

Timing is Important: Risk-aware Fund Allocation based on Time-Series Forecasting

Fuyuan Lyu
McGill University &
MILA - Quebec AI Institute
Montreal, Canada
fuyuan.lyu@mail.mcgill.ca

Lin Feng Du
McGill University
Montreal, Canada
linfeng.du@mail.mcgill.ca

Yunpeng Weng
FiT, Tencent
Shenzhen, China
edwinweng@tencent.com

Qiufang Ying
FiT, Tencent
Shenzhen, China

Zhiyan Xu
FiT, Tencent
Shenzhen, China

Wen Zou
FiT, Tencent
Shenzhen, China

Haolun Wu
McGill University &
MILA - Quebec AI Institute
Montreal, Canada
haolun.wu@mail.mcgill.ca

Xiuqiang He*
College of Big Data and Internet
Shenzhen Technology University
Shenzhen, China
he.xiuqiang@gmail.com

Xing Tang
College of Big Data and Internet
Shenzhen Technology University
Shenzhen, China
xing.tang@hotmail.com

Abstract

Fund allocation has been an increasingly important problem in the financial domain. In reality, we aim to allocate the funds to buy certain assets within a certain future period. Naive solutions such as prediction-only or Predict-then-Optimize approaches suffer from goal mismatch. Additionally, the introduction of the SOTA time series forecasting model inevitably introduces additional uncertainty in the predicted result. To solve both problems mentioned above, we introduce a Risk-aware Time-Series Predict-and-Allocate (RTS-PnO) framework, which holds no prior assumption on the forecasting models. Such a framework contains three features: (i) end-to-end training with objective alignment measurement, (ii) adaptive forecasting uncertainty calibration, and (iii) agnostic towards forecasting models. The evaluation of RTS-PnO is conducted over both online and offline experiments. For offline experiments, eight datasets from three categories of financial applications are used: Currency, Stock, and Cryptos. RTS-PnO consistently outperforms other competitive baselines. The online experiment is conducted on the Cross-Border Payment business at FiT, Tencent, and an 8.4% decrease in regret is witnessed when compared with the product-line approach. The code for the offline experiment is available here¹.

CCS Concepts

• **Information systems** → **Data Mining**; *Decision Support System*.

*Corresponding Author

¹<https://github.com/fuyuanlyu/RTS-PnO>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD '25, Toronto, ON, Canada

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-1454-2/2025/08

<https://doi.org/10.1145/3711896.3737268>

Keywords

Predict-then-Optimize; Time-series Forecasting; Conformal Prediction; Fund Allocation

ACM Reference Format:

Fuyuan Lyu, Lin Feng Du, Yunpeng Weng, Qiufang Ying, Zhiyan Xu, Wen Zou, Haolun Wu, Xiuqiang He, and Xing Tang. 2025. Timing is Important: Risk-aware Fund Allocation based on Time-Series Forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25)*, August 3–7, 2025, Toronto, ON, Canada. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3711896.3737268>

1 Introduction

Fund Allocation has been an increasingly important problem in financial technology. Proper fund allocation can reduce the cost of financial operations. Previous studies allocate funds across different assets, such as stocks and currencies [13]. However, one commonly overlooked aspect is that the price of assets tends to vary rapidly over time, making the timing of acquiring assets equally important. This paper aims to investigate the fund allocation throughout the time dimension. Specifically, the goal is to acquire a certain amount of assets at minimal cost over a period of time.

One naive solution for asset allocation is to adopt SOTA time-series (TS) forecasting models [21, 36] directly to predict the price of the target asset over a period of time and heuristically select the lowest point, given that the price of assets is sequentially ordered as a time series. However, one fundamental problem of such a solution lies in the external constraints. Financial markets usually have auxiliary regulations such as risk management or internal controls. Hence, the lowest point is not necessarily feasible in certain cases.

To deliver a feasible and accountable action, a Predict-then-Optimize (PtO) framework is commonly introduced [8, 30, 33]. The PtO framework intuitively decomposes the task into two sequential steps: predict and optimize. After obtaining the forecasting result under the supervision of future values, it subsequently utilizes off-the-shelf commercial solvers, such as Gurobi [18] or COPT [10], to

obtain solutions with given constraints. Such a design is intuitively based on the hypothesis that a higher prediction accuracy (measured by prediction metrics such as *MSE*) would result in better decision quality (measured by decision metrics such as *Regret*) with theoretical support [3]. The effectiveness of the PtO paradigm has been successfully demonstrated in previous industrial applications, such as courier allocation [33].

However, adopting the PtO paradigm has certain challenges that have not been resolved. First, both empirical [12] and theoretical [7, 16] gaps are witnessed between prediction objectives and business decision goals. The time-series forecasting models aim to predict future prices across all time stamps accurately. In other words, the value of all time stamps contributes equally to the training loss, e.g., *MSE*. In contrast, the optimization process tends to care more about extreme cases, such as the minimal or maximal values. By solving them in two subsequent steps, the PtO paradigm could eventually lead to suboptimal decisions. Additionally, the frequencies of financial data tend to be higher (minute-level), eventually making them harder to predict [4]. Second, such a paradigm overlooks the uncertainty of forecasting. The consequences of a bad financial decision could lead to huge losses for both the company and the customer. Hence, proposing an uncertainty measurement approach that correlates directly with the forecasted result is important. However, these methods tend to suffer from Previous research tends to rely on probabilistic models forecasting both the result and its uncertainty [15, 23]. However, these methods suffer from cumulative errors as they decode each time step recursively. This drawback makes them unsuitable for real-world tasks requiring a long forecasting period. Therefore, an uncertainty quantification measure that is model-agnostic and can forecast long periods directly is desired.

To solve the two problems mentioned above in a unified approach, we propose a risk-aware time-series forecasting Predict-and-Allocate (RTS-PnO) framework to solve the fund allocation problem in the time domain. The RTS-PnO framework features three things: (i) end-to-end trainable with objective alignment, (ii) adaptive uncertainty measurement, and (iii) agnostic towards forecasting models. To alleviate the objective mismatch between prediction and optimization, RTS-PnO adopts the recently proposed Predict-and-Optimize (PnO) paradigm [7], also known as decision-focused learning (DFL). The PnO paradigm directly trains the forecasting model with surrogate losses approximating the feedback from the optimization stage. Though a decrease in prediction accuracy is witnessed under certain cases, an increase in the decision quality can be witnessed [12]. Additionally, inspired by the success of conformal prediction and its extension in the time series domain [24, 34, 35], we propose to measure the forecasting uncertainty adaptively during the training loop. Such a design can iteratively calibrate the uncertainty condition in the surrogate problem, yielding a better decision. Finally, all the designs of RTS-PnO do not make any prior assumptions about the architecture of forecasting models. Therefore, it is easy to update the forecasting models with advanced ones in the future.

Our method is evaluated in both offline and online experiments. For offline experiments, we select eight public benchmarks from three categories of financial applications. RTS-PnO consistently

yields better performances than other baselines. We deploy RTS-PnO on Tencent’s financial platform for the online experiment to support cross-border payment scenarios for WeChat Pay. An average regret reduction of 8.4% is witnessed over 12 time slots compared with the product-line baseline. To sum up, our contributions can be summarized as follows:

- We first study the fund allocation over the time domain, where the prices of assets vary over the time dimension.
- To align the training objective of time-series forecasting models and business criteria, we propose two model-agnostic frameworks, named RTS-PtO and RTS-PnO. RTS-PnO adopts an end-to-end training paradigm driven by the final objective and adaptively calibrates the uncertainty constraint during the process. RTS-PtO adopts a two-stage solution by training a prediction model and then solving the optimization with the fixed uncertainty constraint.
- Extensive evaluation is conducted on both offline benchmarks and online scenarios. Both results prove the effectiveness of the proposed frameworks.

2 Related Work

2.1 Time Series Forecasting

Modern architectures for time series forecasting aim to extend the forecasting horizon and improve long-term accuracy. Inspired by the success of Transformer-based models in capturing long-range dependencies, researchers have explored various adaptations of the Transformer architecture for this task. These include i) reducing computational complexity to sub-quadratic levels using sparse [39] and hierarchical [17] attention, ii) extending the attention mechanism’s point-wise dependency modeling to capture segment-wise [14] and patch-wise dependencies [21, 37], and iii) modifying the attention mechanism to incorporate domain-specific processing techniques [32, 40]. Besides Transformer-based models, modern temporal convolutional networks have also been shown to achieve competitive performance. MICN [29] combines local and global convolutions to better model long sequences, while TimesNet [31] reshapes the 1D series into 2D matrices based on salient periodicities to jointly model intra-period and inter-period variations. In fact, with the recent rise of linear models [36] and MLPs [6], the de facto neural architecture for this task remains undecided. In this work, we demonstrate the wide compatibility of RTS-PtO and RTS-PnO across various model architectures.

One drawback of the above-mentioned methods is the lack of uncertainty quantification. Existing approaches resort to generative modeling [15, 23], which naturally captures data variation. However, these approaches are often limited to short-term prediction, as modeling the joint data probability becomes exponentially difficult. Alternatively, we leverage the conformal prediction framework to characterize uncertainty for longer series [24, 34, 35], which we show empirically can help achieve satisfactory performance across different datasets.

2.2 From PtO To PnO

The predict-then-optimize (PtO) can be viewed as an abstractive problem for many real-world applications, such as portfolio management or power scheduling, requiring both predicting unknown

values and optimizing the target given these unknown values [3, 5]. Such a paradigm has been recently extended to other large-scale applications, such as carrier allocation [33], fund recommendation [27]. However, it is believed that a misalignment of targets exists between prediction and optimization stages. Researchers are increasingly interested in training the prediction model directly targeting the optimization goal, commonly known as predict-and-optimize (PnO) [12, 16, 28] or decision-focused learning [19]. The core challenge is to obtain meaningful gradients for model updating, given the optimization stage. Certain researchers adopt analytical approaches and aim to make the optimization layer differentiable [1, 2]. However, these works tend to rely on strong requirements on the objective functions or constraints, restricting their application scopes in reality. Other researchers [7, 20] instead adopt surrogate loss for the optimization layer and prove its convergence both theoretically and empirically. Our RTS-PnO first extends the application of the predict-and-optimization paradigm to large-scale industrial problems.

3 Methodology

In this section, we introduce the problem formulation of the fund allocation over time in Section 3.1. Section 3.2 details the two-stage solution RTS-PtO based on predict-then-optimize. Section 3.3 details the end-to-end approach RTS-PnO.

3.1 Problem Formulation

In this section, we propose the formal formulation of fund allocation over the time dimension. Without loss of generality, we focus on univariate series. The formulation for multi-variate time series can be derived easily. Suppose the unit price of a certain asset at time step t is defined as p_t , then the unit price of that asset can formulate a time series, denoted as:

$$\underbrace{p_1, p_2, \dots, p_{t-1}, p_t}_{\text{known}} \mid \underbrace{p_{t+1}, p_{t+2}, \dots}_{\text{unknown}}$$

The goal of fund allocation over time is to acquire a certain amount of asset in H future time steps $[p_{t+1}, p_{t+2}, \dots, p_{t+H}]$, at the lowest cost. This can be formulated as:

$$\min \mathbf{a} \times [p_{t+1}, p_{t+2}, \dots, p_{t+H}].$$

Here $\mathbf{a} \in \mathcal{A}$ denotes the allocation results, and $\mathcal{A} \subseteq [0, 1]^H$ represents the feasible allocation space. Again, we can assume the unit amplitude assumption on \mathbf{a} , denoted as $\sum \mathbf{a} = 1$. Therefore, the final objective can be formulated as:

$$\begin{aligned} \min \mathbf{a} \times [p_{t+1}, p_{t+2}, \dots, p_{t+H}]. \\ \text{s.t. } \sum \mathbf{a} = 1, \mathbf{a} \in \mathcal{A}, \mathcal{A} \subseteq [0, 1]^H. \end{aligned} \quad (1)$$

Although the future price $[p_{t+1}, p_{t+2}, \dots, p_{t+H}]$ has ground truth values, these values are unknown by the time t when we make the allocation. Therefore, we need to forecast the future price and denote the predicted result as $[\hat{p}_{t+1}, \hat{p}_{t+2}, \dots, \hat{p}_{t+H}]$. Hence, we propose two solutions to solve the above problem. First is a Predict-then-Optimize (PtO) framework, which treats both \mathbf{a} and $[\hat{p}_{t+1}, \hat{p}_{t+2}, \dots, \hat{p}_{t+H}]$ as independent variables. It makes the prediction first, then optimizes a given the predicted result. Second is a Predict-and-Optimize (PnO) framework, which treats \mathbf{a} as a function

of $[\hat{p}_{t+1}, \hat{p}_{t+2}, \dots, \hat{p}_{t+H}]$ and conducts prediction and optimization simultaneously.

3.2 RTS-PtO: A Two-Stage Solution with Uncertainty Constraints

In this section, we first introduce the Risk-aware Predict-then-Optimize (PtO) framework, which is commonly adopted by similar problems [33]. The PtO solution naturally consists of two steps: (i) predicting the future price of the asset given the historical records and contextual information, and (ii) obtaining the allocation result based on the predicted price. The first step, like other forecasting problems, aims at accurately predicting the price in the future. The second step, on the other hand, can be viewed as solving an optimization problem targeting minimal cost reduction under constraints. Additionally, we propose an additional uncertainty constraint on the forecasted results to avoid over-aggressive decisions.

3.2.1 Prediction Stage. During the first forecasting stage, we aim to forecast the future H steps $[p_{T+1}, \dots, p_{T+H}]$. For simplicity, we denote this target H -step series as y_T . The forecasting model takes the previous M steps $[p_{T-M+1}, \dots, p_{T-1}, p_T]$ as input, simplified as x_T . On certain tasks, additional content information, denoted as c_T , may also be provided. The forecasting model $M(\cdot)$ then predicts the target as follows:

$$\begin{aligned} \hat{y}_T &= M(x_T, c_T), \\ \hat{y}_T &\triangleq [\hat{p}_{T+1}, \dots, \hat{p}_{T+H}], \\ x_T &\triangleq [p_{T-M+1}, \dots, p_{T-1}, p_T], \end{aligned} \quad (2)$$

where \hat{y}_T denotes the predicted result. The training objective of the forecasting model is to reduce the distance between the forecasted value \hat{y}_T and the ground truth y_T . A certain prediction loss \mathcal{L}_p is adopted to measure such distances. Hence, the training objective of the forecasting model can be denoted as follows:

$$\mathcal{L}_p = \frac{1}{|\mathcal{D}|} \min_{M(\cdot)} \sum_{(x_T, y_T, c_T) \in \mathcal{D}} \ell_p(y_T, \hat{y}_T). \quad (3)$$

Note that the Mean-Square-Error (MSE) Loss is widely adopted as the prediction loss $\ell_p(\cdot)$ on each data instance [21, 36].

3.2.2 Optimization Stage. After obtaining the prediction result \hat{y}_T from the well-trained forecasting model $M(\cdot)$, Equation (1) can be viewed as an optimization problem via replacing the parameters y_T with the prediction result \hat{y}_T . Hence, Equation (1) is derived into:

$$\begin{aligned} \min \mathbf{a} \cdot \hat{y}_T \\ \text{s.t. } \sum \mathbf{a} = 1, \mathbf{a} \in \mathcal{A}, \mathcal{A} \subseteq [0, 1]^H. \end{aligned} \quad (4)$$

The above equation can be observed as optimization \mathbf{a} while treating the forecasted result \hat{y}_T as ground truth values. Hence, the accuracy of \hat{y}_T becomes important for the quality of the final decision. However, it is recognized that accurately forecasting time series is not an easy task [4]. To reduce the side effects of inaccurate prediction, we additionally propose forecasting uncertainty constraints that adaptively adjust themselves to constraints on the feasible position of allocation. Suppose a risk measurement can be obtained on each allocation position. As we will elaborate in

Section 3.2.3, the risk vector \mathbf{r} can be represented as:

$$\mathbf{r} \in \mathbb{R}_{\geq 0}^H, \mathbb{R}_{\geq 0} = \{x \in \mathbb{R} \mid x \geq 0\} \quad (5)$$

Hence, we define a new risk-aware feasible space $\mathcal{A}'(\mathbf{r})$ as follows:

$$\mathcal{A}'(\mathbf{r}) = \{\mathbf{a} \cdot \mathbf{r} \leq r_0 \mid \mathbf{a} \in \mathcal{A}\} \quad (6)$$

Here, r_0 is a pre-defined scalar representing the risk tolerance level. A smaller r_0 would lead to a tighter constraint on the forecasting uncertainty and a stronger preference towards forecasting results with high confidence. It is easy to observe that $\mathcal{A}'(\mathbf{r}) \subseteq \mathcal{A}$. Correspondingly, the objective of Equation (4) becomes to solve the following task:

$$\begin{aligned} \mathbf{a}^*(\hat{y}_T) &= \arg \min \mathbf{a}(\hat{y}_T) \cdot \hat{y}_T \\ \text{s.t. } \sum \mathbf{a}(\hat{y}_T) &= 1, \mathbf{a}(\hat{y}_T) \in \mathcal{A}'(\mathbf{r}), \mathcal{A}'(\mathbf{r}) \subseteq [0, 1]^H. \end{aligned} \quad (7)$$

Here $\mathbf{a}^*(\hat{y}_T)$ refers to the optimal allocation under the prediction \hat{y}_T . It is also known as the prescriptive decision [5]. Once the future asset price y_T is known, the optimal allocation results in $\mathbf{a}^*(y_T)$ can be readily obtained by solving Equation (1) as a continuous optimization problem. $\mathbf{a}^*(y_T)$ is also known as the full-information optimal decision [5].

After obtaining both the optimal allocation $\mathbf{a}^*(y_T)$ and prescriptive decision $\mathbf{a}^*(\hat{y}_T)$, we can use the *regret* metric, defined as the cost gap between these two allocation plans, to evaluate the quality of the decision. This can be written as:

$$\text{regret} \triangleq |\mathbf{a}^*(y_T) \cdot y_T - \mathbf{a}^*(\hat{y}_T) \cdot y_T|. \quad (8)$$

A lower regret indicates the predicted allocation $\mathbf{a}^*(\hat{y}_T)$ is closer to the optimal allocation $\mathbf{a}^*(y_T)$, indicating a lower operation cost and a better decision quality.

3.2.3 Uncertainty Quantify via Conformal Prediction. In this section, we focus on how to quantify the uncertainty of an arbitrary forecasting model and, more importantly, how to utilize such uncertainty to guide the training of our framework. Motivated by previous work [24], we adopt conformal prediction to measure the positional uncertainty of time series forecasting. The pseudo-code for positional uncertainty calculation is shown in Algorithm 1.

Algorithm 1 Calculating Positional Uncertainty for Backbone

Input: Calibration Dataset \mathcal{D}_c , coverage rate γ

Output: Positional Uncertainty \mathbf{r}

- 1: Initialize Positional Uncertainty Sets $\epsilon_1 = \{\}, \dots, \epsilon_H = \{\}$
 - 2: **for** for data instance (x_T, y_T, c_T) in Calibration Set \mathcal{D}_c **do**
 - 3: Calculate $\hat{y}_T = [\hat{p}_{T+1}, \dots, \hat{p}_{T+H}]$ given Eq. 2
 - 4: **for** h in $1, \dots, H$ **do**
 - 5: $\epsilon_h \leftarrow \epsilon_h \cup \{|\hat{p}_{T+h} - p_{T+h}|\}$
 - 6: **for** h in $1, \dots, H$ **do**
 - 7: $r_h = \left(\frac{|\mathcal{D}_c|+1}{|\mathcal{D}_c|}\gamma\right)$ - quantile in ϵ_h
 - 8: **Return** $\mathbf{r} = [r_1, r_2, \dots, r_H]$
-

3.2.4 Overall Training Process of RTS-PtO. The pseudo-code for the training of the RTS-PtO framework is shown in Algorithm 2.

Algorithm 2 The Training Process of RTS-PtO Framework

Input: Dataset \mathcal{D} , risk tolerance r_0 , epoch number T

Output: A allocation function $\mathbf{a}^*(\hat{y}_T)$ produce allocation with forecasting result \hat{y}_T

- 1: **for** Epoch $t = 1, \dots, T$ **do**
 - 2: Update the forecasting model M given Eq. 14
 - 3: Obtain the positional uncertainty \mathbf{r} for epoch t given Alg. 1
 - 4: Obtain the allocation feasible space $\mathcal{A}(\mathbf{r})$ given Eq. 5
 - 5: Obtain the prescriptive decision $\mathbf{a}^*(\hat{y}_T)$ given Eq. 7
-

3.3 RTS-PnO: An End-to-end Solution with Adaptive Uncertainty Constraints

In this section, we propose our model-agnostic framework, RTS-PnO. As stated in Section 1, the model differs from the previous two-stage solution in two aspects: (i) an end-to-end training predict-and-optimize paradigm aiming at aligning both the training objective and business goal and (ii) an adaptive risk-aware constraint to mitigate the forecasting error of the prediction model. We will discuss them in the following paragraphs.

3.3.1 End-to-end training with PnO framework. Unlike the PtO framework, the predict-and-optimize (PnO) framework directly trains the model with feedback from the optimization stage. To align both the training process and optimization goal, a surrogate loss is proposed to make the optimization process in the second stage differentiable, denoted as:

$$\mathcal{L}_o = \frac{1}{|\mathcal{D}|} \min_{\mathbf{M}(\cdot)} \sum_{(\hat{a}(\hat{y}_T), \mathbf{a}(y_T)) \in \mathcal{D}} \ell_o(\mathbf{a}^*(\hat{y}_T), \mathbf{a}(y_T)) \quad (9)$$

Then also exist several opinions for the surrogate loss $\mathcal{L}_o(\cdot)$ for the optimization stage. Here, we adopt the SPO+ loss [7], which is widely adopted in classic operation research problems and is verified to have outstanding performances [12, 19]. The SPO+ loss is optimized directly on the prescriptive decision $\mathbf{a}^*(\hat{y}_T)$, denoted as:

$$\ell_o(\mathbf{a}^*(\hat{y}_T)) \triangleq 2\mathbf{a}^*(y_T)\hat{y}_T - \mathbf{a}^*(y_T)y_T + \max_{\mathbf{a} \in \mathcal{A}} \{\mathbf{a}y_T - 2\mathbf{a}\hat{y}_T\}. \quad (10)$$

3.3.2 Adaptive Uncertainty Constraints. Although the SPO+ loss [7] theoretically aligns the training objective and business objective, empirical experiments suggest there is still a significant gap between these two [12]. Hence, it is deemed necessary to introduce the uncertainty constraint as a mitigation towards inaccurate prediction and objective mismatch.

As the risk vector \mathbf{r} reflects the forecasting uncertainty given the position, it is easy to notice that as the parameters within the forecasting model update, its forecasting uncertainty changes accordingly. The empirical studies in Figure 1 on four datasets show that the uncertainty gradually reduces during the training process. Therefore, adopting a fixed constraint like the PtO framework would yield an outdated uncertainty constraint and a suboptimal decision. To solve such a problem, we propose to add an adaptive risk constraint that updates the risk vector \mathbf{r}_t simultaneously after each epoch following Algorithm 1. Such a design would better reflect the current uncertainty of the forecasting model. Additionally, it becomes hard to fix a constant on the risk-tolerance threshold

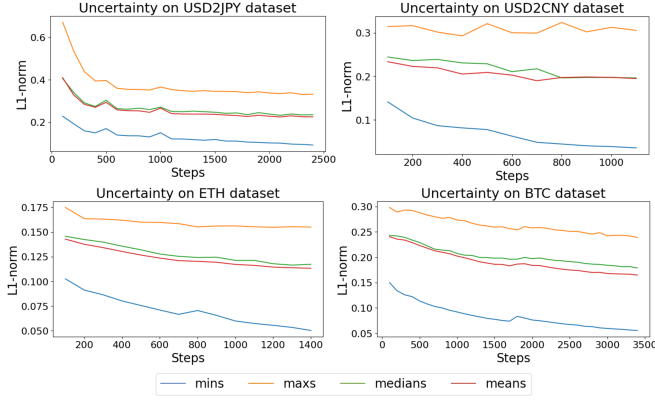


Figure 1: Empirical observation on the trending of uncertainty during the training process. Here, we report the minimum, maximum, mean, and median of the positional uncertainty.

r_0 . Therefore, we define r_0 as the α -quantile of the risk vector \mathbf{r} , denoted as:

$$r_0 = \alpha - \text{quantile in } \mathbf{r} \quad (11)$$

Hence, we define a new risk-aware feasible space $\mathcal{A}'(\mathbf{r}_t)$ as follows:

$$\mathcal{A}'(\mathbf{r}_t) = \{\mathbf{a} \cdot \mathbf{r}_t \leq \alpha - \text{quantile in } \mathbf{r} \mid \mathbf{a} \in \mathcal{A}\} \quad (12)$$

Correspondingly, Equation (10) derives into:

$$\ell_o(\mathbf{a}^*(\hat{y}_T), \mathbf{r}) \triangleq 2\mathbf{a}^*(y_T)\hat{y}_T - \mathbf{a}^*(y_T)y_T + \max_{\mathbf{a} \in \mathcal{A}'(\mathbf{r}_t)} \{\mathbf{a}y_T - 2\mathbf{a}\hat{y}_T\}. \quad (13)$$

The final training objective of RTS-PnO can be written as:

$$\min_{M(\cdot)} \mathcal{L}_o + \beta \cdot \mathcal{L}_p, \quad (14)$$

where the prediction loss \mathcal{L}_p is introduced as a regulator to balance the tradeoff between forecasting accuracy and decision quality, and β denotes its coefficient.

3.3.3 Overall Training Process of RTS-PnO. The pseudo-code for the training of the RTS-PnO framework is shown in Algorithm 3.

Algorithm 3 The Training Process of RTS-PnO Framework

Input: Dataset \mathcal{D} , uncertainty quantile α , loss balancer β , epoch number T

Output: A allocation function $\mathbf{a}^*(\hat{y}_T)$ produce allocation with forecasting result \hat{y}_T , a forecasting model $M(\cdot)$ outputs forecasting result \hat{y}_T

- 1: **for** Epoch $t = 1, \dots, T$ **do**
 - 2: Update the forecasting model M given Eq. 14
 - 3: Update the positional uncertainty \mathbf{r}_t for epoch t given Alg. 1
 - 4: Update the allocation feasible space $\mathcal{A}'(\mathbf{r}_t)$ given Eq. 5
-

4 Experiment

In this section, to comprehensively evaluate our proposed RTS-PnO, we design experiments to answer the following research questions:

- **RQ1:** Can RTS-PnO achieve superior performance in terms of decision quality compared with other baselines?

- **RQ2:** How does the forecasting model influence the overall performance in terms of decision quality?
- **RQ3:** How does the adaptive uncertainty design influence the performance in terms of decision quality?
- **RQ4:** How efficient is RTS-PnO compared to other methods?
- **RQ5:** How does the Predict-and-Optimize design influence the forecasting performance?

4.1 Experimental Setup

4.1.1 Time-Series Forecasting Models as Backbones. In the experiment, we adopt four SOTA time-series forecasting models as the backbone. PatchTST [21] is adopted as the default backbone without specification. DLinear [36], TimesNet [31] and FEDformer [40] are also included as backbones. Below, we include a brief explanation of each backbone involved during the experiments.

- PatchTST [21]: An encoder-only Transformer that operates on patches instead of individual time steps to quadratically reduce computational costs and model patch-wise dependencies.
- DLinear [36]: A fully linear model with a decomposer to separate seasonal and trend signals. The two signals are processed by separate linear transformations and added before output.
- TimesNet [31]: A modern temporal convolutional architecture that reshapes the 1D series into 2D matrices by salient periodicities in each model block to jointly capture intra-period and inter-period dependencies.
- FEDformer [40]: An encoder-decoder Transformer that computes attention in the Fourier space after filtering.

We noticed that some of the models (TimesNet [31] and FEDformer [40]) make use of timestamp information by incorporating special temporal embeddings, whereas other models choose not to. We remove the time stamp information from all models to ensure fair comparisons.

4.1.2 Baselines. In the following experiment, we adopt the following optimization baselines:

- **Forecasting-Only:** The forecasting-only method only makes the decision based on the forecasted result. Specifically, it follows a greedy approach by selecting the lowest k points in the future and evenly distributing the asset quota among these time steps. Here, we adopt Top-1 and Top-5 as baselines.
- **Risk-Avoiding:** The Risk-Avoiding method focuses on the worst-case scenarios. Specifically, it selects the k points with the lowest upper bound for the forecasting and evenly distributes the asset quota among these time steps. It adopts the uncertainty quantification approach in Algorithm 1 to calculate the upper bound. Here, we adopt Top-1 and Top-5 as baselines.

4.1.3 Benchmarks. We evaluate the performance on eight datasets: USD2CNY, USD2JPY, AUD2USD, NZD2USD, S&P 500, Dow Jones, BTC, ETC, from three financial scenarios: Currency, Stock, and Crypto. The statistics of these datasets are shown in Table 2. The processed dataset is available here². Below we further describe the details of each dataset.

- **Currency:** The Currency datasets, USD2CNY, USD2JPY, AUD2USD and NZD2USD, contain 10-minute currency between different currency pairs. The date of this dataset ranges from 2023/07/10

²<https://github.com/fuyuanlyu/RTS-PnO>

Table 1: Main Experiment with PatchTST as Forecasting Model

Category	Dataset	Forecasting-Only				Risk-Avoid				RTS-PtO		RTS-PnO		Relative Improvement	
		Top-1		Top-5		Top-1		Top-5		regret↓	R.R.↓	regret↓	R.R.↓	regret(%)	R.R.(%)
		regret↓	R.R.↓	regret↓	R.R.↓	regret↓	R.R.↓	regret↓	R.R.↓						
Currency	USD2CNY	36.88	5.10	37.00	5.12	35.80	4.95	35.83	4.96	<u>35.74</u>	<u>4.94</u>	31.68	4.38	12.82%	12.79%
	USD2JPY	54.50	34.92	54.21	34.73	<u>49.66</u>	<u>31.90</u>	50.01	32.12	52.11	32.66	48.77	31.25	1.82%	2.08%
	AUD2USD	19.56	29.60	19.92	30.15	<u>19.38</u>	<u>29.36</u>	19.49	29.52	19.48	29.51	19.06	28.84	1.68%	1.80%
	NZD2USD	17.43	28.75	17.66	29.14	<u>16.54</u>	<u>27.29</u>	16.64	27.44	16.82	27.75	15.68	25.85	5.48%	5.57%
Stock	S&P 500	134.99	4.25	135.47	4.24	122.50	3.84	124.24	<u>3.90</u>	126.06	3.94	124.05	<u>3.90</u>	-1.27%	-1.56%
	Dow Jones	1090.88	4.16	1075.79	4.09	<u>1022.73</u>	<u>3.91</u>	1032.21	3.93	1022.90	3.92	997.52	3.82	2.53%	2.36%
Cryptos	BTC	2159.78	4.46	2167.96	4.47	<u>1856.21</u>	<u>3.90</u>	1858.57	3.91	1924.65	3.96	1843.26	3.70	0.70%	5.41%
	ETH	151.14	5.56	149.61	5.48	<u>131.41</u>	4.68	131.42	4.68	138.60	4.96	131.40	<u>4.73</u>	0.00%	-1.07%
Avg. Rank		5.38	5.5	5.63	5.5	2	1.88	3.38	3.13	3.5	3.5	1.13	1.25		

Here **bold** font indicates the best-performed method and underline font indicates the second best-performed method. We also report the average rank of each method across all datasets. Notice that for the Currency category, all *regret* and *R.R.* omit the scaler $\times 10^{-4}$. For the rest category, all *R.R.* omit the scaler $\times 10^{-2}$.

to 2024/07/08. We removed the intervals where the market was closed. Here USD represents the US Dollar, CNY represents the Chinese Yuan, JPY represents the Japanese Yen, AUD represents the Australia Dollar, and NZD represents the New Zeland Dollar.

- Stock [22]: The Stock datasets, S&P 500 and Dow Jowes, contain daily prices from 1990/01/03 to 2024/2/16. It only contains workdays within this period. We use the Open market price as default.
- Cryptos [11]: The original Cryptos data is sampled at a minute frequency. We observed that the series remains relatively stable within each hour, and the first half of the data exhibits minimal variation. Therefore, we retain only the second half of the data and downsample it to an hourly frequency. We use the Open market price as default. Eventually, the ETH dataset contains hourly prices from 2020/07/18 to 2024/07/28, while the BTC dataset contains hourly prices from 2019/11/27 to 2024/07/29.

Table 2: Benchmark Statistics

Category	Dataset	#Times	Max	Min	Mean	Median
Currency	USD2CNY	22968	7.0905	7.3494	7.2267	7.2333
	USD2JPY		1.3740	1.6195	1.4949	1.4922
	USD2SGD		1.3159	1.3757	1.3484	1.3485
	AUD2USD		0.6274	0.6895	0.6559	0.6562
Stock	NZD2USD	8597	0.5774	0.6411	0.6066	0.6087
	S&P 500		295.46	5029.73	1596.80	1270.20
Cryptos	Dow Jones	40932	2365.10	38797.90	13663.80	10846.30
	BTC		4206.86	73705.36	32269.79	29352.15
	ETH	35301	233.72	4853.69	2126.29	1886.80

4.1.4 Metrics. To measure the decision quality of different methods, we adopt the commonly used metric *regret* defined in Equation (8) as the metric [7, 12, 19]. Considering the rapid changing of the asset price in certain datasets, such as BTC and ETH from the Cryptos domain, solely adopting *regret* would favor the model making a good decision in extreme cases. Hence, we additionally adopt the *relative regret*, abbreviated as *R-R*, which denotes the *regret* with the optimal value at that time step. This can be formulated as:

$$R.R. \triangleq \frac{\text{regret}}{\text{optimal cost}} = \frac{|a^*(y_T) \cdot y_T - a^*(\hat{y}_T) \cdot y_T|}{a^*(y_T) \cdot y_T}. \quad (15)$$

For both *regret* and *R.R.*, a lower value indicates the decision is closer to the optimal one, which is naturally a higher-quality decision.

Apart from the decision quality, we also need to measure the forecasting ability in RQ5. Here, we adopt *MSE* and *MAE* as forecasting metrics, following previous works in the time-series forecasting domain [21, 31, 36, 40]. For both *MSE* and *MAE*, a lower value indicates a higher forecasting accuracy.

4.1.5 Implementation Details. In this section, we provide the implementation details for all offline experiments. Our implementation is based on the PyTorch framework. Adam Optimizer is adopted for all setups. We select the learning ratio from {1e-3, 3e-4, 1e-4, 3e-5, 1e-5}. We adopt Gurobi [18] as the solver for optimization problems and borrow the implementation of SPO+ loss [7] from the PyEPO [26] library. All experiments in this section are run on an Nvidia RTX 4090D (24GB) GPU with 8 Intel (R) Xeon (R) Platinum 8481C and 40GB of memory.

4.2 Main Experiment (RQ1)

The overall performance of the proposed RTS-PnO and other baselines on eight benchmarks is reported in Table 1. We summarize our observation below.

First, our RTS-PnO proves to be effective compared with all other baselines in terms of both absolute regret and relative regret. In most datasets, the RTS-PnO method demonstrated improvements in terms of absolute regret (Abs-R) and relative regret (Rel-R). The RTS-PnO framework also achieved the best average ranking in both absolute and relative risks. However, the improvement brought by RTS-PnO differs on various datasets. For instance, on the USD2CNY dataset, RTS-PnO brings 12.82% and 12.79% improvement in terms of absolute and relative regrets. In contrast, on the S&P 500 dataset, RTS-PnO ranks second among all baselines.

Secondly, the RTS-PtO framework outperforms the forecasting-only approach in multiple datasets, especially in the stock market (e.g., S&P 500 and Dow Jones). This indicates that combining prediction and optimization in a two-stage process can improve decision quality.

Thirdly, the risk-avoiding strategy performed well across multiple datasets, particularly in the cryptocurrency and stock domains. For instance, on both S&P 500 and ETH datasets, risk-avoiding strategies rank first and show reductions in absolute and relative regrets, suggesting their effectiveness in highly volatile markets. Moreover, in all cases, the risk-avoiding strategy outperforms its

Table 3: Ablation Study on Forecasting Models

Forecasting Model	Dataset	Forecasting-Only				Risk-Avoiding				RTS-PtO		RTS-PnO		Relative Improvement	
		Top-1		Top-5		Top-1		Top-5		regret↓	R.R.↓	regret↓	R.R.↓	regret(%)	R.R.(%)
		regret↓	R.R.↓	regret↓	R.R.↓	regret↓	R.R.↓	regret↓	R.R.↓						
DLinear	USD2CNY	36.99	5.12	36.73	5.08	35.50	4.91	38.11	5.27	<u>35.31</u>	4.98	34.88	4.81	1.23%	3.50%
	Dow Jones	1103.11	4.21	1128.71	4.24	1036.65	3.96	1075.97	4.08	1073.30	4.10	<u>1042.35</u>	<u>3.98</u>	-0.55%	-0.51%
TimesNet	USD2CNY	39.77	5.50	39.46	5.46	36.83	5.09	37.47	5.18	<u>35.99</u>	<u>4.98</u>	33.73	4.66	6.70%	6.87%
	Dow Jones	1157.76	4.40	1143.82	4.32	<u>1037.71</u>	3.98	1082.45	4.11	1042.67	<u>3.95</u>	972.51	3.74	6.70%	5.61%
FEDFormer	USD2CNY	36.44	5.04	36.89	5.10	36.28	5.02	36.53	5.05	<u>35.94</u>	<u>4.97</u>	32.32	4.47	11.23%	11.19%
	Dow Jones	1087.49	4.15	1100.99	4.19	1065.08	4.05	1078.61	4.09	<u>1043.41</u>	<u>3.98</u>	1010.96	3.82	3.21%	4.19%

Here **bold** font indicates the best-performed method and underline font indicates the second best-performed method. Notice that for the USD2CNY dataset, all *regret* and *R.R.* omit the scaler $\times 10^{-4}$. For the Dow Jones dataset, all *R.R.* omit the scaler $\times 10^{-2}$.

forecasting-based versions, showing its adaptiveness in various cases.

Finally, the advantage of adopting a Top-k decision instead of a Top-1 decision varies in Forecasting-Only and Risk-Avoiding scenarios. In all cases, the Top-1 version outperforms the Top-5 version in the Risk-Avoiding scenario, indicating the conflict between the heuristic risk-avoiding strategy and the uncertainty qualification approach. However, in the Forecasting scenario, the Top-1 and Top-5 strategies vary on different datasets, showing their limitations in making decisions under rapidly changing markets.

4.3 Ablation Study on Forecasting Models (RQ2)

The RTS-PnO framework is proposed as a model-agnostic framework to adopt the rapidly changing time-series forecasting models as baselines seamlessly. This section showcases its compatibility with three different models: DLinear, TimesNet, and FEDformer, each representing one category for the time-series forecasting model. The experiment is conducted on two datasets: USD2CNY as a representative for currency data and Dow Jones as a representative for stock data. The result is shown in Table 3. Based on the results, we can make the following observations:

First, we observe consistent performance improvement like in the previous section. The RTS-PnO framework demonstrates significant performance improvements across various forecasting models, e.g., DLinear, TimesNet, and FEDFormer. For instance, in the USD2CNY dataset, the FEDFormer model achieved improvements of 11.23% and 11.19% in absolute regret (Abs-R) and relative regret (Rel-R), respectively, when using RTS-PnO. This indicates that the RTS-PnO framework is not dependent on a specific model.

Secondly, the selection of the forecasting model also influences the decision quality. For instance, TimesNet yields the best performance on the Dow Jones dataset. Meanwhile, PatchTST outperforms others on USD2CNY datasets. Such an observation further highlights the importance of a model-agnostic framework. The compatibility of such a framework greatly extends its application in various real-world applications.

Thirdly, we also observe that the RTS-PtO framework and risk-avoiding paradigm are competitive approaches across various forecasting models. For instance, Risk-Avoiding with Top-1 decision yields best on Dow Jones with the DLinear model, while the RTS-PtO framework ranks second in four out of six cases.

4.4 Ablation Study on Adaptive Uncertainty Constraint for RTS-PnO (RQ3)

To investigate the effect of the adaptive uncertainty constraint on the decision quality, we conduct the following ablation study. In this study, we replace the adaptive uncertainty constraint in Section 3.3.2 with the fixed uncertainty constraint in Section 3.2.3 and Equation (5). The evaluation is conducted over six datasets, shown in Table 4.

Table 4: Ablation on Uncertainty Constraint

Dataset	PtO		Fixed-PnO		Adaptive-PnO	
	regret↓	R.R.↓	regret↓	R.R.↓	regret↓	R.R.↓
USD2CNY	35.74	4.94	<u>34.66</u>	<u>4.71</u>	31.68	4.38
USD2JPY	52.11	32.66	<u>49.21</u>	<u>31.83</u>	48.77	31.25
S&P 500	<u>126.06</u>	<u>3.94</u>	129.13	4.02	124.05	3.90
Dow Jones	<u>1022.90</u>	<u>3.92</u>	1026.45	3.92	997.52	3.82
BTC	<u>1924.65</u>	<u>3.96</u>	1939.81	4.02	1843.26	3.70
ETH	138.60	4.96	<u>136.66</u>	<u>4.90</u>	131.40	4.73

Here **bold** font indicates the best-performed method and underline font indicates the second best-performed method. Notice that for the USD2CNY and USD2JPY dataset, all *regret* and *R.R.* omit the scaler $\times 10^{-4}$. For the rest datasets, all *R.R.* omit the scaler $\times 10^{-2}$.

We can make the following observations: First, we can easily observe that the PnO framework with adaptive uncertainty quantification constantly outperforms the fixed one. Secondly, it is easy to observe that PnO with fixed uncertainty is sometimes outperformed by its PtO version. Specifically, PtO ranks second on S&P 500, Dow Jones, and BTC datasets, while PnO with fixed uncertainty ranks second on the rest datasets. This is likely because the fixed uncertainty, reflecting the uncertainty of a well-trained forecasting model, misleads the PnO framework during the training phase, particularly during the initial stage. This yields an even worse performance than PtO. Both observations justify the necessity of calibrating the risk vector during the training process.

4.5 Ablation on Efficiency (RQ4)

In this section, we empirically evaluate the efficiency aspect of RTS-PnO, which is also deemed important in practical aspect [12]. Specifically, we evaluate both the training and inference efficiency. The experiment is conducted on five datasets: USD2CNY, USD2JPY, BTC, ETH, and S&P 500. We report the result in mean and variable formats for training and inference per epoch.

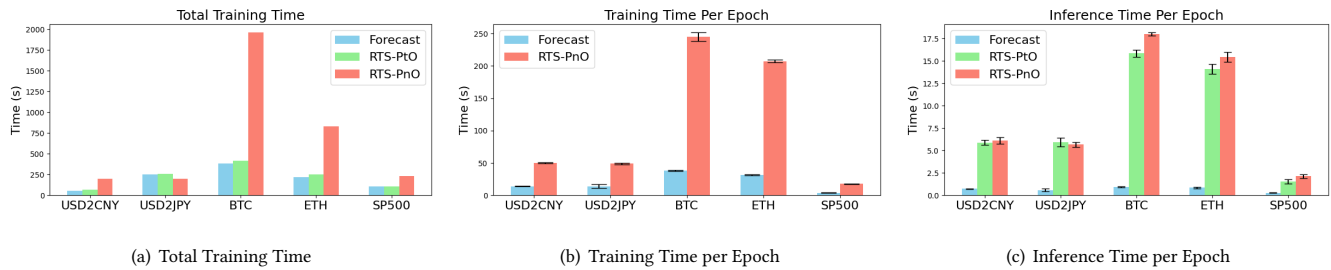


Figure 2: Efficiency Study on RTS-PtO and RTS-PnO.

For the training aspect, we report both total training time and training time per epoch, shown in Figures 2(a) and 2(b), respectively. We can observe that, compared with forecast methods, RTS-PnO does not require much additional training time. This suggests that the time required for the optimization process, compared with that for the training process, is rather neglectable. Additionally, when combining both Figures 2(a) and 2(b), we can observe that RTS-PnO requires much more training time per epoch, but the total training time is relatively at the same level compared to Forecast methods and RTS-PnO in certain cases. This suggests that RTS-PnO can converge to optimal with comparably fewer steps.

For the inference aspect, we can observe from Figure 2(c) that RTS-PnO and RTS-PtO require much more time than Forecast methods. As these two frameworks require solving the optimization process during the inference time, it suggests that the optimization process dramatically slows down the inference speed. This is quite different from the previous observation that the time required for the optimization phase can be neglected compared to training. Such an observation calls for quicker optimization approaches, such as model-based optimization [25].

4.6 Investigation in RTS-PnO from Forecasting Performance (RQ5)

In this section, we evaluate the forecasting performance of the forecasting model trained under the classic prediction paradigm and the proposed RTS-PnO framework. Note that the two-stage solution RTS-PtO has identical forecasting performance compared with the basic forecasting model, as it trains the same forecasting model during the prediction step, and the optimization step does not involve any update of the learned parameters. We report the performance on all three datasets. Both MSE and MAE are adopted to evaluate the forecasting performance, following the custom of time series forecasting [21, 31, 36, 40].

Based on the result in Table 5, we can easily witness a drop in both MAE and MSE after employing RTS-PnO. Such an observation aligns with previous works [12, 19] that the PnO framework improves the decision quality at the cost of prediction accuracy. It also reveals the misalignment between the decision and prediction tasks. However, given the problem setup, the decision tasks play a more important role in the model training.

4.6.1 Potential Amid for reduced forecasting performance. Our observation in the above section is not uncommon. It is commonly

Table 5: Experiment on Forecasting Metrics

Category	Dataset	Prediction		RTS-PnO	
		MSE	MAE	MSE	MAE
Currency	USD2CNY	0.0049	0.0397	0.0053	0.0430
	USD2JPY	0.0383	0.1263	0.1201	0.2796
	AUD2USD	0.0277	0.1220	0.0350	0.1439
	NZD2USD	0.0233	0.1072	0.0327	0.1334
Stock	S&P 500	0.1533	0.2744	0.5567	0.6194
	Dow Jones	0.1184	0.2354	0.3552	0.4815
Criptos	BTC	0.0197	0.0962	0.0953	0.2321
	ETH	0.0213	0.1003	0.1297	0.2608

Here **bold** font indicates the best-performed method and underline font indicates the second best-performed method.

observed that the PnO framework could lower the prediction performance compared to purely prediction modules [12, 19]. Previous researcher tends to focus on the decision quality. The prediction accuracy is also an important metric in evaluating the whole framework. One naive solution for solving the decreased prediction accuracy is to instead view prediction and optimization as a multi-task learning where the prediction and optimization module is the same as the PnO framework, and one additional calibration module is introduced to calibrate the prediction result. The prediction loss is applied to the calibrated result only, instead of the prediction result. General [38] and domain-specific [9, 41] MTL frameworks can be borrowed as potential solutions.

5 Online Experiment

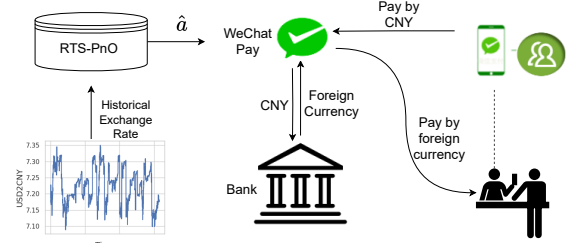


Figure 3: Application of RTS-PnO in the WeChat Pay's Cross-Border Payment Scenario.

We have deployed the proposed RTS-PnO at Tencent’s financial platform to support WeChat Pay’s cross-border payment scenarios. With the rapid increase in the number of Chinese residents traveling abroad, WeChat’s cross-border payment service has now extended to millions of overseas merchants, facilitating the consumption of Chinese tourists abroad.

As illustrated in Figure 3, when consumers make purchases at overseas merchants, the amount in Chinese Yuan (CNY) that consumers are required to pay is calculated based on the exchange rate at the time of the transaction and is deducted from the consumer’s account. Our platform then acquires the corresponding foreign currency amount for the user’s expenditure and settles it to the merchant’s bank account within a specified time window. To effectively manage funding costs, we employ our proposed RTS-PnO to allocate transaction funds within the designated time frame. For example, consider a consumer who makes a purchase of \$100 USD at an overseas merchant when the exchange rate is 6.5 CNY/USD. In this scenario, the consumer will pay 650 CNY. Our platform will then acquire the \$100 USD within the next 24-hour window and settle the amount to the merchant’s bank account. During this period, the exchange rate may fluctuate. To address this, we utilize the RTS-PnO framework to predict and determine the transaction volume to be exchanged each hour, thereby effectively managing funding costs. During the online experiment, no user data is involved. All involved data are obtained from third parties.

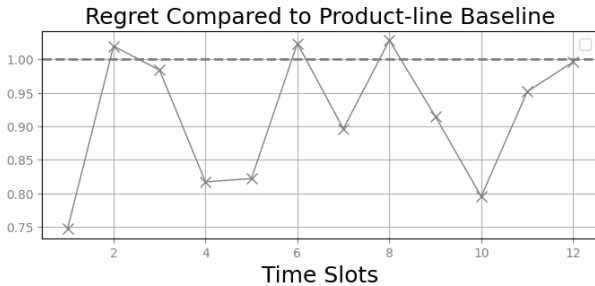


Figure 4: Online Result Compared with Product-line Baseline. The result is measured by the Regret compared with that of the Product-line Baseline. The dashed line represents the Product-line Baseline.

We conducted an online experiment throughout 12 time slots in 2024, during which 50% of the transactions were completed using the PTO framework, while the remaining 50% utilized the RTS-PnO framework. The results show that the regret by RTS-PnO is only 91.6% than that by product-line baseline, indicating that our proposed framework was able to reduce the average regret by 8.4%. The detailed performance over each time slot is shown in Figure 4. It is important to note that this experiment is conducted when acquiring the WeChat Pay cash reserve. It does not affect the amounts paid by the consumers and received by the merchants in any circumstances.

6 Conclusion

In this paper, we study the fund allocation problem extended over the time dimension. Specifically, the target is to acquire a certain

amount of asset within a period, and the price of the unit asset varies from time to time. We propose two solutions: (i) a two-stage solution RTS-PtO with fixed uncertainty constraint and (ii) an end-to-end solution RTS-PnO with adaptive uncertainty constraint. Both constraints are introduced to combine the hard-to-predict nature of time series and improve the decision quality. The evaluation is conducted over eight datasets from three categories of financial data. The proposed methods yield SOTA performance over the majority of the cases. Additionally, the online evaluation conducted over the cross-border payment scenarios of WeChat Pay demonstrates its feasibility in the real world.

Acknowledgement

This research project is partially funded by the support of the SZTU University Research Project (No.20251061020002). This work is done when Fuyuan, Xiuqiang and Xing are working as research intern and full-time employees at FiT, Tencent, respectively.

Ethical

In the online scenario involved in this paper, no user data is involved. The goal of the online task is to generate an allocation plan for buying currency at different time steps. During currency acquisition, all involved data are obtained from third parties (e.g., banks), and privacy regulations are satisfied. After acquiring the currency, the exchange rates for users are uniform and transparent. Neither process evolves any user’s privacy data.

Limitations

Though this paper empirically demonstrates the effectiveness of the proposed RTS-PnO in various benchmarks, this paper inevitably has certain limitations. First, the efficiency is a drawback of RTS-PnO. As illustrated in Figure 2(c), the inference time of RTS-PnO is larger than Forecasting-Only methods. Whether this inference increase is largely dependent on the scenarios. In our online setup, this is not a severe issue, as our online scenarios only require a 10-minute-level decision. However, in other scenarios, such as high-frequency trading, this inference increase might be an issue. Whether RTS-PnO satisfies the efficiency requirements is a domain-specific problem. The practitioners who aim to utilize RTS-PnO are advised to use their own judgment to conclude. Second, the current RTS-PnO is limited to single-variable time-series benchmarks. The evaluation is not conducted on multi-variable time-series benchmarks. However, we believe it can be extended to multi-variable time-series benchmarks.

References

- [1] Akshay Agrawal, Brandon Amos, Shane T. Barratt, Stephen P. Boyd, Steven Diamond, and J. Zico Kolter. 2019. Differentiable Convex Optimization Layers. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*. Curran Associates, Vancouver, BC, Canada, 9558–9570. <https://proceedings.neurips.cc/paper/2019/hash/9ce3c52fc54362e22053399d3181c638-Abstract.html>
- [2] Brandon Amos and J. Zico Kolter. 2017. OptNet: Differentiable Optimization as a Layer in Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017 (Proceedings of Machine Learning Research, Vol. 70)*. PMLR, Sydney, NSW, Australia, 136–145. <http://proceedings.mlr.press/v70/amos17a.html>
- [3] Othman El Balghiti, Adam N. Elmachtoub, Paul Grigas, and Ambuj Tewari. 2019. Generalization Bounds in the Predict-then-Optimize Framework. In *Advances*

- in *Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*. Curran Associates, Vancouver, BC, Canada, 14389–14398. <https://proceedings.neurips.cc/paper/2019/hash/a70145bf8b173e4496b554ce57969e24-Abstract.html>
- [4] Christoph Bergmeir. 2024. Fundamental limitations of foundational forecasting models: The need for multimodality and rigorous evaluation. <https://cbergmeir.com/talks/neurips2024/>
- [5] Dimitris Bertsimas and Nathan Kallus. 2020. From Predictive to Prescriptive Analytics. *Manag. Sci.* 66, 3 (2020), 1025–1044. doi:10.1287/MNSC.2018.3253
- [6] Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. TSMixer: Lightweight MLP-Mixer Model for Multivariate Time Series Forecasting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023*. ACM, Long Beach, CA, USA, 459–469. doi:10.1145/3580305.3599533
- [7] Adam N. Elmachtoub and Paul Grigas. 2022. Smart "Predict, then Optimize". *Manag. Sci.* 68, 1 (2022), 9–26. doi:10.1287/MNSC.2020.3922
- [8] Adam N. Elmachtoub, Jason Cheuk Nam Liang, and Ryan McNellis. 2020. Decision Trees for Decision-Making under the Predict-then-Optimize Framework. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020 (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, Virtual Event, 2858–2867. <http://proceedings.mlr.press/v119/elmachtoub20a.html>
- [9] Shanghua Gao, Teddy Koker, Owen Queen, Tom Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. 2024. UniTS: A Unified Multi-Task Time Series Model. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024*. Curran Associates, Vancouver, BC, Canada. http://papers.nips.cc/paper_files/paper/2024/hash/fe248e22b241ae5a9adf11493c8c12bc-Abstract-Conference.html
- [10] Dongdong Ge, Qi Huangfu, Zizhuo Wang, Jian Wu, and Yinyu Ye. 2022. Cardinal Optimizer (COPT) user guide. <https://guide.coap.online/copt/en-doc>
- [11] Patrick Gendotti. 2024. BTC and ETH 1-min Price History: Coinbase-provided high, low, and close prices, with trading volume. <https://www.kaggle.com/datasets/patrickgendotti/btc-and-eth-1min-price-history>
- [12] Haoyu Geng, Hang Ruan, Runzhong Wang, Yang Li, Yang Wang, Lei Chen, and Junchi Yan. 2024. Benchmarking PtO and PnO Methods in the Predictive Combinatorial Optimization Regime. In *NeurIPS 2024 Datasets and Benchmarks Track*. Curran Associates, Vancouver, British Columbia, Canada. <https://openreview.net/forum?id=cX57Pbw8vS>
- [13] Marcin Kacperczyk, Stijn Van Nieuwerburgh, and Laura Veldkamp. 2016. A rational theory of mutual funds' attention allocation. *Econometrica* 84, 2 (2016), 571–626.
- [14] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyu Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. 2019. Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*. Curran Associates, Vancouver, BC, Canada, 5244–5254. <https://proceedings.neurips.cc/paper/2019/hash/6775a0635c302542da2c32aa19d86be0-Abstract.html>
- [15] Yan Li, Xinjiang Lu, Yaqing Wang, and Dejing Dou. 2022. Generative Time Series Forecasting with Diffusion, Denoise, and Disentanglement. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*. Curran Associates, New Orleans, LA, USA. http://papers.nips.cc/paper_files/paper/2022/hash/91a85f3fb8f570e6be52b333b5ab017a-Abstract-Conference.html
- [16] Heyuan Liu and Paul Grigas. 2021. Risk Bounds and Calibration for a Smart Predict-then-Optimize Method. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021*. Curran Associates, virtual, 22083–22094. <https://proceedings.neurips.cc/paper/2021/hash/b943325cc7b7422d2871b345bf9b067f-Abstract.html>
- [17] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X. Liu, and Schahram Dustdar. 2022. Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting. In *The Tenth International Conference on Learning Representations, ICLR 2022*. OpenReview.net, Virtual Event. <https://openreview.net/forum?id=0EXmFzUn5I>
- [18] G. O. Gurobi. Llc. 2019. Gurobi Optimizer Reference Manual.
- [19] Jayanta Mandi, James Kotary, Senne Berden, Maxime Mulamba, Victor Bucarey, Tias Guns, and Ferdinando Fioretto. 2024. Decision-Focused Learning: Foundations, State of the Art, Benchmark and Future Opportunities. *J. Artif. Intell. Res.* 80 (2024), 1623–1701. doi:10.1613/JAIR.1.5320
- [20] Maxime Mulamba, Jayanta Mandi, Michelangelo Diligenti, Michele Lombardi, Victor Bucarey, and Tias Guns. 2021. Contrastive Losses and Solution Caching for Predict-and-Optimize. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021*. ijcai.org, Virtual Event / Montreal, Canada, 2833–2840. doi:10.24963/IJCAI.2021/390
- [21] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net, Kigali, Rwanda. <https://openreview.net/forum?id=Jbdc0vToc0l>
- [22] Shivesh Prakash. 2024. 34-year Daily Stock Data (1990-2024): Common stocks' daily closing values and trading volumes for quick ML projects. <https://www.kaggle.com/datasets/shiveshprakash/34-year-daily-stock-data>
- [23] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191. doi:10.1016/j.ijforecast.2019.07.001
- [24] Kamile Stankeviciute, Ahmed M. Alaa, and Mihaela van der Schaar. 2021. Conformal Time-series Forecasting. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021*. Curran Associates, Virtual Event, 6216–6228. <https://proceedings.neurips.cc/paper/2021/hash/312f1ba2a72318edaa995a67835fad5-Abstract.html>
- [25] Shiliang Sun, Zehui Cao, Han Zhu, and Jing Zhao. 2020. A Survey of Optimization Methods From a Machine Learning Perspective. *IEEE Trans. Cybern.* 50, 8 (2020), 3668–3681. doi:10.1109/TCYB.2019.2950779
- [26] Bo Tang and Elias B. Khalil. 2024. PyEPO: a PyTorch-based end-to-end predict-then-optimize library for linear and integer programming. *Math. Program. Comput.* 16, 3 (2024), 297–335. doi:10.1007/s12532-024-00255-X
- [27] Xing Tang, Yunpeng Weng, Fuyuan Lyu, Dugang Liu, and Xiuqiang He. 2025. A Predict-Then-Optimize Customer Allocation Framework for Online Fund Recommendation. *arXiv preprint arXiv:2503.03165* (2025).
- [28] Toon Vanderschueren, Tim Verdonck, Bart Baesens, and Wouter Verbeke. 2022. Predict-then-optimize or predict-and-optimize? An empirical evaluation of cost-sensitive learning strategies. *Inf. Sci.* 594 (2022), 400–415. doi:10.1016/j.ins.2022.02.021
- [29] Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. 2023. MICN: Multi-scale Local and Global Context Modeling for Long-term Series Forecasting. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net, Kigali, Rwanda. <https://openreview.net/forum?id=zt53IDUR1U>
- [30] Kai Wang, Sanket Shah, Haipeng Chen, Andrew Perrault, Finale Doshi-Velez, and Milind Tambe. 2021. Learning MDPs from Features: Predict-Then-Optimize for Sequential Decision Making by Reinforcement Learning. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021*. Curran Associates, virtual, 8795–8806. <https://proceedings.neurips.cc/paper/2021/hash/49e863b146f3b5470ee222ee84669b1c-Abstract.html>
- [31] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net, Kigali, Rwanda. https://openreview.net/forum?id=ju_Uqw384Oq
- [32] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021*. Curran Associates, virtual, 22419–22430. <https://proceedings.neurips.cc/paper/2021/hash/bcc0d400288793e8bdcd7c19a8ac0c2b-Abstract.html>
- [33] Kaiwen Xia, Li Lin, Shuai Wang, Haotian Wang, Desheng Zhang, and Tian He. 2023. A Predict-Then-Optimize Couriers Allocation Framework for Emergency Last-mile Logistics. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023*. ACM, Long Beach, CA, USA, 5237–5248. doi:10.1145/3580305.3599766
- [34] Chen Xu and Yao Xie. 2021. Conformal prediction interval for dynamic time-series. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021 (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, Virtual Event, 11559–11569. <http://proceedings.mlr.press/v139/xu21h.html>
- [35] Chen Xu and Yao Xie. 2023. Conformal Prediction for Time Series. *IEEE Trans. Pattern Anal. Mach. Intell.* 45, 10 (2023), 11575–11587. doi:10.1109/TPAMI.2023.3272339
- [36] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are Transformers Effective for Time Series Forecasting?. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023*. AAAI Press, Washington, DC, USA, 11121–11128. doi:10.1609/AAAI.V37I9.26317
- [37] Yunhao Zhang and Junchi Yan. 2023. Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net, Kigali, Rwanda. <https://openreview.net/forum?id=vSvLM2j9eie>
- [38] Yu Zhang and Qiang Yang. 2022. A Survey on Multi-Task Learning. *IEEE Trans. Knowl. Data Eng.* 34, 12 (2022), 5586–5609. doi:10.1109/TKDE.2021.3070203
- [39] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*. AAAI Press, Virtual Event, 11106–11115. doi:10.1609/AAAI.V35I12.17325
- [40] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In *International Conference on Machine Learning, ICML 2022 (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, Baltimore, Maryland,

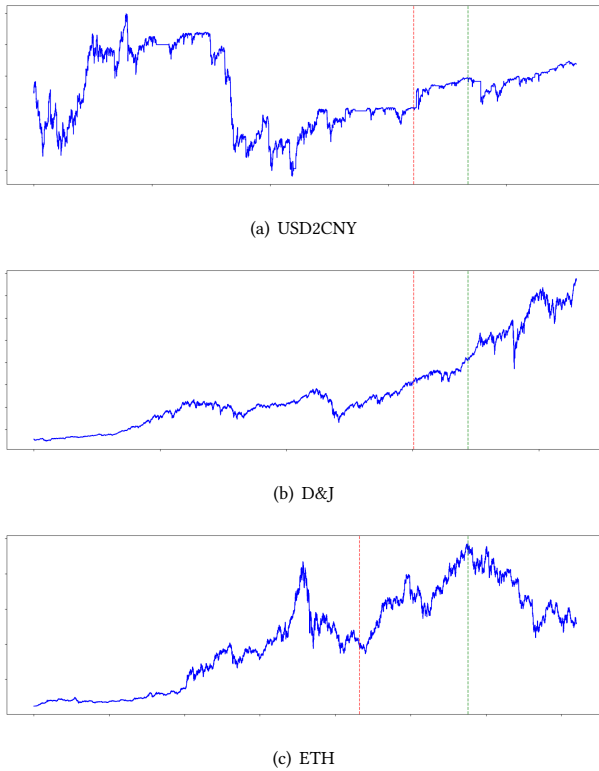


Figure 6: General Trend of Related Datasets. Red and Green lines indicate the margin between training VS validation and validation VS testing sets.

USA, 27268–27286. <https://proceedings.mlr.press/v162/zhou22g.html>
 [41] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. 2023. One Fits All: Power General Time Series Analysis by Pretrained LM. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*. Curran Associates, New Orleans, LA, USA. http://papers.nips.cc/paper_files/paper/2023/hash/86c17de05579cde52025f9984e6e2ebb-Abstract-Conference.html

A Ablation Study on the Hyper-parameter Sensitivity

In this section, we study the sensitivity of α (also r_0) over three benchmarks: USD2CNY, D&J and ETH in Figure 5. We also visualize the general trend of the involved datasets in Figure 6 to better illustrate how to select the optimal α on different benchmarks.

Given the observation in Figure 5, we can conclude that the optimal depends on the dataset. Generally speaking, a smaller α limits the position of feasible time slots. Currency data, such as USD2CNY, is changing relatively smoothly across time. This can be observed in Figure 6(a). It can be implied from Table 2 that the max and min of Currency data are of relatively the same magnitude. So a smaller α can reduce the amount of unreliable predictions being considered during optimization. In contrast, the Stock and Cryptos datasets, represented by D&J and ETH, may vary a lot across time and generally show a stronger trending behaviour, as shown in Figure 6(b) and 6(c) respectively. So a smaller α may eliminate all feasible choices during optimization in the extreme cases. Therefore, we need a larger α to loosen the constraint on prediction uncertainty.

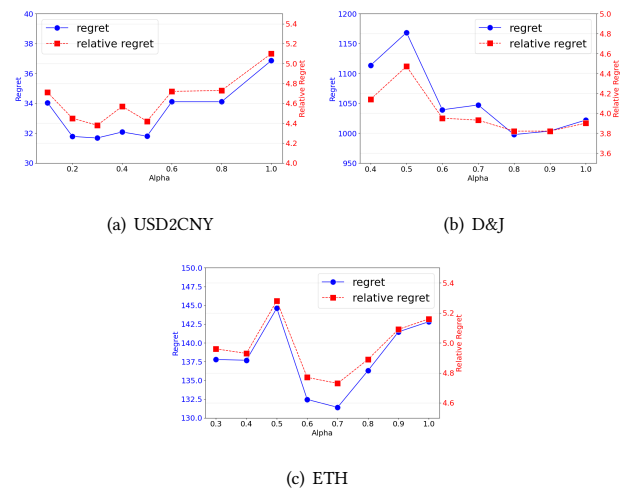


Figure 5: Sensitivity Study on α .