Differentiable Approximations of Fair OWA Optimization

My H Dinh [1] **James Kotary** [1] **Ferdinando Fioretto** [1]

Abstract

Decision processes in AI and operations research often involve parametric optimization problems, whose unknown parameters must be predicted from correlated data. In such settings, the Predict-Then-Optimize (PtO) paradigm trains parametric prediction models end-to-end with the subsequent optimization model. This paper extends PtO to handle optimization of the nondifferentiable Ordered Weighted Averaging (OWA) objectives, known for their ability to ensure fair and robust solutions with respect to multiple objectives. By proposing efficient differentiable approximations of OWA optimization, it provides a framework for integrating fair optimization concepts with parametric prediction under uncertainty.

1. Introduction

The *Predict-Then-Optimize* (PtO) framework [1] models decision-making processes as optimization problems with unspecified parameters $\boldsymbol{c}$, which must be estimated by a machine learning (ML) model, given correlated features $\boldsymbol{z}$. An estimation of $\boldsymbol{c}$ completes the problem's specification, whose solution defines a mapping:

$$\boldsymbol{x}^\star(\boldsymbol{c}) = \underset{\boldsymbol{x}\in\mathcal{S}}{\operatorname{argmax}} \quad f(\boldsymbol{x}, \boldsymbol{c}) \tag{1}$$

The goal is to learn a model $\hat{\boldsymbol{c}} = \mathcal{M}_\theta(\boldsymbol{z})$ from observable features $\boldsymbol{z}$, such that the objective value $f(\boldsymbol{x}^\star(\hat{\boldsymbol{c}}), \boldsymbol{c})$ under ground-truth parameters $\boldsymbol{c}$ is maximized on average. This is common in many applications requiring decision-making under uncertainty, like planning the fastest route through a city with unknown traffic delays or predicting optimal power generation schedules based on demand forecasts.

Optimization of multiple objectives is crucial in contexts requiring a balance of competing goals, especially when fairness is essential in fields like energy systems [2], urban planning [3], and multi-objective portfolio optimization [4], [5]. A common approach is using Ordered Weighted Averaging (OWA) [6] to achieve Pareto-optimal solutions that fairly balance each objective. However, optimizing an OWA objective in PtO is challenging due to its nondifferentiability, which prevents backpropagation through $\boldsymbol{x}^\star(\boldsymbol{c})$ within machine learning models trained by gradient descent. To our knowledge, no prior PtO models encounter a nondifferentiable objective, making this challenge novel.

2. Preliminaries

2.1. Fair OWA and its Optimization

The *Ordered Weighted Average* (OWA) operator [6] is used in various decision-making fields to fairly aggregate multiple objective criteria [7]. Let $\mathbf{y} \in \mathbb{R}^m$ be a vector of m distinct criteria, and $\tau : \mathbb{R}^m \to \mathbb{R}^m$ be the sorting map that orders $\mathbf{y}$ in increasing order. For any $\mathbf{w}$ satisfying $\mathbf{w} \in \mathbb{R}^m$, $\sum_i w_i = 1$, and $\mathbf{w} \geq 0$, the OWA aggregation with weights $\mathbf{w}$ is piecewise-linear in $\mathbf{y}$ [8]:

$$\text{OWA}_{\mathbf{w}}(\mathbf{y}) = \mathbf{w}^T \tau(\mathbf{y}), \tag{2}$$

This paper uses its concave version, *Fair OWA* [9], characterized by weights in descending order: $w_1 > \ldots > w_n > 0$.

The following three properties of Fair OWA functions are crucial for fairly optimizing multiple objectives: **(1)** *Impartiality*: Permutations of a utility vector are equivalent solutions. **(2)** *Equitability*: Marginal transfers from a higher value criterion to a lower one increase the OWA aggregated value. **(3)** *Monotonicity*: $\text{OWA}_{\mathbf{w}}(\mathbf{y})$ is an increasing function of each element of $\mathbf{y}$. This ensures that solutions optimizing the OWA objectives are Pareto Efficient, meaning no criterion can be improved without worsening another [8]. Optimization of aggregation functions that possess these properties leads to *equitably efficient solutions*, which satisfy a rigorously defined notion of fairness [10].

2.2. Predict-Then-Optimize Learning

Our problem setting fits within the PtO framework. Generally, a parametric optimization problem (1) models an optimal decision $\boldsymbol{x}^\star(\boldsymbol{c})$ with respect to unknown parameters $\boldsymbol{c}$ drawn from a distribution $\boldsymbol{c} \sim \mathcal{C}$. While the true value of $\boldsymbol{c}$ is

[1]Department of Computer Science, University of Virginia, Charlottesville, USA. Correspondence to: My H Dinh <fqw2tz@virginia.edu>.

Published at the 2^{nd} Differentiable Almost Everything Workshop at the 41^{st} International Conference on Machine Learning, Vienna, Austria. July 2024.

unknown, correlated feature values $\boldsymbol{z} \sim \mathcal{Z}$ can be observed. The goal is to learn a predictive model $\mathcal{M}_\theta : \mathcal{Z} \to \mathcal{C}$ from features $\boldsymbol{z}$ to estimate problem parameters $\hat{\boldsymbol{c}} = \mathcal{M}_\theta(\boldsymbol{z})$, by maximizing the empirical objective value of the resulting solution under ground-truth parameters. That is,

$$\underset{\theta}{\operatorname{argmax}} \quad \mathbb{E}_{(\boldsymbol{z},\boldsymbol{c})\sim\Omega}\, f\left(\boldsymbol{x}^\star(\mathcal{M}_\theta(\boldsymbol{z})), \boldsymbol{c}\right), \tag{3}$$

where Ω represents the joint distribution between $\mathcal{Z}$ and $\mathcal{C}$.

The above training goal is often achieved by maximizing empirical *Decision Quality* as a loss function [1], defined:

$$\mathcal{L}_{DQ}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = f\left(\boldsymbol{x}^\star(\hat{\boldsymbol{c}}), \boldsymbol{c}\right). \tag{4}$$

Gradient descent training of (3) with $\mathcal{L}_{DQ}$ requires a model of gradient $\frac{\partial \mathcal{L}_{DQ}}{\partial \hat{\boldsymbol{c}}}$, either directly or through chain-rule composition $\frac{\partial \mathcal{L}_{DQ}}{\partial \hat{\boldsymbol{c}}} = \frac{\partial \boldsymbol{x}^\star(\hat{\boldsymbol{c}})}{\partial \hat{\boldsymbol{c}}} \cdot \frac{\partial \mathcal{L}_{DQ}}{\partial \boldsymbol{x}^\star}$. When $\boldsymbol{x}^\star$ is not differentiable, as in OWA optimizations, smooth approximations are required, such as those developed in the next section.

3. End-to-End Learning with Fair OWA Optimization

This paper focuses on scenarios where the objective function f is an ordered weighted average of m linear objective functions, each parameterized by a row of a matrix $\boldsymbol{C} \in \mathbb{R}^{m\times n}$ so that $f(\boldsymbol{x}, \boldsymbol{C}) = \mathrm{OWA}_{\boldsymbol{w}}(\boldsymbol{C}\boldsymbol{x})$ and

$$\boldsymbol{x}^\star(\boldsymbol{C}) = \underset{\boldsymbol{x}\in\mathcal{S}}{\operatorname{argmax}} \quad \mathrm{OWA}_{\boldsymbol{w}}(\boldsymbol{C}\boldsymbol{x}). \tag{5}$$

Note that this methodology extends to cases where the OWA objective is combined with additional smooth terms. For simplicity, the exposition primarily focuses on the pure OWA objective as shown in equation (5).

The goal is to learn a prediction model $\hat{\boldsymbol{C}} = \mathcal{M}_\theta(\boldsymbol{z})$ that maximizes decision quality through gradient descent on problem (3), which requires obtaining its gradients w.r.t. $\hat{\boldsymbol{C}}$:

$$\frac{\partial \mathcal{L}_{DQ}(\hat{\boldsymbol{C}}, \boldsymbol{C})}{\partial \hat{\boldsymbol{C}}} = \underbrace{\frac{\partial \boldsymbol{x}^\star}{\partial \hat{\boldsymbol{C}}}}_{\boldsymbol{J}} \cdot \underbrace{\frac{\partial \mathrm{OWA}_{\boldsymbol{w}}(\boldsymbol{C}\boldsymbol{x}^\star)}{\partial \boldsymbol{x}^\star}}_{\boldsymbol{g}}, \tag{6}$$

where $\boldsymbol{x}^\star$ is evaluated at $\hat{\boldsymbol{C}}$. The main strategy involves determining the OWA function's gradient $\boldsymbol{g}$ and then computing $\boldsymbol{J}\boldsymbol{g}$ by backpropagating $\boldsymbol{g}$ through $\boldsymbol{x}^\star$.

While nondifferentiable, the class of OWA functions is *subdifferentiable*, with subgradients as follows:

$$\frac{\partial}{\partial \boldsymbol{y}} \mathrm{OWA}_{\boldsymbol{w}}(\boldsymbol{y}) = \boldsymbol{w}_{(\sigma^{-1})} \tag{7}$$

where σ are the sorting indices on $\boldsymbol{y}$ [11]. Based on this formula, computing an overall subgradient $\boldsymbol{g} = {}^{\partial}/_{\partial \boldsymbol{x}}\, \mathrm{OWA}_{\boldsymbol{w}}(\boldsymbol{C}\boldsymbol{x})$ is a routine application of the chain rule (via automatic differentiation). We apply the differentiable approximations proposed next to enable its backpropagation through OWA optimization. A schematic illustration highlighting the forward and backward steps required for this process is provided in Figure 1.

4. Differentiable Approximate OWA Optimization

This section introduces two differentiable approximations of the OWA optimization mapping (5). Section 4.1 adapts a quadratic smoothing technique [12], [13] for a discontinuous linear programming model of OWA. Then, Section 4.2 presents an efficient alternative by employing OWA's Moreau envelope approximation. To the best of the author's knowledge, this is the first instance of using objective smoothing via the Moreau envelope as an effective technique for approximating nondifferentiable optimization programs in end-to-end learning.

4.1. OWA LP with Quadratic Smoothing

In [8], it's observed that the OWA optimization (5) can have the following LP formulation when $\boldsymbol{x} \in \mathcal{S}$ is linear:

$$\boldsymbol{x}^\star(\boldsymbol{C}) = \operatorname{argmax}_{\boldsymbol{x}\in\mathcal{S},\boldsymbol{y},z} \quad z \tag{8a}$$

$$\text{s.t.:} \quad \boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} \tag{8b}$$

$$z \le \boldsymbol{w}_\tau \cdot \boldsymbol{y} \quad \forall \tau \in \mathcal{P}_m. \tag{8c}$$

This LP problem is typically solvable with a simplex method. However, its constraints (8c) grows factorially as $m!$, where m is the number of criteria aggregated by OWA.

Our first approach to differentiable OWA optimization combines this LP transformation with the smoothing technique of [12], which forms differentiable approximations to linear programs by adding a scaled Euclidean norm term $\epsilon|\boldsymbol{x}|^2$ to the objective function. This results in a continuous mapping $\boldsymbol{x}^\star(\boldsymbol{c}) = \operatorname{argmax}_{\boldsymbol{A}\boldsymbol{x}\le\boldsymbol{b}} \quad \boldsymbol{c}^T\boldsymbol{x} + \epsilon\|\boldsymbol{x}\|^2$, a quadratic program (QP) which can be differentiated implicitly via its KKT conditions as in [14].

Smoothing by the scaled norm of joint variables $\boldsymbol{x}, \boldsymbol{y}, z$ in 8a leads to a differentiable QP approximation, viable when m is small. This optimization can be solved and differentiated using techniques from [14] or a generic differentiable optimization solver such as [15]:

$$\boldsymbol{x}^\star(\boldsymbol{C}) = \underset{\boldsymbol{x}\in\mathcal{S},\boldsymbol{y},z}{\operatorname{argmax}} \quad z + \epsilon\left(\|\boldsymbol{x}\|_2^2 + \|\boldsymbol{y}\|_2^2 + z^2\right) \tag{9a}$$

$$\text{subject to:} \quad (8b), (8c). \tag{9b}$$

While problem (8) does not fit the exact LP form due to its parameterized constraints (8b), the need for quadratic smoothing (9a) is illustrated experimentally in Section 5.1. The main *disadvantage* of this method is poor scalability in the number of criteria m, due to constraints (8c). Another drawback is that the transformed QP is much harder to solve than its original associated LP problems since quadratic smoothing increases the difficulty of an OWA-equivalent LP problem. These drawbacks motivate the next smoothing

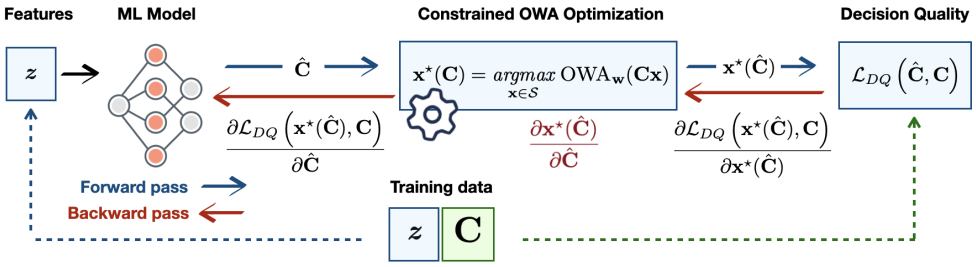

Figure 1: Predict-Then-Optimize for OWA Optimization.

method, which yields a tractable optimization problem by replacing the OWA objective with a smooth approximation.

4.2. Moreau Envelope Smoothing

Instead of adding a quadratic term as in (9), we replace the piecewise linear function $\text{OWA}_{\boldsymbol{w}}$ in (5) with its Moreau envelope, defined for a convex function f as:

$$f^\beta(\mathbf{x}) = \min_{\mathbf{v}} \; f(\mathbf{v}) + \frac{1}{2\beta}\|\mathbf{v}-\mathbf{x}\|^2. \tag{10}$$

Compared to its underlying function f, the Moreau envelope is $\frac{1}{\beta}$ smooth while sharing the same optima [16]. The Moreau envelope-smoothed OWA optimization problem is

$$\boldsymbol{x}^\star(\boldsymbol{C}) = \text{argmax}_{\boldsymbol{x}\in\mathcal{S}} \; \text{OWA}^\beta_{\boldsymbol{w}}(\boldsymbol{Cx}). \tag{11}$$

With its smooth objective function, problem (11) can be solved by gradient-based optimization methods (see Section 5.1), and also differentiated for backpropagation.

Differentiation of (11) is nontrivial since its objective function lacks a closed form. We model its Jacobian by differentiating the fixed-point conditions of a gradient-based solver. To proceed, we first note from [11] that the gradient of the Moreau envelope $\text{OWA}^\beta_{\boldsymbol{w}}$ is equal to a projection:

$$\frac{\partial}{\partial \boldsymbol{x}}\text{OWA}^\beta_{\boldsymbol{w}}(\mathbf{x}) = \text{proj}_{\mathcal{C}(\tilde{\boldsymbol{w}})}\left(\frac{\boldsymbol{x}}{\beta}\right), \tag{12}$$

where $\tilde{\boldsymbol{w}} = -(w_m, \ldots, w_1)$ and the permutahedron $\mathcal{C}(\tilde{\boldsymbol{w}})$ is the convex hull of all permutations of $\tilde{\boldsymbol{w}}$.

The following approach to differentiation of (11) requires differentiation of the function (12). For this, we leverage the differentiable permutahedral projection framework of [17], which was originaly used to implement a soft sorting model. This allows evaluation and differentiation of (12) in $\mathcal{O}(m \log m)$ time, via isotonic regression.

Letting $\mathcal{U}(\boldsymbol{x}, \boldsymbol{C}) = proj_{\mathcal{S}}(\boldsymbol{x} - \alpha \cdot \frac{\partial}{\partial \boldsymbol{x}}\text{OWA}^\beta_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{C}))$, a projected gradient descent step on (11) is $\boldsymbol{x}^{k+1} = \mathcal{U}(\boldsymbol{x}^k, \boldsymbol{C})$. Differentiating the fixed-point conditions of convergence where $\boldsymbol{x}^k = \boldsymbol{x}^{k+1} = \boldsymbol{x}^\star$, and rearranging terms yields a linear system for $\frac{\partial \boldsymbol{x}^\star}{\partial \boldsymbol{C}}$:

$$\left(I - \underbrace{\frac{\partial \mathcal{U}(\boldsymbol{x}^\star, \boldsymbol{C})}{\partial \boldsymbol{x}^\star}}_{\boldsymbol{\Phi}}\right)\frac{\partial \boldsymbol{x}^\star}{\partial \boldsymbol{C}} = \underbrace{\frac{\partial \mathcal{U}(\boldsymbol{x}^\star, \boldsymbol{C})}{\partial \boldsymbol{C}}}_{\boldsymbol{\Psi}} \tag{13}$$

The partial Jacobian matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ above can be found given a differentiable implementation of $\mathcal{U}$. This is achieved by computing the inner gradient $\frac{\partial}{\partial \boldsymbol{x}}\text{OWA}^\beta_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{C})$ via the differentiable permutahedral projection (12), and solving the outer projection mapping $\text{proj}_{\mathcal{S}}$ using a generic differentiable solver such as `cvxpy` [15]. As such, applying $\mathcal{U}$ at a precomputed solution $\boldsymbol{x}^\star(\boldsymbol{C})$ allows $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ to be extracted in PyTorch, in order to solve (13); this process is efficiently implemented via the `fold-opt` library [18].

5. Experiments

This section extends the PtO framework to scenarios with multiple uncertain objectives jointly learned and fairly optimized through OWA aggregation. The *Robust Markowitz Portfolio Optimization* evaluates the differentiable approximations from Section 4 against baseline methods.

The training goal is to maximize empirical decision quality with respect to their Fair OWA aggregation:

$$\mathcal{L}_{DQ}(\hat{\boldsymbol{C}}, \boldsymbol{C}) = \text{OWA}_{\boldsymbol{w}}\left(\boldsymbol{C}\boldsymbol{x}^\star(\hat{\boldsymbol{C}})\right). \tag{14}$$

Evaluations Each model is evaluated based on its ability to train a model $\hat{\boldsymbol{C}} = \mathcal{M}_\theta(z)$ to achieve high decision quality (14) for the OWA-aggregated objective. Results are reported using the *regret* metric of suboptimality, whose 0 corresponds to maximum decision quality:

$$regret(\hat{\boldsymbol{C}}, \boldsymbol{C}) = \text{OWA}^\star_{\boldsymbol{w}}(\boldsymbol{C}) - \text{OWA}_{\boldsymbol{w}}\left(\boldsymbol{C}\boldsymbol{x}^\star(\hat{\boldsymbol{C}})\right) \tag{15}$$

where $\text{OWA}^\star_{\boldsymbol{w}}(\boldsymbol{C})$ is the true optimal value of (5). This experiment evaluates the proposed differentiable approximations (9) and (11), named *OWA-QP* and *OWA-Moreau*. Two common baselines are compared against our methods: **(1)** *Two-stage*: This standard baseline for PtO (3) [1] trains the prediction model $\hat{\boldsymbol{C}} = \mathcal{M}_\theta(\boldsymbol{z})$ by minimizing MSE $\mathcal{L}_{TS}(\hat{\boldsymbol{C}}, \boldsymbol{C}) = \|\hat{\boldsymbol{C}} - \boldsymbol{C}\|^2$ without considering the downstream optimization model, used only at test time. **(2)** *Unweighted sum* (*UWS*): This baseline (Sum-QP) uses an

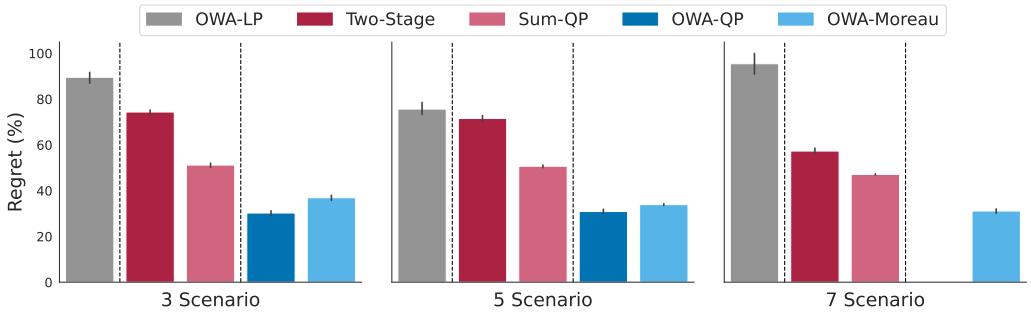

Figure 2: Percentage OWA regret (lower is better) on test set, on robust portfolio problem over 3,5,7 scenarios.

LP model: $\boldsymbol{x}^\star(\boldsymbol{C}) = \operatorname{argmax}_{\boldsymbol{x}\in\mathcal{S}} \mathbf{1}^T(\boldsymbol{C}\boldsymbol{x})$ in end-to-end training, with quadratic smoothing [12] in Section 5.1.

5.1. OWA Optimization Under Uncertainty: Robust Markowitz Portfolio Problem

The classic Markowitz portfolio problem is concerned with constructing an optimal investment portfolio, given future returns $\boldsymbol{c} \in \mathbb{R}^n$ on n assets, which are unknown and predicted from exogenous data. An alternative risk-aware approach considers robustness over scenarios. In [19], m future price scenarios are represented by a matrix $\boldsymbol{C} \in \mathbb{R}^{m\times n}$, where the i^{th} row contains per-asset prices for the i^{th} scenario. Thus, an optimal allocation is modeled as:

$$\boldsymbol{x}^\star(\boldsymbol{C}) = \underset{\boldsymbol{x}\in\Delta_n}{\operatorname{argmax}} \ \ \mathrm{OWA}_{\boldsymbol{w}}(\boldsymbol{C}\boldsymbol{x}). \tag{16}$$

This experiment integrates robust portfolio optimization (16) end-to-end with per-scenario price prediction $\hat{\boldsymbol{C}} = \mathcal{M}_\theta(\boldsymbol{z})$. This experiment's setting is detailed in Appendix B.

Results. Figure 2 shows average percent regret in the OWA objective over the test set (lower is better). The end-to-end training *Sum-QP* outperforms *Two-stage* approach. However, both *OWA-QP* and *OWA-Moreau* achieve substantially higher decision quality. While *OWA-QP* performs slightly better, it cannot scale past 5 scenarios, highlighting the importance of the Moreau envelope smoothing. More results on an alternate dataset can be found in Appendix B.

OWA-LP uses the OWA's equivalent LP as a differentiable optimization without smoothing. Grey bars show non-smoothed OWA LP results implemented with implicit differentiation in `cvxpylayers` [15]. This comparison highlights the accuracy improvement due to quadratic smoothing in OWA-QP. The poor performance of OWA subgradient training under non-smoothed OWA-LP demonstrates the necessity of the proposed approximations in Section 4.

Runtimes of the smoothed models (9) and (11) are compared in Figure 3. Moreau envelope smoothing keep runtimes low as m increases, while the QP approximation suffers past $m = 5$ and encounters memory overflow beyond $m = 6$.

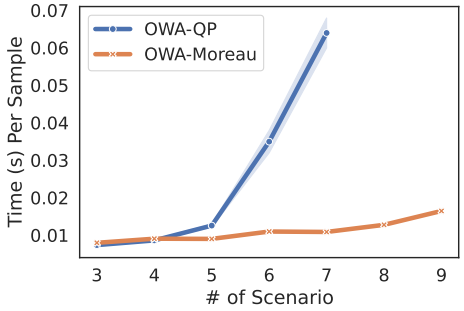

Figure 3: Average solving time of 2 smoothed OWA optimization models, on Robust Portfolio Optimization, over 1000 input samples. Missing data points past 7 scenarios are due to memory overflow as the QP model grows factorially.

6. Related Work

Modern approaches to the Predict-Then-Optimize setting, formalized in Section 2.2, typically maximize decision quality as a loss function, enabled by backpropagation through the mapping $\boldsymbol{c} \to \boldsymbol{x}^\star(\boldsymbol{c})$ defined by (1). When this mapping is differentiable, backpropagation can be performed using differentiable optimization libraries [14], [15], [18], [20]. Otherwise, effective training techniques are typically based on forming continuous approximations of (1), whether by smoothing the objective function [13], [21], [22], introducing random noise [23], [24], or estimation by finite differencing [25]. This paper falls into that category, due to nondifferentiability of the OWA objective, requiring approximation of (1) by differentiable functions. An extended review of related work is provided in Appendix A.

7. Conclusions

This work presents an efficient methodology for integrating Fair OWA optimization with predictive models. This proposal shows the potential of OWA optimization in data-driven decision-making, which has important applications in areas such as risk management and fair resource allocation.

References

[1] J. Mandi, J. Kotary, S. Berden, *et al.*, "Decision-focused learning: Foundations, state of the art, benchmark and future opportunities," *Journal of Artificial Intelligence Research*, vol. TBA, TBA, 2024. DOI: 10.48550/arXiv.2307.13565.

[2] T. Terlouw, T. AlSkaif, C. Bauer, and W. van Sark, "Multi-objective optimization of energy arbitrage in community energy storage systems using different battery technologies," *Applied Energy*, vol. 239, pp. 356–372, 2019, ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2019.01.227. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261919302478.

[3] J. Salas and V. Yepes, "Enhancing sustainability and resilience through multi-level infrastructure planning," *International Journal of Environmental Research and Public Health*, vol. 17, no. 3, p. 962, Feb. 4, 2020. DOI: 10.3390/ijerph17030962.

[4] D. A. Iancu and N. Trichakis, "Fairness and efficiency in multiportfolio optimization," *Operations Research*, vol. 62, no. 6, pp. 1285–1301, 2014. DOI: 10.1287/opre.2014.1310. [Online]. Available: https://doi.org/10.1287/opre.2014.1310.

[5] Y. Chen and A. Zhou, "Multiobjective portfolio optimization via pareto front evolution," *Complex and Intelligent Systems*, vol. 8, pp. 4301–4317, 2022. DOI: 10.1007/s40747-022-00715-8. [Online]. Available: https://doi.org/10.1007/s40747-022-00715-8.

[6] R. R. Yager, "On ordered weighted averaging aggregation operators in multicriteria decisionmaking," in *Readings in Fuzzy Sets for Intelligent Systems*, D. Dubois, H. Prade, and R. R. Yager, Eds., Morgan Kaufmann, 1993, pp. 80–87, ISBN: 978-1-4832-1450-4. DOI: https://doi.org/10.1016/B978-1-4832-1450-4.50011-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9781483214504500110.

[7] R. R. Yager and J. Kacprzyk, *The Ordered Weighted Averaging Operators: Theory and Applications*. Springer Publishing Company, Incorporated, 2012, ISBN: 1461378060.

[8] W. Ogryczak and T. Śliwiński, "On solving linear programs with the ordered weighted averaging objective," *European Journal of Operational Research*, vol. 148, no. 1, pp. 80–91, 2003.

[9] W. Ogryczak, H. Luss, M. Pióro, D. Nace, and A. Tomaszewski, "Fair Optimization and Networks: A Survey," *Journal of Applied Mathematics*, vol. 2014, no. SI08, pp. 1–25, 2014. DOI: 10.1155/2014/612018. [Online]. Available: https://doi.org/10.1155/2014/612018.

[10] M. M. Kostreva and W. Ogryczak, "Linear optimization with multiple equitable criteria," *RAIRO-Operations Research-Recherche Opérationnelle*, vol. 33, no. 3, pp. 275–297, 1999.

[11] V. Do and N. Usunier, "Optimizing generalized gini indices for fairness in rankings," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022, pp. 737–747.

[12] B. Wilder, B. Dilkina, and M. Tambe, "Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization," in *AAAI*, vol. 33, 2019, pp. 1658–1665.

[13] B. Amos, V. Koltun, and J. Z. Kolter, "The limited multi-label projection layer," *arXiv preprint arXiv:1906.08707*, 2019.

[14] B. Amos and J. Z. Kolter, "Optnet: Differentiable optimization as a layer in neural networks," in *ICML*, JMLR. org, 2017, pp. 136–145.

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

[16] A. Beck, *First-order methods in optimization*. SIAM, 2017.

[17] M. Blondel, O. Teboul, Q. Berthet, and J. Djolonga, "Fast differentiable sorting and ranking," in *International Conference on Machine Learning*, PMLR, 2020, pp. 950–959.

[18] J. Kotary, M. H. Dinh, and F. Fioretto, "Backpropagation of unrolled solvers with folded optimization," in *International Joint Conference on Artificial Intelligence*, ijcai.org, 2023, pp. 1963–1970. DOI: 10.24963/ijcai.2023/218. [Online]. Available: https://doi.org/10.24963/ijcai.2023/218.

[19] D. Cajas, "Owa portfolio optimization: A disciplined convex programming framework," *Available at SSRN 3988927*, 2021.

[20] A. Agrawal, S. T. Barratt, S. P. Boyd, E. Busseti, and W. M. Moursi, "Differentiating through a cone program," *Journal of Applied and Numerical Optimization*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:121394814.

[21] B. Wilder, B. Dilkina, and M. Tambe, "Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, vol. 33, 2019, pp. 1658–1665.

[22] J. Mandi and T. Guns, "Interior point solving for lp-based prediction+optimisation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

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

[24] M. Paulus, D. Choi, D. Tarlow, A. Krause, and C. J. Maddison, "Gradient estimation with stochastic softmax tricks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5691–5704, 2020.

[25] M. V. Pogančić, A. Paulus, V. Musil, G. Martius, and M. Rolinek, "Differentiation of blackbox combinatorial solvers," in *International Conference on Learning Representations*, 2019.

[26] J. Kotary, F. Fioretto, P. Van Hentenryck, and B. Wilder, "End-to-end constrained optimization learning: A survey," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021, pp. 4475–4482. DOI: 10.24963/ijcai.2021/610. [Online]. Available: https://doi.org/10.24963/ijcai.2021/610.

[27] S. Gould, B. Fernando, A. Cherian, P. Anderson, R. S. Cruz, and E. Guo, "On differentiating parameterized argmin and argmax problems with application to bi-level optimization," *arXiv preprint arXiv:1607.05447*, 2016.

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

[29] T. Konishi and T. Fukunaga, "End-to-end learning for prediction and optimization with gradient boosting," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 2021, pp. 191–207.

[30] P. Donti, B. Amos, and J. Z. Kolter, "Task-based end-to-end model learning in stochastic optimization," in *NIPS*, 2017, pp. 5484–5494.

[31] A. Nemirovski, "Advances in convex optimization: Conic programming," in *International Congress of Mathematicians*, vol. 1, 2007, pp. 413–444.

[32] E. Busseti, W. M. Moursi, and S. Boyd, "Solution refinement at regular points of conic problems," *Computational Optimization and Applications*, vol. 74, no. 3, pp. 627–643, 2019.

[33] M. C. Grant and S. P. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent advances in learning and control*, Springer, 2008, pp. 95–110.

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

[35] S. Sekhar Sahoo, M. Vlastelica, A. Paulus, V. Musil, V. Kuleshov, and G. Martius, "Gradient backpropagation through combinatorial algorithms: Identity with projection works," *arXiv e-prints*, arXiv–2205, 2022.

[36] J. Kotary, F. Fioretto, P. Van Hentenryck, and Z. Zhu, "End-to-end learning for fair ranking systems," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 3520–3530.

[37] A. N. Elmachtoub and P. Grigas, "Smart "predict, then optimize"," *Management Science*, 2021.

[38] Nasdaq, *Nasdaq end of day us stock prices*, https://data.nasdaq.com/databases/EOD/documentation, Accessed: 2023-08-15, 2022.

[39] A. Martins and R. Astudillo, "From softmax to sparsemax: A sparse model of attention and multi-label classification," in *International conference on machine learning*, PMLR, 2016, pp. 1614–1623.

A. Extended Related Work

Recent literature has been developed around constrained optimization models that are trained end-to-end with machine learning models [26]. In the Predict-Then-Optimize setting, a machine learning model predicts the unknown coefficients of an optimization problem. Then, backpropagation through the optimal solution of the resulting problem allows for end-to-end training of its objective value, under ground-truth coefficients, as a loss function. The primary challenge is backpropagation through the optimization model, for which a variety of alternative techniques have been proposed. Differentiation through constrained argmin problems in the context of machine learning was discussed as early as [27], who proposed first to implicitly differentiate the argmin of a smooth, unconstrained convex function by its first-order optimality conditions, defined when the gradient of the objective function equals zero. This technique is then extended to find approximate derivatives for constrained problems, by applying it to their unconstrained log-barrier approximations. Subsequent approaches applied implicit differentiation to the KKT optimality conditions of constrained problems directly [13], [28], but only on special problem classes such as Quadratic Programs. [29] extend the method of [28], by modeling second-order derivatives of the optimization for training with gradient boosting methods. [30] uses the differentiable quadratic programming solver of [28] to approximately differentiate general convex programs through quadratic surrogate problems. Other problem-specific approaches to analytical differentiation models include ones for sorting and ranking [17], linear programming [22], and convex cone programming [20].

The first general-purpose differentiable optimization solver was proposed in [15], which leverages the fact that any convex program can be converted to a convex cone program [31]. The equivalent cone program is subsequently solved and differentiated following [20], which implicitly differentiates a zero-residual condition representing optimality [32]. A differentiable solver library `cvxpy` is based on this approach, which converts convex programs to convex cone programs by way of their graph implementations as described in [33].

A related line of work concerns end-to-end learning with *discrete* optimization problems, which includes linear programs, mixed-integer programs, and constraint programs. These problem classes often define discontinuous mappings with respect to their input parameters, making their true gradients unhelpful as descent directions in optimization. Accurate end-to-end training can be achieved by *smoothing* the optimization mappings, to produce approximations that yield more useful gradients. A common approach is to augment the objective function with smooth regularizing terms such as Euclidean norm or entropy functions [12], [22], [34]. Others show that similar effects can be produced by applying random noise to the objective [23], [24], or through finite difference approximations [25], [35]. This enables end-to-end learning with discrete structures such as constrained ranking policies [36], shortest paths in graphs [37], and various decision models [12].

B. Portfolio Optimization Experiment

The classic Markowitz portfolio problem is concerned with constructing an optimal investment portfolio, given future returns $\boldsymbol{c} \in \mathbb{R}^n$ on n assets, which are unknown and predicted from exogenous data. A common formulation maximizes future returns subject to a risk limit, modeled as a quadratic covariance constraint. Define the set of valid fractional allocations $\Delta_n = \{\boldsymbol{x} \in \mathbb{R}^n : \mathbf{1}^T\boldsymbol{x} = 1, \boldsymbol{x} \geq 0\}$, then :

$$\boldsymbol{x}^\star(\boldsymbol{c}) = \underset{\boldsymbol{x}\in\Delta_n}{\operatorname{argmax}} \quad \boldsymbol{c}^T\boldsymbol{x} \quad \textit{s.t.:} \quad \boldsymbol{x}^T\boldsymbol{\Sigma}\boldsymbol{x} \leq \delta. \tag{17}$$

where $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$ are the price covariances over n assets. The optimal portfolio allocation (17) as a function of future returns $\boldsymbol{c} \in \mathbb{R}^n$ is differentiable using known methods [15], and is commonly used in evaluation of Predict-Then-Optimize methods [1].

Settings. Historical prices of $n = 50$ assets are obtained from the Nasdaq online database [38] years 2015-2019, and $N = 5000$ baseline asset price samples $\boldsymbol{c}_i$ are generated by adding Gaussian random noise to randomly drawn price vectors. Price scenarios are simulated as a matrix of multiplicative factors uniformly drawn as $\mathcal{U}(0.5, 1.5)^{m\times n}$, whose rows are multiplied elementwise with $\boldsymbol{c}_i$ to obtain $\boldsymbol{C}_i \in \mathbb{R}^{m\times n}$. While future asset prices can be predicted on the basis of various exogenous data including past prices or sentiment analysis, this experiment generates feature vectors $\boldsymbol{z}_i$ using a randomly generated nonlinear feature mapping. The experiment is replicated in three settings which assume $m = 3$, 5, and 7 scenarios.

Two sets of stocks were selected to generate two different datasets based on their average returns across observations. The first set consists of assets from the index with average returns within the 25th to 50th quantile range, while the second set includes assets from the 75th quantile.

Table 1: Hyperparameters

Hyperparameter	Min	Max	Final Value					
			OWA-LP	Two-Stage	Sum-QP	OWA-QP	OWA-Moreau	Sur-QP
learning rate	$1e^{-3}$	$1e^{-1}$	$\mathbf{1e^{-2}}$	$\mathbf{5e^{-3}}$	$\mathbf{1e^{-2}}$	$\mathbf{1e^{-2}}$	$\mathbf{1e^{-2}}$	$\mathbf{1e^{-2}}$
smoothing parameter ϵ	0.1	1.0	**N/A**	**N/A**	1.0	1.0	**N/A**	1.0
smoothing parameter β_0	0.005	10.0	**N/A**	**N/A**	**N/A**	**N/A**	0.05	**N/A**
MSE loss weight λ	0.1	0.5	0.4	**N/A**	0.3	0.4	0.1	0.3

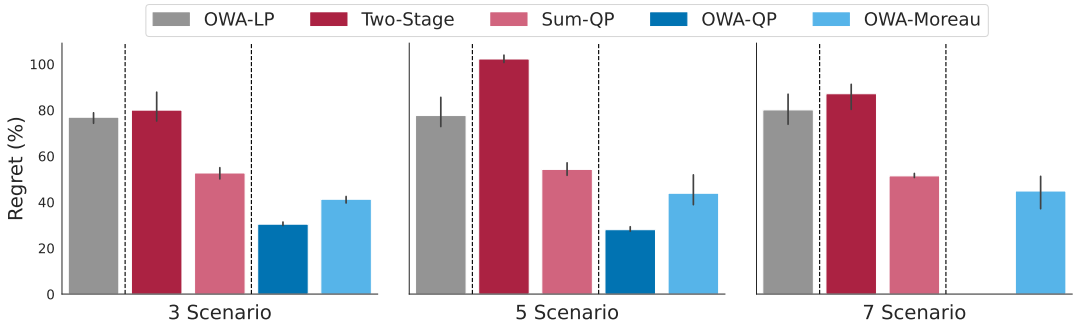

Figure 4: Percentage OWA regret (lower is better) on test set, on robust portfolio problem over 3,5,7 scenarios.

The predictive model $\mathcal{M}_\theta$ is a feedforward neural network with three shared hidden layers followed by one separated hidden layer for each species that is trained using Adam Optimizer and with a batch size of 64. The size of each shared layer is halved, and the output dimension of the separated layer is equal to the number of assets. Hyperparameters were selected as the best-performing on average among those listed in Table 1). Results for each hyperparameter setting are averaged over five random seeds. In the OWA-Moreau model, the forward pass is executed using projected gradient descent for 300, 500, and 750 iterations, respectively, for scenarios with 3, 5, and 7 inputs. The update step size is set to $\gamma = 0.02$.

At test time, $\mathcal{M}_\theta$ is evaluated over a test set for the distribution $(\boldsymbol{z}, \boldsymbol{C}) \in \Omega$, by passing its predictions to a projected subgradient solver of (16).

B.1. Additional Results

Figure 4 and 2 display models' performance on datasets generated from assets with average returns in the 75th quantile and within the 25th-50th percentiles, respectively. The y-axis represents the percentage of regret based on optimal OWA values. A consistent trend is observed in both datasets: end-to-end approaches tend to outperform two-stage approaches. Additionally, our proposed frameworks (*OWA-QP* and *OWA-Moreau*) perform better than *Sum-QP*, with improvements ranging from 5-30%. *OWA-QP* performs better when the number of scenarios is small but struggles to scale beyond 6 scenarios.

B.2. Effect of adding MSE loss

Figure 5 illustrates the impact of combining the Mean Squared Error loss $\mathcal{L}_{MSE}$ in a weighted combination with the decision quality loss $\mathcal{L}_{DQ}$. With the exception of OWA-LP, which exhibited instability, and Two-Stage, already trained with MSE Loss, the addition of MSE resulted in slight enhancements to the regret performance.

B.3. Solution Methods

The OWA portfolio optimization problem (16) is solved at test time, for each compared method, by projected subgradient descent using OWA subgradients (7) and an efficient projection onto the unit simplex Δ as in [39]:

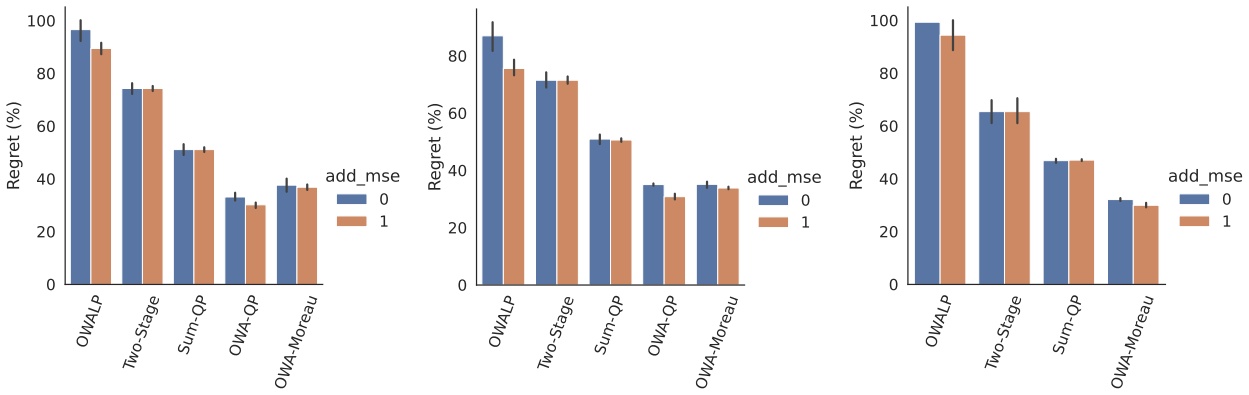

Figure 5: Effect of MSE Loss on differentiable optimization models. From left to right: 3, 5, 7 scenarios

$$\boldsymbol{x}^{k+1} = \text{proj}_{\Delta}\left(\boldsymbol{x}^{k} - \alpha \frac{\partial}{\partial \boldsymbol{x}} \text{OWA}_{\boldsymbol{w}}(\boldsymbol{C}\boldsymbol{x})\right) \tag{18}$$

For the Moreau-envelope smoothed OWA optimization (11) proposed for end-to-end training, the main difference is that its objective function is differentiable (with gradients (12)), which allows solution by a more efficient Frank-Wolfe method [16], whose inner optimization over Δ reduces to the simple argmax function which returns a binary vector with unit value in the highest vector position and 0 elsewhere, which can be computed in linear time:

$$\boldsymbol{x}^{k+1} = \frac{k}{k+2}\boldsymbol{x}^{k} + \frac{2}{k+2}\,\text{argmax}\left(\frac{\partial}{\partial \boldsymbol{x}} \text{OWA}_{\boldsymbol{w}}(\boldsymbol{C}\boldsymbol{x}^{k})\right) \tag{19}$$