TIMING IS IMPORTANT: RISK-AWARE FUND ALLOCATION BASED ON TIME-SERIES FORECASTING

Fuyuan Lyu[1,2], **Linfeng Du**[1], **Yunpeng Weng**[3], **Qiufang Ying**[3], **Zhiyan Xu**[3], **Wen Zou**[3], **Haolun Wu**[1,2], **Xiuqiang He**[4], **Xing Tang**[3]
[1]McGill University, [2]MILA, [3]FiT, Tencent, [4]Shenzhen Technology University
fuyuan.lyu@mail.mcgill.ca, 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 **R**isk-aware **T**ime-**S**eries **P**redict-a**n**d-**A**llocate (RTS-PnO) framework, which holds no prior assumption on the forecasting models. Such a framework contains two 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 eight datasets from three categories of financial applications: Currency, Stock, and Cryptos. RTS-PnO consistently outperforms other competitive baselines. The code is available here[1].

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 (Kacperczyk et al. (2016)). 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 (Zeng et al. (2023); Nie et al. (2023)) 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 (Xia et al. (2023); Elmachtoub et al. (2020); Wang et al. (2021)). The PtO framework intuitively decomposes the task in 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 (Llc. (2019)) or COPT (Ge et al. (2022)), 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 (Balghiti et al. (2019)). The effectiveness of the PtO paradigm has been successfully demonstrated in previous industrial applications, such as courier allocation (Xia et al. (2023)).

However, adopting the PtO paradigm has certain challenges that have not been resolved. First, both empirical (Geng et al. (2024)) and theoretical (Elmachtoub & Grigas (2022); Liu & Grigas (2021))

[1]https://github.com/fuyuanlyu/RTS-PnO

gaps are witnessed between prediction objectives and business decision goals. The time-series forecasting models aim to accurately predict future prices across all time stamps. 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 (Bergmeir (2024)). 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 (Salinas et al. (2020); Li et al. (2022)). However, these methods suffer from cumulative errors as they decode each time step in a recursive manner. 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 unifiedly solve the two problems mentioned above, 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 objective mismatch between prediction and optimization, RTS-PnO adopts the recently proposed Predict-and-Optimize (PnO) paradigm (Elmachtoub & Grigas (2022)), 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 (Geng et al. (2024)). Additionally, inspired by the success of conformal prediction and its extension in the time series domain (Stankeviciute et al. (2021); Xu & Xie (2021; 2023)), 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 on eight public datasets originating from three categories of financial applications, where RTS-PnO consistently yields better performances than other baselines. 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 on 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 eight datasets, proving 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 (Zhou et al. (2021)) and hierarchical (Liu et al. (2022)) attention, ii) extending the attention mechanism's point-wise dependency modelling to capture segment-wise (Li et al. (2019)) and patch-wise dependencies (Nie et al. (2023); Zhang & Yan (2023)), and iii) modifying the attention mechanism to incorporate domain-specific processing techniques (Wu et al. (2021); Zhou et al. (2022)). Besides Transformer-based models, modern temporal convolutional networks have also been shown to achieve competitive performance. MICN (Wang et al. (2023)) combines local and global convolutions to better model long sequences, while TimesNet (Wu et al. (2023)) 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 (Zeng et al. (2023)) and MLPs (Ekambaram et al. (2023)), 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 (Salinas et al. (2020); Li et al. (2022)), which naturally captures data variation. However, these approaches are often limited to short-term prediction, as modelling the joint data probability becomes exponentially difficult. Alternatively, we leverage the conformal prediction framework to characterize uncertainty for longer series (Stankeviciute et al. (2021); Xu & Xie (2021; 2023)), 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 (Bertsimas & Kallus (2020); Balghiti et al. (2019)). Such a paradigm has been recently extended to other large-scale applications, such as carrier allocation (Xia et al. (2023)). 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) (Geng et al. (2024); Vanderschueren et al. (2022); Liu & Grigas (2021)) or decision-focused learning (Mandi et al. (2024)). 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 differentible (Amos & Kolter (2017); Agrawal et al. (2019)). However, these works tend to rely on strong requirements on the objective functions or constraints, restricting their application scopes in reality. Other researchers (Mulamba et al. (2021); Elmachtoub & Grigas (2022)) 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 seciton, we propose the formal formulation of fund allocation over 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 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:

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

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

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

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 conclusion can be formulated as:

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

Although the future price $[p_{t+1}, p_{t+2}, \cdots, p_{t+H}]$ 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 $[\hat{p}_{t+1}, \hat{p}_{t+2}, \cdots, \hat{p}_{t+H}]$. Hence, we propose two solutions to solve the above problem. First is a Predict-then-Optimize (PtO) framework, which treats $\mathbf{a}$ and $[\hat{p}_{t+1}, \hat{p}_{t+2}, \cdots, \hat{p}_{t+H}]$ as independent variables. It make the prediction first, then optimize $\mathbf{a}$ given the predicted result. Second is a Predict-and-Optimize (PnO) framework, which treats $\mathbf{a}$ as a function of $[\hat{p}_{t+1}, \hat{p}_{t+2}, \cdots, \hat{p}_{t+H}]$ and conduct 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 (Xia et al. (2023)). The PtO solution naturally consists of two steps: (i) predicting the future price of asset given the historicial records and contextual information and (ii) obtain the allocation result based on the predicted price. The first step, like other forecasting problems, aims at accurately predicting the prize 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 $[p_{T+1}, \cdots, p_{T+H}]$. For simiplicity, we denote this target H steps series as y_T. The forecasting model takes the previous M steps $[p_{T-M+1}, \cdots, p_{T-1}, p_T]$ 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:

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

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 ground truth y_T. Certain prediction loss $\mathcal{L}_p$ is adopted to measure such distances. Hence, the training objective of forecasting model can be denoted as follows:

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

Note that the Mean-Square-Error (MSE) Loss is widely adopted as the prediction loss $\ell_p(\cdot)$ on each data instance (Nie et al. (2023); Zeng et al. (2023)).

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:

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

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 final decision. However, it is recognized that accurately forecasting time-series is not an easy task (Bergmeir (2024)). To reduce the side-effort of inaccurate prediction, we additionally propose a forecasting uncertainty constraints that adaptively adjust itself to constraint 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:

$$\mathbf{r} \in \mathbb{R}^H_{\geq 0}, \ \mathbb{R}_{\geq 0} = \{x \in \mathbb{R} \mid x \geq 0\} \tag{5}$$

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

$$\mathscr{A}^{'}(\mathbf{r}) = \{\mathbf{a} \cdot \mathbf{r} \leq r_0 \mid \mathbf{a} \in \mathscr{A}\} \tag{6}$$

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 $\mathscr{A}^{'}(\mathbf{r}) \subseteq \mathscr{A}$. Correspondingly, the objective of Equation 4 becomes to solve the following task:

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

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 (Bertsimas & Kallus (2020)). Once the future asset price y_T is known, the optimal allocation result $\mathbf{a}^*(y_T)$ can be readily obtained by solution Equation 1 as an continuous optimization problem. $\mathbf{a}^*(y_T)$ is also known as the full-information optimal decision (Bertsimas & Kallus (2020)).

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

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

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 uncertainty of arbitrary forecasting model, and more importantly, how to unitlize such uncertainty to guide the training of our framework. Motivated by previous work (Stankeviciute et al. (2021)), we adopt the conformal prediction to measure the positional uncertainty of time series forecasting. The pesudo-code for positional uncertainty calculation is shown in Algorithm 1.

Algorithm 1 Calculating Positional Uncertainty for Forecasting Model

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

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.

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 on two aspects: (i) An end-to-end training predict-and-optimize paradigm aiming at aligning both the training objective and business goal, and (ii) Adaptive risk-aware constraint to mitigate the forecasting error of prediction model. We will detailedly discuss them in the following paragraphs.

Algorithm 2 The Training Process of RTS-PtO Framework

Require: Dataset $\mathcal{D}$, risk tolerance r_0, epoch number T
Ensure: A allocation function $\mathbf{a}^*(\hat{y}_T)$ produce allocation with forecasting result $\hat{y}_T$

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

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 the 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:

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

Then also exist several opinions for the surrogate loss $\mathcal{L}_o(\cdot)$ for the optimization stage. Here we adopt the SPO+ loss (Elmachtoub & Grigas (2022)), which is widely adopted in classic operation research problems and is verified to have outstanding performances (Geng et al. (2024); Mandi et al. (2024)). The SPO+ loss is optimized directly on the prescriptive decision $\mathbf{a}^*(\hat{y}_T)$, denoted as:

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

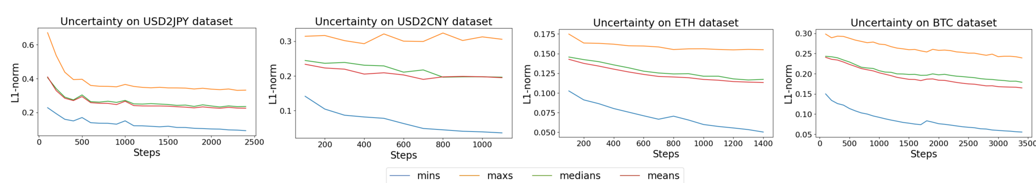


Figure 1: Empirical observation on the trending of uncertainty during the training process. Here, we report the minimum, maximum, mean, and median of the positional uncertainty.

3.3.2 Adaptive Uncertainty Constraints

Although the SPO+ loss (Elmachtoub & Grigas (2022)) theoretically aligns the training objective and business objective, empirical experiments suggest there is still a significant gap between these two (Geng et al. (2024)). Hence, it is deemed necessary to introduce the uncertainty constraint as 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 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 r_0. Therefore, we define r_0 as the α-quantile of the risk vector $\mathbf{r}$, denoted as:

$$r_0 = \alpha - \text{quantile in } \mathbf{r} \tag{11}$$

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

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

Correspondingly, Equation 10 derives into:

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

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

$$\min_{\mathbf{M}(\cdot)} \mathscr{L}_o + \beta \cdot \mathscr{L}_p, \tag{14}$$

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

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

Require: Dataset $\mathscr{D}$, uncertainty quantile α, loss balancer β, epoch number T
Ensure: 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, $\cdots$, 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 $\mathscr{A}(\mathbf{r}_t)$ 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 (Nie et al. (2023)) is adopted as the default backbone without specification. DLinear (Zeng et al. (2023)), TimesNet (Wu et al. (2023)) and FEDformer (Zhou et al. (2022)) are also included as backbones. We further introduce the setup of these backbones in Appendix A.1.

4.1.2 Baselines

In the following experiment, we adopt the following optimization baselines:

- Forecasting-Only: The forecasting-only method only make 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 of different approaches on eight datasets: USD2CNY, USD2JPY, AUD2USD, NZD2USD, S&P 500, Dow Jones, BTC and ETC, from three financial scenarios: Currency, Stock and Crypto. Details about these datasets are shown in Appendix A.2. The statistics of these datasets are shown in Table 4. The processed dataset is available here (Author (s)).

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 (Mandi et al. (2024); Geng et al. (2024); Elmachtoub & Grigas (2022)). Considering the rapid changing of the asset price in certain datasets, such as BTC and ETH from the Cryptos domain, solely adopting *regret* would favour 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:

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

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 (Nie et al. (2023); Wu et al. (2023); Zeng et al. (2023); Zhou et al. (2022)). For both *MSE* and *MAE*, a lower value indicates a higher forecasting accuracy.

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 observations below.

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	35.74	4.94	**31.68**	**4.38**	12.82%	12.79%
	USD2JPY	54.50	34.92	54.21	34.73	49.66	31.90	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	19.38	29.36	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	16.54	27.29	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	3.90	126.06	3.94	124.05	3.90	-1.27%	-1.56%
	Dow Jones	1090.88	4.16	1075.79	4.09	1022.73	3.91	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	1856.21	3.90	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	131.41	**4.68**	131.42	**4.68**	138.60	4.96	**131.40**	4.73	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 $\times 10^{-2}$.

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 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 2. Based on the results, we can make the following observations:

Table 2: 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	35.31	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	1042.35	3.98	-0.55%	-0.51%
TimesNet	USD2CNY	39.77	5.50	39.46	5.46	36.83	5.09	37.47	5.18	35.99	4.98	**33.73**	**4.66**	6.70%	6.87%
	Dow Jones	1157.76	4.40	1143.82	4.32	1037.71	3.98	1082.45	4.11	1042.67	3.95	**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	35.94	4.97	**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	1043.41	3.98	**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 $\times 10^{-2}$.

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 3.

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

Table 3: Ablation on Uncertainty Constraint

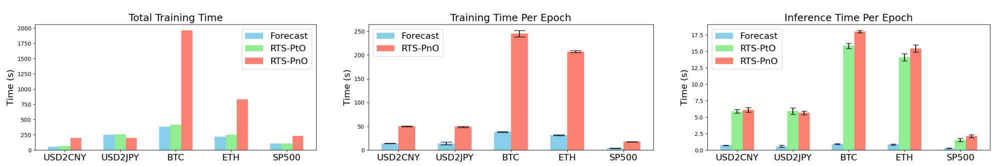
Dataset	USD2CNY		USD2JPY		S & P 500		Dow Jones		BTC		ETH	
	regret↓	R.R.↓	regret↓	R.R.↓	regret↓	R.R.↓	regret↓	R.R.↓	regret↓	R.R.↓	regret↓	R.R.↓
PtO	35.74	4.94	52.11	32.66	126.06	3.94	1022.90	3.92	1924.65	3.96	138.60	4.96
Fixed-PnO	34.66	4.71	49.21	31.83	129.13	4.02	1026.45	3.92	1939.81	4.02	136.66	4.90
Adaptive-PnO	**31.68**	**4.38**	**48.77**	**31.25**	**124.05**	**3.90**	**997.52**	**3.82**	**1843.26**	**3.70**	**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 $\times 10^{-2}$.

ranks second on the rest of the 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 the practical aspect (Geng et al. (2024)). 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.



(a) Total Training Time (b) Training Time per Epoch (c) Inference Time 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 (Sun et al. (2020)).

5 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.

REFERENCES

Akshay Agrawal, Brandon Amos, Shane T. Barratt, Stephen P. Boyd, Steven Diamond, and J. Zico Kolter. Differentiable convex optimization layers. In Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, pp. 9558–9570, Vancouver, BC, Canada, 2019. Curran Associates. URL https://proceedings.neurips.cc/paper/2019/hash/9ce3c52fc54362e22053399d3181c638-Abstract.html.

Brandon Amos and J. Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In Proceedings of the 34th International Conference on Machine Learning, ICML 2017, volume 70 of Proceedings of Machine Learning Research, pp. 136–145, Sydney, NSW, Australia, 2017. PMLR. URL http://proceedings.mlr.press/v70/amos17a.html.

Anonymous Author(s), 2025. URL https://anonymous.4open.science/r/RTS-PnO-53B0/.

Othman El Balghiti, Adam N. Elmachtoub, Paul Grigas, and Ambuj Tewari. 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, pp. 14389–14398, Vancouver, BC, Canada, 2019. Curran Associates. URL https://proceedings.neurips.cc/paper/2019/hash/a70145bf8b173e4496b554ce57969e24-Abstract.html.

Christoph Bergmeir. Fundamental limitations of foundational forecasting models: The need for multimodality and rigorous evaluation, 2024. URL https://cbergmeir.com/talks/neurips2024/.

Dimitris Bertsimas and Nathan Kallus. From predictive to prescriptive analytics. Manag. Sci., 66 (3):1025–1044, 2020. doi: 10.1287/MNSC.2018.3253.

Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 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, pp. 459–469, Long Beach, CA, USA, 2023. ACM. doi: 10.1145/3580305.3599533.

Adam N. Elmachtoub and Paul Grigas. Smart ”predict, then optimize”. Manag. Sci., 68(1):9–26, 2022. doi: 10.1287/MNSC.2020.3922. URL https://doi.org/10.1287/mnsc.2020.3922.

Adam N. Elmachtoub, Jason Cheuk Nam Liang, and Ryan McNellis. Decision trees for decision-making under the predict-then-optimize framework. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, volume 119 of Proceedings of Machine Learning Research, pp. 2858–2867, Virtual Event, 2020. PMLR. URL http://proceedings.mlr.press/v119/elmachtoub20a.html.

Dongdong Ge, Qi Huangfu, Zizhuo Wang, Jian Wu, and Yinyu Ye. Cardinal Optimizer (COPT) user guide. https://guide.coap.online/copt/en-doc, 2022.

Patrick Gendotti, 2024. URL https://www.kaggle.com/datasets/patrickgendotti/btc-and-eth-1min-price-history.

Haoyu Geng, Hang Ruan, Runzhong Wang, Yang Li, Yang Wang, Lei Chen, and Junchi Yan. Benchmarking pto and pno methods in the predictive combinatorial optimization regime. In NeurIPS 2024 Datasets and Benchmarks Track, Vancouver, British Columbia, Canada, 2024. Curran Associates. URL https://openreview.net/forum?id=cX57Pbw8vS.

Marcin Kacperczyk, Stijn Van Nieuwerburgh, and Laura Veldkamp. A rational theory of mutual funds’ attention allocation. Econometrica, 84(2):571–626, 2016.

Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. 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, pp. 5244–5254, Vancouver, BC, Canada, 2019. Curran Associates. URL https://proceedings.neurips.cc/paper/2019/hash/6775a0635c302542da2c32aa19d86be0-Abstract.html.

Yan Li, Xinjiang Lu, Yaqing Wang, and Dejing Dou. 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, New Orleans, LA, USA, 2022. Curran Associates. URL http://papers.nips.cc/paper_files/paper/2022/hash/91a85f3fb8f570e6be52b333b5ab017a-Abstract-Conference.html.

Heyuan Liu and Paul Grigas. 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, pp. 22083–22094, virtual, 2021. Curran Associates. URL https://proceedings.neurips.cc/paper/2021/hash/b943325cc7b7422d2871b345bf9b067f-Abstract.html.

Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X. Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, 2022. OpenReview.net. URL https://openreview.net/forum?id=0EXmFzUn5I.

G. O. Gurobi. Llc. Gurobi optimizer reference manual, 2019.

Jayanta Mandi, James Kotary, Senne Berden, Maxime Mulamba, Victor Bucarey, Tias Guns, and Ferdinando Fioretto. Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. J. Artif. Intell. Res., 80:1623–1701, 2024. doi: 10.1613/JAIR.1.15320.

Maxime Mulamba, Jayanta Mandi, Michelangelo Diligenti, Michele Lombardi, Victor Bucarey, and Tias Guns. Contrastive losses and solution caching for predict-and-optimize. In Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, pp. 2833–2840, Virtual Event / Montreal, Canada, 2021. ijcai.org. doi: 10.24963/IJCAI.2021/390.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, 2023. OpenReview.net. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Shivesh Prakash, 2024. URL https://www.kaggle.com/datasets/shiveshprakash/34-year-daily-stock-data.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. International Journal of Forecasting, 36(3):1181–1191, 2020. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2019.07.001.

Kamile Stankeviciute, Ahmed M. Alaa, and Mihaela van der Schaar. Conformal time-series forecasting. In Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, pp. 6216–6228, Virtual Event, 2021. Curran Associates. URL https://proceedings.neurips.cc/paper/2021/hash/312f1ba2a72318edaaa995a67835fad5-Abstract.html.

Shiliang Sun, Zehui Cao, Han Zhu, and Jing Zhao. A survey of optimization methods from a machine learning perspective. IEEE Trans. Cybern., 50(8):3668–3681, 2020. doi: 10.1109/TCYB.2019.2950779.

Bo Tang and Elias B. Khalil. Pyepo: a pytorch-based end-to-end predict-then-optimize library for linear and integer programming. Math. Program. Comput., 16(3):297–335, 2024. doi: 10.1007/S12532-024-00255-X.

Toon Vanderschueren, Tim Verdonck, Bart Baesens, and Wouter Verbeke. Predict-then-optimize or predict-and-optimize? an empirical evaluation of cost-sensitive learning strategies. Inf. Sci., 594: 400–415, 2022. doi: 10.1016/J.INS.2022.02.021.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. MICN: multi-scale local and global context modeling for long-term series forecasting. In The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, 2023. OpenReview.net. URL https://openreview.net/forum?id=zt53IDUR1U.

Kai Wang, Sanket Shah, Haipeng Chen, Andrew Perrault, Finale Doshi-Velez, and Milind Tambe. 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, pp. 8795–8806, virtual, 2021. Curran Associates. URL https://proceedings.neurips.cc/paper/2021/hash/49e863b146f3b5470ee222ee84669b1c-Abstract.html.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 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, pp. 22419–22430, virtual, 2021. Curran Associates. URL https://proceedings.neurips.cc/paper/2021/hash/bcc0d400288793e8bdcd7c19a8ac0c2b-Abstract.html.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, 2023. OpenReview.net. URL https://openreview.net/forum?id=ju_Uqw384Oq.

Kaiwen Xia, Li Lin, Shuai Wang, Haotian Wang, Desheng Zhang, and Tian He. 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, pp. 5237–5248, Long Beach, CA, USA, 2023. ACM. doi: 10.1145/3580305.3599766.

Chen Xu and Yao Xie. Conformal prediction interval for dynamic time-series. In Proceedings of the 38th International Conference on Machine Learning, ICML 2021, volume 139 of Proceedings of Machine Learning Research, pp. 11559–11569, Virtual Event, 2021. PMLR. URL http://proceedings.mlr.press/v139/xu21h.html.

Chen Xu and Yao Xie. Conformal prediction for time series. IEEE Trans. Pattern Anal. Mach. Intell., 45(10):11575–11587, 2023. doi: 10.1109/TPAMI.2023.3272339.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, pp. 11121–11128, Washington, DC, USA, 2023. AAAI Press. doi: 10.1609/AAAI.V37I9.26317.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, 2023. OpenReview.net. URL https://openreview.net/forum?id=vSVLM2j9eie.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, pp. 11106–11115, Virtual Event, 2021. AAAI Press. doi: 10.1609/AAAI.V35I12.17325.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In International Conference on Machine Learning, ICML 2022, volume 162 of Proceedings of Machine Learning Research, pp. 27268–27286, Baltimore, Maryland, USA, 2022. PMLR. URL https://proceedings.mlr.press/v162/zhou22g.html.

A Experimental Setup

A.1 BackBones

Here, we further detail the setup of these time series forecasting models serving as backbones in this paper. We noticed that some of the models (TimesNet and FEDformer) 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.

- PatchTST (Nie et al. (2023)): An encoder-only Transformer that operates on patches instead of individual time steps to quadratically reduce computational costs and model patch-wise dependencies.
- DLinear (Zeng et al. (2023)): 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 (Wu et al. (2023)): 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 (Zhou et al. (2022)): An encoder-decoder Transformer which computes attention in the Fourier space after filtering.

A.2 Datasets

Below we describe the details of each datasets.

- 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 to 2024/07/08. We removed the intervals where the market was closed.
- Stock (Prakash (2024)): 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 (Gendotti (2024)): 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 4: 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
	NZD2USD		0.5774	0.6411	0.6066	0.6087
Stock	S&P 500	8597	295.46	5029.73	1596.80	1270.20
	Dow Jones		2365.10	38797.90	13663.80	10846.30
Cryptos	BTC	40932	4206.86	73705.36	32269.79	29352.15
	ETH	35301	233.72	4853.69	2126.29	1886.80

A.3 Implementation Details

In this section, we provide the implementation details for all offline experiments. Our implementation (Author (s)) 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 (Llc. (2019)) as the solver for optimization problems and borrow the implementation of SPO+ loss (Elmachtoub & Grigas (2022)) from the PyEPO (Tang & Khalil (2024)) 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. Our experiments are available here (Author (s)).

B Ablation Study

B.1 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 (Zeng et al. (2023); Nie et al. (2023); Zhou et al. (2022); Wu et al. (2023)).

Table 5: Experiment on Forecasting Metrics

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

Here **bold** font indicates the best-performed method and <u>underline</u> font indicates the second best-performed method.

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 (Mandi et al. (2024); Geng et al. (2024)) that the PnO framework improves the decision quality at the cost of prediction accuracy. It also reveals the misalignment between decision and prediction tasks.