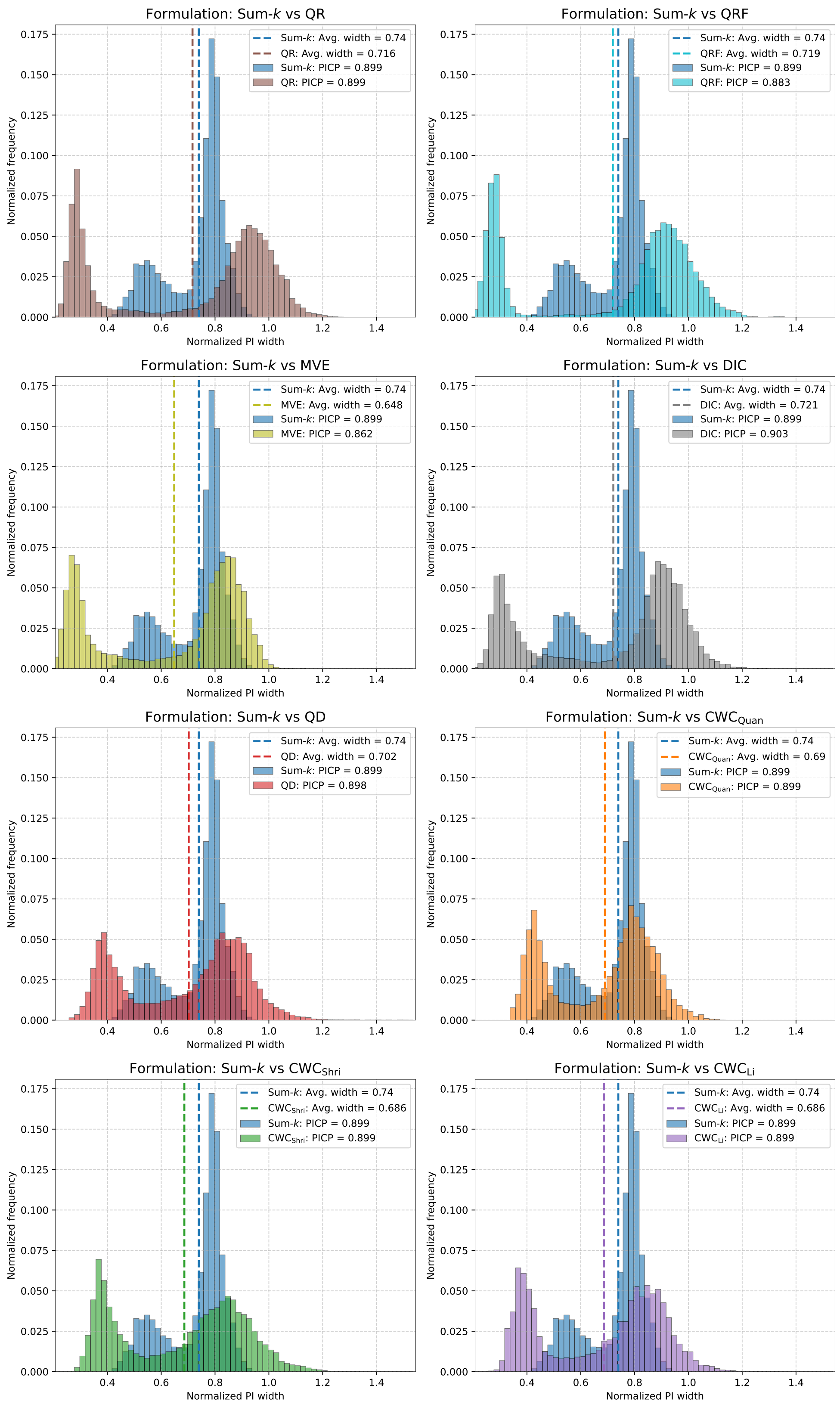


Figure 6.3: Experiment 3 result: Comparison of PI width histogram aggregated across 100 trials in the sum of Gaussian dataset.

6.2 Experiment 4 - Performing Sum-k formulation on solar irradiance forecasting

This experiment shows the application of the Sum-k formulation in estimating PI for the real-world application of solar irradiance forecasting. Due to weather conditions, solar irradiance forecasting involves significant uncertainty that is challenging to handle. Furthermore, it fluctuates significantly due to the randomness of clouds. This application is a good example of data with heteroskedastic and high volatility noise, especially the solar irradiance in Thailand, which is located near the equator. Then, we can demonstrate a significant reduction in large PI widths during the high volatility period in this experiment compared to other methods. In this experiment, the same ANN architectures are used for the Sum-k method and benchmark methods, including QR, QD, and CWC$_{\text{Shri}}$. We also demonstrate the compatibility of the Sum-k formulation with a more complex model, LSTM, to compare performance across different NN architectures.

Objective

This experiment aims to generate a one-hour ahead prediction interval (PI) for solar irradiance from 07:00 to 17:00, maintaining a 0.9 confidence level at a 15-minute resolution, covering four lead times. This forecasting specification is commonly used in the economic dispatch problem, where the uncertainty in solar irradiance provides valuable information to system operators, aiding in decision-making for improved reserve preparation and generation planning. We aim to observe the performance metrics, including PICP, PINAW, PINALW, and Winkler across all lead times when comparing the performance with QR, QD, CWC$_{\text{Shri}}$ and Sum-k using ANN and LSTM. Next, we aim to analyze the characteristics of 15-minute ahead PIs by comparing QD with Sum-k ANN. Additionally, we evaluate the model performance differences by comparing the 15-minute ahead PIs between Sum-k ANN and LSTM. Finally, we demonstrate the application of Sum-k and QD in real-time forecasting by generating one-hour ahead predictions with a four-step forecast from a specific time.

Dataset

We utilized solar irradiance data aggregated from ten solar stations located in Central Thailand, spanning January to December 2023, with a 15-minute resolution. The complete description of the dataset can be found in Section 4.4. In this experiment, we divided all predictors into two groups: lagged regressors and future regressors.

The lagged regressors utilize past observations and are categorized into auto-lagged and exogenous-lagged regressors. In this dataset, solar irradiance measurements were treated as auto-lagged regressors since the target variable is also solar irradiance. Conversely, cloud data was employed as exogenous lagged regressors, and in this experiment, we utilized only the R-channel cloud index CI$_R$. This choice was made due to the strong correlation observed in the pre-analysis. For both lagged regressors, we selected a four-period lag for solar irradiance and cloud index at $t-45, t-30, t-15, t$, based on cloud data availability and forecast specifications.

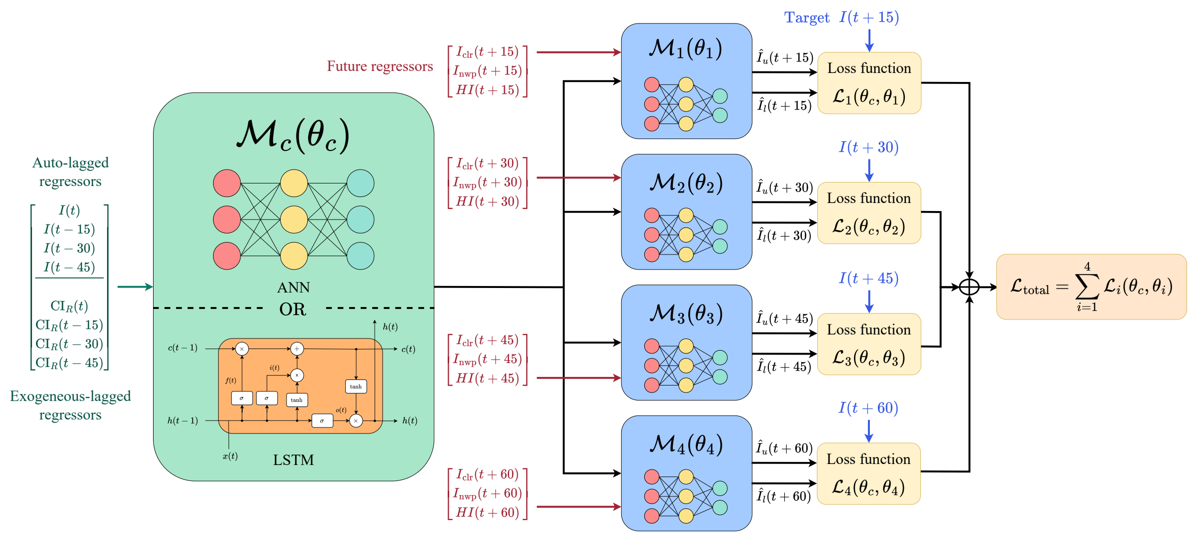
The future regressors refers to exogenous variables assumed to be available for future periods. In this experiment, the future regressors consist of clear-sky irradiance, NWP forecast variable, and hour index. For NWP forecast data, we only selected short-wave irradiance I_{nwp} due to strong correlation. The target variables, solar irradiance, were set as future values corresponding to $I(t+15), I(t+30), I(t+45)$, and $I(t+60)$ due to the forecasting specifications. Consequently, the future regressors are aligned with the forecast times $t+15, t+30, t+45$, and $t+60$.

The data set covers the period from 06:45 to 17:00. Next, the data was divided into training, validation, and test sets. The training set was used to update model parameters, while the validation set was used to monitor validation loss and tune formulation hyper-parameters to achieve the desired coverage probability. The test set was then used for performance evaluation and comparison.

Since the solar irradiance data includes both clear-sky and cloudy days, it impacts model performance differently. Clear skies represent low uncertainty, while cloudy days show high uncertainty. We aim to partition the daily solar irradiance data to ensure that the training, validation, and test sets contain a balanced proportion of different sky conditions. First, we calculated the clear-sky index (k) using the formula $k = I/I_{\text{clr}}$ for each sample, and we averaged it daily to obtain $\overline{k}$. The value of $\overline{k}$ represents the sky condition for that day, approaching one for clear-sky conditions and nearing zero for cloudy conditions. Then, we aim to categorize sky conditions into three groups: clear, partly cloudy, and cloudy. Clear-sky conditions were characterized by the smooth variation of solar irradiance, as a consistent downward pattern usually indicates clear skies. Other conditions were classified by $\overline{k}$: cloudy if $\overline{k} < 0.75$; otherwise, it was considered partly cloudy. The sky conditions were further classified based on $\overline{k}$: a day is considered cloudy if $\overline{k} < 0.75$; otherwise, it was categorized as partly cloudy. According to the classification results, this dataset shows clear-sky, partly cloudy, and cloudy conditions in a ratio of 52:11:37. Then, we divided the dataset into training, validation, and test sets using an 80:10:10 ratio, ensuring an equal distribution of each sky condition. The number of samples in each set is summarized in Table 6.4.

Table 6.4: The number of samples for the training, validation, and test sets in experiment 4.

Sky condition	**Training set**	**Validation set**	**Test set**
Clear-sky	10,469	1,254	1,311
Partly-cloudy	45,738	5,819	5,771
Cloudy	34,848	4,382	4,201
Total	91,055	11,455	11,283



Target $I(t+15)$
Future regressors
$\mathcal{M}_c(\theta_c)$
ANN
OR
LSTM
Auto-lagged regressors
Exogeneous-lagged regressors
$\mathcal{M}_1(\theta_1)$
$\mathcal{M}_2(\theta_2)$
$\mathcal{M}_3(\theta_3)$
$\mathcal{M}_4(\theta_4)$
Loss function $\mathcal{L}_1(\theta_c,\theta_1)$
Loss function $\mathcal{L}_2(\theta_c,\theta_2)$
Loss function $\mathcal{L}_3(\theta_c,\theta_3)$
Loss function $\mathcal{L}_4(\theta_c,\theta_4)$
$\mathcal{L}_{\text{total}} = \sum_{i=1}^{4} \mathcal{L}_i(\theta_c,\theta_i)$

Figure 6.4: The NN architecture used in experiment 4 includes a common model $\mathcal{M}_c$ with received lagged regressors as inputs and a submodel $\mathcal{M}_i$ with received future regressors, where the PI outputs with the target are used to evaluate the loss function.

Model and modified loss function

Model. As our forecasting specification aims to release four-step forecasting, we designed a model as shown in Figure 6.4. The model provides the upper and lower bounds for four lead times, resulting in eight output neurons. We also designed a model that can receive different predictors for each forecasting lead time, allowing the future regressors to align with the forecasted time. Therefore, the model consists of two components: a common model and four separate lead-time-specific submodels. The common model ($\mathcal{M}_c$) is designed to handle lagged regressors including I and CI$_R$. The common model shares its parameter and input layers across all lead times. In this experiment, we explored two choices of $\mathcal{M}_c$: ANN and LSTM architecture. We aim to observe the performance of capturing temporal characteristics in the time series of I and CI$_R$ between two choices. For ANN, $\mathcal{M}_c$ has two hidden layers, each containing 100 neurons. For the LSTM, we configured two layers of LSTM cells, each with a hidden size of 45, ensuring that the number of trainable parameters matches that of the ANN for performance comparison. Next, the separated lead time submodels are denoted as $\mathcal{M}_i$ for $i = 1, 2, \dots, H$ where H is the number of lead times. Each $\mathcal{M}_i$ provides the upper and lower bounds for the i^{th} step-ahead prediction. The future regressors, including $I_{\text{clr}}, I_{\text{nwp}},$, and HI, were merged with the corresponding timestamps of the target variables, as illustrated in Figure 6.4. The reason is that these corresponding timestamps of future regressors directly explain the target variable at those timestamps. We configured each submodel to be identical, including two hidden layers of an ANN, each containing 100 neurons using the ReLU activation function. Additionally, we added a batch normalization layer before the activation function to improve training stability. In conclusion, the model includes 95,808 trainable parameters for the ANN and

99,278 for the LSTM.

Loss function. For each step, the PI and the corresponding target variables are used to evaluate the loss function at each step ($\mathcal{L}_i$), as shown in Figure 6.4. According to the four-step forecast, the model is trained only once, so the loss function for each step forecast is aggregated into the overall loss ($\mathcal{L}_{\text{total}}$) by summing the individual losses into a single value as:

$$\mathcal{L}_{\text{total}}(\theta_c, \theta_1, \theta_2, \theta_3, \theta_4) = \sum_{i=1}^{4} \mathcal{L}_i(\theta_c, \theta_i), \tag{6.1}$$

where θ_c is the parameters of the common model, and θ_i for $i = 1, 2, 3, 4$ corresponds to the parameters of the submodel for the i^{th} lead time.

Experiment setting

For formulation with hyperparameter including QD, CWC$_{\text{Shri}}$, and Sum-k formulation, The γ was tuned to achieve PICP as 0.9 in the validation set across all forecast lead time. For certain values of γ, the PICP performance may vary across different forecast steps. Therefore, we selected the optimal γ where the PICP trend across different forecast steps closely aligns with 0.9. For Sum-k, we set k to 0.3 and vary λ in the validation set. We chose $\lambda = 0.9$ because the PICP is acceptable, and the large PI width remains sufficiently small. For QR, the upper and lower quantiles were set at 0.95 and 0.05, respectively, to maintain a 0.9 confidence level. We utilized ANN as a common model with the same architecture for benchmarked methods and Sum-k. Moreover, we incorporated Sum-k with LSTM as a common model to compare performance across more advanced models, while the submodel architectures remain consistent across all methods. We used the Adam optimizer as a numerical method to solve for optimal model parameters for all methods with a batch size of $0.3N_{\text{training}}$. We set the maximum number of epochs to 2,000 to terminate the training process, along with a patience of 100 for early stopping. We chose the learning rate by observing the loss as it converges during training.

Result and discussion

Table 6.5 compares evaluation metrics, including PICP, PINAW, the Winkler score, PINALW, and the reduction ratio, assessed on the test set for each lead time. For PICP results, all methods that require tuning of γ can achieve the desired coverage probability across all steps, while the PICP for QR falls below 0.9 at the 30-minute ahead. This result indicates that the formulation incorporating the trade-off hyperparameter offers greater flexibility in selecting the operating point to achieve the desired PICP. However, QR, which requires only a confidence level to align with the upper and lower quantiles, does not ensure achieving the desired PICP when losses are aggregated from various lead times. Comparing PI width among the same ANN architecture, Table 6.5 shows that the Sum-k has the lowest PINALW for all lead times and also has the lowest PINAW at 15 and 45 minutes ahead. The lowest PINALW from the Sum-k shows superior performance in reducing the large

PI width. This reduction in PINALW aligns with our goal of imposing greater penalties on larger PI widths. Regarding the Winkler score, QR demonstrates the best performance since the pinball loss used to train QR is equivalent to the Winkler score. This shows that the PI from QR closely matches quantiles 0.95 and 0.05. However, the Sum-k exhibits the highest Winkler, indicating that the PI from Sum-k deviates the most from quantiles 0.95 and 0.05. This result aligns with the findings from experiment 3, where the lowest PI width is typically achieved without aligning the PI to a specific quantile. In many real applications, such as solar forecasting, it is not necessary for PI to match a fixed quantile, as the main focus is on the PI width and PICP.

Figure 6.5 illustrates a performance comparison among all methods across all lead times in the unit of W/m^2. It emphasizes that all methods can achieve the desired PICP with an acceptable value across all lead times except for QR. For the Sum-k, with proper tuning, we can achieve the lowest PINALW while also demonstrating good performance in PINAW. As the number of forecast steps increases, PI widths typically expand due to greater uncertainty in the time series data with longer lead times. Figure 6.5 shows that the Sum-k exhibits a PI width that aligns with this trend, demonstrating a wider PI width as the forecast lead time increases. However, this trend is not assured for all methods, particularly when all lead time losses are combined and trained at the same time. This is evident in the PINAW within 45 and 60 minutes of lead time in Figure 6.5. In the training process, the longest lead time usually involves the highest PI width due to its inherent nature, so the model may prioritize reducing the PI width when minimizing the overall loss.

Figure 6.6 shows a comparison of the time series plot of solar irradiance predicted 15 minutes ahead between QD and Sum-k using the same ANN. The time series plots are displayed separately under different sky conditions to compare the effectiveness of each formulation in various scenarios of uncertainty. For each condition, we select the first four dates where QD provides the widest PINAW, calculated by averaging the PI width for each day to compare PI characteristics. The Sum-k effectively reduces the PI width on partly cloudy and cloudy days, which involve high volatile noise from cloud cover while maintaining a PICP of 0.9. Additionally, with a properly chosen λ in Sum-k, the PI width under clear-sky conditions remains effective, even with lower volatility in noise compared to QD. As observed in clear sky conditions, the PI width from Sum-k is narrower at noon, while it is usually wider in the morning and evening. This result from the Sum-k provides a narrower PI width in high uncertainty data, which usually occurs at noon, while also a wider PI width in low uncertainty region, which usually occurs in the morning and noon.

The performance comparison between Sum-k ANN and LSTM is also demonstrated in Figure 6.7. The LSTM exhibits the lowest PINALW across all lead times, showing some improvement over PINAW in comparison to ANN. Figure 6.5 also shows better results in PINALW than ANN for all lead times. When monitoring the validation loss, the final validation loss at which training terminates is lower for LSTM than for ANN, indicating its superior ability to minimize loss, corresponding to a lower PINALW. The time series comparing the 15-minute-ahead PI of ANN and LSTM from Sum-k loss. We select the first four dates with the highest PINAW in the ANN model for each sky condition. Next, we compare the PI characteristics of ANN and LSTM on these dates when ANN performs poorly. The PI from the two models reveals similar characteristics. When ANN generates large PI widths, the LSTM can decrease these large PI widths. As the same loss function, the overall PI's shape is comparable, although their different performances vary slightly based on the inherent characteristics of the model. In conclusion, this result indicates that Sum-k is suitable for training more complex NN architectures using gradient-based algorithms, allowing the integration of advanced architectures to enhance PI performance.

When PI is applied in a real application, the 4-step ahead PIs are released during a specified time shown in Figure 6.8. This figure compares the PI in forecast mode between QD and Sum-k using the same ANN model. Figure 6.8(a) illustrates instances where the PIs cover the future actual irradiance across all lead times. In this case, Sum-k exhibits a narrower PI width compared to the QD when the chosen date involves some degree of uncertainty. However, since our PI is designed to cover the data with a 0.9 confidence level, there remains a 0.1 probability that the actual future irradiance will fall outside the PI, as illustrated in Figure 6.8(b). This behavior is typical because the PI estimation can only ensure reliability at the confidence level specified by the user. As a result, the PI from Sum-k can successfully achieve PICP at the confidence level with significantly reduction in the large PI widths.

In overall performance in Table 6.5, we quantify the reduction ratio to illustrate the extent to which Sum-k LSTM can reduce the large PI widths compared with benchmarked methods. The reduction of large PI widths of solar irradiance (I) forecasts, ranging from 7.7% to 30.7% across all lead times. Since the generated solar power (P) is linearly proportional to the irradiance (I), several methods can be used to convert I to P. One common approach is to estimate the proportionality constant α in the relationship $P = \alpha I$ using least squares estimation. In Amnuaypongsa et al. (2025), the author proposed a method for estimating solar panel efficiency that accounts for curtailment effects—situations in which the output power P does not ideally follow the irradiance I, but is intentionally reduced to prevent generation from exceeding demand. As a result, reducing the PI width of I leads to a proportional decrease in the uncertainty of solar power forecasts, which could further lower operational costs for reserve preparation.

Table 6.5: Comparison of PI evaluation metrics on the test set of one-hour-ahead solar irradiance forecasting, with a resolution of 15 minutes and controlled PICP at 0.9 in the validation set. **The bold value** indicates the best performance with an acceptable PICP. The reduction ratio represents a relative decrease in PINALW from the Sum-k LSTM reference, calculated as $(X - \text{Ref.})/X$.

15-minute ahead					
Method	**PICP**	**PINAW**	**Winkler**	**PINALW**	**Reduction ratio**
QR	0.912	0.395	**0.484**	0.638	30.7%
QD	0.895	0.345	0.572	0.499	11.3%
CWC$_{\text{Shri}}$	0.895	0.342	0.611	0.501	11.8%
Sum-k ANN	0.892	**0.335**	0.656	0.449	1.6%
Sum-k LSTM	0.892	0.340	0.675	**0.442**	-
30-minute ahead					
Method	**PICP**	**PINAW**	**Winkler**	**PINALW**	**Reduction ratio**
QR	0.859	0.388	**0.547**	0.614	18.9%
QD	0.902	0.399	0.627	0.560	11.2%
CWC$_{\text{Shri}}$	0.902	0.394	0.647	0.556	10.5%
Sum-k ANN	0.893	0.399	0.694	0.523	4.9%
Sum-k LSTM	0.887	**0.377**	0.666	**0.498**	-
45-minute ahead					
Method	**PICP**	**PINAW**	**Winkler**	**PINALW**	**Reduction ratio**
QR	0.898	0.449	**0.569**	0.681	20.9%
QD	0.893	0.458	0.644	0.642	16.2%
CWC$_{\text{Shri}}$	0.900	0.457	0.643	0.653	17.7%
Sum-k ANN	0.894	0.428	0.716	0.563	4.4%
Sum-k LSTM	0.892	**0.412**	0.694	**0.538**	-
60-minute ahead					
Method	**PICP**	**PINAW**	**Winkler**	**PINALW**	**Reduction ratio**
QR	0.889	0.446	**0.579**	0.684	17.9%
QD	0.880	**0.425**	0.676	0.608	7.7%
CWC$_{\text{Shri}}$	0.901	0.442	0.684	0.640	12.2%
Sum-k ANN	0.896	0.454	0.704	0.589	4.7%
Sum-k LSTM	0.892	0.429	0.713	**0.561**	-

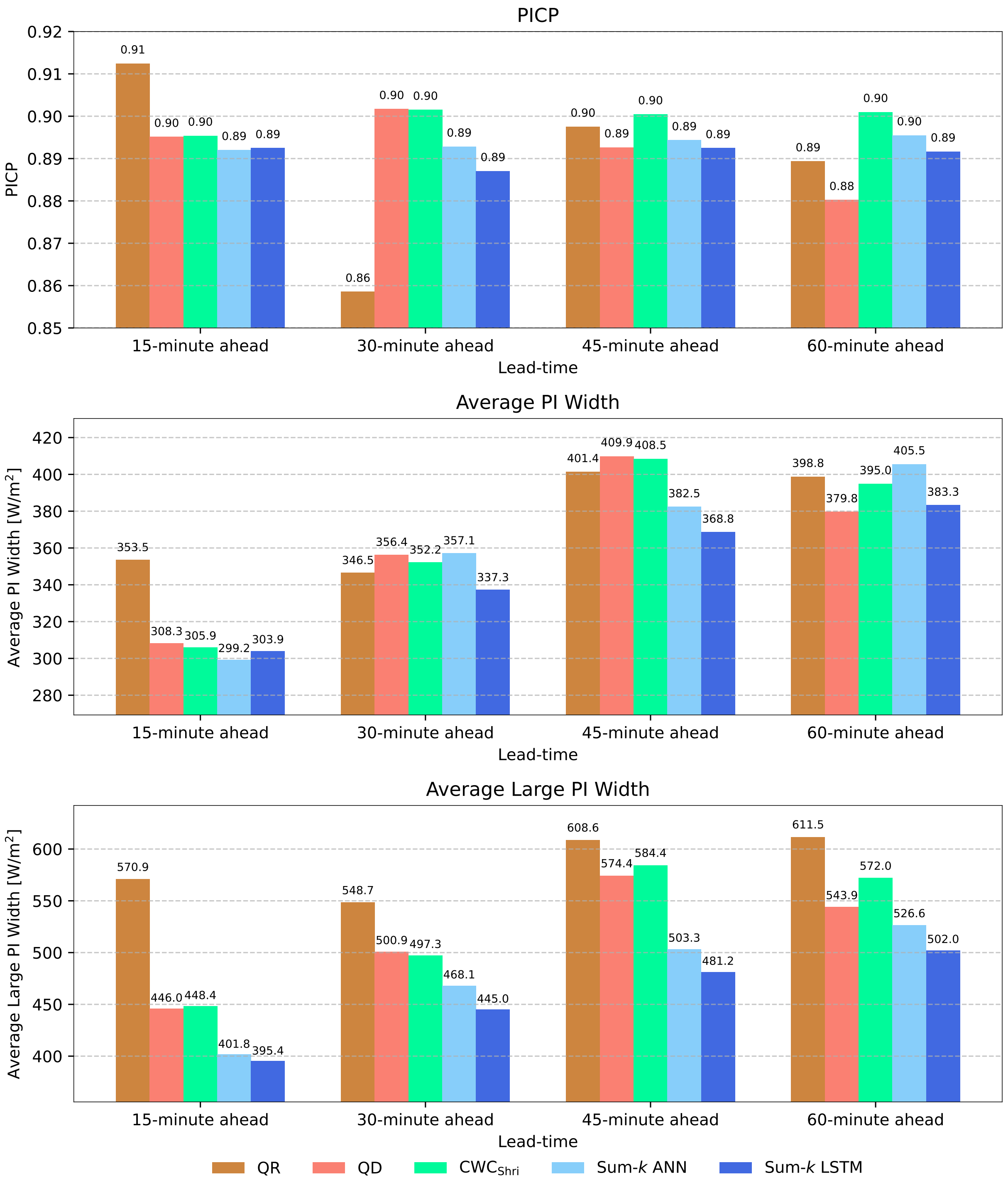


Figure 6.5: Experiment 4 result: Comparison of PI evaluation metrics on the test set in the unit of W/m^2.

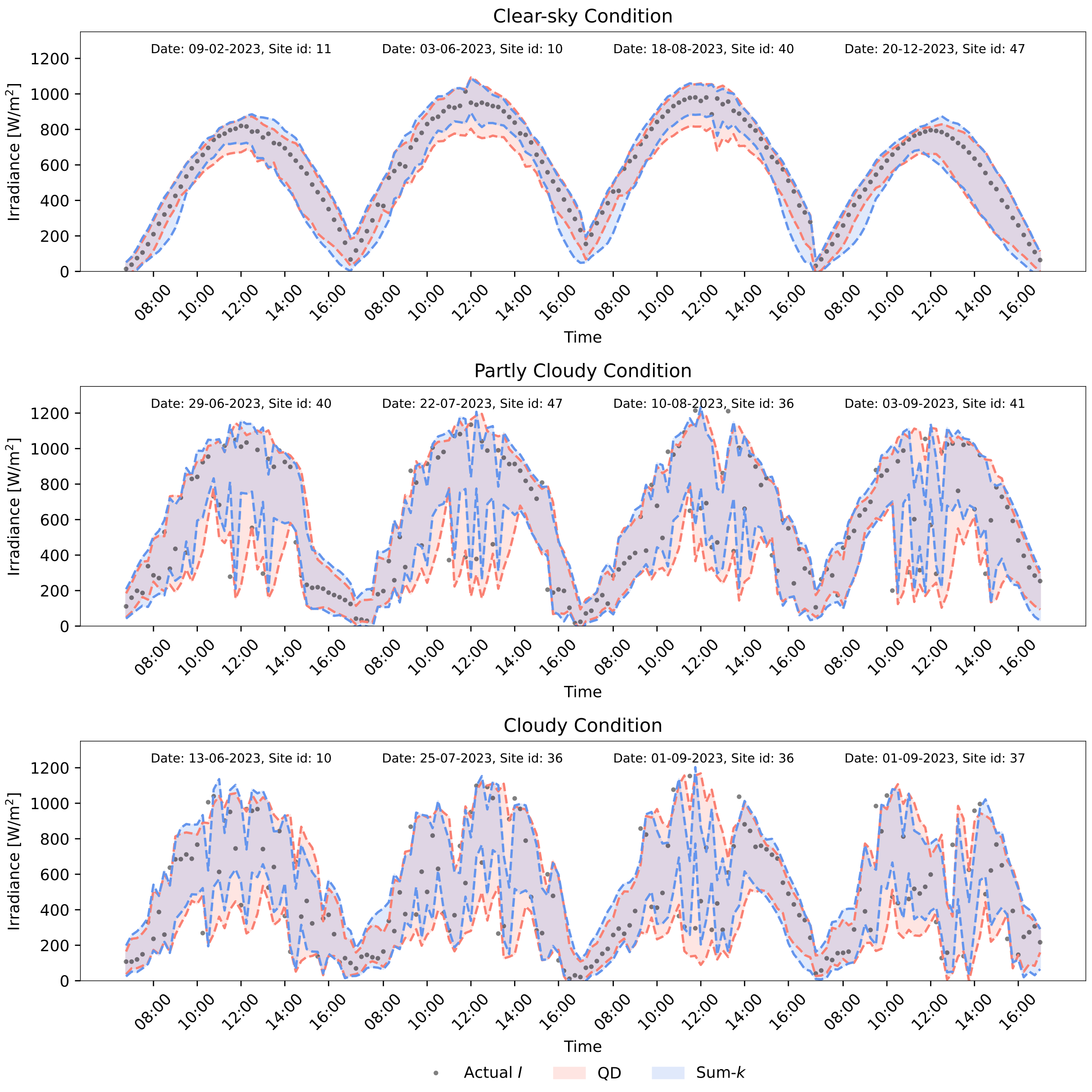


Figure 6.6: Experiment 4 result: A comparison of the 15-minute-ahead PI forecast of solar irradiance, with a confidence level of 0.9, between **Sum-k ANN and QD**.

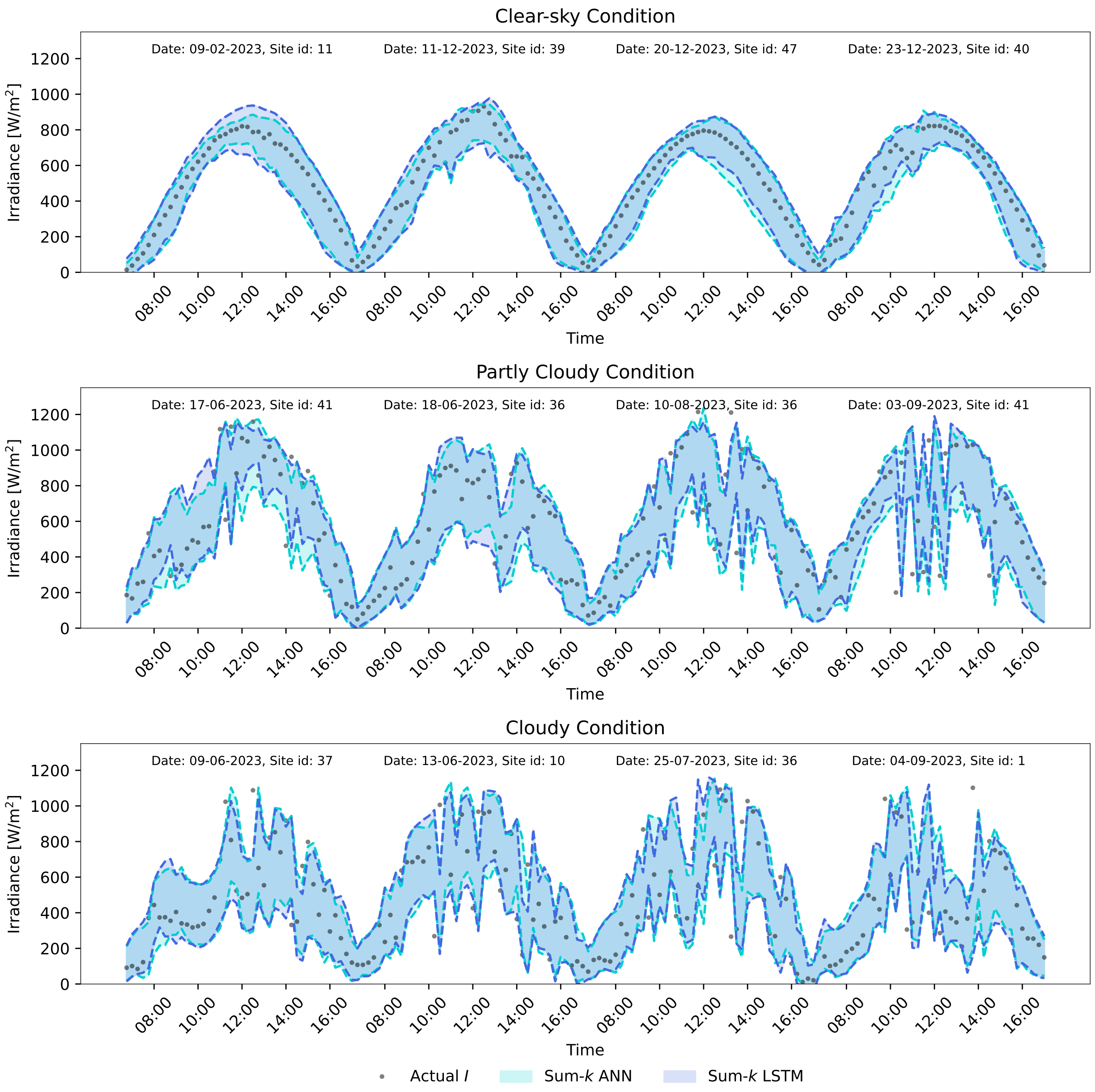
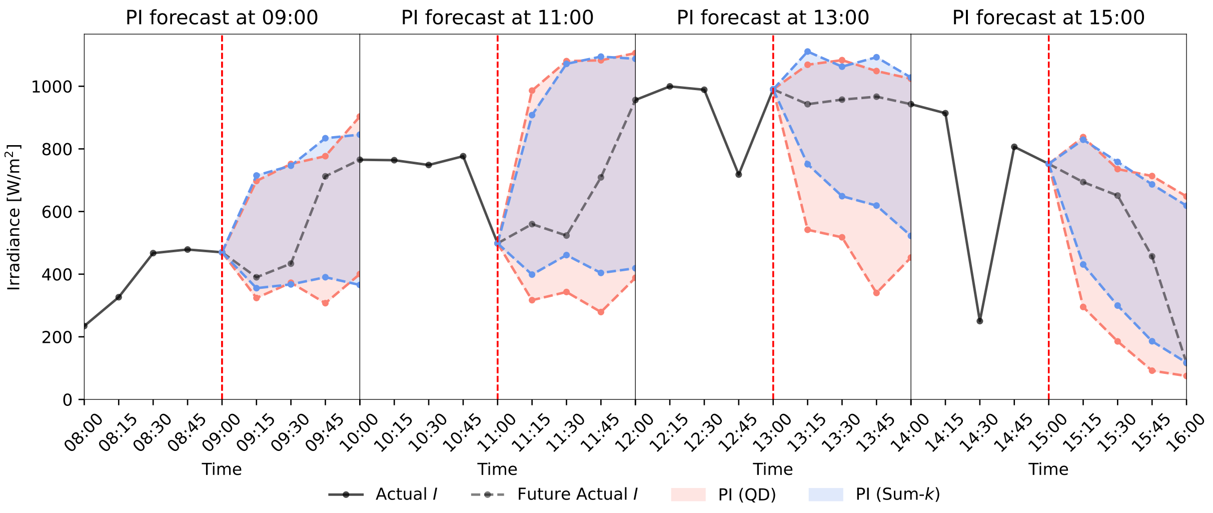
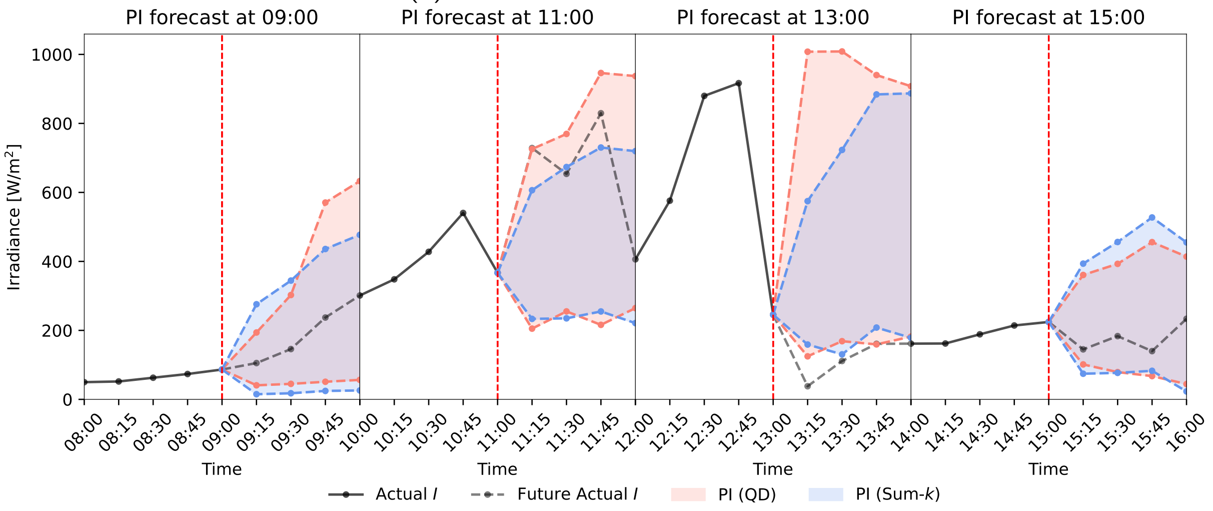


Figure 6.7: Experiment 4 result: A comparison of the 15-minute-ahead PI forecast of solar irradiance, with a confidence level of 0.9, between **Sum-k ANN and Sum-k LSTM**.



(a) Actual future are covered PI.



(b) Actual future are not covered PI.

Figure 6.8: A comparison between the Sum-k and QD formulations of a real-time 4-step PI forecast for solar irradiance, released at varying times .

Chapter VII

EFFECTIVENESS OF THE PROPOSED METHODS ON ENGINEERING SYSTEM APPLICATION

This chapter showcases the effectiveness of the methodology 2 in an engineering system application, divided into two sections. First, we demonstrate the impact of reducing the large PI width from the Sum-k formulation on reserve cost preparation. Second, we show the effect of PI width reduction from the Sum-k formulation in a robust energy management system.

7.1 Cost evaluation in reserve preparation with prediction intervals

As a solar power provider, the point forecast and its associated PI, defined by the upper ($\hat{u}(t)$) and lower ($\hat{l}(t)$) bounds, are utilized to determine the reserve requirements necessary to maintain power balance under uncertainty (Zhao et al., 2021). The point forecast $\hat{y}(t)$ represents the expected solar power generation and is typically regarded as the committed or offered power by the provider. The upper and lower bounds indicate the range within which the actual solar generation ($y(t)$) is expected to fall, with a specified confidence level. In the cost evaluation framework, two types of penalties are considered: provision penalty and deficit penalty, as illustrated in Figure 7.1. The provision penalty accounts for the reserve capacity that must be scheduled ahead of real-time operation. This includes both upward reserve ($r^U(t)$) and downward reserve ($r^D(t)$), calculated as in (7.1). The upward reserve represents the additional power that must be available in case the actual generation falls below the committed power, potentially down to the lower bound $\hat{l}(t)$. This reserve ensures system balance by compensating for shortfalls. Conversely, the downward reserve accounts for scenarios where actual generation may exceed the committed power, potentially reaching the upper bound $\hat{u}(t)$. In such cases, mitigation strategies, such as energy storage or curtailment, must be planned in advance. During real-time operation, the deficit penalty is imposed when the actual power generation falls outside the PI, as defined in (7.2). This directly relates to the PICP. When the actual generation lies below $\hat{l}(t)$, the shortfall is quantified by $r^U_-(t)$, representing the amount of lost load. If the actual generation exceeds $\hat{u}(t)$, the surplus is captured by $r^D_-(t)$, or lost opportunity because it corresponds to the amount of solar curtailment during real-time operation.

$$r^U(t) = \hat{y}(t) - \hat{l}(t), \qquad r^D(t) = \hat{u}(t) - \hat{y}(t) \tag{7.1}$$

$$r^U_-(t) = \max(\hat{l}(t) - y(t), 0), \qquad r^D_-(t) = \max(y(t) - \hat{u}(t), 0) \tag{7.2}$$

Based on the results from chapter 6, we also generated the point forecast of solar irradiance using the same model architecture in Figure 6.4 but with one output node per

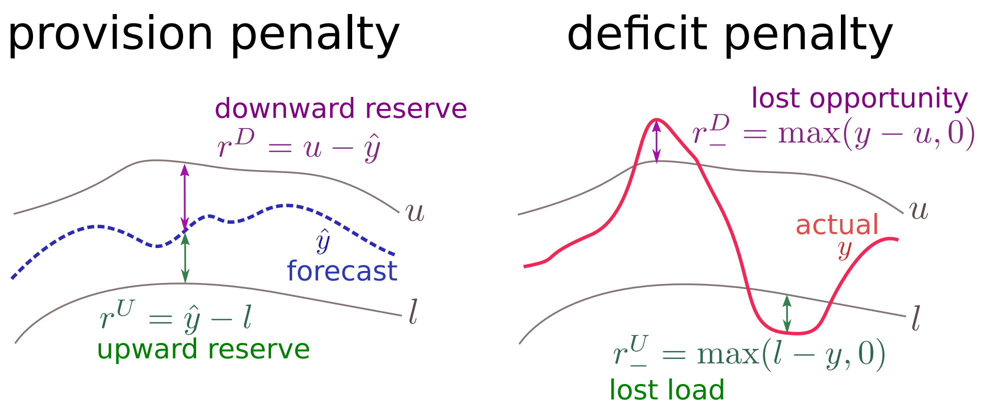


Figure 7.1: Four types of reserves quantification using prediction intervals.

Table 7.1: Reserve quantities in MWh calculated as $r = \sum_{\forall t} r(t)\Delta t$ over 4 months in test set.

	1-step ahead			4-step ahead		
	QD	CWC$_{\text{Shri}}$	Sum-k	QD	CWC$_{\text{Shri}}$	Sum-k
r^U	50,485.2	46,504.2	44,832.9	57,273.3	59,208.5	63,331.0
r_-^U	1,302.0	1,863.2	2,262.9	2,117.6	2,146.5	1,886.1
r^D	36,475.3	39,787.2	39,560.5	49,851.3	52,196.2	51,057.5
r_-^D	1,564.8	1,524.9	1,793.6	1,050.6	906.1	1,277.5

submodel, trained with pinball loss at the 0.5 quantile to estimate the conditional median. Next, we convert the solar irradiance (I) into solar power (P), corresponding to an installed capacity of 100 MW, by using $P = \eta I$ where

$$\eta = \frac{\text{solar panel efficiency (\%)} \cdot \text{area of a panel (sqm)} \cdot \text{desired capacity (W)}}{100 \cdot \text{rated power of a panel (W)}}$$

We evaluate the reserves for each type following (7.1) and (7.2) on the test set and compare them among all methods of PI generation. QR is excluded in the comparison due to crossing PI issue.

The reserves evaluated from each PI construction method for the 1-step (15 min) and 4-step (60 min) ahead are shown in Table 7.1. The results indicate that in the one-step ahead scenario, the Sum-k method achieves the lowest upward reserve, as well as the lowest combined total of upward and downward reserves, which directly corresponds to a narrower PI width. This outcome aligns with the findings in Table 6.5, where Sum-k yields the lowest PINAW in the one-step ahead setting. However, this narrower PI comes with a trade-off regarding the cost of coverage: Sum-k has the highest values of lost load and lost opportunity because its PICP is slightly lower than that of the other methods, as it focuses on reducing PI width rather than maximizing coverage. In contrast, under the four-step ahead scenario, characterized by greater uncertainty, the Sum-k achieves the *lowest lost load*, demonstrating its ability to effectively capture the actual power output in high-uncertainty conditions, while the other methods maintain competitive performance of other reserve quantities.

To evaluate the total operating reserve cost, each type of reserve is weighted by its corresponding price. The cost is then computed as:

$$\text{Total operating reserve cost} = \sum_{\forall t} \left(\pi^U r^U(t) + \pi^D r^D(t) + \pi^U_{-} r^U_{-}(t) + \pi^D_{-} r^D_{-}(t) \right) \Delta t, \quad (7.3)$$

where π^U and π^D represent the provision payments for upward and downward reserves, respectively, while π^U_{-} and π^D_{-} denote the value of lost load and the lost opportunity cost due to curtailment. These pricing parameters are defined according to Table 7.2, based on Zhao et al. (2021). We adopt a two-tier pricing scheme for the upward (π^U) and downward (π^D) reserve penalties to reflect higher costs associated with larger reserve allocations, following the practical approach proposed in Frew et al. (2021). Specifically, a base rate of \$5.5/MWh for upward and \$0.08/MWh for downward reserve is applied when the reserve requirement is less than 40 MW, with a 50% increase in the rate for reserve amounts exceeding this threshold. We also examine two settings for the value of lost load, \$50/MWh and \$500/MWh, to demonstrate how scaling this parameter impacts the total operating reserve cost.

Table 7.2: The reserve price.

Reserve price penalty	π^U	π^D	π^U_{-}	π^D_{-}
Price (\$/MWh)	5.5, 8.25	0.08, 0.12	50, 500	30

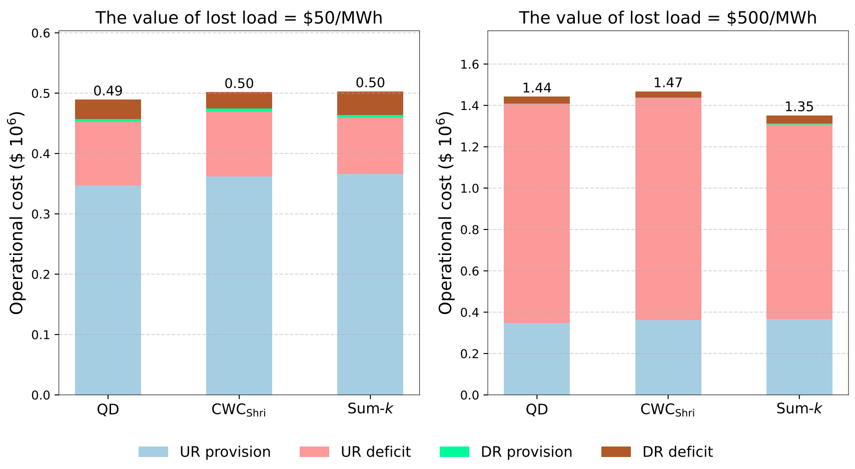


Figure 7.2: The solar power reserve cost estimated using 60-minute ahead forecasts.

We evaluate total operating reserve costs under a 4-step forecasting scenario to assess potential cost implications in a worst-case context, where wider PIs reflect increased uncertainty. Figure 7.2 presents the total operating reserve cost under two different values of lost load (VoLL), with each cost component stacked and compared across different PI estimation methods, *i.e.*, QD, CWC$_{\text{Shri}}$, and Sum-k (proposed). In both VoLL scenarios, the costs of downward reserves and lost opportunities are relatively minor compared to the overall costs, even when reserves are large. This is because managing excessive energy is much cheaper than acquiring extra energy through upward reserves or compensating for lost load. When the VoLL is set at \$50/MWh, the upward reserve cost becomes the dominant factor, making the total costs across all methods relatively similar. However, in realistic power system operations, the VoLL could reach up to \$9,000/MWh (Group) or even in ten thousands (Cartuyvels and Papavasiliou, 2023; California ISO), which strongly penalizes the load that was not served when actual generation falls below the forecasted lower bound. In this context, effective PI construction methods should achieve good r^U and r^D values, resulting in a narrow PI width that accurately reflects the amount of reserve required before real-time operation. However, during actual operation, the PICP for each method must not fall below the confidence level, as the VoLL carries a significantly higher penalty compared to other costs. Maintaining the PICP above the confidence level ensures that the VoLL does not escalate excessively. The result under the \$500/MWh scenario shows that Sum-k achieves the lowest total cost among all methods. This advantage stems from its lower bound more effectively capturing actual generation, thereby minimizing lost load penalties. In summary, this numerical cost evaluation shows that under high-uncertainty scenarios, the Sum-k effectively captures system reliability, leading to a reduction in total operating reserve cost compared to other methods.

7.2 Impact of PI width reduction in robust energy management

This section illustrates a benefit of applying our width penalization in reducing conservatism in robust energy management (EMS). Figure 7.3 shows the elements of a small building energy management system (BEMS) operated as a microgrid that consists of electrical load with peak load of 10 kW, a battery storage unit with 25kWh capacity, and a 5-kW solar rooftop system. The building is connected to the external grid served by the power utility where the outgoing power line from the building is a point of common coupling (PCC). Let $P_{\text{chg}}(t), P_{\text{dchg}}(t)$ be the charging and discharging power of batteries at time t and denote $P_{\text{pv}}(t)$ and P_{load} the generated solar power and electrical consumption occurred at the building, respectively. The power balance equation at the PCC between our system and the outside grid is the total electrical load subtracted by the total generation:

$$P_{\text{net}}(t) = P_{\text{load}}(t) - P_{\text{pv}}(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t). \tag{7.4}$$

When $P_{\text{net}}(t) > 0$, it is required to draw power flowing from the grid to serve the demand, while for $P_{\text{net}}(t) < 0$, excess self generation creates power flowing back to the grid.

Due to uncertain natures of electrical load and solar power, we can develop a probabilistic forecasting model that provides a prediction of net load defined as net load $\triangleq P_{\text{net load}}(t) = P_{\text{load}}(t) - P_{\text{pv}}(t)$. The power balance equation is represented in terms of net load as

$$P_{\text{net}}(t) = P_{\text{net load}}(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t). \tag{7.5}$$

For EMS optimization, $P_{\text{net load}}(t)$, is derived from a forecasting model constructed according to the methodology outlined in this paper. That is, it releases a prediction interval as $[L, U]$

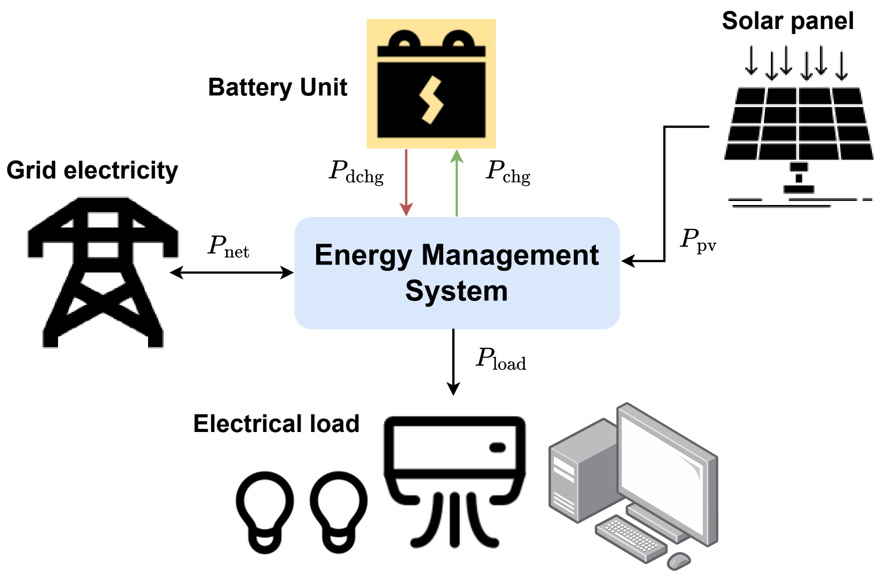


Figure 7.3: Energy management system.

associated with a confidence level of $1 - \delta = 0.9$ where the PI width is penalized to be small.

The robust EMS optimizes battery charging and discharging to maximize profit from selling excess power to the grid while meeting demand. Two formulations are implemented based on distinct constraint sets.

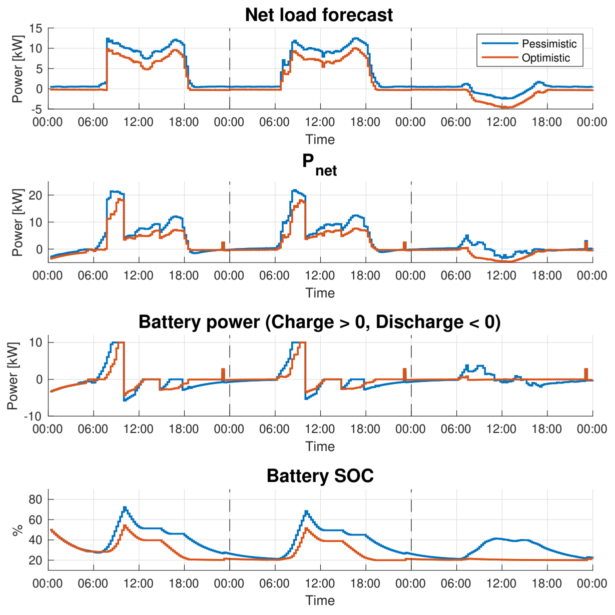
1. **Pessimistic case:** $P_{\text{net}}(t) = U(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t)$. It corresponds to the formulation (B.9) which is a robust EMS that considers the worst-case objective under a box uncertainty set.

2. **Optimistic case:** $P_{\text{net}}(t) = L(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t)$. It corresponds to the formulation (B.11) which is a robust EMS that considers a chance constraint of $P_{\text{net}}(t)$.

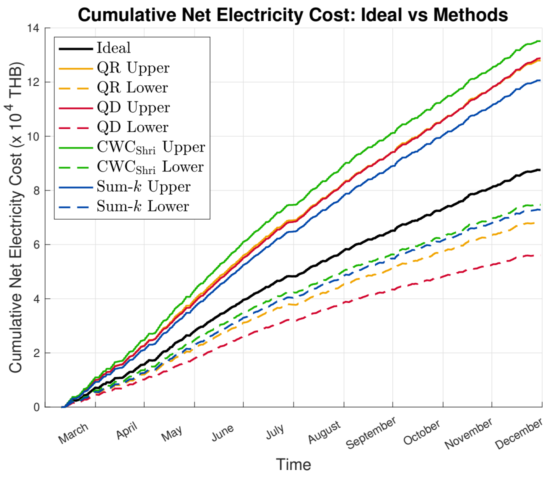
The details of two robust EMS optimizations and implementations over a 10-month period are explained in Appendix B and Appendix C. Maximized profit corresponds to minimized net electricity cost, defined as the cost of grid-purchased electricity minus the value of electricity sold back to the grid. The net electricity cost interval (representing pessimistic and optimistic scenarios) will be compared against intervals obtained using the benchmarked forecasting methods.

Figure 7.4(a) shows an example of battery operation over three consecutive days under rolling robust EMS in two scenarios. For the first two days where the net load are positive, the battery typically charges in the early morning (because the tariff is cheaper) to ensure sufficient state of charge (SoC) for later discharge within the four-hour horizon.

The charging energy is less in the optimistic scenario compared to the pessimistic one. On the third day with negative net load, the optimistic scenario leaves the battery idle and sells all excess energy to the grid. Consequently, this scenario yields a lower net energy (P_{net}) and thus a lower net electricity cost compared to the pessimistic scenario.



(a) Battery operations.



(b) The cumulative cost over 10 months.

Figure 7.4: Robust EMS operation and the net electricity cost

Table 7.4: Deviation of the net electricity cost from the ideal case.

Deviation from ideal	QR	QD	CWC$_{\text{Shri}}$	Sum-k
Pessimistic (%)	46.2	47.2	54.4	37.9
Optimistic (%)	-22.2	-36.0	-14.6	-16.8

Using the *actual* net load in the power balance equation for optimization defines an ideal battery operation. Comparing net electricity costs across pessimistic, optimistic, and ideal scenarios reveals the conservatism of the robust design using PI bounds where smaller deviations are better. Table 7.4 shows non-negative deviations for the pessimistic case (net load overestimation) and negative deviations for the optimistic case (net load underestimation). The Sum-k method demonstrates the smallest sum of absolute deviations, indicating superior PI quality in the robust EMS. Figure 7.4(b) illustrates the cumulative net electricity cost for the pessimistic (PI upper bound) and optimistic (PI lower bound) scenarios. For the upper bound, a PI estimation method that yields a lower net cost indicates superior performance, whereas a higher net cost is preferable for the lower bound. Notably, the Sum-k method exhibits the narrowest overall range between the upper and lower bounds (approximately 48k THB in 10 months) compared to benchmarked methods (60k THB for

QR, 73k THB for QD, 60k THB for $\mathrm{CWC}_{\mathrm{Shri}}$), signifying reduced uncertainty in net cost analysis for EMS.

Integrating PIs into optimal scheduling for renewable energy and EMS can be achieved through various methodologies. For instance, the study by Dong et al. (2022) employed robust model predictive control, utilizing min-max optimization with a budget constraint on renewable energy prediction errors. Their sensitivity analysis revealed a monotonic increase in operating cost in proportion to an expanded prediction error budget, which directly reflects the PI widths. In addition, a robust economic dispatch-unit commitment (EDUC) framework for renewable energy (RE) sources within microgrids was investigated in Aguilar et al. (2024). Their study utilized RE forecasts derived from ensemble Sequence-to-Sequence (Seq2Seq) models, with PIs calculated at varying confidence levels. The findings demonstrate a direct correlation between increased operating costs and wider PIs, as the confidence level increases. These examples showed the influence of PI width on reducing conservatism within robust strategies for RE sources. Consequently, a reduction in PI widths directly correlates with a decrease in operating costs in EMS.

Chapter VIII

CONCLUSION

This thesis presents methodologies for generating prediction intervals (PIs) while addressing two conflicting objectives: reliability (PICP) and sharpness (PI width). A PI consists of upper and lower bounds at a specified confidence level. The goal is to develop optimization frameworks that ensure PICP achieves the desired probability (high PICP) while minimizing PI width. To bridge the gap in the literature, this thesis primarily focuses on reducing excessively large PI widths, which can lower operational costs in various applications, particularly in reserve power generation planning.

We propose two methodologies: pinball-based formulation (methodology 1: Section 3.1) and PICP with width control formulation (methodology 2: Section 3.2). The first methodology consists of three optimization problems, P1, P2, and P3, sharing the same pinball objective with different width control constraints: average width, large width, and maximum width. The first methodology introduces three optimization problems—P1, P2, and P3—that share a pinball loss objective but impose different width control constraints: average width, large width, and maximum width. This approach employs a linear additive model (3.1), allowing the optimization to be formulated as a convex program, which guarantees numerical advantages in reaching the global minimum. The second methodology leverages the Sum-k loss function, which consists of two components: PICP and large PI width (3.8). The Sum-k function penalizes large PI widths by summing the K largest components of the width function (3.9). Additionally, a smooth tanh approximation (3.17) is introduced for the count function in the PICP term, offering a simpler alternative to the commonly used sigmoid function, ensuring differentiability. This methodology is implemented within a nonlinear modeling framework. Given the nonlinear nature of both the loss function and the model, the Sum-k loss is formulated as an unconstrained nonlinear optimization problem, solvable using gradient-based algorithms. All formulations proposed in this work include a hyperparameter, γ, which controls the trade-off between PICP and PI width.

We conduct experiments on both synthetic and real-world solar irradiance forecasting data to evaluate the effectiveness of our proposed methods. The experiments are divided into two parts: Chapter 5 covers the first methodology, while Chapter 6 focuses on the second methodology. For each methodology, we analyze the trade-off between PICP and PI width by varying γ, examine histograms of PI widths to understand their distribution and assess the characteristics of the PI generated by each formulation. For the first methodology, in synthetic data experiments, the trade-off analysis reveals that P3 outperforms other approaches in reducing the maximum PI width while maintaining the desired probability, particularly when the data is highly corrupted by heteroskedastic noise. The histogram of

PI widths demonstrates that P3 induces a two-sided inward shift in the distribution, with decreasing γ leading to reduced variation in PI width. When γ reaches a critical point, all PI widths in P3 become equal due to the activation of the maximum constraint. In the solar irradiance forecasting application, P3 also shows superior, effectively narrowing PI widths during periods of high fluctuations and uncertainty while allowing slight widening in low-uncertainty regions. For the second methodology, in the synthetic data experiment, the trade-off curve shows that the Sum-k approach successfully achieves the target PICP probability while significantly reducing large PI widths. The histogram indicates a two-sided inward shift, where large PI widths are effectively reduced while narrow PIs experience slight widening. In the solar irradiance forecasting application, Sum-k produces the narrowest PI widths among all benchmarked methods in high-uncertainty conditions, such as cloudy weather, while maintaining a competitive average PI width compared to other approaches. Additionally, results demonstrate the compatibility of Sum-k with more complex models, such as LSTMs. We also demonstrated the effectiveness of Methodology 2 in a real engineering system application in Chapter 7, including reserve preparation costs and robust energy management. According to the results, the Sum-k can reduce the reserve preparation cost compared to other methods by lowering the value of the lost load cost, which shows that our formulation can effectively cut reserve costs. Additionally, the results from robust EMS indicated that the Sum-k can estimate a narrower PI of the net load forecast compared to other methods when performing rolling EMS. The findings revealed that the Sum-k can estimate the narrowest range of the cumulative net electricity cost deviation from the ideal, and in the pessimistic case, the Sum-k shows the lowest deviation of the cost from the ideal.

Methodology 1. The advantage of the first methodology stems from the convexity of the proposed optimization problems. By utilizing a linear additive form as a model, the approach offers high interpretability, making it easier to derive insights into the relationships between features and the target variable. However, this model requires extensive feature engineering to effectively represent such nonlinearity present in the data. Additionally, since the methodology addresses two competing objectives—PICP and PI width—the pinball-based formulation provides only an indirect way of optimizing PICP. Minimizing the pinball loss results in a solution that best approximates a specific quantile but does not necessarily ensure the narrowest PI width. Furthermore, the hyperparameter γ must be required to be carefully tuned to balance these objectives, necessitating multiple solving of the optimization problem with different γ values to find the most suitable trade-off.

Methodology 2. This methodology provides a direct approach to optimizing both PICP and PI width by incorporating them into the loss function, ensuring an optimal balance between the two objectives. Since the loss function is nonlinear, it supports the use of nonlinear models, such as neural networks, while maintaining the problem as a nonlinear program. Unlike linear models, nonlinear models can capture complex patterns in the data, which makes them more suitable for addressing intricate relationships in the task. Additionally, the smoothness of the loss function allows for the application of gradient-based optimization, the standard training method for neural networks, enabling the use of more complex models. The Sum-k loss effectively reduces large PI widths by penalizing the sum of the K largest widths while still incorporating the narrow PI widths in the loss. Furthermore, introducing three hyperparameters—γ, k, and λ—enhances the flexibility of the Sum-k formulation, allowing users to fine-tune the PI to meet specific requirements. However, tuning these hyperparameters can be challenging and requires prior knowledge of the data. The parameter γ must be adjusted to achieve the desired PICP probability. Selecting k is difficult in applications where identifying high-uncertainty regions is not straightforward. Similarly, λ needs careful tuning to balance large and narrow PI widths, ensuring an appropriate distribution of PI widths. The presence of multiple hyperparameters increases the complexity of tuning. Moreover, since neural network models are highly nonlinear, they require huge computational resources and result in a loss of interpretability, leading to a lack of understanding of the relationship between predictors and the target variable.

Further implementation. The proposed method addresses conflicting multi-objective optimization by introducing the trade-off hyperparameter γ, which controls the balance between PICP and PI width. However, a key limitation is that selecting the appropriate γ requires multiple solving optimization problems to reach the desired probability. To enhance this multi-task learning approach, a more efficient multi-objective optimization framework could be employed, allowing γ to be dynamically adjusted during training. This adaptive mechanism would enable the model to efficiently converge to an optimal balance between the conflicting objectives without requiring repeated retraining. Additionally, the proposed framework could be extended to integrate point forecasts as an additional model output, with the loss function modified to incorporate regression loss alongside PICP and PI width terms. This expansion would introduce additional objectives, increasing the complexity of the optimization problem. Consequently, more advanced optimization techniques, such as multi-objective algorithms, may be required to effectively handle the increased number of objectives. In practical reserve preparation, based on our experimental results demonstrating the effectiveness of our approach, a one-hour reserve preparation horizon may not provide sufficient time to respond adequately. Extending the horizon to one day ahead offers a more realistic representation of actual reserve planning and allows for more effective preparation.

REFERENCES

Aguilar, D., Quinones, J. J., Pineda, L. R., Ostanek, J., and Castillo, L. (2024). Optimal scheduling of renewable energy microgrids: A robust multi-objective approach with machine learning-based probabilistic forecasting. *Applied Energy*, 369:123548.

Ahmed, R., Sreeram, V., Mishra, Y., and Arif, M. (2020). A review and evaluation of the state-of-the-art in pv solar power forecasting: Techniques and optimization. *Renewable and Sustainable Energy Reviews*, 124:109792.

Alcántara, A., Galván, I. M., and Aler, R. (2022). Direct estimation of prediction intervals for solar and wind regional energy forecasting with deep neural networks. *Engineering Applications of Artificial Intelligence*, 114:105128.

Amnuaypongsa, W., Suparanonrat, Y., Tongamrak, N., and Songsiri, J. (2025). Estimation of solar panel efficiency in the presence of curtailment. In *Proceedings of the 22nd International Conference on Electrical Engineering/Electronics, Computer, Telecommunication, and Information Technology (ECTI-CON)*, pages 1–6. IEEE. To appear.

Anderson, K. S., Hansen, C. W., Holmgren, W. F., Jensen, A. R., Mikofski, M. A., and Driesse, A. (2023). pvlib python: 2023 project update. *Journal of Open Source Software*, 8:5994.

Antonanzas, J., Osorio, N., Escobar, R., Urraca, R., de Pison, F. M., and Antonanzas-Torres, F. (2016). Review of photovoltaic power forecasting. *Solar Energy*, 136:78–111.

Antonanzas-Torres, F., Urraca, R., Polo, J., Perpiñán-Lamigueiro, O., and Escobar, R. (2019). Clear sky solar irradiance models: A review of seventy models. *Renewable and Sustainable Energy Reviews*, 107:374–387.

Apostolopoulou, D., Grève, Z. D., and McCulloch, M. (2018). Robust optimization for hydroelectric system operation under uncertainty. *IEEE Transactions on Power Systems*, 33:3337–3348.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

Bogdan, M., van den Berg, E., Su, W., and Candes, E. (2013). Statistical estimation and testing via the sorted ℓ_1 norm. https://arxiv.org/abs/1310.1969.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Breiman, L. (2001). Random forests. *Machine Learning*, 45:5–32.

Bremnes, J. B. (2004). Probabilistic wind power forecasts using local quantile regression. *Wind Energy*, 7:47–54.

Byrnes, L., Brown, C., Foster, J., and Wagner, L. D. (2013). Australian renewable energy policy: Barriers and challenges. *Renewable Energy*, 60:711–721.

California ISO. Price formation enhancements: Scarcity pricing: Anchoring penalty prices to the value of lost load. https://stakeholdercenter.caiso.com/StakeholderInitiatives/Price-formation-enhancements. Accessed: April 28, 2025.

Cartuyvels, J. and Papavasiliou, A. (2023). Calibration of operating reserve demand curves using a system operation simulator. *IEEE Transactions on Power Systems*, 38(4):3043–3055.

Chen, Y., Yu, S., Lim, C. P., and Shi, P. (2024). A novel interval estimation framework for wind power forecasting using multi-objective gradient descent optimization. *Sustainable Energy, Grids and Networks*, 38:101363.

Chen, Y., Yu, S. S., Lim, C. P., and Shi, P. (2023). Multi-objective estimation of optimal prediction intervals for wind power forecasting. *IEEE Transactions on Sustainable Energy*.

Cordova, S., Rudnick, H., Lorca, A., and Martinez, V. (2018). An efficient forecasting-optimization scheme for the intraday unit commitment process under significant wind and solar power. *IEEE Transactions on Sustainable Energy*, 9:1899–1909.

Davis, S. J., Lewis, N. S., Shaner, M., Aggarwal, S., Arent, D., Azevedo, I. L., Benson, S. M., Bradley, T., Brouwer, J., Chiang, Y.-M., Clack, C. T. M., Cohen, A., Doig, S., Edmonds, J., Fennell, P., Field, C. B., Hannegan, B., Hodge, B.-M., Hoffert, M. I., Ingersoll, E., Jaramillo, P., Lackner, K. S., Mach, K. J., Mastrandrea, M., Ogden, J., Peterson, P. F., Sanchez, D. L., Sperling, D., Stagner, J., Trancik, J. E., Yang, C.-J., and Caldeira, K. (2018). Net-zero emissions energy systems. *Science*, 360.

Désidéri, J.-A. (2009). Multiple-Gradient Descent Algorithm (MGDA). Research Report RR-6953, INRIA.

Dong, X., Zhang, C., and Sun, B. (2022). Optimization strategy based on robust model predictive control for RES-CCHP system under multiple uncertainties. *Applied Energy*, 325:119707.

Dozat, T. (2016). Incorporating Nesterov momentum into Adam. In *Proceedings of the 4th International Conference on Learning Representations*. ICLP workshop.

Duchi, J., Hazan, E., and Singer, Y. (2011a). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159.

Duchi, J., Hazan, E., and Singer, Y. (2011b). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, pages 2121–2159.

Etingov, P. V., Ma, J., Makarov, Y. V., and Subbarao, K. (2012). Online analysis of wind and solar part i: Ramping tool. Technical report, Pacific Northwest National Lab. (PNNL), Richland, WA (United States).

Figueiredo, M. and Nowak, R. (2016). Ordered weighted L1 regularized regression with strongly correlated covariates: Theoretical aspects. In Gretton, A. and Robert, C. C., editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51, pages 930–938. PMLR.

Frew, B., Brinkman, G., Denholm, P., Narwade, V., Stephen, G., Bloom, A., and Lau, J. (2021). Impact of operating reserve rules on electricity prices with high penetrations of renewable energy. *Energy Policy*, 156:112443.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 1050–1059. PMLR.

Galván, I. M., Valls, J. M., Cervantes, A., and Aler, R. (2017). Multi-objective evolutionary optimization of prediction intervals for solar energy forecasting with neural networks. *Information Sciences*, 418-419:363–382.

Gers, F., Schmidhuber, J., and Cummins, F. (1999). Learning to forget: continual prediction with lstm. In *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, volume 2, pages 850–855 vol.2.

Gneiting, T. and Katzfuss, M. (2014). Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1:125–151.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Group, H. E. P. About the operating reserve demand curve and wholesale electric prices. https://hepg.hks.harvard.edu/files/hepg/files/ordcupdate-final.pdf. Accessed: April 10, 2025.

Hernandez-Lobato, J. M. and Adams, R. (2015). Probabilistic backpropagation for scalable learning of bayesian neural networks. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1861–1869. PMLR.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hong, T. and Fan, S. (2016). Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting*, 32:914–938.

Ineichen, P. and Perez, R. (2002). A new airmass independent formulation for the linke turbidity coefficient. *Solar Energy*, 73:151–157.

James, G., Witten, D., Hastie, T., Tibshirani, R., and Taylor, J. (2023). *An Introduction to Statistical Learning*. Springer International Publishing.

Jeon, J. and Taylor, J. W. (2012). Using conditional kernel density estimation for wind power density forecasting. *Journal of the American Statistical Association*, 107(497):66–79.

Kanai, S., Fujiwara, Y., and Iwamura, S. (2017). Preventing gradient explosions in gated recurrent units. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Khosravi, A., Nahavandi, S., and Creighton, D. (2013). Quantifying uncertainties of neural network-based electricity price forecasts. *Applied Energy*, 112:120–129.

Khosravi, A., Nahavandi, S., Creighton, D., and Atiya, A. F. (2011). Lower upper bound estimation method for construction of neural network-based prediction intervals. *IEEE Transactions on Neural Networks*, 22:337–346.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization. https://arxiv.org/abs/1412.6980.

Kitzing, L., Mitchell, C., and Morthorst, P. E. (2012). Renewable energy policies in Europe: Converging or diverging? *Energy Policy*, 51:192–201.

Koenker, R. (2005). *Quantile Regression*. Cambridge University Press.

Lai, Y., Shi, Y., Han, Y., Shao, Y., Qi, M., and Li, B. (2022). Exploring uncertainty in regression neural networks for construction of prediction intervals. *Neurocomputing*, 481:249–257.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30, pages 1–12.

Li, B. and Zhang, J. (2020). A review on the integration of probabilistic solar forecasting in power systems. *Solar Energy*, 210:68–86.

Li, C., Tang, G., Xue, X., Chen, X., Wang, R., and Zhang, C. (2020). The short-term interval prediction of wind power using the deep learning model with gradient descend optimization. *Renewable Energy*, 155:197–211.

Li, R. and Jin, Y. (2018). A wind speed interval prediction system based on multi-objective optimization for machine learning method. *Applied Energy*, 228:2207–2220.

Li, X., Ma, L., Chen, P., Xu, H., Xing, Q., Yan, J., Lu, S., Fan, H., Yang, L., and Cheng, Y. (2022). Probabilistic solar irradiance forecasting based on xgboost. *Energy Reports*, 8:1087–1095.

Lian, C., Zeng, Z., Yao, W., Tang, H., and Chen, C. L. P. (2016). Landslide displacement prediction with uncertainty based on neural networks with random hidden weights. *IEEE Transactions on Neural Networks and Learning Systems*, 27:2683–2695.

Lin, Z., Trivedi, S., and Sun, J. (2021). Locally valid and discriminative prediction intervals for deep learning models. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. S., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 8378–8391. Curran Associates, Inc.

Lorenz, E., Hurka, J., Heinemann, D., and Beyer, H. G. (2009). Irradiance forecasting for the power prediction of grid-connected photovoltaic systems. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2:2–10.

Mattingley, J., Wang, Y., and Boyd, S. (2011). Receding horizon control. *IEEE Control Systems Magazine*, 31(3):52–65.

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

Mir, M., Kabir, H. D., Nasirzadeh, F., and Khosravi, A. (2021). Neural network-based interval forecasting of construction material prices. *Journal of Building Engineering*, 39:102288.

Morales, G. and Sheppard, J. W. (2023). Dual accuracy-quality-driven neural network for prediction interval generation. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–11.

Ni, Q., Zhuang, S., Sheng, H., Kang, G., and Xiao, J. (2017). An ensemble prediction intervals approach for short-term pv power forecasting. *Solar Energy*, 155:1072–1083.

Nix, D. and Weigend, A. (1994). Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, pages 55–60 vol.1. IEEE.

Nowotarski, J. and Weron, R. (2018). Recent advances in electricity price forecasting: A review of probabilistic forecasting. *Renewable and Sustainable Energy Reviews*, 81:1548–1568.

Papadopoulos, G., Edwards, P., and Murray, A. (2001). Confidence estimation methods for neural networks: a practical comparison. *IEEE Transactions on Neural Networks*, 12:1278–1287.

Paulescu, M., Paulescu, E., Gravila, P., and Badescu, V. (2013). *Weather Modeling and Forecasting of PV Systems Operation*. Springer London.

Pearce, T., Brintrup, A., Zaki, M., and Neely, A. (2018). High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4075–4084. PMLR.

Quan, H., Khosravi, A., Yang, D., and Srinivasan, D. (2020). A survey of computational intelligence techniques for wind power uncertainty quantification in smart grids. *IEEE Transactions on Neural Networks and Learning Systems*, 31:4582–4599.

Quan, H., Srinivasan, D., and Khosravi, A. (2014a). Particle swarm optimization for construction of neural network-based prediction intervals. *Neurocomputing*, 127:172–180.

Quan, H., Srinivasan, D., and Khosravi, A. (2014b). Short-term load and wind power forecasting using neural network-based prediction intervals. *IEEE Transactions on Neural Networks and Learning Systems*, 25:303–315.

Romano, Y., Patterson, E., and Candes, E. (2019). Conformalized quantile regression. In *Advances in Neural Information Processing Systems*, volume 32, pages 1–11.

S. Salem, T., Langseth, H., and Ramampiaro, H. (2020). Prediction intervals: Split normal mixture from quality-driven deep ensembles. In Peters, J. and Sontag, D., editors, *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 124 of *Proceedings of Machine Learning Research*, pages 1179–1187. PMLR.

Saeed, A., Li, C., and Gan, Z. (2024). Short-term wind speed interval prediction using improved quality-driven loss based gated multi-scale convolutional sequence model. *Energy*, 300:131590.

Shi, Z., Liang, H., and Dinavahi, V. (2018). Direct interval forecast of uncertain wind power based on recurrent neural networks. *IEEE Transactions on Sustainable Energy*, 9:1177–1187.

Shrivastava, N. A., Khosravi, A., and Panigrahi, B. K. (2015). Prediction interval estimation of electricity prices using PSO-tuned support vector machines. *IEEE Transactions on Industrial Informatics*, 11:322–331.

Solangi, K., Islam, M., Saidur, R., Rahim, N., and Fayaz, H. (2011). A review on global solar energy policy. *Renewable and Sustainable Energy Reviews*, 15:2149–2163.

Suwanwimolkul, S., Tongamrak, N., Thungka, N., Hoonchareon, N., and Songsiri, J. (2025). Deep-learning-based and near real-time solar irradiance map using himawari-8 satellite imageries. *Solar Energy*, 288:113262.

Tieleman, T. and Hinton, G. (2012). Lecture 6.5 rmsprop: Divide the gradient by a running average of its recent magnitude. https://cir.nii.ac.jp/crid/1370017282431050757.

van der Meer, D., Widén, J., and Munkhammar, J. (2018). Review on probabilistic forecasting of photovoltaic power production and electricity consumption. *Renewable and Sustainable Energy Reviews*, 81:1484–1512.

Winkler, R. L. (1972). A decision-theoretic approach to interval estimation. *Journal of the American Statistical Association*, 67:187–191.

Ye, L., Zhou, J., Gupta, H. V., Zhang, H., Zeng, X., and Chen, L. (2016). Efficient estimation of flood forecast prediction intervals via single and multi objective versions of the LUBE method. *Hydrological Processes*, 30:2703–2716.

Zeng, X. and Figueiredo, M. A. T. (2014). Decreasing weighted sorted $\ell_1$regularization. *IEEE Signal Processing Letters*, 21:1240–1244.

Zeng, X. and Figueiredo, M. A. T. (2015). The ordered weighted ℓ_1 norm: Atomic formulation, projections, and algorithms. https://arxiv.org/abs/1409.4271.

Zhang, G., Wu, Y., Wong, K. P., Xu, Z., Dong, Z. Y., and Iu, H. H.-C. (2015). An advanced approach for construction of optimal wind power prediction intervals. *IEEE Transactions on Power Systems*, 30:2706–2715.

Zhang, S., Andrews-Speed, P., Zhao, X., and He, Y. (2013). Interactions between renewable energy policy and renewable energy industrial policy: A critical analysis of china's policy approach to renewable energies. *Energy Policy*, 62:342–353.

Zhang, Y., Wang, J., and Wang, X. (2014). Review on probabilistic forecasting of wind power generation. *Renewable and Sustainable Energy Reviews*, 32:255–270.

Zhao, C., Wan, C., and Song, Y. (2021). Operating reserve quantification using prediction intervals of wind power: An integrated probabilistic forecasting and decision methodology. *IEEE Transactions on Power Systems*, 36:3701–3714.

Zhao, C., Wan, C., and Song, Y. (2022). Cost-oriented prediction intervals: On bridging the gap between forecasting and decision. *IEEE Transactions on Power Systems*, 37:3048–3062.

Zhao, C., Wan, C., Song, Y., and Cao, Z. (2020). Optimal nonparametric prediction intervals of electricity load. *IEEE Transactions on Power Systems*, 35:2467–2470.

Zhou, T., Li, Y., Wu, Y., and Carlson, D. (2021). Estimating uncertainty intervals from collaborating networks. *Journal of Machine Learning Research*, 22:1–47.

Appendix A

LITERATURE REVIEW OF BENCHMARKED EXPERIMENTS

We present the literature review of both the benchmarked method and scheme applied in this thesis.

Performance evaluation. The evaluation metrics of the prediction interval (PI) can be categorized into three groups: coverage probability, PI width, and aggregated metrics. The coverage probability reflects the reliability of the probabilistic forecasting desired to reach a given confidence level $(1 - \delta)$. The popular metrics in this group are PICP and average coverage error (ACE $=$ PICP $- (1 - \delta)$). Next, The PI width is assessed, referring to the sharpness of PI. There are various ways to assess the PI width, such as calculating the average PI width or the root mean square of the PI width. The average PI is equivalent to the ℓ_1 of the PI width, while the root mean square is equivalent to the ℓ_2 of PI width, being more sensitive to larger widths. The aggregated metrics combine both PICP and PI width evaluations into single-value metrics. Examples of aggregated metrics include the Winkler score, coverage width-based criterion (CWC), and the proposed loss, which incorporates both PICP and PI width terms.

Hyperparameters tuning. The methodology for constructing PI involves three groups of hyperparameters: formulation hyperparameters, model hyperparameters, and optimization algorithm hyperparameters, which affect the performance of the PI. First, most of the proposed loss functions have the formulation hyperparameters to control the trade-off between the PICP and PI width. The choice of formulation hyperparameter has a direct impact on both the PICP and the PI width. This hyperparameter is determined based on the specific characteristics of the loss function. For example, setting the formulation hyperparameter to penalize more when the PICP falls below the desired probability can lead to a higher PICP while also increasing the PI width. Next, the model hyperparameters, the neural network model also has the model hyperparameters, such as the number of hidden layers, the number of neurons, or the choices of the activation function. The model hyperparameters have an impact on the complexity of the model. A more complex model can accurately capture the nonlinearity characteristics of the dataset, which may result in a high PICP and narrow PI widths. Then, the algorithm hyperparameters, the optimization algorithm, requires tuning certain hyperparameters. The optimization hyperparameters have an indirect effect on the PICP and PI width. When the optimization hyperparameters are appropriately set, the loss function can converge to the actual minimum point, which reflects the best performance the model can achieve and the actual characteristics of the designed loss. When addressing a highly nonlinear multi-objective problem, different methods are employed, including heuristic and gradient-based approaches. The heuristic algorithm involves optimizing numerous hyperparameters based on the algorithm's definition. On the other hand, in the gradient-based algorithm, optimization hyperparameters, such as the learning rate, need to be selected according to the loss function. The performance of PI is influenced by many hyperparameters, making it challenging to compare it across different methods. To compare our proposed method with others, it's important to establish a fair comparison scheme to determine which methods produce better PI. In the following paragraph, we provide a literature review of how authors compare their methods with the work of others.

Literature reviews. In this part, we provide the literature on benchmark experiments, including the proposed methods, the evaluation metrics, the benchmarked methods, the dataset, and the experiment setting. We summarize the related literature in Table A.1 and Table A.2. We focus on the experiment setting, which describes how they compare the performance of their proposed methods with other methods. From the literature, there are two schemes to benchmark the methods for constructing the PI. We refer to the two schemes as the direct comparison and the controlled PICP comparison as shown in Table A.1 and Table A.2, respectively. Details are provided in the next paragraph.

Direct comparison. The direct comparison approach uses certain hyperparameters following recommendations from the reference work and compares them with the proposed method. This study presents a comparison of various evaluation metrics across three groups. Their proposed method demonstrates a significant improvement in reducing the PI width and performs well in aggregated metrics across multiple datasets, as demonstrated in their study. In Pearce et al. (2018), the authors proposed the quality-driven loss (Loss_{QD}) with formulation hyperparameter, then the formulation hyperparameter was tunned through trial and error and used it consistently for every dataset. The formulation hyperparameters in the benchmarked methods were set based on the reference work. In the studies by Chen et al. (2024) and Chen et al. (2023), a multi-objective gradient descent (MOGD) is used to determine the optimal trade-off hyperparameter for each iteration. This results in a common descent direction for the QD and WL-LUBE losses, respectively. There are two additional hyperparameters for this method that are set as upper and lower bounds for the optimal trade-off hyperparameter during the optimization process. These hyperparameters are obtained through a trial-and-error approach in their study. The formulation hyperparameter in the benchmarked method is set based on the original work and then fine-tuned through trial and error for the specific dataset. Meanwhile, the model hyperparameters and the optimization hyperparameters are kept consistent across all methods. In Li et al. (2020), the authors introduced an adjusted version of CWC_{ori} called CWC_{Li}. They also proposed an LSTM model and compared it with different NN model architectures. The loss function was adjusted by adding two new hyperparameters, resulting in a total of four hyperparameters for CWC_{Li}. The hyperparameters for CWC_{ori} were set based on a reference work, while the two new hyperparameters of CWC_{Li} were chosen through trial and error. The authors then compared their LSTM model, which had its hyperparameters determined through trial and error, with other models whose hyperparameters were based on the reference work. The pattern that the author introduced as an improved version of the formulation, along with the proposed model architecture, can be found in Saeed et al. (2024). The author utilized the formulation hyperparameters and benchmarked model hyperparameters based on the referenced work, while the hyperparameters of the proposed model were chosen through a trial and error process. In Lai et al. (2022), the authors introduced a new loss function comprising two terms with a formulation hyperparameter (λ) to control the trade-off between the PI width and PICP. The first term is composed of PI width and point forecast error derived from the log-likelihood function of the Gaussian distribution. The second term is the coverage term, which is similar to the coverage loss in the QD loss. To select an appropriate hyperparameter set of their loss, the authors use fine-tuning in the learning rate, decay rate, λ, and training epochs to obtain the PICP as the desired probability. They also have the performance analysis of different values of λ on selected two datasets and also suggest that the desired PICP can be obtained by varying λ. The formulation hyperparameters of other benchmarked methods are set based on the reference work. The model architechture is consistent along the methods. The author criticizes the fact that reaching the desired PICP is mentioned as being achievable by adjusting the trade-off hyperparameter, which depends on the dataset. However, only the formulation hyperparameter is varied for their loss, while the formulation hyperparameter of the benchmarked method is kept as a reference suggestion. This results in an unfair comparison, as they adjust the hyperparameters for only their method, but not for the others. In conclusion, the first benchmarked approach may be the unfair approach to comparing different formulations because the suggested hyperparameters in the reference work may not yield the same performance across different datasets. The author could have tuned the hyperparameters for their proposed method to claim better performance through trial and error.

Controlled PICP comparison. The second approach is to adjust the hyperparameters of both the proposed method and the benchmark methods to achieve the controlled PICP. They strive to control the PICP, compare the PI width with aggregated metrics across all methods at this operating point, and present the achievable PICP for each method. The result could lead to significantly better performance in reducing PI width and the aggregated loss across many datasets. It is also necessary to report the PICP for each method, which can be presented as the mean and standard deviation of PICP across various datasets. In the study by Ye et al. (2016), the CWC_{Ye} is proposed with hyperparameters chosen to attain a PICP closest to 0.9. Throughout the experiment, the model structure remains consistent, and the optimization algorithm is selected following the reference. Subsequently, a comparison is made on the performance of reducing the PI width across all methods. This scheme offers a fair comparison of different loss functions, but it only displays the PI width metric, making it difficult to illustrate the various distributions of PI width. In Alcántara et al. (2022), the authors combine model hyperparameters, formulation hyperparameters, and algorithm hyperparameters into the hyperparameter space. Then, they perform a grid search to choose the right set of hyperparameters for each method and dataset. This approach involves an extensive search in the hyperparameter space, which is very computationally expensive.

Additionally, the model hyperparameters should be consistent across all methods, and the optimization hyperparameters should be adjusted based on the loss, not the performance. In S. Salem et al. (2020), they proposed a new alternative version of QD called QD+ by adding mean square error (MSE) and constraint penalization terms. The QD+ loss involves three hyperparameters $\lambda_1, \lambda_2, \xi$. The hyperparameters λ_1 and λ_2 are adjusted using random search to find the best combination. The goal is to achieve a mean PICP equal to the desired probability with a maximum difference of 0.01. The combination with the lowest PINAW and MSE is then selected. For the benchmarked methods, the hyperparameter in QD loss is varied to achieve PICP using the same criterion as their method for a fair comparison.

Our scheme of benchmarking According to our thesis, we aim to propose new formulations that also have a formulation hyperparameter controlling the trade-off between the PICP and PI width. We plan to use the second benchmarking approach to select the formulation hyperparameters that result in the PICP being closest to the desired probability. Then, we will compare the PI width term to evaluate which methods provide the narrower PI width. To demonstrate the different characteristics of PI width from various formulations, we will compare the entire histogram of the PI width that has the same PICP. The histogram of PI width may clearly highlight advantages over just an average width. Furthermore, a few studies have analyzed the characteristics of the PI width based on the formulation. Next, we aim to control the model hyperparameters to be the same across all benchmarked methods. For the optimization hyperparameter, we aim to select the appropriate optimization hyperparameters based on the characteristic of the loss by observing the loss to reach the minimization. The full technical details of our scheme are provided in the experiment setting section.

Table A.1: Summary of literature of direct comparison approach on benchmarked experiments evaluating the performance of the prediction intervals.

Ref.	**Method/ Loss/Model**	**Evaluation metrics**	**Benchmarked methods**	**Hyperparameters tuning**	**Dataset**
Pearce et al. (2018)	Quality-driven PI/ QD/ ANN	PICP, PINAW, Loss_{QD}	MVE-GD, MVE-PSO, LUBE-GD, LUBE-PSO, QD-GD, QD-PSO	**Formulation:** the trade-off hyperparameter is determined through trial and error for each dataset, while the softening factor remains constant across all datasets. **Model:** the number of neurons depends on the size of the dataset **Algorithm:** learning rate, decay rate, and number of training epochs are tuned using random search	Synthetic data and ten open-access datasets

Ref.	Method/ Loss/Model	Evaluation metrics	Benchmarked methods	Hyperparameters tuning	Dataset
Chen et al. (2023)	MOGD WL-LUBE/ WL-LUBE/ BiGRU	PICP, PINAW, ACE, Winkler	MVE, QR, CWC-LUBE, WL-LUBE	**Formulation:** based on the reference work, then fine-tuning **Model:** controlled the same architecture based on reference work **Algorithm:** not available	Wind power
Chen et al. (2024)	MOGD QD/ QD/ BiGRU	PICP, PINAW, ACE, Winkler , Loss_{QD}	MVE, DeepAR, QR, Loss_{HIA}, QD-LUBE	**Formulation:** based on the reference work, then fine-tuning **Model:** controlled the same architecture based on reference work **Algorithm:** batch size = 128, learning rate = 0.001	Wind power
Li et al. (2020)	CWC_{Li}/CWC_{Li}/ LSTM	PICP, PINAW, PINRW, INAD, CWC	Compare among different models: SVM, KELM, ANN, LSTM	**Formulation:** μ and η are determined based on the reference work, while α and β are determined through trial and error. **Model:** trial and error **Algorithm:** means of trial-and-error	Wind power
Saeed et al. (2024)	improved QD/ improved QD/ GMSCSM	PICP, PINRW, INAD, CWC_{Li}, Computational time	LUBE ANN, Bootstrap ELM, QD-GMSCSM, QD_{imp}-LSTM	**Formulation:** based on the reference work **Model:** trial and error **Algorithm:** trial and error	Wind power
Quan et al. (2014a)	CWC_{Quan}/ CWC_{Quan}/ ANN	PICP, PINAW, CWC_{Quan}, Computational time	CWC_{ori}, delta method	**Formulation:** η is set based on trial-and-error **Model:** selected by using k-fold cross-validation method **Algorithm:** not available	6 datasets: a scalar function with heterogeneous noise, a vector function with homogeneous noise, dry blub temperature data, plasma beta-carotene data, two sets of real baggage handling system dataset
Shrivastava et al. (2015)	CWC_{Shri}/ CWC_{Shri}/ SVM	PICP, ACE, PINAW, Winkler, CWC_{Shri}	Naive, QR, CART, LR, Bootstrap, MVE, LUBE NN, SVM-PSO	**Formulation:** trial-and-error and set different for various case studies **Model:** grid search **Algorithm:** not available	Synthetic: Sinc function, Real world data: electricity price

Ref.	Method/ Loss/Model	Evaluation metrics	Benchmarked methods	Hyperparameters tuning	Dataset
Lai et al. (2022)	Uncertainty-based PI (UBPI)/ UBPI/ ANN	PICP, PINAW	QR, QR_{GBDT}, IntPred, QD, QD+	**Formulation:** Random search of λ to reach PICP = 0.95 **Model:** controlled the same across all methods where the number of neurons depends on the size of the dataset **Algorithm:** Adam optimizer with learning rate = 0.2, decay rate = 0.95, training epochs = 2000 which obtained by random search	Synthetic data: two sinusoidal functions, and Real-world data: 9 open-access datasets from UCI

Table A.2: Summary of literature of the controlled PICP comparison approach on benchmarked experiments evaluating the performance of the prediction intervals.

Ref.	Method/ Loss/Model	Evaluation metrics	Benchmarked methods	Hyperparameters tuning	Dataset
Ye et al. (2016)	CWC_{Ye}/CWC_{Ye}/ ANN	PICP, PINAW, PINRW, PIARW	CWC_{ori}, CWC_{Quan}	**Formulation:** η_1, η_2 are selected to achieve a PICP of 90% **Model:** trial and error **Algorithm:** based on the reference work	Flood forecast
Li and Jin (2018)	Multi-objective/ (1 - PICP, 1 - PINAW)/ LSSVM	PICP, ACE, PINAW, Winkler, CWC_{ori}, Computational time, Pareto front	Compare among different multi-objectives optimization algorithms: MOALO, NSGA-II, PESA-II, SPEA-II, MOPSO	**Formulation:** selected the operating point in Pareto from the lowest CWC to be compared **Model:** not available **Algorithm:** controlled to have equal numbers of hyperparameters and the same loss function	Wind speed prediction
Alcántara et al. (2022)	Hypernetwork (HN)/ (PINAW, 1 - PICP)/ ANN	PICP, PINAW, $\frac{\text{PICP}}{\text{PINAW}}$	QR, QD	**Formulation:** Select the operating point in the Pareto front that meets the desired PICP. **Model, algorithm:** Both hyperparameters are aggregated as a set, and a grid search is performed to find the best combination.	Solar and wind energy

Ref.	Method/ Loss/Model	Evaluation metrics	Benchmarked methods	Hyperparameters tuning	Dataset
S. Salem et al. (2020)	QD+ with new aggregation method (SNM)/QD+/ ANN	PICP, PINAW, MSE	SNM-QD+, SEM-QD+, SEM-QD, MVE	**Model:** controlled the same across all methods where the number of neurons depends on the size of the dataset **Formulation, algorithm:** learning rate, decay rate, number of epochs, λ_1, λ_2 are selected when mean PICP is equal to the desired probability with variation no more than 0.01, then select the hyperparameters with lowest PINAW and MSE.	ten open-access datasets from the UCI dataset

Appendix B

ROBUST EMS FORMULATIONS

This section describes the details of robust EMS used in section 7.2. From the system description, the system dynamics and constraints are contributed from the power balance equation and battery conditions as follows.

1. **Power balance equation.**
$$P_{\text{net}}(t) = P_{\text{load}}(t) - P_{\text{pv}}(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t). \tag{B.1}$$

2. **Battery dynamics.** Let SoC(t) be the state-of-charge of the battery at time t that has the dynamics described by
$$\text{SoC}(t+1) = \text{SoC}(t) + \frac{100\%}{\text{BattCapacity}}\left(\eta_c P_{\text{chg}}(t) - \frac{P_{\text{dchg}}(t)}{\eta_d}\right)\Delta t \tag{B.2}$$
where $0 < \eta_c, \eta_d < 1$ are charging and discharging efficiency coefficients, respectively. The SoC variable obeys a linear state-space equation where $P_{\text{chg}}(t)$ and $P_{\text{dchg}}(t)$ act as the inputs. Here, we employ separate variables for charging and discharging powers, P_{chg} and P_{dchg}, that is different from using a single variable P_{batt} with its sign indicating charging/discharging status.

3. **Charge and discharge limit rate.** For $t = 1, 2, \dots, T$,
$$0 \leq P_{\text{chg}}(t) \leq \text{max charge rate}, \quad 0 \leq P_{\text{dchg}}(t) \leq \text{max discharge rate}, \tag{B.3}$$

4. **SoC range.** To save battery life, we keep the SoC within a desired range.
$$\text{SoC}_{\text{min}} \leq \text{SoC}(t) \leq \text{SoC}_{\text{max}}, \quad t = 1, 2, \dots, T. \tag{B.4}$$

The optimization variables in our EMS optimization are:

- $P_{\text{chg}}(t)$, $P_{\text{dchg}}(t)$: battery charging/discharging power, for $t = 1, 2, \dots, T$.
- SoC(t): battery's state of charge for $t = 1, 2, \dots, T$.
- $P_{\text{net}}(t)$: net power between the building and the external grid, for $t = 1, 2, \dots, T$.

Objective function The objective function is a weighted sum of economic and battery objectives: $J_{\text{cost}} + w_b J_{\text{batt}}$ where the weight $w_b > 0$ controls the trade-off between the two objectives.

1. **Economic objective:** Given utility buy rate $b(t)$ and sell-back rate $s(t)$ (THB/kWh), the objective is to minimize the negative profit, which is the difference between energy expense and revenue from selling excess energy.

$$J_{\text{cost}} = \Delta t \sum_{t=1}^{T} b(t)\,\text{max}(0, P_{\text{net}}(t)) - s(t)\,\text{max}(0, -P_{\text{net}}(t)) \tag{B.5}$$

   Note that the revenue is calculated from $P_{\text{net}}(t) < 0$ where the power flows back to the grid, so we employ $\text{max}(0, -P_{\text{net}}(t))$ to represent the portion of negative net power to be sold.

2. **Battery objective:** To avoid abrupt battery charging and discharging, consecutive changes in $P_{\text{chg}}(t)$ and $P_{\text{dchg}}(t)$ are penalized.

$$J_{\text{batt}} = \Delta t \sum_{t=1}^{T-1} |P_{\text{chg}}(t+1) - P_{\text{chg}}(t)| + \Delta t \sum_{t=1}^{T-1} |P_{\text{dchg}}(t+1) - P_{\text{dchg}}(t)|. \tag{B.6}$$

Uncertainty The power balance equation contains P_{load} and P_{pv} taken as problem parameters. Due to uncertain natures of electrical load and solar power, we can develop a probabilistic forecasting model that provides a prediction of net load defined as net load $\triangleq P_{\text{net load}}(t) = P_{\text{load}}(t) - P_{\text{pv}}(t)$. The power balance equation (B.1) is altered to

$$P_{\text{net}}(t) = P_{\text{net load}}(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t). \tag{B.7}$$

To run EMS optimization, $P_{\text{net load}}(t)$ is from a forecasting model that is constructed according to the methodology proposed in this paper. That is, it releases a prediction interval as $[L, U]$ associated with a confidence level of $1-\delta = 0.9$ where the PI width is penalized to be small. In other words, we can define an uncertainty set of the net load as a rectangular box set:

$$\mathcal{U} = \{P_{\text{net load}}(t) \mid L(t) \leq P_{\text{net load}}(t) \leq U(t)\ \}.$$

The power balance equation (B.7) becomes the inequality:

$$L(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t) \leq P_{\text{net}}(t) \leq U(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t). \tag{B.8}$$

Robust EMS with uncertainty set The robust EMS considers the worst-case objective where it can be shown that the maximum of negative profit under uncertainty of net load occurs when the net load achieves its upper bound. The minimization of worst-case negative profit can be described as follows.

$$\begin{array}{ll} \text{minimize} & J_{\text{cost}} + w_b J_{\text{batt}} \\ \text{subject to} & P_{\text{net}}(t) = U(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t), \\ & \text{Battery constraints (B.2),(B.3),(B.4)}\ . \end{array} \tag{B.9}$$

Robust EMS with chance constraint We consider a probabilistic constraint that guarantees the uncertain variable lying within a region. Since we construct a PI of net load associated with a confidence level, we can guarantee that **prob** $(P_{\text{net load}}(t) \in [L(t), U(t)]) = 1-\delta$ and that $P_{\text{net}}(t)$ is affine in $P_{\text{net load}}(t)$. It then follows that the probability that $P_{\text{net}}(t)$ lies within

$$L(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t) \leq P_{\text{net}}(t) \leq U(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t) \tag{B.10}$$

is $1 - \delta$. We can then replace the chance constraint by using (B.10) as a constraint in the robust EMS formulation which suggests that the net power P_{net} lies within an interval. Since the negative profit (B.5) is monotonically increasing with $P_{\text{net}}(t)$, its minimum over (B.10) occurs when $P_{\text{net}}(t)$ achieves its lower bound, $L(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t)$. The robust EMS with a chance constraint can be expressed as

$$\begin{array}{ll} \text{minimize} & J_{\text{cost}} + w_b J_{\text{batt}} \\ \text{subject to} & P_{\text{net}}(t) = L(t) + P_{\text{chg}}(t) - P_{\text{dchg}}(t), \\ & \text{Battery constraints (B.2),(B.3),(B.4)} \ . \end{array} \tag{B.11}$$

Since the objective function and constraints are linear in the optimization variables, the problems (B.9) and (B.11) are linear programs (LP).

Appendix C

ROLLING EMS OPTIMIZATION

We consider the robust EMS that plans the battery operation over a 4-hour horizon with a resolution of 15 minutes and this process repeats every 15 minutes over a period of March-Dec, 2023. The rolling EMS optimization is essentially a robust EMS with MPC (model predictive control) or receding horizon control Mattingley et al. (2011) that works as follows. At time t, the battery operations in H step-ahead, $t+1, t+2, \ldots, t+H$ is planned (here, $H = 16$). The MPC is performed according to the following steps.

1. **Prediction.** A net load forecasting model must provide a PI of H-step ahead. That is,
$$[L(t+1), U(t+1)], [L(t+2), U(t+2)], \ldots, [L(t+H), U(t+H)]$$
are estimated from the forecasting model and used as problem parameters for the robust EMS formulations.

2. **Optimization.** The optimizations (B.9) and (B.11) are solved at time t, by using the estimated PI in the predictive step, and gives the charging/discharging power of the battery at time $t+1, t+2, \ldots, t+H$, denoted as $P_{\text{chg,ems}}(t+k), P_{\text{dchg,ems}}(t+k)$ for $k = 1, \ldots, H$.

3. **Execute.** Profiles of $P_{\text{chg,ems}}(t+k), P_{\text{dchg,ems}}(t+k)$ are taken as the input to update the system dynamics (SoC and net power) but only the first time step is applied. Since the EMS is rolled every 15 minute, so when the time index t moves to $t := t+1$, the realization of net load is revealed, *i.e.*, the actual measurement of electrical load and solar power are available at time $t+1$, we can calculate the actual $P_{\text{net}}(t+1)$ that is obtained from applying the first step of battery action as
$$P_{\text{net}}(t+1) = \text{net load}_{\text{forecast}}(t+1) + P_{\text{chg,ems}}(t+1) - P_{\text{dchg,ems}}(t+1). \tag{C.1}$$
The battery dynamic is also updated according to the first step of battery operation:
$$\text{SoC}(t+2) = \text{SoC}(t+1) + \frac{100\%}{\text{BattCapacity}} \left(\eta_c P_{\text{chg,ems}}(t+1) - \frac{P_{\text{dchg,ems}}(t+1)}{\eta_d} \right) \Delta t \tag{C.2}$$

Repeating this process over the specified period yields profiles of P_{net} and SoC, enabling the calculation of the minimized cumulative net electricity cost over the horizon presented in Section 7.2.

Table C.1: EMS problem parameters and electricity tariffs.

Parameter	Value	Unit	Parameter	Value	Unit
Battery			**Tariff**		
Charging efficiency	0.95	-	Buy rate (22:00-10:00)	2.7	THB
Discharging efficiency	0.88	-	Buy rate (10:00-14:00)	5.7	THB
Max charge rate	5	kW	Buy rate (14:00-18:00)	7	THB
Max discharge rate	5	kW	Buy rate (18:00-22:00)	8	THB
Minimum SoC	20	%	Sell rate (23:00-18:00)	2.2	THB
Maximum SoC	80	%	Sell rate (18:00-23:00)	2.5	THB

The details of system parameters are described in Table C.1. Electricity tariffs are based on Thailand's pricing rate, incorporating a future assumption of higher evening buy rates due to increased electrification demand. The sell rate during on-peak period is slightly higher than the off-peak period.

Appendix D

LIST OF PUBLICATIONS

Two methodologies in this thesis were presented in two publications. The first methodology, which includes three convex formulations in Section 3.1, along with experimental results based on synthetic and real-world data on Chapter 5, was presented in

W. Amnuaypongsa, W. Wangdee, and J. Songsiri, "Probabilistic Solar Power Forecasting Using Multi-Objective Quantile Regression," *2024 18th International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, 2024, pp. 26-31, doi: `https://doi.org/10.1109/PMAPS61648.2024.10667174`.

The second methodology, which includes the Sum-k formulation in Section 3.2, along with experimental results based on synthetic and real-world data in Chapter 6 and also system application Chapter 7, was presented in

W. Amnuaypongsa, W. Wangdee, and J. Songsiri, Neural Network-Based Prediction Interval Estimation with Large Width Penalization for Renewable Energy Forecasting and System Applications; doi: `https://doi.org/10.1016/j.ecmx.2025.101119` (Accepted for publication in Energy Conversion and Management: X (EM: X) journal).

Additionally, the methodology for estimating solar panel efficiency under curtailment has also been presented as an alternative approach for converting solar irradiance to solar power, as shown in

W. Amnuaypongsa, Y. Suparanonrat, N. Tongamrak, and J. Songsiri, "Estimation of Solar Panel Efficiency in the Presence of Curtailment,"*The 22nd International Conference on Electrical Engineering/Electronics, Computer, Telecommunication, and Information Technology (ECTI-CON 2025)*, 2025, pp. 1-6.

Biography

Worachit Amnuaypongsa was born and raised in Samut Prakan. He completed his primary and secondary education at Assumption Samutprakan School. In 2022, he earned a bachelor's degree in Electrical Engineering with first-class honors from Chulalongkorn University, Thailand. In 2023, he began pursuing a master's degree in Electrical Engineering at Chulalongkorn University, affiliated with the Data Analytics, Optimization, and Control Research Laboratory. His master's studies are fully supported by the CUEE High-Performance Scholarship. His research interests include optimization, statistical learning, machine learning, deep learning, and uncertainty quantification.