Differentiable Distributionally Robust Optimization Layers

Xutao Ma [1] **Chao Ning** [1 †] **Wenli Du** [2]

Abstract

In recent years, there has been a growing research interest in decision-focused learning, which embeds optimization problems as a layer in learning pipelines and demonstrates a superior performance than the prediction-focused approach. However, for distributionally robust optimization (DRO), a popular paradigm for decision-making under uncertainty, it is still unknown how to embed it as a layer, *i.e.*, how to differentiate decisions with respect to an ambiguity set. In this paper, we develop such differentiable DRO layers for generic mixed-integer DRO problems with parameterized second-order conic ambiguity sets and discuss its extension to Wasserstein ambiguity sets. To differentiate the mixed-integer decisions, we propose a novel dual-view methodology by handling continuous and discrete parts of decisions via different principles. Specifically, we construct a differentiable energy-based surrogate to implement the dual-view methodology and use importance sampling to estimate its gradient. We further prove that such a surrogate enjoys the asymptotic convergency under regularization. As an application of the proposed differentiable DRO layers, we develop a novel decision-focused learning pipeline for contextual distributionally robust decision-making tasks and compare it with the prediction-focused approach in experiments.

1. Introduction

In real-world scenarios, decision-making problems are typically affected by uncertainties. Therefore, machine learning techniques are usually leveraged to predict the behavior of the uncertainty, and then this prediction is passed to an optimization problem to derive decisions (Ning & You, 2019). Conventionally, the learning model is trained by minimizing a prediction loss, *i.e.*, in a prediction-focused way.

In recent years, decision-focused learning, also known as smart predict-and-optimize in operations research (Elmachtoub & Grigas, 2022), has received much research interest (Mandi et al., 2023; Sadana et al., 2023). Different from prediction-focused learning, decision-focused learning aims to train a learning model that minimizes a decision loss, *i.e.*, improving the decision quality. To implement decision-focused learning, differentiable optimization layers play the central role of passing gradient information from the decision back to the learning model, and this is achieved by differentiating the decision with respect to the learning target.

From the learning side, the learning target of most differentiable optimization layer research is a point prediction of uncertain quantity, and some research learns to predict the distribution of uncertainty. However, in prior research, the robustness of prediction is typically ignored, so the decision made based on this prediction is also in lack of robustness. As an emerging paradigm for robust decision-making, distributionally robust optimization (DRO) has seen a boom in both theory and applications in recent years (Delage & Ye, 2010; Wiesemann et al., 2014; Mohajerin Esfahani & Kuhn, 2018; Rahimian & Mehrotra, 2022). Therefore, to improve decision quality while preserving robustness, developing a differentiable DRO layer to learn the ambiguity set in a decision-focused way is highly desired but has not been investigated yet.

From the optimization side, most differentiable optimization layer research focuses on either pure continuous or pure discrete decisions. However, the decisions in practical problems are typically mixed-integer. Only Ferber et al. (2020) developed a mixed-integer linear program (MILP) layer. However, their approach relies on the specific solution structure of linear program (LP). Therefore, how to differentiate the mixed-integer decisions for generic mixed-integer convex optimization remains an unsolved problem.

To fill the aforementioned research gaps, this paper develops the first differentiable DRO layers with mixed-integer decisions. That is, the learning target is an ambiguity set

[†]Corresponding Author [1]Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China [2]The Key Laboratory of Smart Manufacturing in Energy Chemical Process, Ministry of Education, East China University of Science and Technology, Shanghai 200237, China. Correspondence to: Chao Ning <chao.ning@sjtu.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

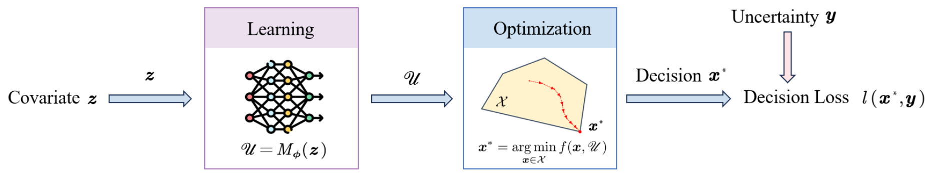
Covariate z
z
Learning
$\mathcal{U} = M_\phi(z)$
$\mathcal{U}$
Optimization
$\mathcal{X}$
x^*
$x^* = \arg\min_{x\in\mathcal{X}} f(x, \mathcal{U})$
Decision x^*
Uncertainty y
Decision Loss $l(x^*, y)$

Figure 1. Sequential learning and decision-making pipeline.

and the output decisions are mixed-integer. The ambiguity set we mainly focus on is the class of parameterized second-order conic (SOC) ambiguity set (Bertsimas et al., 2019), which is widely adopted in various applications (Zhou et al., 2019; Zhang et al., 2022; Yang et al., 2023), and Wasserstein ambiguity set is also discussed in Appendix C.

The major contributions of this paper are summarized as follows.

- We develop the first generic differentiable DRO layers, which enable integrating learning and distributionally robust decision-making via gradient descent.
- We propose a novel dual-view methodology to differentiate the mixed-integer decisions. We note that this methodology can be applied to develop any mixed-integer convex optimization layer, not limited to the proposed DRO layers.
- We construct a differentiable energy-based surrogate value function to implement the dual-view methodology and use importance sampling to estimate its gradient. In theory, we prove that such a surrogate enjoys the asymptotic convergency under regularization.
- As an application of the proposed differentiable DRO layers, we develop a novel decision-focused learning pipeline, which is of interest in its own right, for contextual distributionally robust decision-making tasks and compare it with the prediction-focused approach in experiments.

2. Related Literature

We first review existing work on differentiable optimization layers with pure continuous and pure discrete decisions.

Convex optimization layers. To differentiate continuous decisions of constrained optimization, the basic idea is to apply the implicit differentiation theorem to the optimality conditions. Following this idea, Amos & Kolter (2017) successfully differentiated through constrained quadratic programs. Differentiating through LP was achieved by adding regulation terms in Wilder et al. (2019) and Mandi & Guns (2020). For linear conic programming, the optimality condition was derived by leveraging the homogeneous self-dual embedding technique (Busseti et al., 2019), and then the implicit differentiation was applied (Agrawal et al., 2019b). Finally, Agrawal et al. (2019a) aggregated all these work and developed the differentiable convex optimization layer package *cvxpylayers*.

Combinatorial optimization layers. To handle the non-differentiability of discrete decisions, Berthet et al. (2020) developed differentiable surroagte solution by adding perturbation. Similar ideas also appeared in Niepert et al. (2021) and Pogančić et al. (2020). We refer to Dalle et al. (2022) for a review of this perturbation technique.

Aside from the differentiable optimization layer approach that manages to differentiate the decision, some research constructs a surrogate loss to circumvent difficulty.

Surrogate loss approach. The seminal work Elmachtoub & Grigas (2022) developed a surrogate SPO$^+$ loss. Shah et al. (2022) and Zharmagambetov et al. (2023) constructed a training-based surrogate loss. Kong et al. (2022) developed a surrogate loss for stochastic programming (SP) by using an energy-based model. For combinatorial optimization problems, Mulamba et al. (2020) and Mandi et al. (2022) constructed surrogate loss functions by maximizing the probability of the ground-truth optimal decision.

From the perspective of the learning target, most of the work mentioned above only considered point prediction, except for Donti et al. (2017) and Kong et al. (2022), which learned conditional distributions. Chenreddy et al. (2022) and Sun et al. (2023) investigated prediction-focused learning methods for uncertainty sets, and Wang et al. (2023) developed a learning method for robust optimization (RO) based on an augmented Lagrangian method. Very recently, Chenreddy & Delage (2024) developed an end-to-end learning method for robust optimization.

Perhaps the most relevant work to this paper is Costa & Iyengar (2023), which to the best of our knowledge is the only research on distributionally robust decision-focused learning. However, their framework presumes the uncertainty distribution to have a residual structure and only applies to specific financial problems with continuous decisions. On the contrary, our differentiable DRO layers apply to general distributions and a broad family of optimization problems with mixed-integer decisions.

3. Background

In this section, we provide some background information on the topic of this paper.

3.1. Decision-Making under Uncertainty

A typical sequential learning and decision-making pipeline is shown in Figure 1, where the decision-maker first leverages a learning model $M_{\boldsymbol{\phi}}$ to predict some information $\mathscr{U}$ concerning the uncertainty $\boldsymbol{y}$ from covariate $\boldsymbol{z}$. Such information $\mathscr{U}$ can be a point prediction, conditional distribution, uncertainty set, or ambiguity set of the uncertainty $\boldsymbol{y}$.

Subsequently, the decision-maker takes $\mathscr{U}$ as a parameter and solves a constrained optimization problem $\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x},\mathscr{U})$ to derive the decision $\boldsymbol{x}^*$. Depending on the form of $\mathscr{U}$, this constrained optimization problem can be deterministic optimization, SP, RO, or DRO.

Finally, after the decision is made, the uncertainty $\boldsymbol{y}$ is revealed and the decision loss $l(\boldsymbol{x}^*,\boldsymbol{y})$ is realized.

3.2. Decision-Focused Learning

In the conventional prediction-focused approach, the learning model is trained independently of the subsequent optimization process. On the contrary, in decision-focused learning, the learning model is trained by directly minimizing the decision loss, which can be formally expressed as the following bilevel problem.

$$\begin{aligned} \min_{\boldsymbol{\phi}\in\Phi} \quad & \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\, l\big(\boldsymbol{x}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),\boldsymbol{y}\big) \\ \text{s.t.} \quad & \boldsymbol{x}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})) = \arg\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x},\mathscr{U}=M_{\boldsymbol{\phi}}(\boldsymbol{z})) \end{aligned} \tag{1}$$

where $\boldsymbol{\phi}$ is the parameter of the learning model $M_{\boldsymbol{\phi}}$ we want to train, $\mathbb{P}$ is the joint distribution of covariate $\boldsymbol{z}$ and uncertainty $\boldsymbol{y}$, and here we assume (1) is well-defined, *i.e.*, the solution set of the argmin operator is a singleton.

To optimize this bilevel problem by gradient descent, it necessitates the computation of the following gradient.

$$\begin{aligned} &\frac{\partial l\big(\boldsymbol{x}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),\boldsymbol{y}\big)}{\partial\boldsymbol{\phi}} \\ =&\frac{\partial l\big(\boldsymbol{x}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),\boldsymbol{y}\big)}{\partial\boldsymbol{x}^*}\frac{\partial\boldsymbol{x}^*}{\partial M_{\boldsymbol{\phi}}(\boldsymbol{z})}\frac{\partial M_{\boldsymbol{\phi}}(\boldsymbol{z})}{\partial\boldsymbol{\phi}} \end{aligned} \tag{2}$$

where the first and last terms are easy to compute. However, the existence of argmin operator poses great difficulty in the computation of the middle term $\frac{\partial\boldsymbol{x}^*}{\partial\mathscr{U}}$, so the goal of a differentiable optimization layer is to compute this term.

In this paper, we aim to develop a differentiable DRO layer, *i.e.*, the learning target $\mathscr{U}$ is an ambiguity set and $\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x},\mathscr{U})$ is a DRO problem. Therefore, the goal is to develop a method to differentiate the mixed-integer decision $\boldsymbol{x}^*$ with respect to the ambiguity set $\mathscr{U}$, *i.e.*, computing $\frac{\partial\boldsymbol{x}^*}{\partial\mathscr{U}}$.

3.3. Distributionally Robust Optimization

The DRO takes an ambiguity set as the parameter and outputs a decision by optimizing the following problem.

$$\boldsymbol{x}^*(\mathscr{U}) = \arg\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x},\mathscr{U}) := \max_{\mathbb{P}\in\mathscr{U}} \mathbb{E}_{\boldsymbol{y}\sim\mathbb{P}}[c(\boldsymbol{x},\boldsymbol{y})] \tag{3}$$

where the cost function c is usually taken as the decision loss l and f is often referred to as 'worst-case expectation'.

4. Differentiable Distributionally Robust Optimization Layers

Since the space of all ambiguity sets is infinite-dimensional, directly learning in this space is generally computationally impossible. Therefore, we focus on the class of parameterized SOC ambiguity sets, which stem from the well-known SOC ambiguity set (Bertsimas et al., 2019).

To define the parameterized SOC ambiguity set, we first introduce the following differentiable parameterized second-order cone representable set, which is an extension of the conventional second-order cone representable set (see Appendix A.1).

Definition 4.1. A set $\mathscr{W}(\boldsymbol{\theta}) \subset \mathbb{R}^K$ is a differentiable parameterized second-order cone representable set with parameter $\boldsymbol{\theta}$ if there exists a collection of J second-order cone inequalities such that

$$\boldsymbol{y}\in\mathscr{W}(\boldsymbol{\theta}) \Leftrightarrow \exists\boldsymbol{v}: \boldsymbol{A}_j(\boldsymbol{\theta})\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{v}\end{bmatrix} - \boldsymbol{b}_j(\boldsymbol{\theta}) \geq_{L^{m_j}} \boldsymbol{0}, \forall j\in[J]$$

where L^{m_j} represents a m_j dimensional second-order cone and matrixes $\boldsymbol{A}_j(\boldsymbol{\theta})$ and vectors $\boldsymbol{b}_j(\boldsymbol{\theta})$ are differentiable functions of $\boldsymbol{\theta}$.

Now we define the parameterized SOC ambiguity set.

Definition 4.2. An ambiguity set $\mathscr{U}(\boldsymbol{\theta})$ is a parameterized SOC ambiguity set with parameter $\boldsymbol{\theta}$ if it can be expressed as follows.

$$\mathscr{U}(\boldsymbol{\theta}) = \left\{\mathbb{P}\;\middle|\; \begin{array}{c}\mathbb{P}(\Xi)=1\\ \mathbb{E}_{\mathbb{P}}[g_i(\boldsymbol{y},\boldsymbol{\alpha}_i)]\leq\sigma_i, \forall i\in[I]\end{array}\right\} \tag{4}$$

where $\mathbb{P}$ is a distribution of $\boldsymbol{y}$, $\boldsymbol{\theta}=(\boldsymbol{\alpha}_1,\sigma_1,\cdots,\boldsymbol{\alpha}_I,\sigma_I)$, support $\Xi\subset\mathbb{R}^K$ of the uncertainty is a second-order cone representable set, and the epigraph of each g_i,

$$\text{epi}\, g_i = \{(\boldsymbol{y},u)|u\geq g_i(\boldsymbol{y},\boldsymbol{\alpha}_i)\} \tag{5}$$

is a differentiable parameterized second-order cone representable set with parameter $\boldsymbol{\alpha}_i$

By selecting functions g_i, the parameterized SOC ambiguity set can characterize a variety of distributional features. We present some examples of the parameterized SOC ambiguity set in Appendix A.2 and see also Bertsimas et al. (2019).

For the cost function $c(\boldsymbol{x}, \boldsymbol{y})$ in Equation (3), we consider both the single- and two-stage cost functions.

Assumption 4.3. The cost function $c(\boldsymbol{x}, \boldsymbol{y})$ admits a single-stage formulation (i) without recourse or a two-stage formulation (ii) with relatively complete recourse as follows.

(i). Single-stage formulation: $c(\boldsymbol{x}, \boldsymbol{y}) = \sum_{k=1}^{K} c_k(\boldsymbol{x}) y_k$, where the epigraph of each function c_k is a second-order cone representable set and the uncertainty $\boldsymbol{y}$ is required to be non-negative.

(ii). Two-stage formulation: $c(\boldsymbol{x}, \boldsymbol{y})$ is the optimal value of an LP, *i.e.*,

$$c(\boldsymbol{x}, \boldsymbol{y}) = \min_{\boldsymbol{\gamma} \geq 0} \boldsymbol{q}^T \boldsymbol{\gamma} \text{ s.t. } \boldsymbol{T}(\boldsymbol{y})\boldsymbol{x} + \boldsymbol{W}\boldsymbol{\gamma} = h(\boldsymbol{y}) \tag{6}$$

where $\boldsymbol{T}(\boldsymbol{y}) = \boldsymbol{T}_0 + \sum_{k=1}^{K} \boldsymbol{T}_k y_k$, $h(\boldsymbol{y}) = \boldsymbol{h}_0 + \sum_{k=1}^{K} \boldsymbol{h}_k y_k$, and the constraint in (6) is feasible for all $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{y} \in \Xi$.

To derive a tractable reformulation of the DRO problem (3), we need the following regularity assumption on the parameterized SOC ambiguity set, and the detailed explanation of Assumption 4.4 is presented in Appendix A.3.

Assumption 4.4. Slater's condition holds for the parameterized SOC ambiguity set $\mathscr{U}(\boldsymbol{\theta})$.

Now we can state the following reformulation theorem.

Theorem 4.5. *Suppose $\mathscr{U}(\boldsymbol{\theta})$ is a parameterized SOC ambiguity set and Assumption 4.3 and Assumption 4.4 hold, then the worst-case expectation $f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta})) = \max_{\mathbb{P} \in \mathscr{U}(\boldsymbol{\theta})} \mathbb{E}_{\boldsymbol{y} \sim \mathbb{P}}[c(\boldsymbol{x}, \boldsymbol{y})]$ is a linear second-order cone program.*

Proof. See Appendix B.1 □

Since the parameterized SOC ambiguity set is determined by the finite-dimensional parameter $\boldsymbol{\theta}$, it suffices to use the learning model $M_{\boldsymbol{\phi}}(\boldsymbol{z})$ to learn the parameter $\boldsymbol{\theta}$, *i.e.*,

$$\boldsymbol{\theta} = M_{\boldsymbol{\phi}}(\boldsymbol{z}). \tag{7}$$

Therefore, the goal of the differentiable DRO layer comes down to computing $\frac{\partial \boldsymbol{x}^*}{\partial \boldsymbol{\theta}}$.

By Theorem 4.5, the worst-case expectation function f is a linear second-order cone programming. Therefore, if the decision $\boldsymbol{x}$ is continuous, the gradient $\frac{\partial \boldsymbol{x}^*}{\partial \boldsymbol{\theta}}$ can be directly computed by the technique of differentiating through a cone program (Agrawal et al., 2019b). We formally state this result in Theorem 4.6.

Theorem 4.6. *Suppose conditions in Theorem 4.5 hold and $\boldsymbol{x}$ is a continuous variable with $\mathcal{X}$ a second-order cone representable set, then $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta}))$ is differentiable with respect to $\boldsymbol{\theta}$.*

Proof. See Appendix B.2 □

However, when the decision $\boldsymbol{x}$ is mixed-integer, it is inherently non-differentiable due to the discreteness of integer variables. Therefore, it is necessary to develop a new methodology to handle mixed-integer decisions.

4.1. Dual-View of Mixed-Integer Decisions

Existing research on differentiable optimization layers all views the decision $\boldsymbol{x}^*$ as a function of the parameter and handles it by the principle of automatic differentiation. However, this view does not work for discrete decisions. Although some research manages to differentiate the discrete decisions by adding perturbations (Berthet et al., 2020), these approaches are still restricted to integer linear programming.

Alternatively, by examining the bilevel formulation (1) of the whole decision-focused learning task, we notice that the decision-making process is the lower-level problem. Therefore, the decision $\boldsymbol{x}^*$ can be viewed as a constraint, and we can handle it via the principle of constrained optimization.

By the above observations, we propose the following dual-view methodology to address mixed-integer decisions.

Dual-View Methodology:

I The continuous part of the decisions is viewed as a function of parameters and handled via the principle of automatic differentiation.

II The discrete part of the decisions is viewed as a constraint of the whole bilevel learning problem and handled via the principle of constrained optimization.

In this dual-view methodology, we have already established part I in Theorem 4.6. To better illustrate the idea of part II, we make the following assumptions and notations.

Assumption 4.7. $\boldsymbol{x} = (\boldsymbol{x}_d, \boldsymbol{x}_c)$ is a mixed-integer variable with discrete part $\boldsymbol{x}_d \in \{0,1\}^{n_1}$ and continuous part $\boldsymbol{x}_c \in \mathbb{R}^{n_2}$. The feasible region of $\boldsymbol{x}$ is $\mathcal{X} = \overline{\mathcal{X}} \cap (\{0,1\}^{n_1} \otimes \mathbb{R}^{n_2})$, where $\overline{\mathcal{X}}$ is a second-order cone representable set.

We denote by $\mathcal{X}_d$ the feasible region of the discrete part of variable $\boldsymbol{x}$, *i.e.*, $\mathcal{X}_d = \{\boldsymbol{x}_d \in \{0,1\}^{n_1} | \exists \boldsymbol{x}_c \in \mathbb{R}^{n_2} : (\boldsymbol{x}_d, \boldsymbol{x}_c) \in \mathcal{X}\}$, and by $\mathcal{X}_c(\boldsymbol{x}_d)$ the feasible region of the continuous part variable $\boldsymbol{x}_c$ given the integer part $\boldsymbol{x}_d \in \mathcal{X}_d$, *i.e.*, $\mathcal{X}_c(\boldsymbol{x}_d) = \{\boldsymbol{x}_c \in \mathbb{R}^{n_2} | (\boldsymbol{x}_d, \boldsymbol{x}_c) \in \mathcal{X}\}$.

The next assumption ensures that the bilevel problem (1) is well-defined, and see Appendix A.4 for a detailed discussion of this assumption.

Assumption 4.8. (i) For all $\boldsymbol{\phi} \in \Phi$, $\boldsymbol{z} \in \mathcal{Z}$, and $\boldsymbol{x}_d \in \mathcal{X}_d$, the optimal continuous solution

$$\boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z})) := \arg\min_{\boldsymbol{x}_c \in \mathcal{X}_c(\boldsymbol{x}_d)} f\big((\boldsymbol{x}_d, \boldsymbol{x}_c), M_{\boldsymbol{\phi}}(\boldsymbol{z})\big)$$

is unique.

(ii) For all $\boldsymbol{\phi} \in \Phi$, the optimal integer solution

$$\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})) := \arg\min_{\boldsymbol{x}_d \in \mathcal{X}_d} f\big((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z})), M_{\boldsymbol{\phi}}(\boldsymbol{z})\big)$$

is unique almost surely, *i.e.*,

$$\forall \boldsymbol{\phi} \in \Phi, \mathbb{P}_{\boldsymbol{z}}\big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})) \text{ is a singleton}\big) = 1$$

where $\mathbb{P}_{\boldsymbol{z}}$ is the marginal distribution of covariate $\boldsymbol{z}$.

By the above assumptions, the bilevel problem (1) can be reformulated as follows.

Corollary 4.9. *Suppose Assumption 4.7 and Assumption 4.8 hold, the decision-focused learning can be formulated as*

$$\begin{aligned} \min_{\boldsymbol{\phi}\in\Phi} \; & \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\, l\Big(\big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), \boldsymbol{x}_c^*(\boldsymbol{x}_d^*, M_{\boldsymbol{\phi}}(\boldsymbol{z}))\big), \boldsymbol{y}\Big) \\ \textit{s.t. } & \boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})) \\ & \quad = \arg\min_{\boldsymbol{x}_d \in \mathcal{X}_d} f\big((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}))), M_{\boldsymbol{\phi}}(\boldsymbol{z})\big) \end{aligned} \tag{8}$$

From the bilevel form (8) we can see more clearly the idea of dual-view methodology. The optimal continuous decision $\boldsymbol{x}_c^*$ is embedded as a function of the integer decision and learning target $M_{\boldsymbol{\phi}}(\boldsymbol{z})$. On the contrary, the optimal integer solution $\boldsymbol{x}_d^*$ is explicitly expressed as a constraint.

Let $\mathcal{R}(\boldsymbol{\phi})$ denote the value function of problem (8), *i.e.*,

$$\mathcal{R}(\boldsymbol{\phi}) := \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\, l\Big(\big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), \boldsymbol{x}_c^*(\boldsymbol{x}_d^*, M_{\boldsymbol{\phi}}(\boldsymbol{z}))\big), \boldsymbol{y}\Big)$$

Then problem (8) is equivalent to $\min_{\boldsymbol{\phi}\in\Phi} \mathcal{R}(\boldsymbol{\phi})$.

Following Part II in the dual-view methodology, we handle the constrained optimization (8) by approximating the value function sequentially. That is, we want to construct a sequence of differentiable surrogate value functions $\mathcal{R}_\lambda(\boldsymbol{\phi})$ such that $\mathcal{R}_\lambda(\boldsymbol{\phi}) \to \mathcal{R}(\boldsymbol{\phi})$ in some sense.

4.2. Energy-Based Surrogate Value Function

To construct such a surrogate function $\mathcal{R}_\lambda(\boldsymbol{\phi})$, we first construct point surrogate function $r_\lambda(M_{\boldsymbol{\phi}}(\boldsymbol{z}), \boldsymbol{y})$ for each point $(\boldsymbol{z}, \boldsymbol{y})$ and then define $\mathcal{R}_\lambda(\boldsymbol{\phi})$ as

$$\mathcal{R}_\lambda(\boldsymbol{\phi}) = \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\, r_\lambda(M_{\boldsymbol{\phi}}(\boldsymbol{z}), \boldsymbol{y}) \tag{9}$$

We construct the point surrogate function $r_\lambda(\boldsymbol{\phi}, \boldsymbol{z}, \boldsymbol{y})$ by leveraging the energy-based model. Specifically, we assign each feasible integer decision $\boldsymbol{x}_d \in \mathcal{X}_d$ the following energy function $E(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}), \lambda)$, where λ is a positive scalar.

$$\begin{aligned} E(\boldsymbol{x}_d, & M_{\boldsymbol{\phi}}(\boldsymbol{z}), \lambda) \\ &= \exp\left(-\frac{f((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z})), M_{\boldsymbol{\phi}}(\boldsymbol{z}))}{\lambda}\right) \end{aligned} \tag{10}$$

Based on the energy function, we can define a distribution $p(\boldsymbol{x}_d | M_{\boldsymbol{\phi}}(\boldsymbol{z}), \lambda)$ over $\mathcal{X}_d$.

$$p(\boldsymbol{x}_d | M_{\boldsymbol{\phi}}(\boldsymbol{z}), \lambda) = \frac{E(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}), \lambda)}{\sum_{\boldsymbol{x}_d' \in \mathcal{X}_d} E(\boldsymbol{x}_d', M_{\boldsymbol{\phi}}(\boldsymbol{z}), \lambda)} \tag{11}$$

We then construct point surrogate function $r_\lambda(\boldsymbol{\phi}, \boldsymbol{z}, \boldsymbol{y})$ as follows.

$$\begin{aligned} & r_\lambda(M_{\boldsymbol{\phi}}(\boldsymbol{z}), \boldsymbol{y}) \\ & := \mathbb{E}_{\boldsymbol{x}_d \sim p(\boldsymbol{x}_d | M_{\boldsymbol{\phi}}(\boldsymbol{z}), \lambda)} l((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}))), \boldsymbol{y}) \\ & = \sum_{\boldsymbol{x}_d \in \mathcal{X}_d} p(\boldsymbol{x}_d | M_{\boldsymbol{\phi}}(\boldsymbol{z}), \lambda) l((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}))), \boldsymbol{y}) \end{aligned} \tag{12}$$

By the above construction, we notice that the optimal integer solution $\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))$ has the largest energy, so the corresponding probability $p(\boldsymbol{x}_d^* | M_{\boldsymbol{\phi}}(\boldsymbol{z}), \lambda)$ is also the highest. When $\lambda \to 0^+$, this probability will converge to 1, and $r_\lambda(\boldsymbol{\phi}, \boldsymbol{z}, \boldsymbol{y})$ will also converge to the true decision loss $l\big((\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), \boldsymbol{x}_c^*(\boldsymbol{x}_d^*, M_{\boldsymbol{\phi}}(\boldsymbol{z}))), \boldsymbol{y}\big)$.

To further establish convergence results of the surrogate value function $\mathcal{R}_\lambda(\boldsymbol{\phi})$, we need the following continuity assumptions and the concept of epi-convergence.

Assumption 4.10. The decision loss $l((\boldsymbol{x}_d, \boldsymbol{x}_c), \boldsymbol{y})$ is bounded and continuous in $\boldsymbol{x}_c$. For any $\boldsymbol{z} \in \mathcal{Z}$, $f((\boldsymbol{x}_d, \boldsymbol{x}_c), M_{\boldsymbol{\phi}}(\boldsymbol{z}))$ is continuous in $\boldsymbol{x}_c$ and $\boldsymbol{\phi}$.

Assumption 4.11. For all $\boldsymbol{x}_d \in \mathcal{X}_d$ and $\boldsymbol{z} \in \mathcal{Z}$, the optimal continuous decision $\boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}))$ is continuous in $\boldsymbol{\phi}$.

We note that Assumption 4.10 is easily satisfied. For Assumption 4.11, since $\boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}))$ is differentiable with respect to $\boldsymbol{\theta} = M_{\boldsymbol{\phi}}(\boldsymbol{z})$ by Theorem 4.6, Assumption 4.11 simply requires the continuity of $M_{\boldsymbol{\phi}}$ in its parameter $\boldsymbol{\phi}$.

Definition 4.12 (Bonnans & Shapiro (2013), p.41)**.** A sequence of functions $\{\mathcal{R}_n(\boldsymbol{\phi})\}$ epi-converges to a function $\mathcal{R}(\boldsymbol{\phi})$ if and only if $\forall \boldsymbol{\phi} \in \Phi$, condition (i) and (ii) hold.

(i) For any sequence $\{\boldsymbol{\phi}_n\}$ converges to $\boldsymbol{\phi}$, $\liminf_{n\to\infty} \mathcal{R}_n(\boldsymbol{\phi}_n) \geq \mathcal{R}(\boldsymbol{\phi})$

(ii) There exists a sequence $\{\boldsymbol{\phi}_n\}$ converging to $\boldsymbol{\phi}$ such that $\limsup_{n\to\infty} \mathcal{R}_n(\boldsymbol{\phi}_n) \leq \mathcal{R}(\boldsymbol{\phi})$

The epi-convergence of $\mathcal{R}_\lambda(\boldsymbol{\phi})$ and asymptotic convergence of optimal solution are established in Theorem 4.13.

Theorem 4.13. *Suppose Assumption 4.7, Assumption 4.8, Assumption 4.10, and Assumption 4.11 hold, then for any sequence $\lambda_n \searrow 0^+$ as $n \to \infty$, the following two assertions hold for the energy-based surrogate value function $\mathcal{R}_\lambda(\boldsymbol{\phi})$.*

(i) $\mathcal{R}_{\lambda_n}(\boldsymbol{\phi})$ epi-converges to $\mathcal{R}(\boldsymbol{\phi})$ as $n \to \infty$.

(ii) if $\boldsymbol{\phi}_{\lambda_{n_k}} \in \arg\min_{\boldsymbol{\phi}\in\Phi} \mathcal{R}_{\lambda_{n_k}}(\boldsymbol{\phi})$ for some sub-sequence $\{n_k\} \subset \mathbb{N}$ and $\{\boldsymbol{\phi}_{\lambda_{n_k}}\}$ converges to a point $\boldsymbol{\phi}^*$, then

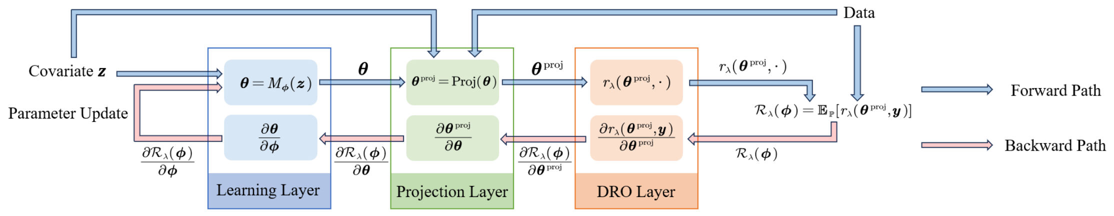
Data
Covariate $\boldsymbol{z}$
Parameter Update
$\boldsymbol{\theta} = M_\phi(\boldsymbol{z})$
$\frac{\partial \boldsymbol{\theta}}{\partial \boldsymbol{\phi}}$
$\frac{\partial \mathcal{R}_\lambda(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}}$
Learning Layer
$\boldsymbol{\theta}$
$\frac{\partial \mathcal{R}_\lambda(\boldsymbol{\phi})}{\partial \boldsymbol{\theta}}$
$\boldsymbol{\theta}^{\text{proj}} = \text{Proj}(\boldsymbol{\theta})$
$\frac{\partial \boldsymbol{\theta}^{\text{proj}}}{\partial \boldsymbol{\theta}}$
Projection Layer
$\boldsymbol{\theta}^{\text{proj}}$
$\frac{\partial \mathcal{R}_\lambda(\boldsymbol{\phi})}{\partial \boldsymbol{\theta}^{\text{proj}}}$
$r_\lambda(\boldsymbol{\theta}^{\text{proj}}, \cdot)$
$\frac{\partial r_\lambda(\boldsymbol{\theta}^{\text{proj}}, \boldsymbol{y})}{\partial \boldsymbol{\theta}^{\text{proj}}}$
DRO Layer
$r_\lambda(\boldsymbol{\theta}^{\text{proj}}, \cdot)$
$\mathcal{R}_\lambda(\boldsymbol{\phi}) = \mathbb{E}_{\mathbb{P}}[r_\lambda(\boldsymbol{\theta}^{\text{proj}}, \boldsymbol{y})]$
$\mathcal{R}_\lambda(\boldsymbol{\phi})$
Forward Path
Backward Path

Figure 2. Decision-focused learning pipeline for contextual distributionally robust decision-making.

$\boldsymbol{\phi}^* \in \arg\min_{\boldsymbol{\phi}\in\Phi} \mathcal{R}(\boldsymbol{\phi})$ *and* $\lim_{k\to\infty} \inf_{\boldsymbol{\phi}\in\Phi} \mathcal{R}_{\lambda_{n_k}}(\boldsymbol{\phi}) = \inf_{\boldsymbol{\phi}\in\Phi} \mathcal{R}(\boldsymbol{\phi})$

Proof. See Appendix B.3 □

4.3. Gradient Estimation

According to Theorem 4.13, the optimal parameter $\boldsymbol{\phi}$ of the learning model $M_{\boldsymbol{\phi}}$ can be derived by optimizing the surrogate value functions $\mathcal{R}_\lambda(\boldsymbol{\phi})$ sequentially. We next show in Theorem 4.14 that the surrogate value function $\mathcal{R}_\lambda(\boldsymbol{\phi})$ is differentiable, so it can be optimized via gradient descent.

Theorem 4.14. *Suppose conditions in Theorem 4.5 and Theorem 4.13 hold, then $\mathcal{R}_\lambda(\boldsymbol{\phi})$ is differentiable with respect to $\boldsymbol{\phi}$ and the gradient is*

$$\frac{\partial \mathcal{R}_\lambda(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}} = \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\left[\frac{\partial r_\lambda(\boldsymbol{\theta}, \boldsymbol{y})}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{\theta}}{\partial \boldsymbol{\phi}}\right] \tag{13}$$

where $\boldsymbol{\theta} = M_{\boldsymbol{\phi}}(\boldsymbol{z})$ is the learning target, and $\frac{\partial r_\lambda(\boldsymbol{\theta},\boldsymbol{y})}{\partial \boldsymbol{\theta}}$ can be computed by

$$\begin{aligned}
&\frac{\partial r_\lambda(\boldsymbol{\theta}, \boldsymbol{y})}{\partial \boldsymbol{\theta}} \\
&= \mathbb{E}_{\boldsymbol{x_d}\sim p}\left[\frac{E^{'}(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)} l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big)\right] - \\
&\mathbb{E}_{\boldsymbol{x_d}\sim p}\left[\frac{E^{'}(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}\right] \mathbb{E}_{\boldsymbol{x_d}\sim p}\left[l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big)\right] \\
&+ \mathbb{E}_{\boldsymbol{x_d}\sim p}\left[\frac{\partial l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big)}{\partial \boldsymbol{x_c}^*} \frac{\partial \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right]
\end{aligned} \tag{14}$$

where $p = p(\boldsymbol{x}_d|\boldsymbol{\theta}, \lambda)$ and $E^{'}(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda) = \frac{\partial E(\boldsymbol{x_d},\boldsymbol{\theta},\lambda)}{\partial \boldsymbol{\theta}}$.

Proof. See Appendix B.4 □

Note that in the last term of Equation (14), $\frac{\partial \boldsymbol{x_c}^*(\boldsymbol{x_d},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ is the gradient of continuous decision with respect to the parameter, which is exactly what we develop in Theorem 4.6.

The gradient (14) can be estimated by sampling from distribution $p(\boldsymbol{x}_d|\boldsymbol{\theta}, \lambda)$. However, direct sampling from $p(\boldsymbol{x}_d|\boldsymbol{\theta}, \lambda)$ necessitates the computation of the normalizer in Equation (11), which requires the calculation of the energy function of all the feasible integer solutions.

To avoid this problem, we adopt the self-normalized importance sampling method (See Appendix A.5). To construct a proposal distribution q that resembles $p(\boldsymbol{x}_d|\boldsymbol{\theta}, \lambda)$, we first derive T integer solutions $\mathscr{C} = \{\boldsymbol{x_d}^1, \cdots, \boldsymbol{x_d}^T\}$ with the largest energy functions by solving f for T times (See Appendix A.6 for readers not familiar with this oracle) and then construct the proposal distribution q as follows.

$$\begin{aligned}
q(\boldsymbol{x_d}) &= \frac{E(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{\sum_{t\in[T]} E(\boldsymbol{x_d}^t, \boldsymbol{\theta}, \lambda) + M(|\mathcal{X}_d| - T)}, \forall \boldsymbol{x_d} \in \mathscr{C} \\
q(\boldsymbol{x_d}) &= \frac{M}{\sum_{t\in[T]} E(\boldsymbol{x_d}^t, \boldsymbol{\theta}, \lambda) + M(|\mathcal{X}_d| - T)}, \forall \boldsymbol{x_d} \notin \mathscr{C}
\end{aligned} \tag{15}$$

where M is a constant that can be understood as the energy of other integer solutions.

Therefore, each term in Equation (14) can be estimated unbiasedly by sampling from q.

5. Application: Contextual Distributionally Robust Decision-Making

As an application of the differentiable DRO layers in Section 4, we develop a decision-focused learning pipeline for contextual distributionally robust decision-making tasks (Bertsimas & Van Parys, 2022; Wang et al., 2021; Yang et al., 2022).

In this paper, we mainly focus on and develop a decision-focused learning method for DRO with SOC ambiguity set, but in fact, the proposed DRO Layer technique can also be extended to the Wasserstein ambiguity set and we discuss this issue in Appendix C.

5.1. Decision-Focused Learning Pipeline

The proposed pipeline is illustrated in Figure 2. A learning model $M_{\boldsymbol{\phi}}$ is first leveraged to learn the ambiguity set parameter $\boldsymbol{\theta}$ from covariate $\boldsymbol{z}$. However, the output parameter $\boldsymbol{\theta}$ provided by the learning model can lead to an empty ambiguity set, *i.e.*, $\mathscr{U}(\boldsymbol{\theta}) = \varnothing$, and this problem typically happens when the learning model is a neural network (NN).

To fix this problem, we add a projection layer after the learning layer. The projection layer takes $\boldsymbol{\theta}$ as input and outputs $\boldsymbol{\theta}^{\text{proj}}$ such that $\mathscr{U}(\boldsymbol{\theta}^{\text{proj}})$ is always non-empty. To achieve this, we construct the projection layer as follows.

$$\boldsymbol{\theta}^{\text{proj}} := \arg\min_{\boldsymbol{\theta}^{\text{proj}}} \left\| \boldsymbol{\theta}^{\text{proj}} - \boldsymbol{\theta} \right\| \text{ s.t. } \mathbb{Q}_{\boldsymbol{z}} \in \mathscr{U}(\boldsymbol{\theta}^{\text{proj}}) \quad (16)$$

In the constraint of (16), we explicitly require that a distribution $\mathbb{Q}_{\boldsymbol{z}}$ lies in the parameterized SOC ambiguity set $\mathscr{U}(\boldsymbol{\theta}^{\text{proj}})$, which ensures the non-emptyness of $\mathscr{U}(\boldsymbol{\theta}^{\text{proj}})$. This distribution $\mathbb{Q}_{\boldsymbol{z}}$ should be understood as an estimation of the conditional distribution of uncertainty $\boldsymbol{y}$ given covariate $\boldsymbol{z}$.

To construct such a conditional distribution estimation $\mathbb{Q}_{\boldsymbol{z}}$, we take the idea in Bertsimas & Kallus (2020), which constructed the conditional distribution from data $(\boldsymbol{z}_n, \boldsymbol{y}_n), n \in [N]$ in a weighted sample average way as follows.

$$\mathbb{Q}_{\boldsymbol{z}} = \sum_{n=1}^{N} \omega_n(\boldsymbol{z}) \delta_{\boldsymbol{y}_n}, \omega_n(\boldsymbol{z}) \geq 0, \sum_{n=1}^{N} \omega_n(\boldsymbol{z}) = 1 \quad (17)$$

where $\delta_{[\cdot]}$ is the Dirac delta function.

In (17), the weight $\omega_n(\boldsymbol{z})$ can be intuitively understood as a measurement of closeness between $\boldsymbol{z}$ and data $\boldsymbol{z}_n$. Some research papers provide such weight functions to choose from (Bertsimas & Kallus, 2020; Kallus & Mao, 2023), for example, the k-nearest-neighbors weight function.

With the formulation (17) of $\mathbb{Q}_{\boldsymbol{z}}$, the constraint of (16) is equivalent to

$$\sum_{n=1}^{N} \omega_n(\boldsymbol{z}) g_i(\boldsymbol{y}_n, \boldsymbol{\alpha}_i) \leq \sigma_i, \ \forall i \in [I] \quad (18)$$

Further, if $g_i, i \in [I]$ are selected as in Appendix A.2, then (18) can be reformulated into finitely many second-order cone constraints (See Appendix A.7 for this result). Therefore, the projection layer is a convex optimization layer.

After projection, $\boldsymbol{\theta}^{\text{proj}}$ is fed into the DRO layer to construct the surrogate value function $\mathcal{R}_\lambda(\boldsymbol{\phi})$, which is usually estimated by data of a certain batch size. Then, the back-propagation and parameter update processes are conducted.

5.2. Prediction-Focused Pre-training

If the learning model is very complicated, for example, a deep neural network, it can be hard to train it directly via the decision-focused learning pipeline. Therefore, to facilitate convergence, we first pre-train the learning model in a prediction-focused fashion.

In Definition 4.2, function $g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i)$ captures distributional features of the uncertainty $\boldsymbol{y}$. Therefore, lower $\mathbb{E}_{\boldsymbol{y}\sim\mathbb{Q}}[g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i)]$ can be deemed as better characterization of these distributional features, *i.e.*, better prediction.

By the above observation, we define the loss function of prediction-focused pre-training as

$$\text{Loss} = \sum_{i=1}^{I} \left\| \mathbb{E}_{\mathbb{Q}_{\boldsymbol{z}}}[g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i)] \right\| + \left\| \mathbb{E}_{\mathbb{Q}_{\boldsymbol{z}}}[g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i)] - \sigma_i \right\|$$

where $\mathbb{Q}_{\boldsymbol{z}}$ is defined in Equation (17).

6. Experiments

To validate the effectiveness of the proposed differentiable DRO layers, we conduct experiments on a toy example and the portfolio management problem[1].

In both experiments, problems are formulated in a contextual DRO setting and thus we can apply the decision-focused learning pipeline developed in Section 5. The detailed experiment setup is presented in Appendix D.

6.1. Toy Example: Multi-item Newsvendor Problem

We consider a multi-item newsvendor problem (19) where two options are provided for buying each item, *i.e.*, retail and wholesale. The wholesale price $\boldsymbol{a}_i^d$ is lower than the retail price $\boldsymbol{a}_i^c$ but it can only be sold at a fixed amount $\boldsymbol{v}_i$.

$$\begin{aligned} \min_{\boldsymbol{x}^c, \boldsymbol{x}^d} & \left\{ \sum_{i=1}^{n} a_i^c x_i^c + a_i^d v_i x_i^d + b_i (y_i - x_i^c - v_i x_i^d)^+ \right. \\ & \left. \qquad + d_i (x_i^c + v_i x_i^d - y_i)^+ \right\} \\ \text{s.t.} & \qquad x_i^c \geq 0, \ x_i^d \in \{0, 1\}, \ \forall i \in [n] \end{aligned} \quad (19)$$

where y_i is the demand of item i, the continuous variable $\boldsymbol{x}^c$ denotes amount of item bought from retail, integer variable $\boldsymbol{x}^d$ denotes the wholesale option, $\boldsymbol{b}$ is the unit price of additional ordering, and $\boldsymbol{d}$ is the unit holding cost.

To characterize the uncertainty in demand y_i, we consider the following three types of parameterized SOC ambiguity sets, where first- and second-order moment features are characterized.

$$\mathscr{U}_{\text{I}}(\boldsymbol{\mu}_{\text{I}}, \sigma_{\text{I}}) = \left\{ \mathbb{P} \,\middle|\, \begin{array}{c} \mathbb{P}(\Xi) = 1 \\ \mathbb{E}_{\mathbb{P}}[\|\boldsymbol{y} - \boldsymbol{\mu}_{\text{I}}\|_1] \leq \sigma_{\text{I}} \end{array} \right\} \quad \text{(SOC-I)}$$

$$\mathscr{U}_{\text{II}}(\boldsymbol{\mu}_{\text{II}}, \sigma_{\text{II}}) = \left\{ \mathbb{P} \,\middle|\, \begin{array}{c} \mathbb{P}(\Xi) = 1 \\ \mathbb{E}_{\mathbb{P}}\left[\|\boldsymbol{y} - \boldsymbol{\mu}_{\text{II}}\|_2^2\right] \leq \sigma_{\text{II}} \end{array} \right\} \quad \text{(SOC-II)}$$

$$\mathscr{U}_{\text{III}}(\boldsymbol{\mu}_{\text{I}}, \sigma_{\text{I}}, \boldsymbol{\mu}_{\text{II}}, \sigma_{\text{II}}) = \mathscr{U}_{\text{I}}(\boldsymbol{\mu}_{\text{I}}, \sigma_{\text{I}}) \bigcap \mathscr{U}_{\text{II}}(\boldsymbol{\mu}_{\text{II}}, \sigma_{\text{II}}) \quad \text{(SOC-III)}$$

[1]Source code of all the experiments is available at https://github.com/DOCU-Lab/Differentiable_DRO_Layers.

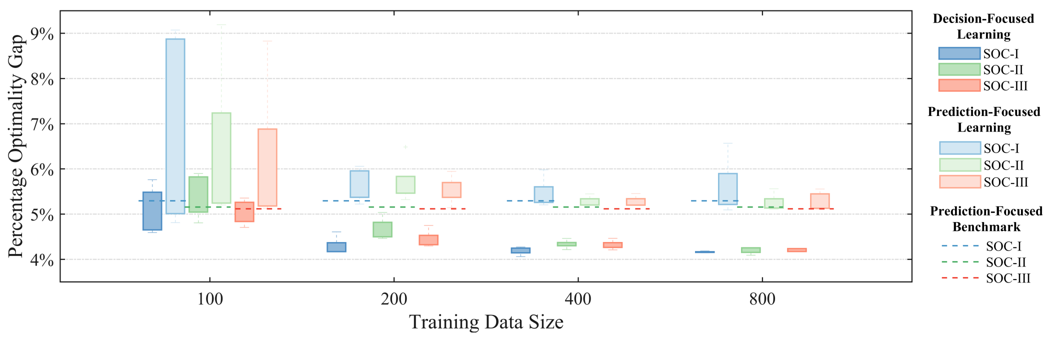

Figure 3. Experimental results on the multi-item newsvendor problem.

The ambiguity set parameters are learned by NN in all the experiments. We compare the proposed decision-focused learning method (Section 5.1) with the prediction-focused learning method, which is described in Section 5.2.

To fully verify the superior performance of the decision-focused learning method, we further compare it with the prediction-focused benchmark. The parameters of the ambiguity set in the prediction-focused benchmark are selected with the full knowledge of the conditional distribution $p(\boldsymbol{y}|\boldsymbol{z})$. For example, in SOC-I, the parameters $\boldsymbol{\mu}_{\mathrm{I}}$ and σ_{I} are directly set to the mean and first-order absolute central moment of the conditional distribution $p(\boldsymbol{y}|\boldsymbol{z})$. Therefore, the performance of the prediction-focused benchmark is the 'optimal' performance that the prediction-focused learning method can expect.

In experiments, we take $n = 4$ and conduct 10 runs for cases of different training data sizes. We use the percentage optimality gap, whose definition can be found in Appendix D, to measure the performance of each method. The experiment results are presented in Figure 3.

By Figure 3, the performance of prediction-focused learning will converge to the prediction-focused benchmark as the training data size grows. On the contrary, by directly minimizing the decision loss, the proposed decision-focused learning method demonstrates better performance. Quantitatively, the proposed decision-focused learning method demonstrates average improvements of 21.4%, 18.7%, and 18.1% compared with the prediction-focused benchmark in the three ambiguity sets, respectively.

6.2. Portfolio Management Problem

As mentioned in the literature review, Costa & Iyengar (2023) also developed an end-to-end DRO method for portfolio management problems, but their method only applies to pure continuous decision cases. Therefore, in this part, we make a direct comparison with the method proposed in Costa & Iyengar (2023) on the portfolio management problem with pure continuous decisions, and then we further conduct experiments with mixed-integer decisions.

6.2.1. Portfolio Management Problem with Pure Continuous Decisions

The portfolio management problem (20) with pure continuous decisions aims to select the optimal portfolio $\boldsymbol{x} \in \mathbb{R}^n$ that minimizes the cost and the uncertainty comes from the return $\boldsymbol{y}$.

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & -\boldsymbol{y}^T\boldsymbol{x} \\ \text{s.t.} \quad & \mathbf{1}^T\boldsymbol{x} = 1,\ \boldsymbol{x} \geq 0 \end{aligned} \tag{20}$$

To characterize the uncertainty in return $\boldsymbol{y}$, we consider the following parameterized SOC ambiguity set.

$$\mathscr{U}(\boldsymbol{\mu}, \boldsymbol{\sigma}) = \left\{ \mathbb{P} \left| \begin{array}{c} \mathbb{P}(\Xi) = 1 \\ \mathbb{E}_{\mathbb{P}}\left[\|\boldsymbol{y}_i - \boldsymbol{\mu}_i\|_2^2 \right] \leq \boldsymbol{\sigma}_i, \forall i \in [n] \end{array} \right. \right\} \tag{21}$$

In experiments, we take the dimension n of assets to 40. For learning the ambiguity set parameter, we use the same 2-layer neural network in both our method and the method proposed in Costa & Iyengar (2023) for a fair comparison.

We use 2,500 data for training, 500 data for validation, and 1,000 data for testing. The average percentage profit of the proposed decision-focused learning method, prediction-focused learning method, and the method proposed in Costa & Iyengar (2023) are 0.289, 0.082, and 0.268, respectively. We also plot the wealth evolution in Figure 4. This superiority in performance is ascribed to the fact that a restrictive assumption on the structure of uncertainty distribution is presumed in the method of Costa & Iyengar (2023) while our method applies to general distributions.

6.2.2. Portfolio Management Problem with Mixed-Integer Decisions

We further conduct experiments on the mixed-integer portfolio management problem (22), where some of the assets

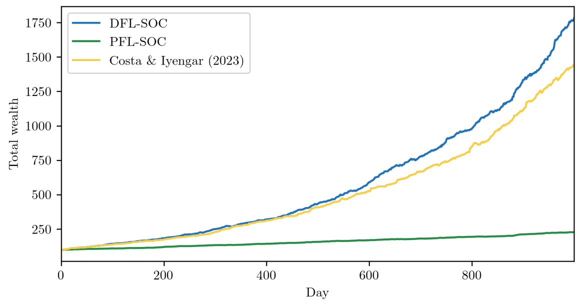

Figure 4. Wealth evolution on a 40-dimensional continuous portfolio management problem using decision-focused learning, prediction-focused learning, and method proposed in Costa & Iyengar (2023). (In the legend, 'DFL' stands for decision-focused learning, and 'PFL' stands for prediction-focused learning.)

are only allowed to be bought with either a fixed amount or 0. Thus, the decisions on these assets are binary variables.

$$\begin{aligned} \min_{\boldsymbol{x}_c, \boldsymbol{x}_d} \quad & -\boldsymbol{y}_c^T \boldsymbol{x}_c - \boldsymbol{y}_d^T \operatorname{diag}(\boldsymbol{v}) \boldsymbol{x}_d \\ \text{s.t.} \quad & \mathbf{1}^T \boldsymbol{x}_c + \boldsymbol{v}^T \boldsymbol{x}_d = 1, \ \boldsymbol{x}_c \geq 0, \boldsymbol{x}_d \in \{0,1\} \end{aligned} \tag{22}$$

where $\boldsymbol{y}_c, \boldsymbol{y}_d$ are returns corresponding to assets with continuous decisions $\boldsymbol{x}_c$ and binary decisions $\boldsymbol{x}_d$, and $\boldsymbol{v}$ denotes the fixed amount of assets allowed to be bought.

In experiments, the SOC ambiguity set is set to (21) if not explicitly specified and the number of binary decisions is set to $\frac{1}{5}$ of the number of the problem dimension.

Performance with Different Dimensions: We compare the performance of the proposed decision-focused learning method with prediction-focused learning method on mixed-integer portfolio management problems of different dimensions, and the results are presented in Table 1, where problem dimension refers to the number of assets. Generally, the performance gap between decision-focused and predict-focused learning methods scales as the problem dimension becomes larger. The advantage of decision-focused method becomes more apparent as the problem dimension increases.

Table 1. Average percentage profit of decision-focused learning and prediction-focused learning on mixed-integer portfolio management problems with different dimensions.

Problem dimension	Method		Improvement
	DFL	PFL	
20	0.1561	0.0583	167%
40	0.1763	0.0634	178%
60	0.2145	0.0479	347%

Performance with Different SOC Ambiguity Sets: The the SOC ambiguity set (4) is determined by the constraints g_i, which significantly affect the performance. Therefore, we conduct experiments on 60-dimensional mixed-integer portfolio management problems with three SOC ambiguity sets with different numbers of constraints. The detailed information of these three ambiguity sets is presented in Appendix D.

The performance of decision-focused and prediction-focused learning methods using these three types of ambiguity sets are presented in Table 2. With more complex ambiguity sets, both decision-focused and prediction-focused methods have better performance, and the improvement for decision-focused learning generally grows.

Table 2. Average percentage profit of decision-focused learning and prediction-focused learning on 60-dimensional mixed-integer portfolio management problem with different SOC ambiguity sets.

SOC constraint No.	Method		Improvement
	DFL	PFL	
15	0.0762	0.0357	113%
30	0.1349	0.0491	174%
60	0.2145	0.0479	347%

7. Discussion of Limitations

The major limitation of applying the proposed DRO-Layer method to large-scale problems lies in the heavy computational burden. Specifically, the decision-focused learning pipeline we developed in Section 4 can be decomposed into four processes: 1. learning layer; 2. projection layer; 3. solving MICP; 4. DRO Layer. Processes 2 and 4 are built on Cvxpylayers (Agrawal et al., 2019a), and Process 3 is built on commercial solver Gurobi. To test the computational efficiency and scalability, we conduct experiments and present the computational time of each of these four processes in Appendix E.

It is noteworthy that both Cvxpylayers and Gurobi are built on CPU rather than on GPU, thereby leading to computational inefficiency and relatively weak scalability inevitably. If GPU training is allowable in these software, we believe the computation will not be a burden.

8. Conclusion

We developed the first generic differentiable DRO layers, where a novel dual-view methodology was proposed to handle the mixed-integer decision via distinct principles. Based on the proposed differentiable DRO layers, we further developed a decision-focused learning pipeline for contextual DRO problems and verified its effectiveness in experiments.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 62103264, and in part by the National Natural Science Foundation of China (Basic Science Center Program) under Grant 61988101.

Impact Statement

This paper presents work whose goal is to advance the integration of DRO and machine learning. None of the potential impacts of our work we feel must be specifically highlighted here.

References

Agrawal, A., Amos, B., Barratt, S., Boyd, S., Diamond, S., and Kolter, J. Z. Differentiable convex optimization layers. *Advances in neural information processing systems*, 32, 2019a.

Agrawal, A., Barratt, S. T., Boyd, S. P., Busseti, E., and Moursi, W. M. Differentiating through a cone program. *Journal of Applied and Numerical Optimization*, 2019b.

Amos, B. and Kolter, J. Z. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pp. 136–145. PMLR, 2017.

Ben-Tal, A. and Nemirovski, A. *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*. SIAM, 2001.

Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J.-P., and Bach, F. Learning with differentiable pertubed optimizers. *Advances in neural information processing systems*, 33:9508–9519, 2020.

Bertsimas, D. and Kallus, N. From predictive to prescriptive analytics. *Management Science*, 66(3):1025–1044, 2020.

Bertsimas, D. and Van Parys, B. Bootstrap robust prescriptive analytics. *Mathematical Programming*, 195(1): 39–78, 2022.

Bertsimas, D., Sim, M., and Zhang, M. Adaptive distributionally robust optimization. *Management Science*, 65(2): 604–618, 2019.

Bonnans, J. F. and Shapiro, A. *Perturbation analysis of optimization problems*. Springer Science & Business Media, 2013.

Busseti, E., Moursi, W. M., and Boyd, S. Solution refinement at regular points of conic problems. *Computational Optimization and Applications*, 74:627–643, 2019.

Chenreddy, A. R. and Delage, E. End-to-end conditional robust optimization. *ArXiv*, abs/2403.04670, 2024.

Chenreddy, A. R., Bandi, N., and Delage, E. Data-driven conditional robust optimization. *Advances in Neural Information Processing Systems*, 35:9525–9537, 2022.

Costa, G. and Iyengar, G. N. Distributionally robust end-to-end portfolio construction. *Quantitative Finance*, 23(10): 1465–1482, 2023.

Dalle, G., Baty, L., Bouvier, L., and Parmentier, A. Learning with combinatorial optimization layers: a probabilistic approach. *arXiv preprint arXiv:2207.13513*, 2022.

Delage, E. and Ye, Y. Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations research*, 58(3):595–612, 2010.

Donti, P., Amos, B., and Kolter, J. Z. Task-based end-to-end model learning in stochastic optimization. *Advances in neural information processing systems*, 30, 2017.

Elmachtoub, A. N. and Grigas, P. Smart "predict, then optimize". *Management Science*, 68(1):9–26, 2022.

Ferber, A., Wilder, B., Dilkina, B., and Tambe, M. Mipaal: Mixed integer program as a layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 1504–1511, 2020.

Kallus, N. and Mao, X. Stochastic optimization forests. *Management Science*, 69(4):1975–1994, 2023.

Kong, L., Cui, J., Zhuang, Y., Feng, R., Prakash, B. A., and Zhang, C. End-to-end stochastic optimization with energy-based model. *Advances in Neural Information Processing Systems*, 35:11341–11354, 2022.

Mandi, J. and Guns, T. Interior point solving for lp-based prediction+ optimisation. *Advances in Neural Information Processing Systems*, 33:7272–7282, 2020.

Mandi, J., Bucarey, V., Tchomba, M. M. K., and Guns, T. Decision-focused learning: through the lens of learning to rank. In *International Conference on Machine Learning*, pp. 14935–14947. PMLR, 2022.

Mandi, J., Kotary, J., Berden, S., Mulamba, M., Bucarey, V., Guns, T., and Fioretto, F. Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. *arXiv preprint arXiv:2307.13565*, 2023.

Mohajerin Esfahani, P. and Kuhn, D. Data-driven distributionally robust optimization using the wasserstein metric: performance guarantees and tractable reformulations. *Mathematical Programming*, 171(1-2):115–166, 2018.

Mulamba, M., Mandi, J., Diligenti, M., Lombardi, M., Bucarey, V., and Guns, T. Contrastive losses and solution caching for predict-and-optimize. *arXiv preprint arXiv:2011.05354*, 2020.

Niepert, M., Minervini, P., and Franceschi, L. Implicit mle: backpropagating through discrete exponential family distributions. *Advances in Neural Information Processing Systems*, 34:14567–14579, 2021.

Ning, C. and You, F. Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming. *Computers & Chemical Engineering*, 125:434–448, 2019.

Pogančić, M. V., Paulus, A., Musil, V., Martius, G., and Rolinek, M. Differentiation of blackbox combinatorial solvers. In *International Conference on Learning Representations*, 2020.

Rahimian, H. and Mehrotra, S. Frameworks and results in distributionally robust optimization. *Open Journal of Mathematical Optimization*, 3:1–85, 2022.

Sadana, U., Chenreddy, A., Delage, E., Forel, A., Frejinger, E., and Vidal, T. A survey of contextual optimization methods for decision making under uncertainty. *arXiv preprint arXiv:2306.10374*, 2023.

Shah, S., Wang, K., Wilder, B., Perrault, A., and Tambe, M. Decision-focused learning without decision-making: Learning locally optimized decision losses. *Advances in Neural Information Processing Systems*, 35:1320–1332, 2022.

Shapiro, A., Dentcheva, D., and Ruszczynski, A. *Lectures on stochastic programming: modeling and theory*. SIAM, 2021.

Sun, C., Liu, L., and Li, X. Predict-then-calibrate: A new perspective of robust contextual lp. *arXiv preprint arXiv:2305.15686*, 2023.

Wang, I., Becker, C., Van Parys, B., and Stellato, B. Learning for robust optimization. *arXiv preprint arXiv:2305.19225*, 2023.

Wang, T., Chen, N., and Wang, C. Distributionally robust prescriptive analytics with wasserstein distance. *arXiv preprint arXiv:2106.05724*, 2021.

Wiesemann, W., Kuhn, D., and Sim, M. Distributionally robust convex optimization. *Operations research*, 62(6): 1358–1376, 2014.

Wilder, B., Dilkina, B., and Tambe, M. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 1658–1665, 2019.

Yang, J., Zhang, L., Chen, N., Gao, R., and Hu, M. Decision-making with side information: A causal transport robust approach. *Optimization Online*, 2022.

Yang, Y., Yin, Y., Wang, D., Ignatius, J., Cheng, T., and Dhamotharan, L. Distributionally robust multi-period location-allocation with multiple resources and capacity levels in humanitarian logistics. *European Journal of Operational Research*, 305(3):1042–1062, 2023.

Zhang, J., Li, Y., and Yu, G. Emergency relief network design under ambiguous demands: A distributionally robust optimization approach. *Expert Systems with Applications*, 208:118139, 2022.

Zharmagambetov, A., Amos, B., Ferber, A., Huang, T., Dilkina, B., and Tian, Y. Landscape surrogate: Learning decision losses for mathematical optimization under partial information. *arXiv preprint arXiv:2307.08964*, 2023.

Zhou, Y., Shahidehpour, M., Wei, Z., Li, Z., Sun, G., and Chen, S. Distributionally robust unit commitment in coordinated electricity and district heating networks. *IEEE Transactions on Power Systems*, 35(3):2155–2166, 2019.

A. Supplementary Background Material

In this section, we give supplementary information on DRO with SOC ambiguity set in A.1, A.2, and A.3. Most of these materials are selected from Ben-Tal & Nemirovski (2001) and Bertsimas et al. (2019).

In Appendix A.4, A.5, A.6, and A.7, detailed information on our method are provided.

A.1. Second-Order Cone Representable Set

[Ben-Tal & Nemirovski (2001), p.86] A set $\mathscr{W} \subset \mathbb{R}^n$ is a second-order cone representable set if there exists J second-order cone inequalities of the form

$$\boldsymbol{A}_j \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{v} \end{bmatrix} - \boldsymbol{b}_j \geq_{L^{m_j}} \boldsymbol{0}, \forall j \in [J] \tag{23}$$

such that

$$\boldsymbol{y} \in \mathscr{W} \iff \exists \boldsymbol{v} : \boldsymbol{A}_j \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{v} \end{bmatrix} - \boldsymbol{b}_j \geq_{L^{m_j}} \boldsymbol{0}, \forall j \in [J] \tag{24}$$

where L^m is the m dimensional second-order cone:

$$L^m = \left\{ \boldsymbol{y} = (y_1, \cdots, y_m)^T \in \mathbb{R}^m \middle| y_m \geq \sqrt{y_1^2 + \cdots + y_{m-1}^2} \right\} \tag{25}$$

A.2. Examples of Parameterized SOC Ambiguity Set

The parameterized SOC ambiguity set is determined by its constraints $\mathbb{E}_{\mathbb{P}}[g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i)] \leq \sigma_i$, and here we present some choices of function g and the parameter $\boldsymbol{\alpha}$.

1. $g = \boldsymbol{\mu}^T \boldsymbol{y}$ with vector $\boldsymbol{\mu}$ as the parameter.
2. $g = |\boldsymbol{\mu}^T \boldsymbol{y} - h|$ with vector $\boldsymbol{\mu}$ and scalar h as the parameter.
3. $g = (|\boldsymbol{\mu}^T \boldsymbol{y} - h|)^p$ for some rational $p \geq 1$ with vector $\boldsymbol{\mu}$ and scalar h as the parameter.
4. $g = ((\boldsymbol{\mu}^T \boldsymbol{y} - h)^+)^2 = (\max\{0, \boldsymbol{\mu}^T \boldsymbol{y} - h\})^2$ vector $\boldsymbol{\mu}$ and scalar h as the parameter.
5. $g = \|\boldsymbol{A}\boldsymbol{y} - \boldsymbol{\mu}\|_p$ for some rational $p \geq 1$ norm $\|\cdot\|_p$ with matrix $\boldsymbol{A}$ and vector $\boldsymbol{\mu}$ as the parameter.

More examples can be constructed by taking the maximum, *i.e.*, $g = \max_{l \in [L]} g_i$, and non-negtive sum, *i.e.*, $g = \sum_{l=1}^{L} \lambda_i g_i$ for $\lambda_i \geq 0$. See Ben-Tal & Nemirovski (2001) for more operators that preserve the second-order cone representable property of g.

A.3. Slater's Condition for Parameterized SOC Ambiguity Set

According to Proposition 1 in Bertsimas et al. (2019), the parameterized SOC ambiguity set $\mathscr{U}(\boldsymbol{\theta})$ (defined in Definition 4.2) can be equivalently reformulated as follows.

$$\mathscr{U}(\boldsymbol{\theta}) = \left\{ \#_{\boldsymbol{y}}\mathbb{Q} \middle| \begin{array}{c} (\boldsymbol{y}, \boldsymbol{u}) \sim \mathbb{Q} \\ \mathbb{E}_{\mathbb{Q}}[u_i] \leq \sigma_i, \forall i \in [I] \\ \mathbb{Q}(\mathscr{V}) = 1 \end{array} \right\} \tag{26}$$

where each dimension u_i of variable $\boldsymbol{u}$ corresponds to the constraint g_i in $\mathscr{U}(\boldsymbol{\theta})$, distribution $\mathbb{Q}$ is on the space of $(\boldsymbol{y}, \boldsymbol{u})$, $\#_{\boldsymbol{y}}\mathbb{Q}$ is the marginal distribution of $\mathbb{Q}$ on dimension $\boldsymbol{y}$, and the support $\mathscr{V}$ is defined as

$$\mathscr{V} = \{(\boldsymbol{y}, \boldsymbol{u}) | \boldsymbol{y} \in \Xi, g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i) \leq u_i, \forall i \in [I]\} = \{(\boldsymbol{y}, \boldsymbol{u}) | \boldsymbol{y} \in \Xi\} \bigcap_{i=1}^{I} \{(\boldsymbol{y}, \boldsymbol{u}) | g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i) \leq u_i\} \tag{27}$$

By Definition 4.2, $\{(\boldsymbol{y}, \boldsymbol{u}) | \boldsymbol{y} \in \Xi\}$ is second-order cone representable set, and each $\{(\boldsymbol{y}, \boldsymbol{u}) | g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i) \leq u_i\}$ are differentiable parameterized second-order representable sets. Therefore, $\mathscr{V}$ is also a differentiable parameterized second-order representable set, so $\mathscr{V}$ can be represented by finitely many second-order cone constraints.

$$\mathscr{V} = \left\{ (\boldsymbol{y}, \boldsymbol{u}) \middle| \exists \boldsymbol{v} : \boldsymbol{A}_j(\boldsymbol{\theta}) \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} - \boldsymbol{b}_j(\boldsymbol{\theta}) \geq_{L^{m_j}} \boldsymbol{0}, \forall j \in [J] \right\} \tag{28}$$

The Slater's condition requires that there exist a $(\boldsymbol{y}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ such that $u_i^* < \sigma_i,\ \forall i \in [I]$ and

$$\boldsymbol{A}_j(\boldsymbol{\theta}) \begin{bmatrix} \boldsymbol{y}^* \\ \boldsymbol{u}^* \\ \boldsymbol{v}^* \end{bmatrix} - \boldsymbol{b}_j(\boldsymbol{\theta}) >_{L^{m_j}} \mathbf{0}, \forall j \in [J] \tag{29}$$

A.4. Discussion of Assumption 4.8

Assumption (i) ensures the uniqueness of the continuous decision, which is common in differentiable optimization layer research (Agrawal et al., 2019a).

For the discrete decision $\boldsymbol{x}_d$, since its feasible region $\mathcal{X}_d$ is finite, we can not require the uniqueness of $\boldsymbol{x}_d$ for all $\boldsymbol{\theta} \in \Theta$, where $\boldsymbol{\theta} = M_{\boldsymbol{\phi}}(\boldsymbol{z})$. For example, if we consider the combinatorial optimization problem, *i.e.*,

$$\boldsymbol{x}_d = \underset{\boldsymbol{x}_d \in \{0,1\}^n}{\arg\min}\ f(\boldsymbol{x}_d, \boldsymbol{\theta}) := \boldsymbol{\theta}^T \boldsymbol{x}_d,\ \text{s.t. } \mathbf{1}^T \boldsymbol{x}_d = 1 \tag{30}$$

The solution to this problem is not unique when the prediction $\boldsymbol{\theta}$ has multiple minimum elements.

However, we note that for problem (30), the $\boldsymbol{\theta}$ leading to multiple solutions has measure zero in its space Θ, and this property holds for a lot of problems. Therefore, we require in assumption (ii) the uniqueness to hold almost surely with respect to the marginal distribution $\mathbb{P}_{\boldsymbol{z}}$ of $\boldsymbol{z}$.

Specifically, in combinatorial optimization problem (30), if the learning model is a linear one, *i.e.*, $\boldsymbol{\theta} = \boldsymbol{A}\boldsymbol{z} + \boldsymbol{b}$ with $\boldsymbol{\phi} = (\boldsymbol{A}, \boldsymbol{b})$, where the rows of matrix $\boldsymbol{A}$ are all different, and covariate $\boldsymbol{z}$ has marginal distribution absolutely continuous with respect to the Lebesgue measure, then (ii) is satisfied.

We further note that (ii) is also implicitly assumed in Pogančić et al. (2020).

A.5. Self-Normalized Importance Sampling

Suppose we want to compute the expectation of a random variable $J(\boldsymbol{x}_d)$, *i.e.*,

$$\mathbb{E}_{\boldsymbol{x_d} \sim p(\boldsymbol{x_d}|\boldsymbol{\theta},\lambda)}[J(\boldsymbol{x_d})] \tag{31}$$

where $p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda)$ is defined as in Equation (11).

The importance sampling aims to compute $\mathbb{E}_{\boldsymbol{x_d} \sim p(\boldsymbol{x_d}|\boldsymbol{\theta},\lambda)}[J(\boldsymbol{x_d})]$ by leveraging a proposal distribution q which is absolutely continuous with respect to $p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda)$. With proposal distribution q, the above expectation can be computed by

$$\mathbb{E}_{\boldsymbol{x_d} \sim p(\boldsymbol{x_d}|\boldsymbol{\theta},\lambda)}[J(\boldsymbol{x_d})] = \frac{\mathbb{E}_{\boldsymbol{x_d} \sim q(\boldsymbol{x_d})}\left[\frac{E(\boldsymbol{x_d},\boldsymbol{\theta},\lambda)}{q(\boldsymbol{x_d})} J(\boldsymbol{x_d})\right]}{\mathbb{E}_{\boldsymbol{x_d} \sim q(\boldsymbol{x_d})}\left[\frac{E(\boldsymbol{x_d},\boldsymbol{\theta},\lambda)}{q(\boldsymbol{x_d})}\right]}$$

Therefore, by sampling from the known distribution q, the expectation can be estimated unbiasedly.

A.6. Oracles to Get T Optimal Integer Solutions

Suppose that we want to solve T optimal integer solutions of the following mixed-integer linear cone program.

$$\begin{gathered} \min_{\boldsymbol{x}_d, \boldsymbol{x}_c, \boldsymbol{v}} \boldsymbol{a}^T \boldsymbol{x}_d + \boldsymbol{b}^T \boldsymbol{x}_c + \boldsymbol{c}^T \boldsymbol{v} \\ \text{s.t. } \boldsymbol{x}_d \in \{0,1\}^{n_d},\ \boldsymbol{x}_c \in \mathbb{R}^{n_c} \\ \boldsymbol{A}_i^T \boldsymbol{x}_d + \boldsymbol{B}_i^T \boldsymbol{x}_c + \boldsymbol{C}_i^T \boldsymbol{v} \leq_{\mathscr{K}_i} \mathbf{0}, \forall i \in [I] \end{gathered} \tag{32}$$

Such mixed-integer linear cone program can be solved by commercial solvers like *gurobi*. We solve this program and get the optimal integer solution $\boldsymbol{x}_d^1$, and then we add the following constraint to problem (32).

$$\left(\mathbf{1} - 2\boldsymbol{x}_d^1\right)^T \boldsymbol{x}_d + \mathbf{1}^T \boldsymbol{x}_d^1 \geq 1 \tag{33}$$

Constraint (33) only cuts out $\boldsymbol{x}_d^1$ from the feasible region. Therefore, by solving (32) with extra constraint (33) we will get the second optimal solution $\boldsymbol{x}_d^2$. Then we cut out $\boldsymbol{x}_d^2$ to get $\boldsymbol{x}_d^3$.

Repeat the above process for T times and we will get T optimal integer solutions.

A.7. Analysis of the Projection Layer

If $g_i, i \in [I]$ in the parameterized SOC ambiguity set are selected as what we give in Appendix A.2, then the epigraph of $g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i)$ with respect to $\boldsymbol{\alpha}_i$, *i.e.*,

$$\{(\boldsymbol{\alpha}_i, u) | \boldsymbol{\alpha}_i \in \mathcal{A}_i, u \geq g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i)\} \tag{34}$$

is also a second-order cone representable set.

By Ben-Tal & Nemirovski (2001) p.91, this representability is preserved by a non-negative sum, so

$$\sum_{n=1}^{N} \omega_n(\boldsymbol{z}) g_i(\boldsymbol{y}_n, \boldsymbol{\alpha}_i) \leq \sigma_i \tag{35}$$

can be expressed by finitely many second-order cone constraints.

B. Proofs

B.1. Proof of Theorem 4.5

When the cost function $c(\boldsymbol{x}, \boldsymbol{y})$ is of the form (ii) in Assumption 4.3, then Theorem 4.5 coincides with Theorem 1 in Bertsimas et al. (2019).

For the case the cost function $c(\boldsymbol{x}, \boldsymbol{y})$ is of the form (i) in Assumption 4.3, the proof is quite similar. Since in this case $c(\boldsymbol{x}, \boldsymbol{y})$ is linear in $\boldsymbol{y}$, according to Theorem 1 in Bertsimas et al. (2019), the worst-case expectation $f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta})) = \max_{\mathbb{P} \in \mathscr{U}(\boldsymbol{\theta})} \mathbb{E}_{\boldsymbol{y} \sim \mathbb{P}}[c(\boldsymbol{x}, \boldsymbol{y})]$ is equivalent to

$$f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta})) = \min_{r, \boldsymbol{\beta}} r + \boldsymbol{\beta}^T \boldsymbol{\sigma} \tag{36}$$

$$\text{s.t. } r \geq c(\boldsymbol{x}, \boldsymbol{y}) - \boldsymbol{\beta}^T \boldsymbol{u}, \ \forall (\boldsymbol{y}, \boldsymbol{u}) \in \mathscr{V}, \boldsymbol{y} \geq \mathbf{0} \tag{37}$$

$$\boldsymbol{\beta} \geq \mathbf{0} \tag{38}$$

where $\mathscr{V}$ is defined in Equation (27).

Since Assumption 4.4 holds, we can leverage the duality theory to reformulate constraint (37) as follows.

$$r \geq \max_{\boldsymbol{y}, \boldsymbol{u} \in \mathscr{V}, \boldsymbol{y} \geq \mathbf{0}} c(\boldsymbol{x}, \boldsymbol{y}) - \boldsymbol{\beta}^T \boldsymbol{u} \iff r \geq \max_{\boldsymbol{y}, \boldsymbol{u} \in \mathscr{V}, \boldsymbol{y} \geq \mathbf{0}} \sum_{k=1}^{K} c_k(\boldsymbol{x}) y_k - \boldsymbol{\beta}^T \boldsymbol{u} \tag{39}$$

$$\iff r \geq \max_{\boldsymbol{y} \geq \mathbf{0}} \min_{\boldsymbol{\eta}_j \geq_{L^{m_j}} \mathbf{0}} \left\{ \sum_{k=1}^{K} c_k(\boldsymbol{x}) y_k - \boldsymbol{\beta}^T \boldsymbol{u} + \sum_{j=1}^{J} \boldsymbol{\eta}_j^T \left(\boldsymbol{A}_j^{\boldsymbol{y}}(\boldsymbol{\theta}) \boldsymbol{y} + \boldsymbol{A}_j^{\boldsymbol{u}}(\boldsymbol{\theta}) \boldsymbol{u} + \boldsymbol{A}_j^{\boldsymbol{v}}(\boldsymbol{\theta}) \boldsymbol{v} - \boldsymbol{b}_j(\boldsymbol{\theta}) \right) \right\} \tag{40}$$

$$\iff r \geq \min_{\boldsymbol{\eta}_j \geq_{L^{m_j}} \mathbf{0}} \max_{\boldsymbol{y} \geq \mathbf{0}} \left\{ \sum_{k=1}^{K} c_k(\boldsymbol{x}) y_k - \boldsymbol{\beta}^T \boldsymbol{u} + \sum_{j=1}^{J} \boldsymbol{\eta}_j^T \left(\boldsymbol{A}_j^{\boldsymbol{y}}(\boldsymbol{\theta}) \boldsymbol{y} + \boldsymbol{A}_j^{\boldsymbol{u}}(\boldsymbol{\theta}) \boldsymbol{u} + \boldsymbol{A}_j^{\boldsymbol{v}}(\boldsymbol{\theta}) \boldsymbol{v} - \boldsymbol{b}_j(\boldsymbol{\theta}) \right) \right\} \tag{41}$$

$$\iff r \geq -\sum_{j=1}^{J} \boldsymbol{\eta}_j^T \boldsymbol{b}_j(\boldsymbol{\theta}), -\sum_{j=1}^{J} (\boldsymbol{A}_j^{\boldsymbol{y}}(\boldsymbol{\theta}))^T \boldsymbol{\eta}_j \geq \begin{bmatrix} c_1(\boldsymbol{x}) \\ \vdots \\ c_K(\boldsymbol{x}) \end{bmatrix}, \sum_{j=1}^{J} (\boldsymbol{A}_j^{\boldsymbol{u}}(\boldsymbol{\theta}))^T \boldsymbol{\eta}_j = \boldsymbol{\beta}, \sum_{j=1}^{J} (\boldsymbol{A}_j^{\boldsymbol{v}}(\boldsymbol{\theta}))^T \boldsymbol{\eta}_j = \mathbf{0}, \boldsymbol{\eta}_j \geq_{L^{m_j}} \mathbf{0} \tag{42}$$

where $(\boldsymbol{A}_j^{\boldsymbol{y}}(\boldsymbol{\theta}), \boldsymbol{A}_j^{\boldsymbol{u}}(\boldsymbol{\theta}), \boldsymbol{A}_j^{\boldsymbol{v}}(\boldsymbol{\theta})) = \boldsymbol{A}_j(\boldsymbol{\theta})$ and $\boldsymbol{b}_j(\boldsymbol{\theta})$ are defined in Equation (28), and (41) hold by duality theory since the Slater's condition holds by Assumption 4.4.

The second term in (42) can be reformulated as

$$-\sum_{j=1}^{J}(\boldsymbol{A}_j^{\boldsymbol{y}}(\boldsymbol{\theta})\boldsymbol{e}_k)^T\boldsymbol{\eta}_j \geq c_k(\boldsymbol{x}),\ \forall k \in [K] \tag{43}$$

where $\boldsymbol{e}_k$ is the vector where the kth element is 1 and other elements are 0. Since by Assumption 4.3 the epigraph of $c_i(\boldsymbol{x})$ is a second-order cone representable set, thus each $-\sum_{j=1}^{J}(\boldsymbol{A}_j^{\boldsymbol{y}}(\boldsymbol{\theta})\boldsymbol{e}_k)^T\boldsymbol{\eta}_j \geq c_k(\boldsymbol{x})$ can be expressed by finitely many second-order cone constraints.

Therefore, Theorem 4.5 holds for both the cases in Assumption 4.3.

B.2. Proof of Theorem 4.6

By Theorem 4.5, the worst-case expectation $f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta}))$ is a linear second-order cone program and all the constraints are linear in $\boldsymbol{x}$, so under continuous assumption of $\boldsymbol{x}$, the DRO problem

$$\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta})) \tag{44}$$

is also a linear second-order cone program.

Specifically, by Appendix B.1, if the cost function $c(\boldsymbol{x}, \boldsymbol{y})$ is of the form (i) in Assumption 4.3, (44) is equivalent to

$$\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta})) = \min_{\boldsymbol{x}\in\mathcal{X}, r, \boldsymbol{\beta}} r + \boldsymbol{\beta}^T\boldsymbol{\sigma} \tag{45}$$

$$\text{s.t. } r \geq -\sum_{j=1}^{J}\boldsymbol{\eta}_j^T\boldsymbol{b}_j(\boldsymbol{\theta}) \tag{46}$$

$$-\sum_{j=1}^{J}(\boldsymbol{A}_j^{\boldsymbol{y}}(\boldsymbol{\theta})\boldsymbol{e}_k)^T\boldsymbol{\eta}_j \geq c_k(\boldsymbol{x}),\ \forall k \in [K] \tag{47}$$

$$-\sum_{j=1}^{J}(\boldsymbol{A}_j^{\boldsymbol{u}}(\boldsymbol{\theta}))^T\boldsymbol{\eta}_j = \boldsymbol{\beta} \tag{48}$$

$$\sum_{j=1}^{J}(\boldsymbol{A}_j^{\boldsymbol{v}}(\boldsymbol{\theta}))^T\boldsymbol{\eta}_j = \boldsymbol{0}, \boldsymbol{\eta}_j \geq_{L^{m_j}} \boldsymbol{0} \tag{49}$$

$$\boldsymbol{\beta} \geq \boldsymbol{0}, \boldsymbol{\eta}_j \geq_{L^{m_j}} \boldsymbol{0}, \forall j \in [J] \tag{50}$$

where constraint (47) is defined in Equation (43) in Appendix B.1 and can be reformulated into finitely many second-order cone constraints that are linear in $\boldsymbol{x}$.

If the cost function $c(\boldsymbol{x}, \boldsymbol{y})$ is of the form (ii) in Assumption 4.3, a similar reformulation can be derived.

The cone programming $\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta}))$ takes $\boldsymbol{A}_j(\boldsymbol{\theta}) = (\boldsymbol{A}_j^{\boldsymbol{y}}(\boldsymbol{\theta}), \boldsymbol{A}_j^{\boldsymbol{u}}(\boldsymbol{\theta}), \boldsymbol{A}_j^{\boldsymbol{v}}(\boldsymbol{\theta}))$ and $\boldsymbol{b}_j(\boldsymbol{\theta})$ as its parameter. Then by Agrawal et al. (2019b), the optimal value $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta}))$ is differentiable with respect to parameter $\boldsymbol{A}_j(\boldsymbol{\theta})$ and $\boldsymbol{b}_j(\boldsymbol{\theta})$. Further, by Definition 4.1, $\boldsymbol{A}_j(\boldsymbol{\theta})$ and $\boldsymbol{b}_j(\boldsymbol{\theta})$ are differentiable with respect to $\boldsymbol{\theta}$.

Therefore, $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}, \mathscr{U}(\boldsymbol{\theta}))$ is differentiable with respect to $\boldsymbol{\theta}$.

B.3. Proof of Theorem 4.13

(i) It suffices to prove that for any sequence $\lambda_n \searrow 0$, $\boldsymbol{\phi} \in \Phi$, and sequence $\boldsymbol{\phi}_n \to \boldsymbol{\phi}$, the following equality holds

$$\lim_{n\to\infty} \mathcal{R}_{\lambda_n}(\boldsymbol{\phi}_n) = \mathcal{R}(\boldsymbol{\phi}) \tag{51}$$

For ease of notation, we define

$$f^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z})) = \min_{\boldsymbol{x}_c\in\mathcal{X}_c(\boldsymbol{x}_d)} f((\boldsymbol{x}_d, \boldsymbol{x}_c), M_{\boldsymbol{\phi}}(\boldsymbol{z})) = f\big((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}})), M_{\boldsymbol{\phi}}(\boldsymbol{z})\big) \tag{52}$$

By Assumption 4.10, $f((\boldsymbol{x}_d, \boldsymbol{x}_c), M_{\boldsymbol{\phi}}(\boldsymbol{z}))$ is continuous in $\boldsymbol{x}_c$ and $\boldsymbol{\phi}$, and by Assumption 4.11, $\boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}))$ is also continuous in $\boldsymbol{\phi}$, so $f^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}))$ is continuous in $\boldsymbol{\phi}$.

If for a pair of $(\boldsymbol{z}, \boldsymbol{\phi})$ the optimal integer solution $\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})) = \arg\min_{\boldsymbol{x}_d \in \mathcal{X}_d} f^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}))$ is unique, then it holds that

$$\min_{\boldsymbol{x}_d \in \mathcal{X}_d / \{\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))\}} f^*\Big(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z})\Big) - f^*\Big(\boldsymbol{x}_d^*, M_{\boldsymbol{\phi}}(\boldsymbol{z})\Big) > 0 \tag{53}$$

Then by the continuity of $f^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}}(\boldsymbol{z}))$ in $\boldsymbol{\phi}$, there exists a $\epsilon > 0$ such that

$$\forall \overline{\boldsymbol{\phi}} \in \left\{ \overline{\boldsymbol{\phi}} \,\middle|\, \|\overline{\boldsymbol{\phi}} - \boldsymbol{\phi}\| \le \epsilon \right\}, \min_{\boldsymbol{x}_d \in \mathcal{X}_d / \{\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))\}} f^*\Big(\boldsymbol{x}_d, M_{\overline{\boldsymbol{\phi}}}(\boldsymbol{z})\Big) - f^*\Big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), M_{\overline{\boldsymbol{\phi}}}(\boldsymbol{z})\Big) \ge \epsilon \tag{54}$$

By the above observation, we further define

$$Y(\boldsymbol{\phi}, N) = \left\{ \boldsymbol{z} \,\middle|\, \min_{\boldsymbol{x}_d \in \mathcal{X}_d / \{\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))\}} f^*\Big(\boldsymbol{x}_d, M_{\overline{\boldsymbol{\phi}}}(\boldsymbol{z})\Big) - f^*\Big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), M_{\overline{\boldsymbol{\phi}}}(\boldsymbol{z})\Big) \ge \frac{1}{N}, \forall \overline{\boldsymbol{\phi}} \in \left\{ \overline{\boldsymbol{\phi}} \,\middle|\, \|\overline{\boldsymbol{\phi}} - \boldsymbol{\phi}\| \le \frac{1}{N} \right\} \right\} \tag{55}$$

It is obvious that $Y(\boldsymbol{\phi}, 1) \subset Y(\boldsymbol{\phi}, 2) \subset \cdots \subset Y(\boldsymbol{\phi}, N) \subset \cdots$.

By the argument concerning (54), we conclude that if inequality (53) holds for a pair of $(\boldsymbol{z}, \boldsymbol{\phi})$, then $\boldsymbol{z} \in Y(\boldsymbol{\phi}, N)$ for sufficiently large N.

Since by Assumption 4.8, for any $\boldsymbol{\phi} \in \Phi$, inequality (53) holds almost surely, thus we have

$$\mathbb{P}\bigg(\bigcup_{N=1}^{\infty} Y(\boldsymbol{\phi}, N) \bigg) = 1 \tag{56}$$

By Assumption 4.10, l is bounded, and we denote this bounded by Ψ.

Therefore, for any $\epsilon > 0$, there exist a $N_{\epsilon/\Psi}$ such that

$$\mathbb{P}(Y(\boldsymbol{\phi}, N_{\epsilon/\Psi})) \ge 1 - \epsilon/\Psi \tag{57}$$

Therefore, when $n \ge N_{\epsilon/\Psi}$, we have

$$
\begin{aligned}
&|\mathcal{R}_{\lambda_n}(\boldsymbol{\phi}_n) - \mathcal{R}(\boldsymbol{\phi})| && (58)\\
&= \bigg| \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}} \bigg[\sum_{\boldsymbol{x}_d \in \mathcal{X}_d} p(\boldsymbol{x}_d | M_{\boldsymbol{\phi}_n}(\boldsymbol{z}), \lambda_n) l((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}_n}(\boldsymbol{z}))), \boldsymbol{y}) \\
&\qquad - l\bigg(\Big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), \boldsymbol{x}_c^*(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), M_{\boldsymbol{\phi}}(\boldsymbol{z})) \Big), \boldsymbol{y} \bigg) \bigg] \bigg| && (59)\\
&\le \bigg| \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}} \bigg\{ \mathbb{1}\big(\boldsymbol{z} \in Y(\boldsymbol{\phi}, N_{\epsilon/\Psi})\big) \bigg[\sum_{\boldsymbol{x}_d \in \mathcal{X}_d} p(\boldsymbol{x}_d | M_{\boldsymbol{\phi}_n}(\boldsymbol{z}), \lambda_n) l((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}_n}(\boldsymbol{z}))), \boldsymbol{y}) \\
&\qquad - l\bigg(\Big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), \boldsymbol{x}_c^*(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), M_{\boldsymbol{\phi}}(\boldsymbol{z})) \Big), \boldsymbol{y} \bigg) \bigg] \bigg\} \bigg| \\
&\quad + \bigg| \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}} \bigg[\mathbb{1}\big(\boldsymbol{z} \notin Y(\boldsymbol{\phi}, N_{\epsilon/\Psi})\big) \sum_{\boldsymbol{x}_d \in \mathcal{X}_d} p(\boldsymbol{x}_d | M_{\boldsymbol{\phi}_n}(\boldsymbol{z}), \lambda_n) l((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}_n}(\boldsymbol{z}))), \boldsymbol{y}) \bigg] \bigg| \\
&\quad + \bigg| \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}} \bigg[\mathbb{1}\big(\boldsymbol{z} \notin Y(\boldsymbol{\phi}, N_{\epsilon/\Psi})\big) l\bigg(\Big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), \boldsymbol{x}_c^*(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), M_{\boldsymbol{\phi}}(\boldsymbol{z})) \Big), \boldsymbol{y} \bigg) \bigg] \bigg| && (60)\\
&\le \bigg| \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}} \bigg\{ \mathbb{1}\big(\boldsymbol{z} \in Y(\boldsymbol{\phi}, N_{\epsilon/\Psi})\big) \bigg[\sum_{\boldsymbol{x}_d \in \mathcal{X}_d} p(\boldsymbol{x}_d | M_{\boldsymbol{\phi}_n}(\boldsymbol{z}), \lambda_n) l((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, M_{\boldsymbol{\phi}_n}(\boldsymbol{z}))), \boldsymbol{y}) \\
&\qquad - l\bigg(\Big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), \boldsymbol{x}_c^*(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})), M_{\boldsymbol{\phi}}(\boldsymbol{z})) \Big), \boldsymbol{y} \bigg) \bigg] \bigg\} \bigg| + 2\frac{\epsilon}{\Psi}\Psi && (61)
\end{aligned}
$$

where $\mathbb{1}$ is the indicator function.

In (61), for $\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))$,

$$p\Big(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))\Big|M_{\boldsymbol{\phi}_n}(\boldsymbol{z}),\lambda_n\Big) = \frac{\exp\left(-\frac{f^*\left(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),M_{\boldsymbol{\phi}_n}(\boldsymbol{z})\right)}{\lambda_n}\right)}{\sum_{\boldsymbol{x_d}'\in\mathcal{X}_d}\exp\left(-\frac{f^*\left(\boldsymbol{x}_d',M_{\boldsymbol{\phi}_n}(\boldsymbol{z})\right)}{\lambda_n}\right)} \tag{62}$$

$$= \frac{1}{1+\sum_{\boldsymbol{x_d}'\in\mathcal{X}_d/\left\{\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))\right\}}\exp\left(-\frac{f^*\left(\boldsymbol{x}_d',M_{\boldsymbol{\phi}_n}(\boldsymbol{z})\right)-f^*\left(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),M_{\boldsymbol{\phi}_n}(\boldsymbol{z})\right)}{\lambda_n}\right)} \tag{63}$$

Since $\boldsymbol{\phi}_n\to\boldsymbol{\phi}$, then for sufficiently large n, $|\boldsymbol{\phi}_n-\boldsymbol{\phi}|\le\frac{1}{N_{\epsilon/\Psi}}$. Therefore, for $\boldsymbol{z}\in Y(\boldsymbol{\phi},N_{\epsilon/\Psi})$,

$$f^*\Big(\boldsymbol{x}_d^{'},M_{\boldsymbol{\phi}_n}(\boldsymbol{z})\Big)-f^*\Big(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),M_{\boldsymbol{\phi}_n}(\boldsymbol{z})\Big)\ge\frac{1}{N_{\epsilon/\Psi}},\forall\boldsymbol{x_d}^{'}\ne\boldsymbol{x_d}^* \tag{64}$$

Therefore,

$$p\Big(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))\Big|M_{\boldsymbol{\phi}_n}(\boldsymbol{z}),\lambda_n\Big)\ge\frac{1}{1+\sum_{\boldsymbol{x_d}'\in\mathcal{X}_d/\left\{\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))\right\}}\exp(-\frac{1}{\lambda_nN_{\epsilon/\Psi}})}=\frac{1}{1+(|\mathcal{X}_d|-1)\exp(-\frac{1}{\lambda_nN_{\epsilon/\Psi}})} \tag{65}$$

Since $\lambda_n\searrow0^+$, we have

$$\lim_{n\to\infty}p\Big(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z}))\Big|M_{\boldsymbol{\phi}_n}(\boldsymbol{z}),\lambda_n\Big)=1 \tag{66}$$

So

$$\lim_{n\to\infty}\sum_{\boldsymbol{x}_d\in\mathcal{X}_d}p(\boldsymbol{x}_d|M_{\boldsymbol{\phi}_n}(\boldsymbol{z}),\lambda_n)l((\boldsymbol{x}_d,\boldsymbol{x}_c^*(\boldsymbol{x}_d,M_{\boldsymbol{\phi}_n}(\boldsymbol{z}))),\boldsymbol{y})=l\Big(\Big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),\boldsymbol{x}_c^*(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),M_{\boldsymbol{\phi}}(\boldsymbol{z}))\Big),\boldsymbol{y}\Big) \tag{67}$$

Or equally

$$\lim_{n\to\infty}\left\{\sum_{\boldsymbol{x}_d\in\mathcal{X}_d}p(\boldsymbol{x}_d|M_{\boldsymbol{\phi}_n}(\boldsymbol{z}),\lambda_n)l((\boldsymbol{x}_d,\boldsymbol{x}_c^*(\boldsymbol{x}_d,M_{\boldsymbol{\phi}_n}(\boldsymbol{z}))),\boldsymbol{y})-l\Big(\Big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),\boldsymbol{x}_c^*(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),M_{\boldsymbol{\phi}}(\boldsymbol{z}))\Big),\boldsymbol{y}\Big)\right\}=0 \tag{68}$$

Since l is bounded, by the bounded convergence theorem, we have

$$\begin{aligned}\lim_{n\to\infty}\Bigg|\mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\Bigg\{\mathbb{1}\big(\boldsymbol{z}\in Y(\boldsymbol{\phi},N_{\epsilon/\Psi})\big)\Bigg[\sum_{\boldsymbol{x}_d\in\mathcal{X}_d}p(\boldsymbol{x}_d|M_{\boldsymbol{\phi}_n}(\boldsymbol{z}),\lambda_n)l((\boldsymbol{x}_d,\boldsymbol{x}_c^*(\boldsymbol{x}_d,M_{\boldsymbol{\phi}_n}(\boldsymbol{z}))),\boldsymbol{y})\\-l\Big(\Big(\boldsymbol{x}_d^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),\boldsymbol{x}_c^*(\boldsymbol{x_d}^*(M_{\boldsymbol{\phi}}(\boldsymbol{z})),M_{\boldsymbol{\phi}}(\boldsymbol{z}))\Big),\boldsymbol{y}\Big)\Bigg]\Bigg\}\Bigg|=0\end{aligned} \tag{69}$$

Therefore,

$$\lim_{n\to\infty}|\mathcal{R}_{\lambda_n}(\boldsymbol{\phi}_n)-\mathcal{R}(\boldsymbol{\phi})|\le2\epsilon \tag{70}$$

Since ϵ can be selected arbitrarily small, we have

$$\lim_{n\to\infty}|\mathcal{R}_{\lambda_n}(\boldsymbol{\phi}_n)-\mathcal{R}(\boldsymbol{\phi})|=0 \tag{71}$$

Therefore, $\mathcal{R}_{\lambda_n}(\boldsymbol{\phi})$ epi-converges to $\mathcal{R}(\boldsymbol{\phi})$ as $n\to\infty$.

(ii) Since $\mathcal{R}_{\lambda_n}(\boldsymbol{\phi})$ epi-converges to $\mathcal{R}(\boldsymbol{\phi})$, (ii) can be immediately derived by applying Proposition 4.6 in Bonnans & Shapiro (2013)

B.4. Proof of Theorem 4.14

We first prove Equation (14). By the chain rule,

$$\frac{\partial r_\lambda(\boldsymbol{\theta}, \boldsymbol{y})}{\partial \boldsymbol{\theta}} = \sum_{\boldsymbol{x_d} \in \mathcal{X}_d} \left\{ \frac{\partial p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda)}{\partial \boldsymbol{\theta}} l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big) + p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda) \frac{\partial l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big)}{\partial \boldsymbol{x_c}^*} \frac{\partial \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\} \tag{72}$$

By Assumption 4.7, the feasible region $\mathcal{X}_c(\boldsymbol{x}_d)$ given $\boldsymbol{x}_d$ is a second-order cone representable set. Therefore, by Theorem 4.6, the optimal continuous solution $\boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})$ is differentiable with respect to $\boldsymbol{\theta}$, so the last gradient term $\frac{\partial \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ in Equation (72) is well-defined.

Since the energy function

$$E(\boldsymbol{x}_d, \boldsymbol{\theta}, \lambda) = \exp\left(-\frac{f\big((\boldsymbol{x}_d, \boldsymbol{x}_c^*(\boldsymbol{x}_d, \boldsymbol{\theta})), \boldsymbol{\theta}\big)}{\lambda}\right) \tag{73}$$

and $\boldsymbol{x}_c^*(\boldsymbol{x}_d, \boldsymbol{\theta}))$ is differentiable, thus $E(\boldsymbol{x}_d, \boldsymbol{\theta}, \lambda)$ is also differentiable with respect to $\boldsymbol{\theta}$.

Therefore, the first gradient term $\frac{\partial p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda)}{\partial \boldsymbol{\theta}}$ in Equation (72) is well-defined since

$$p(\boldsymbol{x}_d|\boldsymbol{\theta}, \lambda) = \frac{E(\boldsymbol{x}_d, \boldsymbol{\theta}, \lambda)}{\sum_{\boldsymbol{x}'_d \in \mathcal{X}_d} E(\boldsymbol{x}'_d, \boldsymbol{\theta}, \lambda)} \tag{74}$$

Let $Z(\boldsymbol{\theta}, \lambda)$ denote the normalizer $\sum_{\boldsymbol{x}'_d \in \mathcal{X}_d} E(\boldsymbol{x}'_d, \boldsymbol{\theta}, \lambda)$. By the chain rule, we have

$$\frac{\partial p(\boldsymbol{x}_d|\boldsymbol{\theta}, \lambda)}{\partial \boldsymbol{\theta}} = \frac{\partial \frac{E(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{\sum_{\boldsymbol{x_d}' \in \mathcal{X}_d} E(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)}}{\partial \boldsymbol{\theta}} = \frac{E'(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{Z(\boldsymbol{\theta}, \lambda)} - \frac{E(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{Z(\boldsymbol{\theta}, \lambda)} \frac{\sum_{\boldsymbol{x_d}'} E'(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)}{Z(\boldsymbol{\theta}, \lambda)} \tag{75}$$

$$= p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda) \frac{E'(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)} - p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda) \left[\sum_{\boldsymbol{x_d}' \in \mathcal{X}_d} p(\boldsymbol{x_d}'|\boldsymbol{\theta}, \lambda) \frac{E'(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)} \right] \tag{76}$$

$$= p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda) \left\{ \frac{E'(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)} - \mathbb{E}_{\boldsymbol{x_d}' \sim p(\boldsymbol{x_d}'|\boldsymbol{\theta}, \lambda)} \left[\frac{E'(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)} \right] \right\} \tag{77}$$

By combining Equation (72) and Equation (77), we have

$$\begin{aligned} \frac{\partial r_\lambda(\boldsymbol{\theta}, \boldsymbol{y})}{\partial \boldsymbol{\theta}} = &\sum_{\boldsymbol{x_d} \in \mathcal{X}_d} p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda) \frac{E'(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)} l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big) \\ &- \mathbb{E}_{\boldsymbol{x_d}' \sim p(\boldsymbol{x_d}'|\boldsymbol{\theta}, \lambda)} \left[\frac{E'(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}', \boldsymbol{\theta}, \lambda)} \right] \left(\sum_{\boldsymbol{x_d} \in \mathcal{X}_d} p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda) l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big) \right) \\ &+ \sum_{\boldsymbol{x_d} \in \mathcal{X}_d} p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda) \frac{\partial l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big)}{\partial \boldsymbol{x_c}^*} \frac{\partial \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \end{aligned} \tag{78}$$

$$\begin{aligned} = &\mathbb{E}_{\boldsymbol{x_d} \sim p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda)} \left[\frac{E'(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)} l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big) \right] \\ &- \mathbb{E}_{\boldsymbol{x_d} \sim p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda)} \left[\frac{E'(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)}{E(\boldsymbol{x_d}, \boldsymbol{\theta}, \lambda)} \right] \mathbb{E}_{\boldsymbol{x_d} \sim p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda)} \left[l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big) \right] \\ &+ \mathbb{E}_{\boldsymbol{x_d} \sim p(\boldsymbol{x_d}|\boldsymbol{\theta}, \lambda)} \left[\frac{\partial l\Big(\big(\boldsymbol{x_d}, \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})\big), \boldsymbol{y}\Big)}{\partial \boldsymbol{x_c}^*} \frac{\partial \boldsymbol{x_c}^*(\boldsymbol{x_d}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right] \end{aligned} \tag{79}$$

Therefore, we have derived Equation (14).

Since $r_\lambda(\boldsymbol{\theta}, \boldsymbol{y}) = r_\lambda(M_{\boldsymbol{\phi}}(\boldsymbol{z}), \boldsymbol{y})$ is bounded by Assumption 4.10, then according to Theorem 9.56 in Shapiro et al. (2021),

$$\frac{\partial \mathcal{R}_\lambda(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}} = \frac{\partial \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\, r_\lambda(M_{\boldsymbol{\phi}}(\boldsymbol{z}), \boldsymbol{y})}{\partial \boldsymbol{\phi}} = \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\left[\frac{\partial r_\lambda(M_{\boldsymbol{\phi}}(\boldsymbol{z}), \boldsymbol{y})}{\partial \boldsymbol{\phi}}\right] = \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\left[\frac{\partial r_\lambda(\boldsymbol{\theta}, \boldsymbol{y})}{\partial \boldsymbol{\theta}}\frac{\partial \boldsymbol{\theta}}{\partial \boldsymbol{\phi}}\right] \tag{80}$$

C. Extension to Wasserstein DRO Layer

Here we present how to extend the proposed DRO Layer to Wasserstein-based DRO with a learnable radius.

In the non-contextual setting, the reference distribution of the Wasserstein ambiguity set is typically set to an empirical distribution of N data points. In the contextual setting, we take the idea in Bertsimas & Kallus (2020) and set the conditional empirical distribution $\widehat{\mathbb{P}}(\boldsymbol{z})$ as a weighted sum of data points, *i.e.*

$$\widehat{\mathbb{P}}(\boldsymbol{z}) = \sum_{n=1}^{N} \omega_n(\boldsymbol{z})\delta_{\boldsymbol{y}_n} \tag{81}$$

where $\delta_{[\cdot]}$ is the Dirac delta function and $\omega_n(\boldsymbol{z}), n \in [N]$ are the weights satisfying $\sum_{n\in[N]} \omega_n(\boldsymbol{z}) = 1$.

Let $\epsilon_\theta(\boldsymbol{z})$ be the learnable radius with parameter θ. Then, the Wasserstein ambiguity set is

$$\mathscr{U}\left(\widehat{\mathbb{P}}(\boldsymbol{z}), \epsilon_\theta(\boldsymbol{z})\right) = \left\{\mathbb{P}\,\middle|\,\mathbb{P}(\Xi) = 1, d_W\left(\widehat{\mathbb{P}}(\boldsymbol{z}), \mathbb{P}\right) \le \epsilon_\theta(\boldsymbol{z})\right\}, \tag{82}$$

and the Wasserstein DRO problem is

$$\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\mathbb{P}\in\mathscr{U}\left(\widehat{\mathbb{P}}(\boldsymbol{z}),\epsilon_\theta(\boldsymbol{z})\right)} \mathbb{E}_{\mathbb{P}}[c(\boldsymbol{x}, \boldsymbol{y})] \tag{WDRO}$$

where $\boldsymbol{x}$ is the mixed-integer decision variable satisfying Assumption 4.7 and $c(\boldsymbol{x}, \boldsymbol{y})$ is the cost function satisfying Assumption 4.3.

As outlined in the paper, the procedure of the proposed DRO Layer is presented as follows.

$$\boldsymbol{z} \xrightarrow{\theta} \mathscr{U}\left(\widehat{\mathbb{P}}(\boldsymbol{z}), \epsilon_\theta(\boldsymbol{z})\right) \longrightarrow \text{Problem (WDRO)} \xrightarrow{\text{Solve (WDRO) for } T \text{ times}} \text{Construct proposal distribution}$$
$$\xrightarrow{\text{Importance sampling}} \text{Compute gradient (14)} \rightarrow \theta \text{ update}$$

Therefore, in order to learn the ambiguity set in a decision-focused style, we only need to ensure that

(i) The problem (WDRO) is a mixed-integer linear cone programming.
(ii) When the integer part of the decision variable (*i.e.*, $\boldsymbol{x}_d$) is fixed, the problem (WDRO) is a linear cone programming.

where condition (i) allows us to use a commercial solver like Gurobi to solve problem (WDRO) for T times so that we can construct the proposal distribution, and condition (ii) allows us to derive the gradient of continuous variables with respect to learnable parameter in computing gradient (14).

In fact, these two conditions are satisfied for Wasserstein DRO, and we formally present this result in the following corollary.

Corollary C.1. *If the mixed-integer decision variable $\boldsymbol{x}$ satisfies Assumption 4.7 and the cost function $c(\boldsymbol{x}, \boldsymbol{y})$ satisfies Assumption 4.8, then condition (i) and (ii) hold for the problem (WDRO).*

Proof. The proof is straightforward by combining techniques in Wasserstein DRO (Mohajerin Esfahani & Kuhn, 2018) and second-order cone programming (Ben-Tal & Nemirovski, 2001).

In fact, since Assumption 4.7 holds, we only need to show that condition (ii) holds when the decision variable $\boldsymbol{x}$ is pure continuous and the feasible region $\mathcal{X}$ is second-order cone representable. For simplicity, we prove the corollary for bilinear cost function

$$c(\boldsymbol{x}, \boldsymbol{y}) = \max_{k\in[K]} \boldsymbol{x}^T \boldsymbol{T}_k \boldsymbol{y} \tag{83}$$

and proof for general cost functions satisfying Assumption 4.3 is quite similar but with heavier notations.

According to Mohajerin Esfahani & Kuhn (2018), the problem (WDRO) can be reformulated as

$$\begin{aligned} \inf_{\boldsymbol{x}\in\mathcal{X},\lambda,s_n,\boldsymbol{\gamma}_{nk}} \quad & \lambda\epsilon_\theta(\boldsymbol{z}) + \sum_{n=1}^{N}\omega_n(\boldsymbol{z})s_n \\ \text{s.t. } s_n \geq & \sup_{\boldsymbol{y}\in\Xi}\left(\boldsymbol{x}^T\boldsymbol{T}_k\boldsymbol{y} - \boldsymbol{\gamma}_{nk}^T\boldsymbol{y}\right) + \boldsymbol{\gamma}_{nk}^T\boldsymbol{y}_n,\ \forall n\in[N],\forall k\in[K] \\ & \|\boldsymbol{\gamma}_{nk}\|_* \leq \lambda,\ \forall n\in[N],\forall k\in[K] \end{aligned} \tag{84}$$

Since the uncertainty support Ξ is a second-order cone representable set, it can be formulated as follows.

$$\Xi = \left\{\boldsymbol{y} \mid \exists\boldsymbol{v} \text{ s.t. } \boldsymbol{A}_j^{\boldsymbol{y}}\boldsymbol{y} + \boldsymbol{A}_j^{\boldsymbol{v}} - \boldsymbol{b}_j \geq_{L^{m_j}} \boldsymbol{0}, \forall j\in[J]\right\} \tag{85}$$

By leveraging expression (85), we can further reformulate (84) as

$$\begin{aligned} \inf_{\boldsymbol{x}\in\mathcal{X},\lambda,s_n,\boldsymbol{\gamma}_{nk},\boldsymbol{\eta}_{nkj}} \quad & \lambda\epsilon_\theta(\boldsymbol{z}) + \sum_{n=1}^{N}\omega_n(\boldsymbol{z})s_n \\ \text{s.t. } s_n \geq & \sum_{j\in[J]} -\boldsymbol{b}_j^T\boldsymbol{\eta}_{nkj},\ \forall n\in[N],\forall k\in[K] \\ \boldsymbol{T}_k^T\boldsymbol{x} - \boldsymbol{\gamma}_{nk} + & \sum_{j\in[J]}\boldsymbol{A}_j^{\boldsymbol{y}T}\boldsymbol{\eta}_{nkj} = \boldsymbol{0},\ \forall n\in[N],\forall k\in[K] \\ & \sum_{j\in[J]}\boldsymbol{A}_j^{\boldsymbol{v}T}\boldsymbol{\eta}_{nkj} = \boldsymbol{0},\ \forall n\in[N],\forall k\in[K] \\ & \boldsymbol{\eta}_{nkj} \geq_{L^{m_j}} \boldsymbol{0},\ \forall n\in[N],\forall k\in[K],\forall j\in[J] \\ & \|\boldsymbol{\gamma}_{nk}\|_* \leq \lambda,\ \forall n\in[N],\forall k\in[K] \end{aligned} \tag{86}$$

Problem (86) is already in the form of a linear cone programming. □

D. Experiment Setup

D.1. Toy Example: Multi-item Newsvendor Problem

We measure the performance by the following percentage optimality gap.

$$\text{Percentage Optimality Gap} = \frac{\mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\ l(\boldsymbol{x}^{\text{learning}}(\boldsymbol{z}),\boldsymbol{y}) - \mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\ l(\boldsymbol{x}^*(\boldsymbol{z}),\boldsymbol{y})}{\mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\ l(\boldsymbol{x}^*(\boldsymbol{z}),\boldsymbol{y})} \tag{87}$$

where $\boldsymbol{x}^{\text{learning}}(\boldsymbol{z})$ is the decision given by the learning method and $\boldsymbol{x}^*(\boldsymbol{z})$ is the optimal decision derived by solving the following problem.

$$\boldsymbol{x}^*(\boldsymbol{z}) = \arg\min_{\boldsymbol{x}\in\mathcal{X}}\mathbb{E}_{\boldsymbol{y}\sim p(\boldsymbol{y}|\boldsymbol{z})}l(\boldsymbol{x},\boldsymbol{y}) \tag{88}$$

In computing the percentage optimality gap, we compute $\mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\ l(\boldsymbol{x}^{\text{learning}}(\boldsymbol{z}),\boldsymbol{y})$ by sample average approximation using 1×10^5 pairs of $(\boldsymbol{z},\boldsymbol{y})$ i.i.d. sampled from $\mathbb{P}$. Since

$$\mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\ l(\boldsymbol{x}^*(\boldsymbol{z}),\boldsymbol{y}) = \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_{\boldsymbol{z}}}\left[\min_{\boldsymbol{x}\in\mathcal{X}}\mathbb{E}_{\boldsymbol{y}\sim p(\boldsymbol{y}|\boldsymbol{z})}\ l(\boldsymbol{x},\boldsymbol{y})\right] \tag{89}$$

we follow a two step approach to compute $\mathbb{E}_{(\boldsymbol{z},\boldsymbol{y})\sim\mathbb{P}}\ l(\boldsymbol{x}^*(\boldsymbol{z}),\boldsymbol{y})$. The outer expectation in Equation (89) is approximated by sample average approximation using 400 $\boldsymbol{z}$ i.i.d. sampled from the marginal distribution $\mathbb{P}_{\boldsymbol{z}}$. The inner stochastic program $\min_{\boldsymbol{x}\in\mathcal{X}}\mathbb{E}_{\boldsymbol{y}\sim p(\boldsymbol{y}|\boldsymbol{z})}\ l(\boldsymbol{x},\boldsymbol{y})$ is also solved by sample average approximation using 200 $\boldsymbol{y}$ i.i.d. sampled from the conditional distribution $p(\boldsymbol{y}|\boldsymbol{z})$.

In experiments, we use k-nearest-neighbors weight function to construct the distribution $\mathbb{Q}_{\boldsymbol{z}}$ (defined in Equation (17)) in the projection layer, *i.e.*,

$$\omega_n(\boldsymbol{z}) = \frac{1}{K},\ \text{if } \boldsymbol{z}_n \text{ is in the k-nearest-neighbor of } \boldsymbol{z}\text{. Else, } \omega_n(\boldsymbol{z}) = 0. \tag{90}$$

In the experiments, we select K as $\frac{N}{20}$, where N is the training data size.

We use neural networks to learn the ambiguity set parameters $\boldsymbol{\mu}_{\mathrm{I}}, \sigma_{\mathrm{I}}, \boldsymbol{\mu}_{\mathrm{II}}, \sigma_{\mathrm{II}}$, the architecture are presented as follows.

$$\boldsymbol{\mu}_{\mathrm{I}}: \ \mathrm{FC}(1,60) \to \mathrm{FC}(60,60) \to \mathrm{FC}(60,4) \tag{91}$$

$$\sigma_{\mathrm{I}}: \ \mathrm{FC}(1,60) \to \mathrm{FC}(60,60) \to \mathrm{FC}(60,1) \tag{92}$$

$$\boldsymbol{\mu}_{\mathrm{II}}: \ \mathrm{FC}(1,60) \to \mathrm{FC}(60,60) \to \mathrm{FC}(60,4) \tag{93}$$

$$\sigma_{\mathrm{II}}: \ \mathrm{FC}(1,60) \to \mathrm{FC}(60,60) \to \mathrm{FC}(60,1) \tag{94}$$

where FC(m_1, m_2) represents full connection layer with m_1 inputs and m_2 outputs.

Since $\boldsymbol{\mu}$ and σ are learned by different neural networks, in the pre-training, we first train the parameter $\boldsymbol{\mu}$ by minimizing $\left\|\mathbb{E}_{\mathbb{Q}_{\boldsymbol{z}}}[g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i)]\right\|$ and then train σ by minimizing $\left\|\mathbb{E}_{\mathbb{Q}_{\boldsymbol{z}}}[g_i(\boldsymbol{y}, \boldsymbol{\alpha}_i)] - \sigma_i\right\|$.

In conducting experiments, we vary the training data size N from 100 to 800, and in each case, 10 runs are conducted. In each case, the size of the validation data set is set to $\frac{N}{5}$. In generating data, we first sample covariate $\boldsymbol{z}$, and then sample $\boldsymbol{y}$ conditioned on $\boldsymbol{z}$.

In the multi-item newsvendor problem (19) with $n = 4$, we set $\boldsymbol{a}^c = (0.25, 0.5, 0.75, 1), \boldsymbol{a}^d = 0.95 \times \boldsymbol{a}^c, \boldsymbol{v} = (10.0, 13.0, 16.0, 19.0), \boldsymbol{b} = (2.0, 4.0, 6.0, 8.0), \boldsymbol{d} = (0.5, 1.0, 1.5, 2.0)$. The covariate $\boldsymbol{z}$ follows the uniform distribution on $[0, 1]$, and conditioned on $\boldsymbol{z}$, we set the distribution of demand $\boldsymbol{y}$ by

$$y_1 \sim \mathcal{N}\left(8 + 6(z-0.2)^2, \frac{1}{1+8|z-0.2|}\right), y_2 \sim \mathcal{N}\left(11 + 6(z-0.4)^2, \frac{1}{1+8|z-0.4|}\right)$$
$$y_3 \sim \mathcal{N}\left(14 + 6(z-0.6)^2, \frac{1}{1+8|z-0.6|}\right), y_4 \sim \mathcal{N}\left(17 + 6(z-0.8)^2, \frac{1}{1+8|z-0.8|}\right)$$

D.2. Portfolio Management Problem

In the experiment on the portfolio management problem, the covariate $\boldsymbol{z}$ is a vector of 5 dimensions, and the returns of n assets are generated by the following non-linear model.

$$\boldsymbol{y} = \boldsymbol{a} + \boldsymbol{B}\boldsymbol{z} + \boldsymbol{C}\boldsymbol{e} + \boldsymbol{D}\mathrm{flat}(\boldsymbol{z}\boldsymbol{z}^T) + \|\boldsymbol{z}\|_1 \mathrm{diag}(\boldsymbol{s})\boldsymbol{g} \tag{95}$$

where matrixes $\boldsymbol{B} \in \mathbb{R}^{n\times 5}, \boldsymbol{C} \in \mathbb{R}^{n\times 3}, \boldsymbol{D} \in \mathbb{R}^{n\times 25}$ and vector $\boldsymbol{s} \in \mathbb{R}^n$ are randomly picked parameters, $\boldsymbol{e} \in \mathbb{R}^3$ and $\boldsymbol{g} \in \mathbb{R}^n$ are random variables independent of covariate $\boldsymbol{z} \in \mathbb{R}^5$, and $\|\cdot\|_1$ is the 1-norm.

We use 2,500 data for training, 500 data for validation, and 1,000 data for testing. The distribution $\mathbb{Q}_{\boldsymbol{z}}$ is also set to k-nearest distribution as in (90), and K is set to 20.

In the gradient estimation, we solve the DRO 3 times to construct (15) and use 4 samples to estimate the gradient term by importance sampling. For the energy parameter λ in (10), we initially set it to 10, and subsequently reduce it by one-third every 30 epochs.

D.2.1. Portfolio Management Problem with Pure Continuous Decisions

We use the following two-layer full-connected neural network to learn the ambiguity set parameter in both our method and the method proposed in Costa & Iyengar (2023).

$$\mathrm{FC}(5,22) \to \mathrm{FC}(22,27) \to \mathrm{FC}(27,m) \tag{96}$$

where m is the dimension of the ambiguity set parameter.

D.2.2. Portfolio Management Problem with Mixed-Integer Decisions

In problem (22), the number of binary decisions is set to $\frac{n}{5}$ where n is the number of assets, and the problem parameter $\boldsymbol{v} \in \mathbb{R}^{n/5}$ is set to $[1/n, \cdots, 1/n]$.

To analyze the impact of the number of constraints in the SOC ambiguity set on the performance of the proposed decision-focused learning, we conduct experiments on the 60-dimensional mixed-integer portfolio management problem and consider the following three types of SOC ambiguity sets.

$$\mathscr{U}_{15}(\boldsymbol{\mu},\boldsymbol{\sigma}) = \left\{ \mathbb{P} \,\middle|\, \begin{array}{c} \mathbb{P}(\Xi) = 1 \\ \mathbb{E}_{\mathbb{P}}\left[\sum_{j=1}^{4} \left\| \boldsymbol{y}_{4(i-1)+j} - \boldsymbol{\mu}_{4(i-1)+j} \right\|_2^2 \right] \leq \boldsymbol{\sigma}_i, \forall i \in [15] \end{array} \right\} \tag{97}$$

$$\mathscr{U}_{30}(\boldsymbol{\mu},\boldsymbol{\sigma}) = \left\{ \mathbb{P} \,\middle|\, \begin{array}{c} \mathbb{P}(\Xi) = 1 \\ \mathbb{E}_{\mathbb{P}}\left[\sum_{j=1}^{2} \left\| \boldsymbol{y}_{2(i-1)+j} - \boldsymbol{\mu}_{2(i-1)+j} \right\|_2^2 \right] \leq \boldsymbol{\sigma}_i, \forall i \in [30] \end{array} \right\} \tag{98}$$

$$\mathscr{U}_{60}(\boldsymbol{\mu},\boldsymbol{\sigma}) = \left\{ \mathbb{P} \,\middle|\, \begin{array}{c} \mathbb{P}(\Xi) = 1 \\ \mathbb{E}_{\mathbb{P}}\left[\left\| \boldsymbol{y}_i - \boldsymbol{\mu}_i \right\|_2^2 \right] \leq \boldsymbol{\sigma}_i, \forall i \in [60] \end{array} \right\} \tag{99}$$

Intuitively, $\mathscr{U}_{15}$ constrains 4 dimensions together and thus leads to 15 constraints, $\mathscr{U}_{30}$ constrains 2 dimensions together and thus leads to 30 constraints, and $\mathscr{U}_{60}$ constrains different dimensions individually and leads to 60 constraints.

E. Discussion of Limitations

The decision-focused learning pipeline we developed in Section 4 can be decomposed into four processes: 1. learning layer; 2. projection layer; 3. solving MICP; 4. DRO Layer.

Processes 2 and 4 are built on Cvxpylayers (Agrawal et al., 2019a), and Process 3 is built on commercial solver Gurobi. Processes 1, 2, and 4 participate in both the forward and backward path, and solving MICP is only involved in the forward path.

To test the scalability of our method, we test the running time for a batch of 100 instances on large-scale problems, and the results are presented in Table 3. The experiments are conducted on a laptop with an i7 CPU and 32G RAM. In Table 3, the running time for solving MICP and DRO Layer pertains to $T = 1$ and $S = 1$, so the total computational time for these two processes should be multiplied by T and S, respectively.

From Table 3, we can see the computational time is very long in high-dimensional problems. However, we note that this computational inefficiency is due to the inefficiency of Cvxpylayers and Gurobi, because both of them are built on **CPU** rather than on GPU.

Table 3. Running time for a batch of 100 instances on problems with different dimensions. (T stands for the number of solutions we derive to construct the proposal distribution, and S stands for the number of sampling to estimate gradient (14).)

Dimension	Forward path				Backward propagation
	learning layer	projection layer	solving MICP ($\times T$)	DRO Layer ($\times S$)	
100	< 1ms	16.5s	8.4s	1.2s	9.8s
200	< 1ms	19.6s	12.8s	1.4s	13.1s
400	< 1ms	44.0s	24.1s	5.8s	90.5s
800	< 1ms	82.9s	47.6s	15.3s	358.7s