SOREL: A STOCHASTIC ALGORITHM FOR SPECTRAL RISKS MINIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

The spectral risk has wide applications in machine learning, especially in real-world decision-making, where people are concerned with more than just average model performance. By assigning different weights to the losses of different sample points, rather than the same weights as in the empirical risk, it allows the model's performance to lie between the average performance and the worst-case performance. In this paper, we propose SOREL, the first stochastic gradient-based algorithm with convergence guarantees for spectral risks minimization. Previous approaches often rely on smoothing the spectral risk by adding a strongly concave function, thereby lacking convergence guarantees for the original spectral risk. We theoretically prove that our algorithm achieves a near-optimal rate of $\widetilde{O}(1/\sqrt{\epsilon})$ to obtain an ϵ-optimal solution in terms ϵ. Experiments on real datasets show that our algorithm outperforms existing ones in most cases, both in terms of runtime and sample complexity.

1 INTRODUCTION

In modern machine learning, model training heavily relies on minimizing the empirical risk. This ensures that the model have high average performance. Given a training set of n sample points $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^n \subset \mathcal{X}$, the goal of the empirical risk minimization is to solve

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} R(\boldsymbol{w}) = (1/n)\sum_{i=1}^n \ell_i(\boldsymbol{w}).$$

Here, $\boldsymbol{w} \in \mathbb{R}^d$ is the parameter vector of the model, $\ell_i(\boldsymbol{w}) = \ell(\boldsymbol{w}, \boldsymbol{x}_i)$ is the loss of the i-th sample, and $\ell : \mathbb{R}^d \times \mathcal{X} \to \mathbb{R}$ is the loss function. However, as machine learning models are deployed in real-world scenarios, the evaluation metrics for these models become more diverse, including factors such as fairness or risk aversion.

In this paper, we consider a generalized aggregation loss function: the spectral risk, which is of the form

$$R_{\boldsymbol{\sigma}}(\boldsymbol{w}) = \sum_{i=1}^n \sigma_i \ell_{[i]}(\boldsymbol{w}).$$

Here $\ell_{[1]}(\cdot) \leq \cdots \leq \ell_{[n]}(\cdot)$ denotes the order statistics of the empirical loss distribution, and $0 \leq \sigma_1 \leq \cdots \leq \sigma_n, \sum_{i=1}^n \sigma_i = 1$. In form, the spectral risk penalizes the occurrence of extreme losses by assigning greater weights to extreme losses. When $\sigma_i = 1/n$, the spectral risk reduces to the empirical risk. When $\sigma_n = 1$ and $\sigma_i = 0$ for $i = 1, \ldots, n-1$, the spectral risk becomes the maximum loss. Therefore, the spectral risk measures the model's performance between the average case and the worst case. By assigning different values to σ_i, the spectral risk encompasses a wide range of aggregated loss functions that have broad applications in fields such as machine learning and finance. Common spectral risks include Conditional Value at Risk (CVaR) or the average of top-k loss (Artzner, 1997; Rockafellar & Uryasev, 2000), Exponential Spectral Risk Measure (ESRM) (Cotter & Dowd, 2006), and Extremal Spectral Risk (Extremile) (Daouia et al., 2019). Their specific forms are shown in Table 1 (Mehta et al., 2022).

Despite the broad applications of spectral risks, optimization methods for spectral risks are still limited. In particular, for large-scale optimization problems, there is currently a lack of stochastic

Table 1: Different spectral risks and the corresponding weights.

Spectral Risks	**Parameter**	σ_i
α-CVaR	$0 < \alpha < 1$	$\begin{cases} \frac{1}{n\alpha}, & i > \lceil n(1-\alpha) \rceil \\ 1 - \frac{\lfloor n\alpha \rfloor}{n\alpha}, & \lfloor n(1-\alpha) \rfloor < i < \lceil n(1-\alpha) \rceil \\ 0, & \text{otherwise} \end{cases}$
ρ-ESRM	$\rho > 0$	$e^{-\rho}\left(e^{\rho\frac{i}{n}} - e^{\rho\frac{i-1}{n}}\right) / \left(1 - e^{-\rho}\right)$
r-Extremile	$r \geq 1$	$\left(\frac{i}{n}\right)^r - \left(\frac{i-1}{n}\right)^r$

algorithms with convergence guarantees for the spectral risk minimization. Indeed, the weight of each sample point depends on the entire training set, introducing complex dependencies and thus making the optimization process challenging. Existing algorithms either abandon the convergence guarantee to the minimum of the spectral risk problem due to the difficulty of obtaining unbiased subgradient estimates (Levy et al., 2020; Kawaguchi & Lu, 2020), or turn to optimize the smooth regularized spectral risk (Mehta et al., 2024; 2022), which lacks convergence guarantees for the original spectral risk. Given the widespread application of the spectral risk in machine learning and the lack of stochastic algorithms for the spectral risk minimization, we are committed to developing stochastic algorithms with convergence guarantees for the spectral risk minimization.

Our Contributions. In this paper, we propose the **S**tochastic **O**ptimizer for Spectral **R**isks minimization with traj**e**ctory Stabi**l**ization (SOREL). i) We propose SOREL, the first stochastic algorithm with convergence guarantees for the spectral risk minimization problem. In particular, SOREL stabilizes the trajectory of the primal variable to the optimal point. ii) Theoretically, we prove that SOREL achieves a near-optimal rate of $\widetilde{O}(1/\sqrt{\epsilon})$ to obtain an ϵ-optimal solution in terms of the squared distance to the optimal point ϵ for spectral risks with a strongly convex regularizer. This matches the known lower bound of $\Omega(1/\sqrt{\epsilon})$ in the deterministic setting (Ouyang & Xu, 2021). iii) Experimentally, SOREL outperforms existing baselines in most cases, both in terms of runtime and sample complexity.

2 Related Work

Statistical Properties of the Spectral Risk. As a type of risk measures, the spectral risk assigns higher weights to the tail distribution and has been profoundly studied in the financial field (Artzner et al., 1999; Rockafellar & Uryasev, 2013; He et al., 2022). Recently, statistical properties of the spectral risk have been investigated by many works in the field of learning theory. Specifically, Mehta et al. (2022); A. & Bhat (2022) demonstrate that the discrete form of spectral risks converges to the spectral risk of the overall distribution, controlled by the Wasserstein distance. Holland & Haress (2022); Khim et al. (2020); Holland & Haress (2021) also consider the learning bounds of spectral risks.

Applications. The spectral risk is widely applied in various fields of finance and machine learning. In some real-world tasks, the worst-case loss is as important as the average-case loss, such as medical imaging (Xu et al., 2022) or robotics (Sharma et al., 2020). The spectral risk minimization can be viewed as a risk-averse learning strategy. In the domain of fair machine learning, different subgroups are classified by sensitive features (e.g., gender and race). Subgroups with higher losses may be treated unfairly in decision-making. By penalizing samples with higher losses, the model's performance across different subgroups can meet certain fairness criteria (Williamson & Menon, 2019), such as demographic parity (Dwork et al., 2012) and equalized odds (Hardt et al., 2016). In the field of distributionally robust optimization, the sample distribution may deviate from a uniform distribution, which can be modeled by reweighting the samples (Chen & Paschalidis, 2020). Mehta et al. (2024) adopts the spectral risk measures as the uncertainty set of the shifted distribution, which is similar to the form of the spectral risk minimization.

In practical applications, people can choose different types of spectral risks based on actual needs. For example, CVaR is popular in the context of portfolio optimization (Rockafellar & Uryasev, 2000), as well as reinforcement learning (Zhang et al., 2024; Chow et al., 2017). Levy et al. (2020) uses the CVaR measure as the uncertainty set in distributionally robust optimization, and their optimization problem is the same as the spectral risk minimization problem that uses CVaR as the spectral risk. Other applications of spectral risks include reducing test errors and mitigating the impact of outliers (Maurer et al., 2021; Kawaguchi & Lu, 2020; Fan et al., 2017), to name a few.

Existing Optimization Methods. There have been many algorithms to optimize CVaR, a special case of spectral risks, including both deterministic (Rockafellar & Uryasev, 2000) and stochastic algorithms (Fan et al., 2017; Curi et al., 2020). For the spectral risk, deterministic methods such as subgradient methods have convergence guarantees, although they are considered algorithms with slow convergence rate. Xiao et al. (2023) propose an Alternating Direction Method of Multipliers (ADMM) type method for the minimization of the rank-based loss, inspired by Cui et al. (2024). Other deterministic methods reformulate this problem into a minimax problem (Thekumparampil et al., 2019; Hamedani & Aybat, 2021; Khalafi & Boob, 2023). However, these methods require calculating $O(n)$ function values and gradients at each iteration, posing significant limitations when solving large-scale problems.

Stochastic algorithms for solving the spectral risk minimization problems are still limited. Some algorithms forgo convergence to the true minimum of the spectral risk (Levy et al., 2020; Kawaguchi & Lu, 2020). Other methods modify the objective to minimize a smooth approximation of the spectral risk by adding a strongly concave term with a coefficient ν (Mehta et al., 2022; 2024). The smaller ν is, the closer the approximation is to the original spectral risk. Mehta et al. (2024) propose the Prospect algorithm and prove that it achieves linear convergence for any $\nu > 0$. Furthermore, if the loss of each sample is different at the optimal point, then the optimal value of the smooth approximation of the spectral risk is the same as the optimal value of the original spectral risk as long as ν is below a certain positive threshold. However, in practice, these conditions are difficult to verify. The convergence of these algorithms still lacks guarantees for original spectral risks minimization. Other stochastic algorithms, including Hamedani & Jalilzadeh (2023); Yan et al. (2019), designed for solving general minimax problems, have a slower convergence rate of $O(1/\epsilon)$ in terms of ϵ to obtain an ϵ-optimal solution. In this paper, we propose SOREL for the original spectral risk minimization problems and achieve a near-optimal convergence rate in terms of ϵ.

3 Algorithm

We consider stochastic optimization of the spectral risk combined with a strongly convex regularizer:

$$\min_{\boldsymbol{w}} \underbrace{\sum_{i=1}^{n} \sigma_i \ell_{[i]}(\boldsymbol{w})}_{R_{\boldsymbol{\sigma}}(\boldsymbol{w})} + g(\boldsymbol{w}). \tag{1}$$

Firstly, we make the basic assumption about the individual loss function ℓ_i and the regularizer g.

Assumption 1 *The individual loss function $\ell_i : \mathbb{R}^d \to \mathbb{R}$ is convex, G-Lipschitz continuous and L-smooth for all $i \in \{1, \ldots, n\}$. The regularizer $g : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$ is proper, lower semicontinuous and μ-strongly convex.*

Assumption 1 is a standard assumption in the literature on stochastic optimization (Nemirovski et al., 2009; Davis & Drusvyatskiy, 2019), especially in the field of the spectral risk minimization (Kawaguchi & Lu, 2020; Holland & Haress, 2022; Levy et al., 2020; Mehta et al., 2022; 2024). The logistic loss satisfies this assumption. The least-square loss satisfies this assumption as long as the iterative sequence lies in a bounded sublevel set. The assumption of strong convexity of g is very common, for example, the l_2 regularization is widely used in machine learning.

3.1 Challenges of Stochastic Optimization for Spectral Risks

In this section, we describe the challenges of the spectral risk minimization problem and the techniques to solve them.

Biases of Stochastic Subgradient Estimators. From convex analysis (Wang et al., 2023, Lemma 10), we know that the subgradient of $R_{\boldsymbol{\sigma}}$ is

$$\partial R_{\boldsymbol{\sigma}}(\boldsymbol{w}) = \mathrm{Conv}\left\{\bigcup_{\pi}\left\{\sum_{i=1}^{n}\sigma_i\nabla\ell_{\pi(i)}(\boldsymbol{w}) : \ell_{\pi(1)}(\boldsymbol{w}) \leq \cdots \leq \ell_{\pi(n)}(\boldsymbol{w})\right\}\right\},$$

where Conv denotes the convex hull of a set, and π is a permutation that arranges $\ell_1, \ldots, \ell_n$ in ascending order. Note that $R_{\boldsymbol{\sigma}}(\boldsymbol{w})$ is non-smooth. Indeed, when there exist $i \neq j$ such that $\ell_i(\boldsymbol{w}) = \ell_j(\boldsymbol{w})$, $\partial R_{\boldsymbol{\sigma}}(\boldsymbol{w})$ contains multiple elements.

The subgradient of $R_{\boldsymbol{\sigma}}$ is related to the ordering of $\ell_1, ..., \ell_n$. We cannot obtain an unbiased subgradient estimator of $\partial R_{\boldsymbol{\sigma}}$ if we use only a mini-batch with m ($m < n$) sample points. For example, when $m = 1$, we randomly sample i uniformly from $\{1, \ldots, n\}$. The subgradient estimator $\nabla\ell_i(\boldsymbol{w})$ is unbiased only if $\sigma_i = 1/n$. For general $\boldsymbol{\sigma}$, unfortunately, to obtain an unbiased subgradient estimator of $\partial R_{\boldsymbol{\sigma}}$, we have to compute n loss function values and then determine the ranking of ℓ_i among the n losses (or the weight corresponding to the i-th sample point). However, computing $O(n)$ losses at each step is computationally heavy. To remedy this, we next design an algorithm that first uses a minimax reformulation of Problem (1) and then alternately updates the weights of each sample point and $\boldsymbol{w}$ using a primal-dual method.

Equivalently, we can rewrite $R_{\boldsymbol{\sigma}}(\boldsymbol{w})$ in the following form

$$R_{\boldsymbol{\sigma}}(\boldsymbol{w}) = \max_{\boldsymbol{\lambda}\in\Pi_{\boldsymbol{\sigma}}}\sum_{i=1}^{n}\lambda_i\ell_i(\boldsymbol{w}), \tag{2}$$

where $\Pi_{\boldsymbol{\sigma}} = \{\Pi\boldsymbol{\sigma} : \Pi\mathbf{1} = \mathbf{1}, \Pi^\top\mathbf{1} = \mathbf{1}, \Pi \in [0,1]^{n\times n}\}$ is the permutahedron associated with $\boldsymbol{\sigma}$, i.e., the convex hull of all permutations of $\boldsymbol{\sigma}$, and $\mathbf{1}$ is the all-one vector (Blondel et al., 2020). Then Problem (1) can be rewritten as

$$\min_{\boldsymbol{w}}\max_{\boldsymbol{\lambda}\in\Pi_{\boldsymbol{\sigma}}} L(\boldsymbol{w}, \boldsymbol{\lambda}) = \sum_{i=1}^{n}\lambda_i\ell_i(\boldsymbol{w}) + g(\boldsymbol{w}). \tag{3}$$

Next, we use a primal-dual method to solve Problem (3). Specifically, we iteratively update $\boldsymbol{w}$ and $\boldsymbol{\lambda}$:

$$\boldsymbol{\lambda}_{k+1} = \arg\max_{\boldsymbol{\lambda}\in\Pi_{\boldsymbol{\sigma}}}\sum_{i=1}^{n}\lambda_i\ell_i(\boldsymbol{w}_k) - \frac{1}{2\eta_k}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_k\|^2, \tag{4}$$

$$\boldsymbol{w}_{k+1} = \arg\min_{\boldsymbol{w}} P_k(\boldsymbol{w}) := \sum_{i=1}^{n}\lambda_{i,k+1}\ell_i(\boldsymbol{w}) + g(\boldsymbol{w}) + \frac{1}{2\tau_k}\|\boldsymbol{w} - \boldsymbol{w}_k\|^2. \tag{5}$$

Steps (4) and (5) can be seen as alternatingly solving the min problem and the max problem in (3) with proximal terms.

Stabilizing the Optimization Trajectory. To update $\boldsymbol{\lambda}_{k+1}$, one may naturally think of solving Problem (2): $\boldsymbol{\lambda}_{k+1} = \arg\max_{\boldsymbol{\lambda}\in\Pi_{\boldsymbol{\sigma}}}\sum_{i=1}^{n}\lambda_i\ell_i(\boldsymbol{w}_k)$, similar to methods in Mehta et al. (2022; 2024) with smoothing coefficient $\nu = 0$. However, since Problem (2) is merely convex, the solution $\boldsymbol{\lambda}$ lacks continuity with respect to $\boldsymbol{w}$, that is, a small change in $\boldsymbol{w}$ could lead to a large change in $\boldsymbol{\lambda}$. Indeed, it is often the case that there are multiple optimal solutions for (2) when there exists $i \neq j$ such that $\ell_i(\boldsymbol{w}) = \ell_j(\boldsymbol{w})$, and in this case, an arbitrary small perturbation of $\boldsymbol{w}$ will lead to a different value of λ_i. As shown in Figure 1, this can cause $\boldsymbol{w}$ to oscillate near points where some losses are the same and prevents the convergence of the algorithm. We also provide a toy example in Appendix C to further illustrate this difficulty. Therefore, the proximal term $\frac{1}{2\eta_k}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_k\|^2$ is added in (4) to prevent excessive changes in $\boldsymbol{\lambda}$ and stabilize the trajectory of the primal variable, where $\eta_k > 0$ controls the extent of its variation.

Stochastic Optimization for the Primal Variable. We use a stochastic algorithm to approximately solve (5). Through the minimax reformulation in (5), we avoid directly calculating the stochastic subgradient of $R_{\boldsymbol{\sigma}}(\boldsymbol{w})$, which requires computing all loss function values to obtain the

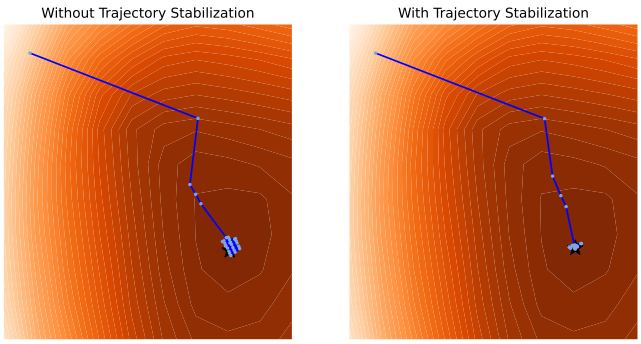

Figure 1: The level set plot of 2D least-square regression with primal-dual optimization trajectories described in Section 3.1. The max subproblem does not have a proximal term (**left**) or has a proximal term (**right**). The min subproblem does not have a proximal term. The black star represents the optimal point. The sample points are obtained by projecting the `yacht` dataset onto $\mathbb{R}^2$ using PCA.

corresponding sample weight λ_i. Additionally, since $\boldsymbol{\lambda}_{k+1}$ is fixed, the finite sum part of the objective function in (5) is smooth, allowing us to use variance reduction (VR), a commonly used technique in stochastic optimization (Shalev-Shwartz & Zhang, 2013; Roux et al., 2012; Johnson & Zhang, 2013; Defazio et al., 2014), to accelerate our stochastic algorithm. In contrast, since $R_{\boldsymbol{\sigma}}(\boldsymbol{w})$ is non-smooth, as previously mentioned, VR cannot be used to directly solve Problem (1). For smooth convex functions in the form of the finite sum, many methods such as SVRG (Johnson & Zhang, 2013), SAGA (Defazio et al., 2014), and SARAH (Nguyen et al., 2017) can enable stochastic methods to achieve the convergence rate of deterministic methods. We apply the proximal stochastic gradient descent with a generalized VR method inspired by SVRG to approximately solve (5), which will be presented in Section 3.2 in detail. Thanks to its strong convexity, Problem (5) can be solved efficiently.

Similar to (4), we add a proximal term $\frac{1}{2\tau_k}\|\boldsymbol{w}-\boldsymbol{w}^k\|^2$ in (5) where $\tau_k > 0$ is the proximal parameter. The proximal parameter τ_k is crucial for the convergence proof of our algorithm. By carefully choosing $\tau_k = O(1/k)$, the updates of $\boldsymbol{w}$ become more stringent as the algorithm progresses, and SOREL can achieve a near optimal rate of $\widetilde{O}(1/\sqrt{\epsilon})$ in terms of ϵ.

3.2 The SOREL Algorithm

Our proposed algorithm SOREL is summarized in Algorithm 1. The specific values for the parameters θ_k, η_k, τ_k and m_k in Algorithm 1 will be given in Section 4. In Line 2 the algorithm initializes $\boldsymbol{\lambda}_0$ by solving Problem (2). In Lines 8-15, the algorithm computes the stochastic gradient and update $\boldsymbol{w}$ for a fixed $\boldsymbol{\lambda}$, as described in Section 3.1. Additionally, we compute the full gradient of $\boldsymbol{w}$ every m_k updates to reduce the variance. In Lines 4-5, we update $\boldsymbol{\lambda}$. Note that we replace $\ell_i(\boldsymbol{w}_k)$ with $\ell_i(\boldsymbol{w}_k) + \theta_k\left(\ell_i(\boldsymbol{w}_k) - \ell_i(\boldsymbol{w}_{k-1})\right)$ to accelerate the algorithm. This can be seen as a momentum term, a widely used technique in smooth optimization (Tseng, 1998; Liu et al., 2020; Gitman et al., 2019; Sutskever et al., 2013), where $\theta_k > 0$ is the momentum parameter.

Define the proximal operator $\operatorname{prox}_h(\bar{\boldsymbol{x}}) := \arg\min_{\boldsymbol{x}} h(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2$ for a function h. In Line 15, we apply the proximal stochastic gradient descent step. We assume that $\operatorname{prox}_{g+\frac{1}{2}\|\cdot\|^2}(\cdot)$ is easy to compute, which is the case for many commonly used regularizers g, such as the l_1 norm and the elastic net regularization (Zou & Hastie, 2005). If g is differentiable, we can replace the proximal stochastic gradient with stochastic gradient: $\boldsymbol{w}_{k,t+1} = \boldsymbol{w}_{k,t} - \alpha\left(\boldsymbol{d}_{k,t} + \frac{1}{\tau_k}\left(\boldsymbol{w}_{k,t} - \boldsymbol{w}_k\right) + \nabla g(\boldsymbol{w}_{k,t})\right)$. This will not affect the convergence or convergence rate of the algorithm as long as ∇g is Lipschitz continuous and the step size α is small enough. In Line 5, we need to compute the projection onto $\Pi_{\boldsymbol{\sigma}}$. For an ordered vector, projecting onto the permutahedron takes $O(n)$ operations using the Pool Adjacent Violators Algorithm (PAVA)(Lim & Wright, 2016). In SOREL, we need to first sort n elements of the projected vector and then compute the projection onto $\Pi_{\boldsymbol{\sigma}}$, which takes a total of $O(n \log n)$ operations.

In practice, we set T_k and m_k to n in Lines 8 and 9, meaning the algorithm updates $\boldsymbol{\lambda}$ once it traverses the training set. We also set the reference point $\bar{\boldsymbol{w}}$ and the output $\boldsymbol{w}_{k+1}$ in Lines 10 and 17 to be the last vector of the previous epoch rather than the average vector, as with most practical algorithms (Johnson & Zhang, 2013; Zhu & Hazan, 2016; Cutkosky & Orabona, 2019; Babanezhad et al., 2015; Gower et al., 2020). In this way, SOREL only requires computing the full batch gradient once for each update of $\boldsymbol{\lambda}$, and becomes single-loop in Lines 8- 16. This makes the algorithm more concise and parameters easier to tune.

Additionally, in the Appendix D, we provide the SOREL algorithm with mini-batching.

Algorithm 1 SOREL

1: **Input:** initial $\boldsymbol{w}_0$, $\boldsymbol{w}_{-1} = \boldsymbol{w}_0$, $\boldsymbol{\sigma}$, and learning rate α, $\{\theta_k\}_{k=0}^{K-1}$, $\{\eta_k\}_{k=0}^{K-1}$, $\{\tau_k\}_{k=0}^{K-1}$, $\{m_k\}_{k=0}^{K-1}$ and $\{T_k\}_{k=0}^{K-1}$.
2: $\boldsymbol{\lambda}_0 = \arg\min_{\boldsymbol{\lambda}\in\Pi_{\boldsymbol{\sigma}}} -\boldsymbol{\ell}(\boldsymbol{w}_0)^\top \boldsymbol{\lambda}$.
3: **for** $k = 0, \ldots, K-1$ **do**
4: $\quad \boldsymbol{v}_k = (1+\theta_k)\boldsymbol{\ell}(\boldsymbol{w}_k) - \theta_k \boldsymbol{\ell}(\boldsymbol{w}_{k-1})$.
5: $\quad \boldsymbol{\lambda}_{k+1} = \arg\min_{\boldsymbol{\lambda}\in\Pi_{\boldsymbol{\sigma}}} -\boldsymbol{v}_k^\top \boldsymbol{\lambda} + \frac{1}{2\eta_k}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^k\|^2$.
6: $\quad \boldsymbol{w}_{k,0} = \boldsymbol{w}_k$, $\bar{\boldsymbol{w}} = \boldsymbol{w}_k$.
7: $\quad \bar{\boldsymbol{g}} = \sum_{i=1}^n \lambda_{i,k+1} \nabla \ell_i(\bar{\boldsymbol{w}})$.
8: $\quad$ **for** $t = 1, \ldots, T_k$ **do**
9: $\quad\quad$ **if** $t \bmod m_k = 0$ **then**
10: $\quad\quad\quad \bar{\boldsymbol{w}} = \frac{1}{m_k}\sum_{j=t-m_k+1}^{t} \boldsymbol{w}_{k,j}$.
11: $\quad\quad\quad \bar{\boldsymbol{g}} = \sum_{i=1}^n \lambda_{i,k+1} \nabla \ell_i(\bar{\boldsymbol{w}})$.
12: $\quad\quad$ **end if**
13: $\quad\quad$ Sample i_t uniformly from $\{1, \ldots, n\}$,
14: $\quad\quad \boldsymbol{d}_{k,t} = n\lambda_{i_t,k+1}\nabla \ell_{i_t}(\boldsymbol{w}_{k,t}) - n\lambda_{i_t,k+1}\nabla \ell_{i_t}(\bar{\boldsymbol{w}}) + \bar{\boldsymbol{g}}$
15: $\quad\quad \boldsymbol{w}_{k,t+1} = \mathrm{Prox}_{\alpha\left(g + \frac{1}{2\tau_k}\|\cdot - \boldsymbol{w}_k\|^2\right)}\left\{\boldsymbol{w}_{k,t} - \alpha \boldsymbol{d}_{k,t}\right\}$.
16: $\quad$ **end for**
17: $\quad \boldsymbol{w}_{k+1} = \frac{1}{m_k}\sum_{j=T_k-m_k+1}^{T_k} \boldsymbol{w}_{k,j}$.
18: **end for**
19: **Output:** $\boldsymbol{w}_K$.

4 Theoretical Analysis

For convenience, we consider that T_k (will be determined in Theorem 1) is large enough so that $\boldsymbol{w}_k$ is a δ_k-optimal solution of $P_k(\boldsymbol{w})$, that is, $\mathbb{E}_k P_k(\boldsymbol{w}_{k+1}) - \min_{\boldsymbol{w}} P_k(\boldsymbol{w}) \le \delta_k$. Here, $\mathbb{E}_k$ represents the conditional expectation with respect to the random sample points used to compute $\boldsymbol{w}_{k+1}$ given $\boldsymbol{w}_k, \ldots, \boldsymbol{w}_0$. Then, we can provide a one-step analysis of the outer loop of SOREL. We use $L(\boldsymbol{w}, \boldsymbol{\lambda}) = \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\boldsymbol{w}) + g(\boldsymbol{w})$ in the analysis for simplicity. The cnvergence analysis for SOREL with mini-batching is presented in Appendix D.

Lemma 1 *Suppose Assumption 1 holds. Let $\{\boldsymbol{w}_k\}$ and $\{\boldsymbol{\lambda}_k\}$ be the sequences generated by Algorithm 1. Then for any $\boldsymbol{w} \in \mathbb{R}^d$, $\boldsymbol{\lambda} \in \Pi_{\boldsymbol{\sigma}}$ and $D = G/\mu$, the following inequality holds,*

$$
\begin{aligned}
&\mathbb{E}_k\ \{L(\boldsymbol{w}_{k+1}, \boldsymbol{\lambda}) - L(\boldsymbol{w}, \boldsymbol{\lambda}_{k+1})\} \\
\le &\mathbb{E}_k \Bigg\{ \langle \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1}, \boldsymbol{\ell}(\boldsymbol{w}_{k+1})\rangle + \frac{1}{2\eta_k}\left[\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_k\|^2 - \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1}\|^2 - \|\boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda}_k\|^2\right] \\
&+ \frac{1}{2\tau_k}\left[\|\boldsymbol{w} - \boldsymbol{w}_k\|^2 - \|\boldsymbol{w} - \boldsymbol{w}_{k+1}\|^2 - \|\boldsymbol{w}_{k+1} - \boldsymbol{w}_k\|^2\right] - \frac{\mu}{2}\|\boldsymbol{w} - \boldsymbol{w}_{k+1}\|^2 \\
&+ \langle \boldsymbol{v}_k, \boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda}\rangle + \delta_k + \sqrt{\frac{(\tau_k^{-1} + \mu)\delta_k}{2}}(D^{-1}\|\boldsymbol{w} - \boldsymbol{w}_{k+1}\|^2 + D) \Bigg\}.
\end{aligned} \tag{6}
$$

Next, we try to telescope the terms on the right hand side of (6) by multiplying each term by γ_k. By choosing appropriate parameters in Algorithm 1 to satisfy some conditions (will be discussed in

Appendix A), we can ensure that the adjacent terms indexed by $k = 0, \ldots, K-1$ can be canceled out during summation. Then we can achieve the desired convergence result.

Theorem 1 *Suppose Assumption 1 holds. Set* $\gamma_k = k+1$, $\eta_k = \frac{\mu(k+1)}{8nG^2}$, $\theta_k = \frac{k}{k+1}$, $\tau_k = \frac{4}{\mu(k+1)}$, $\delta_k = D^2 \min\left(\frac{\mu}{8(k+5)}, \mu(k+1)^{-6}\right)$, $D = G/\mu$, *the step-size* $\alpha = \frac{1}{12L}$, $m_k = \frac{384L}{(k+5)\mu} + 2$ *and* $T_k = O(m_k \log \frac{1}{\delta_k})$ *in Algorithm 1. Let* $\boldsymbol{w}^\star$ *be the optimal solution of Problem (1). Then we have* $\mathbb{E}\|\boldsymbol{w}_K - \boldsymbol{w}^\star\|^2 = O\left(\frac{nG^2}{\mu^2 K^2}\right)$.

Corollary 1 *Under the same conditions in Theorem 1, we obtain an output* $\boldsymbol{w}_K$ *of Algorithm 1 such that* $\mathbb{E}\|\boldsymbol{w}_K - \boldsymbol{w}^\star\|^2 \leq \epsilon$ *in a total sample complexity of* $O\left(\frac{n^{\frac{3}{2}}G}{\mu\sqrt{\epsilon}} \log \frac{\sqrt{n}}{G\sqrt{\epsilon}} + \frac{L}{\mu} \log \frac{\sqrt{n}}{G\sqrt{\epsilon}} \log \frac{\sqrt{n}G}{\mu\sqrt{\epsilon}}\right)$.

Our algorithm achieves a near-optimal convergence rate of $\widetilde{O}(1/\sqrt{\epsilon})$ in terms of ϵ, which matches the lower bound of $\Omega(1/\sqrt{\epsilon})$ in the deterministic setting up to a logarithmic term (Ouyang & Xu, 2021). This is the first near-optimal stochastic method for solving the spectral risk minimization problem. Previously, Mehta et al. (2022; 2024) add a strongly concave term with respect to $\boldsymbol{\lambda}$ in $L(\boldsymbol{w}, \boldsymbol{\lambda})$ and achieve a linear convergence rate for the perturbed problem. One may set the coefficient of the strongly concave term ν to $O(\epsilon)$, obtaining an ϵ-optimal solution for the original spectral risk minimization problem. However, this approach has drawbacks: it leads to a worse sample complexity of $\widetilde{O}(1/\epsilon)$ (Palaniappan & Bach, 2016) or even $\widetilde{O}(1/\epsilon^3)$ (Mehta et al., 2024); additionally, to achieve an ϵ-optimal solution, the step size would need to be set to $O(\epsilon)$, resulting in very small steps that perform poorly in practice. In contrast, SOREL's step size is independent of ϵ.

Remark 1 In Lines 10 and 17 of Algorithm 1, we set the reference point $\bar{\boldsymbol{w}}$ and the output $\boldsymbol{w}_{k+1}$ to the average of the previous epoch. Instead, we can also set them to be the last vector of the previous epoch, which aligns with practical implementation. For theoretical completeness, we may compute the full gradient $\bar{\boldsymbol{g}}$ in Line 11 at each step t with probability p instead of once per epoch (every m_k steps), as done in (Kulunchakov & Mairal, 2019; Hofmann et al., 2015; Kovalev et al., 2020). However, these methods are beyond the scope of this paper.

5 Experiments

In this section, we compare our proposed algorithm SOREL with existing baselines for solving the spectral risk minimization problem. In addition to the precision of the optimizers during training, we also explore fairness and distribution shift metrics on the test set. We focus more on the performance of an optimizer during the training process; therefore, we do not pursue state-of-the-art test metrics due to potential overfitting issues.

We train linear models with l_2 regularization in all experiments. We adopt a wide variety of spectral risks, including ESRM, Extremile, and CVaR. Baseline methods include SGD (Mehta et al., 2022) with a minibatch size of 64, LSVRG (Mehta et al., 2022), and Prospect (Mehta et al., 2024). Note that although both LSVRG and Prospect add a strongly concave term with coefficient ν to smooth the original spectral risk, they have been observed to exhibit linear convergence for the original spectral risk minimization problem in practice without the strongly concave term (Mehta et al., 2022; 2024). Consequently, we set $\nu = 0$ in our experiments. Detailed experimental settings are provided in Appendix B.

5.1 Least-Squares Regression

Five tabular regression benchmarks are used for the least squares loss: `yacht` (Tsanas & Xifara, 2012), `energy` (Baressi Šegota et al., 2019), `concrete` (Yeh, 2006), `kin8nm` (Akujuobi & Zhang, 2017), `power` (Tüfekci, 2014). We compare the suboptimality versus passes (the number of samples divided by n) and runtime. The suboptimality is defined as

$$\text{Suboptimality}(\boldsymbol{w}_k) = \frac{R_{\boldsymbol{\sigma}}(\boldsymbol{w}_k) + g(\boldsymbol{w}_k) - R_{\boldsymbol{\sigma}}(\boldsymbol{w}^\star) - g(\boldsymbol{w}^\star)}{R_{\boldsymbol{\sigma}}(\boldsymbol{w}_0) + g(\boldsymbol{w}_0) - R_{\boldsymbol{\sigma}}(\boldsymbol{w}^\star) - g(\boldsymbol{w}^\star)},$$

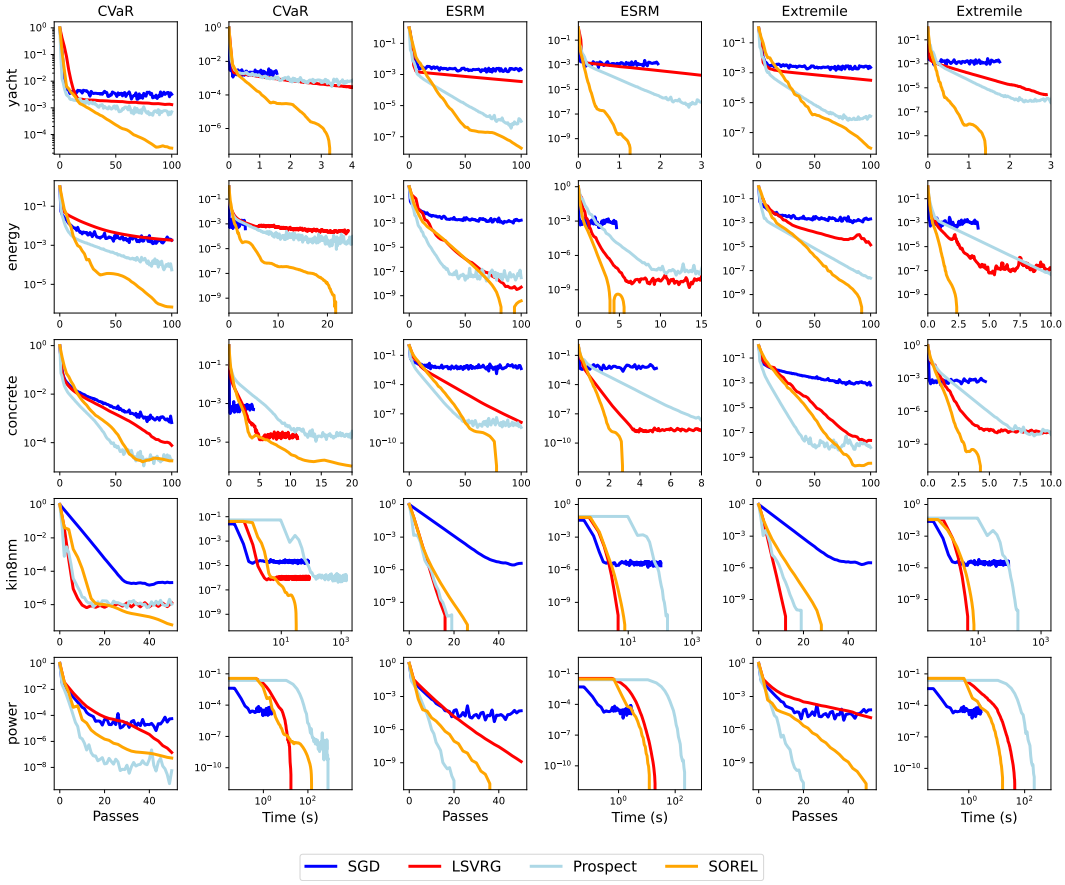

Figure 2: Suboptimality of spectral risks for different algorithms **without mini-batching**. The x-axis represents the effective number of samples used by the algorithm divided by n (odd columns) or CPU time (even columns). Each row corresponds to the same dataset, and each column corresponds to the same type of the spectral risk.

where $\boldsymbol{w}^\star$ is calculated by L-BFGS (Nocedal & Wright, 1999).

Results. Figure 2 compares the training curves of our method with other baselines across various datasets and the spectral risk settings. In terms of sample complexity and runtime, SOREL outperforms other baselines in most cases; SOREL also achieve comparable results in the `kin8nm` dataset. In the `power` dataset, the sample complexity of Prospect is better than that of SOREL. However, the runtime of SOREL is significantly shorter than that of Prospect due to the fact that Prospect needs the calculation of projections onto the permutahedron with $O(n)$ operations each step. As expected, SGD fails to converge due to its inherent bias (Mehta et al., 2022). Although Mehta et al. (2024) discusses the equivalence of minimizing the smoothed spectral risk and the original spectral risk when losses at the optimal point are different from each other, we find that LSVRG and Prospect often fail to reach the true optimal point, indicating limitations of these methods. In contrast, SOREL converges to the true optimal point in all settings (suboptimality less than 0 means the solution's accuracy is higher than L-BFGS).

5.2 Fair Machine Learning

In this experiment, we explore the role of the spectral risks in enhancing fairness in machine learning, as studied in Williamson & Menon (2019). We use the `law` and `acs` datasets. `law` refers to the **Law School Admissions Council's National Longitudinal Bar Passage Study**, which is used for the regression task of predicting a student's GPA (Wightman, 1998). `acs` is derived from **US Census surveys**, which is used for the classification task of predicting whether an adult is employed (Ding et al., 2021). All algorithm are implemented using mini-batching in this experiment.

Assume a source distribution (Y, X, A), where Y is the true label, X represents the available features, and $A \in \{0, 1\}$ is the binary sensitive attribute. Let $\hat{Y} = f(X, A)$ be the model's prediction. For binary classification problems, we consider the fairness metric of Equal Opportunity (EO) defined by

$$\text{EO} = \text{P}\{\widehat{Y} = 1 \mid A = 0, Y = 1\} - \text{P}\{\widehat{Y} = 1 \mid A = 1, Y = 1\}.$$

For regression tasks, we consider the absolute mean difference (SMD) as the fairness metric defined by

$$\text{SMD} = \left|\mathbb{E}\left[\hat{Y} \mid A = 1\right] - \mathbb{E}\left[\hat{Y}\right]\right| + \left|\mathbb{E}\left[\hat{Y} \mid A = 0\right] - \mathbb{E}\left[\hat{Y}\right]\right|.$$

Intuitively, if the EO and SMD are close to 0, the model does not discriminate with respect to A. In both datasets, we set race as the sensitive attribute.

Results. Tables 2 shows the results of using different spectral risks on the `acs` and `law` datasets, respectively. ERM represents the empirical risk. We find that using spectral risks instead of the empirical risk does improve the fairness metrics of the model, and in most cases, a lower suboptimality indicates better fairness of the model. In the `acs` classification task, SOREL significantly outperforms other algorithms in terms of both fairness metrics and suboptimality. For ESRM, SOREL's suboptimality surpasses that of L-BFGS, while the fairness metric of SGD is worse than the baseline under the ERM setting, possibly due to poor performance of SGD when optimizing the objective function. Additionally, CVaR and Extremile are more effective at reducing EO, compared to Extremile. In the `law` regression task, there is no significant difference in SMD improvement among LSVRG, Prospect, and SOREL, but all perform better than SGD. However, SOREL achieves the lowest suboptimality, and its suboptimality is lower than that of L-BFGS under both the ESRM and Extremile settings. Furthermore, training curves in Appendix B show that SOREL can reach low suboptimality in the shortest amount of time.

Table 2: Results of different algorithms on `acs` and `law`. The values in the ERM row represent the mean fairnes metrics (values closer to 0 indicate better fairness) on the test set. The first to third rows for each spectral risk (except ERM) represent, respectively: the mean fairness metrics on the test set, relative fairness metric improvements (%) from ERM, and training suboptimality.

Datasets	`acs`				`law`			
ERM	0.02092				0.05188			
	SGD	LSVRG	Prospect	SOREL	SGD	LSVRG	Prospect	SOREL
CVaR	0.00645 69.17 4.29e-4	0.00816 60.99 7.23e-4	0.00634 69.69 2.12e-4	0.00551 73.66 2.31e-6	0.04019 22.53 3.80e-3	0.03896 24.90 5.88e-3	0.03893 24.96 6.71e-4	0.03890 25.02 2.95e-5
ESRM	0.02469 – 1.47e-3	0.01842 11.95 3.33e-4	0.01840 12.05 4.38e-6	0.01770 15.39 -2.38e-8	0.04184 19.35 7.60e-4	0.04122 20.55 1.13e-4	0.04123 20.53 1.52e-7	0.04123 20.53 -1.13e-7
Extremile	0.00424 79.73 5.11e-3	0.00377 81.98 7.90e-3	0.00237 88.67 6.69e-4	0.00130 93.97 1.14e-4	0.04416 14.88 1.63e-4	0.04377 15.63 6.14e-4	0.04380 15.57 8.14e-6	0.04381 15.56 -2.12e-7

5.3 Out-of-Distribution Generalization

In this subsection, we explore the role of the spectral risk in enhancing model robustness under distribution shift. We use CVaR and Extremile as the spectral risks. Levy et al. (2020) uses the CVaR measure as the uncertainty set, and their optimization problem is the same as the spectral risk minimization problem (1) that uses CVaR as the spectral risk. We use the `amazon` dataset preprocessed by Mehta et al. (2024) for the multi-class classification task, which consists of feature representations generated by BERT (Devlin et al., 2019) from the original dataset. `amazon` refers to the **Amazon Reviews** dataset (Ni et al., 2019), which includes textual reviews of products along with their corresponding ratings from one to five, with different reviewers for the training and test sets. We evaluate the worst group classification error (Sagawa et al., 2020) on the test set. Each group is classified based on the true labels. All algorithm are implemented using mini-batching in this experiment.

We also explore the impact of distribution shift on fairness metrics in Section 5.2. In Ding et al. (2021), it is observed that training and testing on different states lead to unpredictable results. We use

data from California as the training data and train models using ERM and CVaR as loss functions, respectively. We then test the models on four other states.

Results. Figure 3 shows the results of using CVaR and Extremile spectral risks on `amazon`. SOREL achieves the best worst group classification error in both settings. For CVaR, under similar suboptimality, SOREL reaches the minimum worst group classification error, indicating better generalization performance. Moreover, SOREL is the only algorithm that can converge to the true optimal solution under the CVaR setting. Additionally, SOREL demonstrates optimal or near-optimal convergence rates in both spectral risk settings.

Figure 4 shows the results of models tested on four other states. The circles represent model's performance in California (in-distribution). Models' performance in other states (out-of-distribution) is indeed hard to predict. Notably, models trained with ERM often fail to meet the expected fairness metrics in other states. However, models trained with CVaR often achieves higher test accuracy and better fairness metrics. Moreover, the models trained with SOREL achieve the best or nearly the best EO and test accuracy.

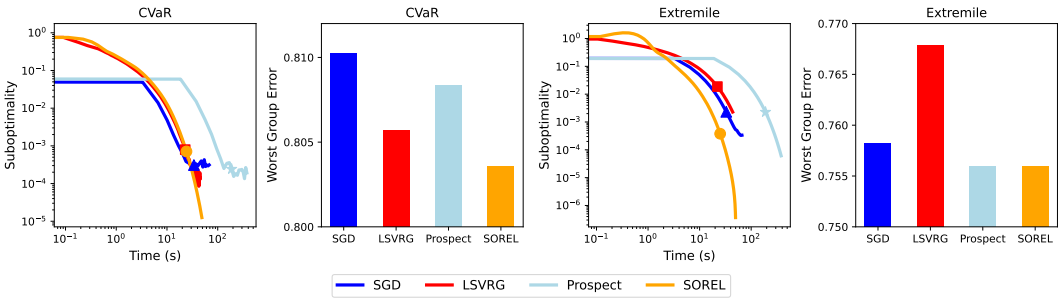

Figure 3: Training curves and worst group classification errors of different algorithms on the `amazon` dataset. The suboptimality at the 500 th pass (where we evaluate the worst group error) is marked on the training curves. The training curves are extended to illustrate convergence.

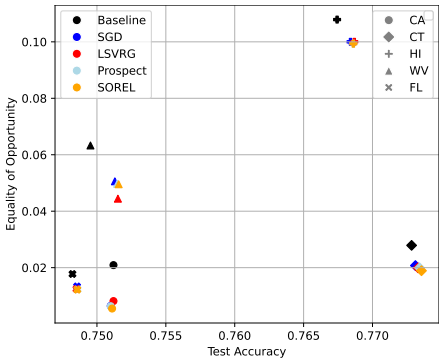

Figure 4: Model performance under geographic distribution shift. The models are trained on the state CA and tested on other states. Dots of different colors (except black) represent the results of using different optimization algorithms to solve the CVaR minimization problem. Baseline refers to the results using ERM as the loss function.

6 Conclusion

We have proposed SOREL, the first stochastic algorithm with convergence guarantees for the spectral risk minimization problems. We have proved that SOREL achieves a near-optimal rate of $\widetilde{O}(1/\sqrt{\epsilon})$. In experiments, SOREL outperforms existing baselines in terms of sample complexity and runtime in most cases.

Future work includes exploring convergence of SOREL for nonconvex problems, and investigating broader applications of the spectral risk in areas such as fairness and distributionally robust optimization.

REFERENCES

Prashanth L. A. and Sanjay P. Bhat. A Wasserstein Distance Approach for Concentration of Empirical Risk Estimates. *Journal of Machine Learning Research (JMLR)*, 23:238:1–238:61, 2022.

Uchenna Akujuobi and Xiangliang Zhang. Delve: A Dataset-Driven Scholarly Search and Analysis System. *SIGKDD Explorations*, 19(2):36–46, 2017.

Philippe Artzner. Thinking coherently. *Risk*, 10:68–71, 1997.

Philippe Artzner, Freddy Delbaen, Jean-Marc Eber, and David Heath. Coherent measures of risk. *Mathematical finance*, 9(3):203–228, 1999.

Reza Babanezhad, Mohamed Osama Ahmed, Alim Virani, Mark Schmidt, Jakub Konecný, and Scott Sallinen. Stopwasting My Gradients: Practical SVRG. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2251–2259, 2015.

Sandi Baressi Šegota, Nikola Anđelić, Jan Kudláček, and Robert Čep. Artificial neural network for predicting values of residuary resistance per unit weight of displacement. *Pomorski zbornik*, 57 (1):9–22, 2019.

Mathieu Blondel, Olivier Teboul, Quentin Berthet, and Josip Djolonga. Fast Differentiable Sorting and Ranking. In *International Conference on Machine Learning (ICML)*, pp. 950–959, 2020.

Digvijay Boob, Qi Deng, and Guanghui Lan. Level constrained first order methods for function constrained optimization. *Mathematical Programming*, 2024.

Ruidi Chen and Ioannis Ch Paschalidis. Distributionally robust learning. *Foundations and Trends® in Optimization*, 4(1-2):1–243, 2020.

Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-Constrained Reinforcement Learning with Percentile Risk Criteria. *Journal of Machine Learning Research (JMLR)*, 18:167:1–167:51, 2017.

John Cotter and Kevin Dowd. Extreme spectral risk measures: An application to futures clearinghouse margin requirements. *Journal of Banking & Finance*, 30(12):3469–3485, 2006.

Xiangyu Cui, Rujun Jiang, Yun Shi, Rufeng Xiao, and Yifan Yan. Decision making under cumulative prospect theory: An alternating direction method of multipliers. *INFORMS Journal on Computing*, 2024.

Sebastian Curi, Kfir Y. Levy, Stefanie Jegelka, and Andreas Krause. Adaptive Sampling for Stochastic Risk-Averse Learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Ashok Cutkosky and Francesco Orabona. Momentum-Based Variance Reduction in Non-Convex SGD. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 15210–15219, 2019.

Abdelaati Daouia, Irène Gijbels, and Gilles Stupfler. Extremiles: A New Perspective on Asymmetric Least Squares. *Journal of the American Statistical Association*, 114(527):1366–1381, 2019.

Damek Davis and Dmitriy Drusvyatskiy. Stochastic Model-Based Minimization of Weakly Convex Functions. *SIAM Journal on Optimization*, 29(1):207–239, 2019.

Aaron Defazio, Francis R. Bach, and Simon Lacoste-Julien. Saga: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 1646–1654, 2014.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North*, pp. 4171–4186. Association for Computational Linguistics, 2019.

Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring Adult: New Datasets for Fair Machine Learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 6478–6490, 2021.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel. Fairness through awareness. In *Innovations in Theoretical Computer Science (ITCS)*, pp. 214–226, 2012.

Yanbo Fan, Siwei Lyu, Yiming Ying, and Bao-Gang Hu. Learning with Average Top-k Loss. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 497–505, 2017.

Igor Gitman, Hunter Lang, Pengchuan Zhang, and Lin Xiao. Understanding the Role of Momentum in Stochastic Gradient Methods. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 9630–9640, 2019.

Robert M. Gower, Mark Schmidt, Francis R. Bach, and Peter Richtárik. Variance-Reduced Methods for Machine Learning. *Proceedings of the IEEE*, 108(11):1968–1983, 2020.

Erfan Yazdandoost Hamedani and Necdet Serhat Aybat. A Primal-Dual Algorithm with Line Search for General Convex-Concave Saddle Point Problems. *SIAM Journal on Optimization*, 31(2): 1299–1329, 2021.

Erfan Yazdandoost Hamedani and Afrooz Jalilzadeh. A stochastic variance-reduced accelerated primal-dual method for finite-sum saddle-point problems. *Computational Optimization and Applications*, 85(2):653–679, 2023.

Moritz Hardt, Eric Price, and Nati Srebro. Equality of Opportunity in Supervised Learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 3315–3323, 2016.

Xue Dong He, Steven Kou, and Xianhua Peng. Risk measures: robustness, elicitability, and backtesting. *Annual Review of Statistics and Its Application*, 9(1):141–166, 2022.

Thomas Hofmann, Aurélien Lucchi, Simon Lacoste-Julien, and Brian McWilliams. Variance Reduced Stochastic Gradient Descent with Neighbors. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2305–2313, 2015.

Matthew J. Holland and El Mehdi Haress. Learning with risk-averse feedback under potentially heavy tails. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 892–900, 2021.

Matthew J. Holland and El Mehdi Haress. Spectral risk-based learning using unbounded losses. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1871–1886, 2022.

Rie Johnson and Tong Zhang. Accelerating Stochastic Gradient Descent using Predictive Variance Reduction. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 315–323, 2013.

Kenji Kawaguchi and Haihao Lu. Ordered SGD: A New Stochastic Optimization Framework for Empirical Risk Minimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 669–679, 2020.

Mohammad Khalafi and Digvijay Boob. Accelerated Primal-Dual Methods for Convex-Strongly-Concave Saddle Point Problems. In *International Conference on Machine Learning (ICML)*, pp. 16250–16270, 2023.

Justin Khim, Liu Leqi, Adarsh Prasad, and Pradeep Ravikumar. Uniform Convergence of Rank-weighted Learning. In *International Conference on Machine Learning (ICML)*, pp. 5254–5263, 2020.

Dmitry Kovalev, Samuel Horváth, and Peter Richtárik. Don't Jump Through Hoops and Remove Those Loops: Svrg and Katyusha are Better Without the Outer Loop. In *International Conference on Algorithmic Learning Theory (ALT)*, pp. 451–467, 2020.

Andrei Kulunchakov and Julien Mairal. Estimate Sequences for Variance-Reduced Stochastic Composite Optimization. In *International Conference on Machine Learning (ICML)*, pp. 3541–3550, 2019.

Daniel Levy, Yair Carmon, John C. Duchi, and Aaron Sidford. Large-Scale Methods for Distributionally Robust Optimization. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Cong Han Lim and Stephen J. Wright. Efficient Bregman Projections onto the Permutahedron and Related Polytopes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1205–1213, 2016.

Yanli Liu, Yuan Gao, and Wotao Yin. An Improved Analysis of Stochastic Gradient Descent with Momentum. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Andreas Maurer, Daniela Angela Parletta, Andrea Paudice, and Massimiliano Pontil. Robust Unsupervised Learning via L-statistic Minimization. In *International Conference on Machine Learning (ICML)*, pp. 7524–7533, 2021.

Ronak Mehta, Vincent Roulet, Krishna Pillutla, and Zaïd Harchaoui. Distributionally Robust Optimization with Bias and Variance Reduction. *The Twelfth International Conference on Learning Representations*, abs/2310.13863, 2024.

Ronak R. Mehta, Vincent Roulet, Krishna Pillutla, Lang Liu, and Zaïd Harchaoui. Stochastic Optimization for Spectral Risk Measures. In *International Conference on Artificial Intelligence and Statistics*, pp. 10112–10159, 2022.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.

Lam M. Nguyen, Jie Liu, Katya Scheinberg, and Martin Takác. Sarah: A Novel Method for Machine Learning Problems Using Stochastic Recursive Gradient. In *International Conference on Machine Learning (ICML)*, pp. 2613–2621, 2017.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019.

Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

Yuyuan Ouyang and Yangyang Xu. Lower complexity bounds of first-order methods for convex-concave bilinear saddle-point problems. *Mathematical Programming*, 185(1-2):1–35, 2021.

Balamurugan Palaniappan and Francis R. Bach. Stochastic Variance Reduction Methods for Saddle-Point Problems. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 1408–1416, 2016.

R Tyrrell Rockafellar and Stan Uryasev. The fundamental risk quadrangle in risk management, optimization and statistical estimation. *Surveys in Operations Research and Management Science*, 18(1-2):33–53, 2013.

R. Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *The Journal of Risk*, 2(3):21–41, 2000.

Nicolas Le Roux, Mark Schmidt, and Francis R. Bach. A Stochastic Gradient Method with an Exponential Convergence Rate for Finite Training Sets. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2672–2680, 2012.

Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. Distributionally Robust Neural Networks. In *International Conference on Learning Representations (ICLR)*, 2020.

Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(1), 2013.

Vishnu D. Sharma, Maymoonah Toubeh, Lifeng Zhou, and Pratap Tokekar. Risk-Aware Planning and Assignment for Ground Vehicles using Uncertain Perception from Aerial Vehicles. In *IEEE/RJS International Conference on Intelligent RObots and Systems (IROS)*, pp. 11763–11769, 2020.

Ilya Sutskever, James Martens, George E. Dahl, and Geoffrey E. Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning (ICML)*, pp. 1139–1147, 2013.

Kiran Koshy Thekumparampil, Prateek Jain, Praneeth Netrapalli, and Sewoong Oh. Efficient Algorithms for Smooth Minimax Optimization. In *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 12659–12670, 2019.

Athanasios Tsanas and Angeliki Xifara. Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. *Energy and Buildings*, 49:560–567, 2012.

Paul Tseng. An incremental gradient (-projection) method with momentum term and adaptive stepsize rule. *SIAM Journal on Optimization*, 8(2):506–531, 1998.

Pınar Tüfekci. Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. *International Journal of Electrical Power & Energy Systems*, 60:126–140, 2014.

Peng Wang, Rujun Jiang, Qingyuan Kong, and L. Balzano. Proximal DC Algorithm for Sample Average Approximation of Chance Constrained Programming: Convergence and Numerical Results. *arXiv*, 2023.

Linda F Wightman. Lsac national longitudinal bar passage study. lsac research report series. 1998.

Robert C. Williamson and Aditya Krishna Menon. Fairness risk measures. In *International Conference on Machine Learning (ICML)*, pp. 6786–6797, 2019.

Lin Xiao and Tong Zhang. A Proximal Stochastic Gradient Method with Progressive Variance Reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

Rufeng Xiao, Yuze Ge, Rujun Jiang, and Yifan Yan. A Unified Framework for Rank-based Loss Minimization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Ziyue Xu, Andriy Myronenko, Dong Yang, Holger R. Roth, Can Zhao, Xiaosong Wang, and Daguang Xu. *Clinical-Realistic Annotation for Histopathology Images with Probabilistic Semi-supervision: A Worst-Case Study*. Springer Nature Switzerland, 2022.

Yan Yan, Yi Xu, Qihang Lin, Lijun Zhang, and Tianbao Yang. Stochastic Primal-Dual Algorithms with Faster Convergence than O(1/√T) for Problems without Bilinear Structure. *ArXiv*, abs/1904.10112, 2019.

I-Cheng Yeh. Analysis of Strength of Concrete Using Design of Experiments and Neural Networks. *Journal of Materials in Civil Engineering*, 18(4):597–604, 2006.

Qiyuan Zhang, Shu Leng, Xiaoteng Ma, Qihan Liu, Xueqian Wang, Bin Liang, Yu Liu, and Jun Yang. Cvar-Constrained Policy Optimization for Safe Reinforcement Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

Zeyuan Allen Zhu and Elad Hazan. Variance Reduction for Faster Non-Convex Optimization. In *International Conference on Machine Learning (ICML)*, pp. 699–707, 2016.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2):301–320, 2005.

A PROOFS

First, we provide an auxiliary lemma. This is an extension of Boob et al. (2024, Lemma 8).

Lemma 2 *Let $\bar{\boldsymbol{x}}$ be an ϵ-approximate solution of $\min_x\{g(\boldsymbol{x}) + \frac{\lambda}{2}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2\}$ in expectation, where $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex, $\mu \geq 0$. Then for any $D>0$*

$$\mathbb{E}\left\{g(\bar{\boldsymbol{x}}) - g(\boldsymbol{x})\right\} \leq \mathbb{E}\left\{\frac{\lambda}{2}\left[\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 - \|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2 - \|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|^2\right] - \frac{\mu}{2}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2\right\} + \sqrt{\frac{(\lambda+\mu)\epsilon}{2}}D^{-1}\mathbb{E}\|\bar{\boldsymbol{x}} - \boldsymbol{x}\|^2 + \sqrt{\frac{(\lambda+\mu)\epsilon}{2}}D + \epsilon.$$

Proof: Let $\boldsymbol{x}^\star = \arg\min_x\{g(\boldsymbol{x}) + \frac{\lambda}{2}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2\}$. By $(\mu+\lambda)$-strong convexity of $g(\cdot) + \frac{\lambda}{2}\|\cdot - \hat{\boldsymbol{x}}\|^2$ we have

$$\begin{aligned} g(\boldsymbol{x}) + \frac{\lambda}{2}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 &\geq g(\boldsymbol{x}^\star) + \frac{\lambda}{2}\|\boldsymbol{x}^\star - \hat{\boldsymbol{x}}\|^2 + \frac{\mu+\lambda}{2}\|\boldsymbol{x} - \boldsymbol{x}^\star\|^2, \\ g(\boldsymbol{x}^\star) - g(\boldsymbol{x}) &\leq \frac{\lambda}{2}\left[\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 - \|\boldsymbol{x}^\star - \hat{\boldsymbol{x}}\|^2 - \|\boldsymbol{x}^\star - \boldsymbol{x}\|^2\right] - \frac{\mu}{2}\|\boldsymbol{x} - \boldsymbol{x}^\star\|^2. \end{aligned} \tag{7}$$

By the definition of $\bar{\boldsymbol{x}}$ we have

$$\mathbb{E}\{g(\bar{\boldsymbol{x}}) + \frac{\lambda}{2}\|\bar{\boldsymbol{x}} - \hat{\boldsymbol{x}}\|^2\} \leq g(\boldsymbol{x}^\star) + \frac{\lambda}{2}\|\boldsymbol{x}^\star - \hat{\boldsymbol{x}}\|^2 + \epsilon \tag{8}$$

Combining (7) and (8) gives

$$\begin{aligned} &\mathbb{E}\left\{g(\bar{\boldsymbol{x}}) - g(\boldsymbol{x})\right\} \leq \frac{\lambda}{2}\left[\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 - \|\boldsymbol{x}^\star - \boldsymbol{x}\|^2 - \mathbb{E}\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|^2\right] - \frac{\mu}{2}\|\boldsymbol{x} - \boldsymbol{x}^\star\|^2 + \epsilon \qquad (9) \\ &= \mathbb{E}\left\{\frac{\lambda}{2}\left[\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 - \|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2 - \|\bar{\boldsymbol{x}} - \hat{\boldsymbol{x}}\|^2\right] - \frac{\mu}{2}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2\right. \\ &\quad \left. + \frac{\lambda+\mu}{2}\left[\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2 - \|\boldsymbol{x} - \boldsymbol{x}^\star\|^2\right] + \epsilon\right\} \\ &\leq \mathbb{E}\left\{\frac{\lambda}{2}\left[\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 - \|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2 - \|\bar{\boldsymbol{x}} - \hat{\boldsymbol{x}}\|^2\right] - \frac{\mu}{2}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2\right. \\ &\quad \left. + (\lambda+\mu)\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|\|\bar{\boldsymbol{x}} - \boldsymbol{x}^\star\| + \epsilon\right\}, \qquad (10) \end{aligned}$$

where the last inequality is due to the fact that $\|a\|^2 - \|b\|^2 \leq -2\langle a, b-a\rangle \leq 2\|a\|\|b-a\|$.

Let $\boldsymbol{x} = \bar{\boldsymbol{x}}$ in (9), and take the expectation with respect to $\bar{\boldsymbol{x}}$. Then we have

$$\frac{\lambda+\mu}{2}\mathbb{E}\|\boldsymbol{x}^\star - \bar{\boldsymbol{x}}\|^2 \leq \epsilon.$$

By Hölder's inequality we have

$$\begin{aligned} \mathbb{E}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|\|\bar{\boldsymbol{x}} - \boldsymbol{x}^\star\| &\leq \left(\mathbb{E}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2\right)^{\frac{1}{2}}\left(\mathbb{E}\|\boldsymbol{x}^\star - \bar{\boldsymbol{x}}\|^2\right)^{\frac{1}{2}} \\ &\leq \frac{1}{2}\left(\mathbb{E}\|\boldsymbol{x}^\star - \bar{\boldsymbol{x}}\|^2\right)^{\frac{1}{2}}\left(D + D^{-1}\mathbb{E}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2\right). \\ &\leq \frac{1}{2}\sqrt{\frac{2\epsilon}{\lambda+\mu}}\left(D + D^{-1}\mathbb{E}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2\right). \end{aligned}$$

Combining the above results and (10) we get the desired result. □

Consider solving the problem from Line 8 to Line 16 in Algorithm 1 while updating $\boldsymbol{w}$:

$$\min_{\boldsymbol{w}} P_k(\boldsymbol{w}) := g(\boldsymbol{w}) + \boldsymbol{\lambda}_{k+1}^\top \boldsymbol{\ell}(\boldsymbol{w}) + \frac{1}{2\eta_k}\|\boldsymbol{w} - \boldsymbol{w}_k\|^2.$$

The following lemma provides the error between $\boldsymbol{w}_{k+1}$ and $\arg\min_{\boldsymbol{w}} P_k(\boldsymbol{w})$.

Lemma 3 *Let* $P_k(\boldsymbol{w}) := \boldsymbol{\lambda}_{k+1}^\top \boldsymbol{\ell}(\boldsymbol{w}) + g(\boldsymbol{w}) + \frac{1}{2\tau_k}\|\boldsymbol{w} - \boldsymbol{w}_k\|^2$. *Set* $\alpha < \frac{L}{4}$, $m_k = \Theta(\frac{L\tau_k}{\mu\tau_k+1})$ *and* $T_k = O(m_k \log \frac{1}{\epsilon})$ *in Algorithm 1. The overall sample complexity of obtaining an* ϵ*-approximate solution such that* $\mathbb{E}P_k(\boldsymbol{w}^{k+1}) - \min_{\boldsymbol{w}} P_k(\boldsymbol{w}) \le \epsilon$ *is* $O\left(\left(n + \frac{L\tau_k}{\mu\tau_k+1}\right)\log\frac{1}{\epsilon}\right)$. *Moreover, we can set* $\alpha = \frac{1}{12L}$ *and* $m_k = \frac{96L}{\mu+\tau_k^{-1}} + 2$ *in practice.*

Proof: First note that $\boldsymbol{\lambda}^{k+1\top}\boldsymbol{\ell}(\boldsymbol{w})$ is L-smooth since

$$\|\sum_{i=i}^{n}\lambda_{k+1,i}\ell_i(\boldsymbol{x}) - \sum_{i=1}^{n}\lambda_{k+1,i}\ell_i(\boldsymbol{y})\| \le \sum_{i=1}^{n}\lambda_{k+1,i}\|\ell_i(\boldsymbol{x}) - \ell_i(\boldsymbol{y})\| \le L\|\boldsymbol{x}-\boldsymbol{y}\|,$$

for $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$. In the last inequality we use $\sum_i \lambda_{k+1,i} \le 1$ due to $\boldsymbol{\lambda}_{k+1} \in \Pi_{\boldsymbol{\sigma}}$ and L-smoothness of ℓ_i. Moreover, it is not hard to see that $P_k(\boldsymbol{w})$ is $\mu + \tau_k^{-1}$-strongly convex. By Xiao & Zhang (2014, Theorem 1) we get the desired result. □

A.1 Proof of Lemma 1

Proof: From the update of $\boldsymbol{\lambda}_{k+1}$ and Lemma 2 we have

$$0 \le \frac{1}{2\eta_k}\left[\|\boldsymbol{\lambda}-\boldsymbol{\lambda}_k\|^2 - \|\boldsymbol{\lambda}-\boldsymbol{\lambda}_{k+1}\|^2 - \|\boldsymbol{\lambda}_{k+1}-\boldsymbol{\lambda}_k\|^2\right] + \langle \boldsymbol{v}_k, \boldsymbol{\lambda}_{k+1}-\boldsymbol{\lambda}\rangle. \tag{11}$$

From the update of $\boldsymbol{w}_{k+1}$ and Lemma 2 we have

$$\begin{aligned}
&\mathbb{E}_k\left\{g(\boldsymbol{w}_{k+1}) + \langle \boldsymbol{\lambda}_{k+1}, \boldsymbol{\ell}(\boldsymbol{w}_{k+1})\rangle - g(\boldsymbol{w}) - \langle \boldsymbol{\lambda}_{k+1}, \boldsymbol{\ell}(\boldsymbol{w})\rangle\right\} \\
&\le \mathbb{E}_k\left\{\frac{1}{2\tau_k}\left[\|\boldsymbol{w}-\boldsymbol{w}_k\|^2 - \|\boldsymbol{w}-\boldsymbol{w}_{k+1}\|^2 - \|\boldsymbol{w}_{k+1}-\boldsymbol{w}_k\|^2\right] - \frac{\mu}{2}\|\boldsymbol{w}-\boldsymbol{w}_{k+1}\|^2\right. \\
&\quad \left. +\delta_k + \sqrt{\frac{(\tau_k^{-1}+\mu)\delta_k}{2}}(D^{-1}\|\boldsymbol{w}-\boldsymbol{w}_{k+1}\|^2 + D)\right\}.
\end{aligned} \tag{12}$$

Taking the conditional expectation $\mathbb{E}_k$ of both sides of (11) and summing with (12) we obtain that

$$\begin{aligned}
&\mathbb{E}_k\left\{L(\boldsymbol{w}_{k+1}, \boldsymbol{\lambda}) - L(\boldsymbol{w}, \boldsymbol{\lambda}_{k+1})\right\} \\
=&\mathbb{E}_k\left\{g(\boldsymbol{w}_{k+1}) + \langle \boldsymbol{\lambda}, \boldsymbol{\ell}(\boldsymbol{w}_{k+1})\rangle - g(\boldsymbol{w}) - \langle \boldsymbol{\lambda}_{k+1}, \boldsymbol{\ell}(\boldsymbol{w})\rangle\right\} \\
\le&\mathbb{E}_k\left\{\langle \boldsymbol{\lambda}-\boldsymbol{\lambda}_{k+1}, \boldsymbol{\ell}(\boldsymbol{w}_{k+1})\rangle + \frac{1}{2\eta_k}\left[\|\boldsymbol{\lambda}-\boldsymbol{\lambda}_k\|^2 - \|\boldsymbol{\lambda}-\boldsymbol{\lambda}_{k+1}\|^2 - \|\boldsymbol{\lambda}_{k+1}-\boldsymbol{\lambda}_k\|^2\right]\right. \\
&+ \frac{1}{2\tau_k}\left[\|\boldsymbol{w}-\boldsymbol{w}_k\|^2 - \|\boldsymbol{w}-\boldsymbol{w}_{k+1}\|^2 - \|\boldsymbol{w}_{k+1}-\boldsymbol{w}_k\|^2\right] + \langle \boldsymbol{v}_k, \boldsymbol{\lambda}_{k+1}-\boldsymbol{\lambda}\rangle \\
&\left. -\frac{\mu}{2}\|\boldsymbol{w}-\boldsymbol{w}_{k+1}\|^2 + \delta_k + \sqrt{\frac{(\tau_k^{-1}+\mu)\delta_k}{2}}(D^{-1}\|\boldsymbol{w}-\boldsymbol{w}_{k+1}\|^2 + D)\right\}.
\end{aligned} \tag{13}$$

□

A.2 Proof of Theorem 1

Lemma 4 *Under the same assumptions as Lemma 1, for any $\boldsymbol{w} \in \mathbb{R}^d$ and $\boldsymbol{\lambda} \in \Pi_{\boldsymbol{\sigma}}$, we have*

$$
\begin{aligned}
&\mathbb{E}_k \left\{ L(\boldsymbol{w}_{k+1}, \boldsymbol{\lambda}) - L(\boldsymbol{w}, \boldsymbol{\lambda}_{k+1}) \right\} \\
\leq &\mathbb{E}_k \left\{ \frac{1}{2\eta_k} \left[\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_k\|^2 - \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1}\|^2 \right] + \frac{1}{2\tau_k} \|\boldsymbol{w} - \boldsymbol{w}_k\|^2 - \frac{1}{2} \left(\frac{1}{\tau_k} + \mu \right) \|\boldsymbol{w} - \boldsymbol{w}_{k+1}\|^2 \right. \\
&+ \langle \boldsymbol{\ell}(\boldsymbol{w}_{k+1}) - \boldsymbol{\ell}(\boldsymbol{w}_k), \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1} \rangle - \theta_k \langle \boldsymbol{\ell}(\boldsymbol{w}_k) - \boldsymbol{\ell}(\boldsymbol{w}_{k-1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_k \rangle \\
&- \frac{1}{2} \left[\frac{1}{\eta_k} - \theta_k \frac{\sqrt{n} G}{\alpha_k} \right] \|\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k+1}\|^2 - \frac{1}{2\tau_k} \|\boldsymbol{w}_k - \boldsymbol{w}_{k+1}\|^2 + \frac{\sqrt{n} G \theta_k \alpha_k}{2} \|\boldsymbol{w}_k - \boldsymbol{w}_{k-1}\|^2 \\
&\left. + \delta_k + \sqrt{\frac{(\tau_k^{-1} + \mu)\delta_k}{2}} (D^{-1} \|\boldsymbol{w} - \boldsymbol{w}_{k+1}\|^2 + D) \right\}.
\end{aligned}
\tag{14}
$$

Proof: First, we have

$$
\begin{aligned}
&\langle \boldsymbol{v}_k, \boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda} \rangle \\
= &\langle \boldsymbol{\ell}(\boldsymbol{w}_k) + \theta_k \left(\boldsymbol{\ell}(\boldsymbol{w}_k) - \boldsymbol{\ell}(\boldsymbol{w}_{k-1}) \right), \boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda} \rangle \\
= &- \langle \boldsymbol{\ell}(\boldsymbol{w}_{k+1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1} \rangle + \langle \boldsymbol{\ell}(\boldsymbol{w}_{k+1}) - \boldsymbol{\ell}(\boldsymbol{w}_k), \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1} \rangle \\
&- \theta_k \langle \boldsymbol{\ell}(\boldsymbol{w}_k) - \boldsymbol{\ell}(\boldsymbol{w}_{k-1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_k \rangle - \theta_k \langle \boldsymbol{\ell}(\boldsymbol{w}_k) - \boldsymbol{\ell}(\boldsymbol{w}_{k-1}), \boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k+1} \rangle.
\end{aligned}
$$

Then we obtain that

$$
\begin{aligned}
&\langle \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1}, \boldsymbol{\ell}(\boldsymbol{w}_{k+1}) \rangle + \langle \boldsymbol{v}_k, \boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda} \rangle \\
&\leq \langle \boldsymbol{\ell}(\boldsymbol{w}_{k+1}) - \boldsymbol{\ell}(\boldsymbol{w}_k), \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1} \rangle - \theta_k \langle \boldsymbol{\ell}(\boldsymbol{w}_k) - \boldsymbol{\ell}(\boldsymbol{w}_{k-1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_k \rangle \\
&\quad - \theta_k \langle \boldsymbol{\ell}(\boldsymbol{w}_k) - \boldsymbol{\ell}(\boldsymbol{w}_{k-1}), \boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k+1} \rangle.
\end{aligned}
\tag{15}
$$

Next we bound the last term on the right-hand side of (15):

$$
\begin{aligned}
&\langle \boldsymbol{\ell}(\boldsymbol{w}_k) - \boldsymbol{\ell}(\boldsymbol{w}_{k-1}), \boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k+1} \rangle \\
\leq &\sqrt{n} G \|\boldsymbol{w}_k - \boldsymbol{w}_{k-1}\| \|\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k+1}\| \\
\leq &\frac{\sqrt{n} G \alpha_k}{2} \|\boldsymbol{w}_k - \boldsymbol{w}_{k-1}\|^2 + \frac{\sqrt{n} G}{2\alpha_k} \|\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k+1}\|^2,
\end{aligned}
\tag{16}
$$

where the first inequality is due to the G-Lipschitz continuity of ℓ_i and in the second inequality we use Young's inequality with $\alpha_k > 0$.

Combing (15) and (16) we obtain that

$$
\begin{aligned}
&\langle \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1}, \boldsymbol{\ell}(\boldsymbol{w}_{k+1}) \rangle + \langle \boldsymbol{v}_k, \boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda} \rangle \\
&\leq \langle \boldsymbol{\ell}(\boldsymbol{w}_{k+1}) - \boldsymbol{\ell}(\boldsymbol{w}_k), \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1} \rangle - \theta_k \langle \boldsymbol{\ell}(\boldsymbol{w}_k) - \boldsymbol{\ell}(\boldsymbol{w}_{k-1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_k \rangle \\
&\quad + \frac{\sqrt{n} G \alpha_k \theta_k}{2} \|\boldsymbol{w}_k - \boldsymbol{w}_{k-1}\|^2 + \frac{\sqrt{n} G \theta_k}{2\alpha_k} \|\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k+1}\|^2.
\end{aligned}
\tag{17}
$$

Taking the conditional expectation $\mathbb{E}_k$ of both sides of (17) and combing it with Lemma 1 we get the desired result. □

We remark that α_k does not need to be computed in the actual algorithm but only exists in the theoretical analysis. Next, we try to telescope the terms on the right hand side of (14) by multiplying each term by γ_k. To ensure that the adjacent terms in the sequence $k = 0, \ldots, K-1$ can be canceled out during summation, we need the parameters of the algorithm to satisfy the following conditions.

Condition 1 *For $k = 0, 1, ...$, the following conditions for parameters in the analysis and Algorithm 1:*

$$\frac{\gamma_{k+1}}{\eta_{k+1}} \leq \frac{\gamma_k}{\eta_k}, \tag{18a}$$

$$\frac{\gamma_{k+1}}{\tau_{k+1}} \leq \gamma_k \left(\frac{1}{\tau_k} + \mu - \sqrt{2(\mu + \tau_k^{-1})\delta_k} D^{-1} \right), \tag{18b}$$

$$\gamma_k = \gamma_{k+1}\theta_{k+1}, \tag{18c}$$

$$\sqrt{n}G\alpha_{k+1} \leq \frac{1}{\tau_k}, \tag{18d}$$

$$\theta_k \frac{\sqrt{n}G}{\alpha_k} \leq \frac{1}{\eta_k}. \tag{18e}$$

Lemma 5 *Assume Assumption 1 holds and Condition 1 is satisfied. Then for all $\boldsymbol{w} \in \mathbb{R}^d$ and $\boldsymbol{\lambda} \in \Pi_{\boldsymbol{\sigma}}$ we have*

$$\frac{\gamma_K}{2\tau_K}\mathbb{E}\|\boldsymbol{w}^\star - \boldsymbol{w}_K\|^2 \leq \frac{\gamma_0}{2\eta_0}\|\boldsymbol{\lambda}^\star - \boldsymbol{\lambda}_0\|^2 + \frac{\gamma_0}{2\tau_0}\|\boldsymbol{w}^\star - \boldsymbol{w}_0\|^2 + \sum_{k=0}^{K-1} \left(\delta_k\gamma_k + \frac{\gamma_k}{2}\sqrt{2(\mu + \tau_k^{-1})\delta_k} D \right),$$

where $\boldsymbol{w}^\star = \arg\min_{\boldsymbol{w}} R_{\boldsymbol{\sigma}}(\boldsymbol{w}) + g(\boldsymbol{w})$ and $\boldsymbol{\lambda}^\star = \boldsymbol{\sigma}_{\pi^{-1}}$. Here, π is the permutation that arranges $\ell_1(\boldsymbol{w}^\star), \ldots, \ell_n(\boldsymbol{w}^\star)$ in ascending order, that is, $\ell_{\pi(1)}(\boldsymbol{w}^\star) \leq \cdots \leq \ell_{\pi(n)}(\boldsymbol{w}^\star)$.

Proof: Taking expectations with respect to $\boldsymbol{w}_k, \ldots, \boldsymbol{w}_1$ in (14) and using the law of total expectation yields

$$\begin{aligned}
&\mathbb{E}\left\{L(\boldsymbol{w}_{k+1}, \boldsymbol{\lambda}) - L(\boldsymbol{w}, \boldsymbol{\lambda}_{k+1})\right\} \\
\leq &\mathbb{E}\left\{ \frac{1}{2\eta_k}\left[\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_k\|^2 - \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1}\|^2\right] - \frac{1}{2}\left[\frac{1}{\eta_k} - \frac{\sqrt{n}G\theta_k}{\alpha_k}\right]\|\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k+1}\|^2 \right. \\
&+ \langle \boldsymbol{\ell}(\boldsymbol{w}_{k+1}) - \boldsymbol{\ell}(\boldsymbol{w}_k), \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1}\rangle - \theta_k\langle \boldsymbol{\ell}(\boldsymbol{w}_k) - \boldsymbol{\ell}(\boldsymbol{w}_{k-1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_k\rangle \\
&+ \frac{1}{2\tau_k}\|\boldsymbol{w} - \boldsymbol{w}_k\|^2 - \frac{1}{2}\left(\frac{1}{\tau_k} + \mu - \sqrt{2(\mu + \tau_k^{-1})\delta_k} D^{-1}\right)\|\boldsymbol{w} - \boldsymbol{w}_{k+1}\|^2 \\
&\left. - \frac{1}{2\tau_k}\|\boldsymbol{w}_k - \boldsymbol{w}_{k+1}\|^2 + \frac{\sqrt{n}G\theta_k\alpha_k}{2}\|\boldsymbol{w}_k - \boldsymbol{w}_{k-1}\|^2 + \delta_k + \frac{1}{2}\sqrt{2(\mu + \tau_k^{-1})\delta_k} D \right\}.
\end{aligned} \tag{19}$$

Multiplying both sides of (19) by γ_k and summing over $k = 0$ to $K - 1$ we obtain that

$$
\begin{aligned}
&\sum_{k=0}^{K-1} \gamma_k \mathbb{E}\left\{L(\boldsymbol{w}_{k+1}, \boldsymbol{\lambda}) - L(\boldsymbol{w}, \boldsymbol{\lambda}_{k+1})\right\} \\
\leq \mathbb{E}\Bigg\{ & \frac{\gamma_0}{2\eta_0}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_0\|^2 + \sum_{k=0}^{K-2} \underbrace{\frac{1}{2}\left(\frac{\gamma_{k+1}}{\eta_{k+1}} - \frac{\gamma_k}{\eta_k}\right)}_{A} \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1}\|^2 - \frac{\gamma_{K-1}}{2\eta_{K-1}}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_K\|^2 \\
& + \frac{\gamma_0}{2\tau_0}\|\boldsymbol{w} - \boldsymbol{w}_0\|^2 + \sum_{k=0}^{K-2} \underbrace{\frac{1}{2}\left[\frac{\gamma_{k+1}}{\tau_{k+1}} - \gamma_k\left(\frac{1}{\tau_k} + \mu - \sqrt{2(\mu + \tau_k^{-1})\delta_k} D^{-1}\right)\right]}_{B} \|\boldsymbol{w} - \boldsymbol{w}_{k+1}\|^2 \\
& - \frac{\gamma_{K-1}}{2}\left(\frac{1}{\tau_{K-1}} + \mu - \sqrt{2(\mu + \tau_{K-1}^{-1})\delta_{K-1}} D^{-1}\right)\|\boldsymbol{w} - \boldsymbol{w}_K\|^2 \\
& + \sum_{k=0}^{K-2} \underbrace{(\gamma_k - \gamma_{k+1}\theta_{k+1})}_{C} \langle \boldsymbol{\ell}(\boldsymbol{w}_{k+1}) - \boldsymbol{\ell}(\boldsymbol{w}_k), \boldsymbol{\lambda} - \boldsymbol{\lambda}_{k+1}\rangle + \gamma_{K-1}\langle \boldsymbol{\ell}(\boldsymbol{w}_K) - \boldsymbol{\ell}(\boldsymbol{w}_{K-1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_K\rangle \\
& + \frac{1}{2}\sum_{k=0}^{K-2} \underbrace{\left(\gamma_{k+1}\theta_{k+1}\alpha_{k+1}\sqrt{n}G - \frac{\gamma_k}{\tau_k}\right)}_{D} \|\boldsymbol{w}_k - \boldsymbol{w}_{k+1}\|^2 - \frac{\gamma_{K-1}}{2\tau_{K-1}}\|\boldsymbol{w}_K - \boldsymbol{w}_{K-1}\|^2 \\
& + \frac{1}{2}\sum_{k=0}^{K-1} \underbrace{\left[-\gamma_k\left(\frac{1}{\eta_k} - \theta_k\frac{\sqrt{n}G}{\alpha_k}\right)\right]}_{E} \|\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k+1}\|^2 + \sum_{k=0}^{K-1}\left(\delta_k\gamma_k + \frac{\gamma_k}{2}\sqrt{2(\mu + \tau_k^{-1})\delta_k} D\right)\Bigg\}.
\end{aligned}
$$

Here we use $\boldsymbol{\ell}(\boldsymbol{w}_0) - \boldsymbol{\ell}(\boldsymbol{w}_{-1}) = 0$ by $\boldsymbol{w}_0 = \boldsymbol{w}_{-1}$ and $\boldsymbol{\lambda}_0 = \boldsymbol{\lambda}_{-1}$. By Condition 1, we have $A, B, D, E \leq 0$ and $C = 0$.

Then we have

$$
\begin{aligned}
&\sum_{k=0}^{K-1} \gamma_k \mathbb{E}\left\{L(\boldsymbol{w}_{k+1}, \boldsymbol{\lambda}) - L(\boldsymbol{w}, \boldsymbol{\lambda}_{k+1})\right\} \\
\leq \mathbb{E}\Bigg\{ & \frac{\gamma_0}{2\eta_0}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_0\|^2 - \frac{\gamma_{K-1}}{2\eta_{K-1}}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_K\|^2 + \frac{\gamma_0}{2\tau_0}\|\boldsymbol{w} - \boldsymbol{w}_0\|^2 \\
& - \frac{\gamma_{K-1}}{2}\left(\frac{1}{\tau_{K-1}} + \mu - \sqrt{2(\mu + \tau_{K-1}^{-1})\delta_{K-1}} D^{-1}\right)\|\boldsymbol{w} - \boldsymbol{w}_K\|^2 \\
& + \gamma_{K-1}\langle \boldsymbol{\ell}(\boldsymbol{w}_K) - \boldsymbol{\ell}(\boldsymbol{w}_{K-1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_K\rangle - \frac{\gamma_{K-1}}{2\tau_{K-1}}\|\boldsymbol{w}_K - \boldsymbol{w}_{K-1}\|^2 \\
& + \sum_{k=0}^{K-1}\left(\delta_k\gamma_k + \frac{\gamma_k}{2}\sqrt{2(\mu + \tau_k^{-1})\delta_k} D\right)\Bigg\}.
\end{aligned} \tag{20}
$$

Next we bound $\gamma_{K-1}\langle \boldsymbol{\ell}(\boldsymbol{w}_K) - \boldsymbol{\ell}(\boldsymbol{w}_{K-1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_K\rangle$ similar to (16). We have

$$
\langle \boldsymbol{\ell}(\boldsymbol{w}_K) - \boldsymbol{\ell}(\boldsymbol{w}_{K-1}), \boldsymbol{\lambda} - \boldsymbol{\lambda}_K\rangle \leq \frac{\sqrt{n}G\alpha_K}{2}\|\boldsymbol{w}_K - \boldsymbol{w}_{K-1}\|^2 + \frac{1}{2}\frac{\sqrt{n}G}{\alpha_K}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_K\|^2.
$$

Taking the expectation and plugging this into (20), we obtain that

$$\begin{aligned}
&\sum_{k=0}^{K-1}\gamma_k\mathbb{E}\left\{L(\boldsymbol{w}_{k+1},\boldsymbol{\lambda})-L(\boldsymbol{w},\boldsymbol{\lambda}_{k+1})\right\}\\
\leq&\mathbb{E}\left\{\frac{\gamma_0}{2\eta_0}\|\boldsymbol{\lambda}-\boldsymbol{\lambda}_0\|^2-\frac{1}{2}\underbrace{\left[\frac{\gamma_{K-1}}{\eta_{K-1}}-\gamma_{K-1}\frac{\sqrt{n}G}{\alpha_K}\right]}_{\tilde{A}}\|\boldsymbol{\lambda}-\boldsymbol{\lambda}_K\|^2\right.\\
&+\frac{\gamma_0}{2\tau_0}\|\boldsymbol{w}-\boldsymbol{w}_0\|^2-\underbrace{\frac{\gamma_{K-1}}{2}\left(\frac{1}{\tau_{K-1}}+\mu-\sqrt{2(\mu+\tau_{K-1}^{-1})\delta_{K-1}}D^{-1}\right)}_{\tilde{B}}\|\boldsymbol{w}-\boldsymbol{w}_K\|^2\\
&\left.+\underbrace{\frac{\gamma_{K-1}}{2}\left(\alpha_K\sqrt{n}G-\frac{1}{\tau_{K-1}}\right)}_{\tilde{C}}\|\boldsymbol{w}_K-\boldsymbol{w}_{K-1}\|^2+\sum_{k=0}^{K-1}\left(\delta_k\gamma_k+\frac{\gamma_k}{2}\sqrt{2(\mu+\tau_k^{-1})\delta_k}D\right)\right\}.
\end{aligned}\tag{21}$$

We analyze $\tilde{A}$-$\tilde{D}$ under Condition 1:

$$\begin{aligned}
&\tilde{A}\overset{(18a)}{\geq}\left[\frac{\gamma_K}{\eta_K}-\gamma_{K-1}\frac{\sqrt{n}G}{\alpha_K}\right]\overset{(18c)}{=}\gamma_K\left[\frac{1}{\eta_K}-\theta_K\frac{\sqrt{n}G}{\alpha_K}\right]\overset{(18e)}{\geq}0,\\
&\tilde{B}\overset{(18b)}{\geq}\frac{\gamma_K}{2\tau_K},\\
&\tilde{C}\overset{(18d)}{\leq}0.
\end{aligned}$$

We obtain that

$$\begin{aligned}
&\sum_{k=0}^{K-1}\gamma_k\mathbb{E}\left\{L(\boldsymbol{w}_{k+1},\boldsymbol{\lambda})-L(\boldsymbol{w},\boldsymbol{\lambda}_{k+1})\right\}\\
\leq&\frac{\gamma_0}{2\eta_0}\|\boldsymbol{\lambda}-\boldsymbol{\lambda}_0\|^2+\frac{\gamma_0}{2\tau_0}\|\boldsymbol{w}-\boldsymbol{w}_0\|^2-\frac{\gamma_K}{2\tau_K}\mathbb{E}\|\boldsymbol{w}-\boldsymbol{w}_K\|^2+\sum_{k=0}^{K-1}\left(\delta_k\gamma_k+\frac{\gamma_k}{2}\sqrt{2(\mu+\tau_k^{-1})\delta_k}D\right).
\end{aligned}\tag{22}$$

For any $\boldsymbol{w}\in\mathbb{R}^d$ and $\boldsymbol{\lambda}\in\Pi_{\boldsymbol{\sigma}}$, we have $L(\boldsymbol{w}^\star,\boldsymbol{\lambda}^\star)=\max_{\boldsymbol{\lambda}\in\Pi_{\boldsymbol{\sigma}}}L(\boldsymbol{w}^\star,\boldsymbol{\lambda})\geq L(\boldsymbol{w}^\star,\boldsymbol{\lambda})$. On the other hand, we have $L(\boldsymbol{w},\boldsymbol{\lambda}^\star)\geq L(\boldsymbol{w}^\star,\boldsymbol{\lambda}^\star)=\min_{\boldsymbol{w}}L(\boldsymbol{w},\boldsymbol{\lambda}^\star)$. Thus we obtain that $L(\boldsymbol{w}_{k+1},\boldsymbol{\lambda}^\star)-L(\boldsymbol{w}^\star,\boldsymbol{\lambda}_{k+1})\geq 0$ for $\forall k=0,\ldots,K-1$. Let $\boldsymbol{w}=\boldsymbol{w}^\star$ and $\boldsymbol{\lambda}=\boldsymbol{\lambda}^\star$ in (22) we get the desired result. □

Now we are ready to prove Theorem 1. By choosing appropriate parameters in Algorithm 1 to satisfy Condition 1, we can achieve the desired convergence rate.

Proof of Theorem 1.

Proof: First, we obtain an δ_k approximate solution to (11) through T_k updates to $\boldsymbol{w}$ in Algorithm 1 by Lemma 3. We then verify that Condition 1 is satisfied by the parameters.

It is not hard to see that $\frac{\gamma_{k+1}}{\gamma_k}=\frac{\eta_{k+1}}{\eta_k}=\frac{k+2}{k+1}$ and $\theta_{k+1}=\frac{\gamma_k}{\gamma_{k+1}}=\frac{k+1}{k+2}$. Thus (18a) and (18c) are satisfied.

Since $\delta_k\leq\frac{\mu}{8(k+5)}D^2$, we have $\sqrt{2(\mu+\tau_k^{-1})\delta_k}D^{-1}\leq\sqrt{2\mu(1+\frac{k+1}{4})\frac{\mu D^2}{8(k+5)}}D^{-1}\leq\frac{\mu}{4}$. Then we obtain that

$$\frac{\gamma_{k+1}}{\gamma_k\tau_{k+1}}=\frac{k+2}{4}\mu+\frac{k+2}{4(k+1)}\mu\leq\frac{k+4}{4}\mu,$$

and

$$\frac{1}{\tau_k}+\mu-\sqrt{2(\mu+\tau_k^{-1})\delta_k}D^{-1}\geq\frac{k+1}{4}\mu+\mu-\frac{\mu}{4}=\frac{k+4}{4}\mu.$$

Thus (18b) holds.

Furthermore, (18d) and (18e) hold due to $\sqrt{n}G\alpha_{k+1} = nG^2\eta_k = \frac{k+1}{8}\mu \leq \frac{k+1}{4}\mu = \frac{1}{\tau_k}$ and $\theta_k \frac{\sqrt{n}G}{\alpha_k} = \frac{\eta_{k-1}}{\eta_k}\frac{\sqrt{n}G}{\sqrt{n}G\eta_{k-1}} = \frac{1}{\eta_k}$.

Now Condition 1 is satisfied. By Lemma 5, we have

$$\frac{\gamma_K}{2\tau_K}\mathbb{E}\left\|\boldsymbol{w}-\boldsymbol{w}_K\right\|^2 \leq \frac{\gamma_0}{2\eta_0}\|\boldsymbol{\lambda}^\star-\boldsymbol{\lambda}_0\|^2+\frac{\gamma_0}{2\tau_0}\|\boldsymbol{w}^\star-\boldsymbol{w}_0\|^2+\sum_{k=0}^{K-1}\left(\delta_k\gamma_k+\frac{\gamma_k}{2}\sqrt{2(\mu+\tau_k^{-1})\delta_k}D\right).$$

Since $\delta_k \leq \mu(k+1)^{-6}D^2$, we have $\sum_{k=0}^{\infty}\delta_k\gamma_k \leq \mu D^2\sum_{k=0}^{\infty}(k+1)^{-5} \leq \frac{\mu}{4}D^2$, and

$$\sum_{k=0}^{\infty}\gamma_k\sqrt{(\mu+\tau_k^{-1})\delta_k}D \leq \frac{\sqrt{2}\mu}{4}D^2\sum_{k=0}^{\infty}(k+1)^{-2}\sqrt{k+5} \leq \frac{\sqrt{2}\mu}{4}D^2\sum_{k=0}^{\infty}(k+1)^{-2}\left(\sqrt{k+1}+2\right) \leq \sqrt{2}\mu D^2.$$

Finally, by $\frac{\gamma_K}{2\tau_K} = \frac{\mu(K+1)^2}{8}$, $\tau_0 = \frac{4}{\mu}$, $\eta_0 = \frac{\mu}{8nG^2}$ and $D = \frac{G}{\mu}$, we get the desired result. □

A.3 Proof of Corollary 1

Proof: Recall that $\tau_k = \frac{4}{\mu(k+1)}$. It is not hard to see that $\frac{L\tau_k}{\mu\tau_k+1} = \frac{4L}{\mu(k+5)} \leq \frac{4L}{\mu(k+1)}$. By Lemma 3, we get a δ_k approximate solution with the sample complexity of $C_{\boldsymbol{w}_{k+1}} = O\left(\left(n+\frac{L}{\mu(k+1)}\right)\log\left(\delta_k^{-1}\right)\right)$. We set $\delta_k = D^2\min\left(\frac{\mu}{8(k+5)}, \mu(k+1)^{-6}\right) = \frac{G^2}{\mu}(k+1)^{-6}$ for $k \geq 1$. And $\delta_0 = \mu/40$. The total sample complexity is

$$\begin{aligned}\sum_{k=0}^{K-1}C_{\boldsymbol{w}_{k+1}} &= \sum_{k=0}^{K-1}O\left(\left(n+\frac{L}{\mu(k+1)}\right)\left(\log(k+1)+\log\left(\frac{\mu}{G^2}\right)\right)\right)\\ &= O\left(nK\log\frac{\mu K}{G^2}+\frac{L}{\mu}\log K\log\frac{\mu K}{G^2}\right).\end{aligned}$$

In the last equality, we calculate $\sum_{k=1}^{K}\frac{\log k}{k} = O\left((\log K)^2\right)$, $\sum_{k=1}^{K}\log k = O\left(K\log K\right)$ and $\sum_{k=1}^{K}\frac{1}{k} = O(\log K)$. By Theorem 1, to achieve an ϵ-optimal solution, we need $K = O\left(\frac{\sqrt{n}G}{\mu\sqrt{\epsilon}}\right)$. Therefore, the total sample complexity is $O\left(\frac{n^{\frac{3}{2}}G}{\mu\sqrt{\epsilon}}\log\frac{\sqrt{n}}{G\sqrt{\epsilon}}+\frac{L}{\mu}\log\frac{\sqrt{n}}{G\sqrt{\epsilon}}\log\frac{\sqrt{n}G}{\mu\sqrt{\epsilon}}\right)$. □

B Experimental Details and Additional Experimental Results

We now outline the details of our experimental setup. Our experimental setup mainly follows that of Mehta et al. (2024).

Datasets. We use the same five datasets from the regression task in Section 5.1 and the `amazon` dataset in Section 5.3 as in Mehta et al. (2024). The statistical characteristics are summarized in Table 3.

Other two datasets in Section 5.2 are as follows:

1. `acs`: predicting whether an American adult is employed.
2. `law`: predicting a student's GPA.

In the experiments, we normalize the features of the sample matrix $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ so that each feature has a mean of 0 and a variance of 1. The test sets are normalized using the statistics of the training set.

Dataset	# features	# samples	Source
yacht	6	244	Tsanas & Xifara (2012)
energy	8	614	Baressi Šegota et al. (2019)
concrete	8	824	Yeh (2006)
kin8nm	8	6,553	Akujuobi & Zhang (2017)
power	4	7,654	Tüfekci (2014)
acs	16	10000	Ding et al. (2021)
law	10	20800	Wightman (1998)
amazon	535	20000	Mehta et al. (2024),Ni et al. (2019)

Table 3: Statistical details of real datasets and sources.

Objectives. We use linear models in our experiments. For spectral risks, we adopt three types: ESRM, Extremile, and CVaR, as specified in Table 1. Additionally, we set the regularizer $g(\boldsymbol{w})$ to $\frac{\mu}{2}\|\boldsymbol{w}\|^2$ with $\mu = \frac{1}{n}$, where n donates the number of sample points in the taining set. Thus, Problem (1) can be written as

$$\min_{\boldsymbol{w}} \sum_{i=1}^{n} \sigma_i \ell_{[i]}(\boldsymbol{w}) + \frac{\mu}{2}\|\boldsymbol{w}\|^2,$$

where $\ell_i(\cdot)$ is the loss function, which will be chosen in different forms for different tasks.

Hyperparameter Selection. We use similar hyperparameter selection method as in Mehta et al. (2024). We set the batch size for SGD to 64. For the selection of step size α, we set the random seed $s \in \{1, \ldots, S\}$. For a single seed s, we calculate the average training loss of the last ten epochs, donated by $L_s(\alpha)$. We choose α that minimizes $\frac{1}{S}\sum_{s=1}^{S} L_s(\alpha)$, where $\alpha \in \{1\times10^{-4}, 3\times10^{-4}, 1\times10^{-3}, 3\times10^{-3}, 1\times10^{-2}, 3\times10^{-2}, 1\times10^{-1}, 3\times10^{-1}\}$. For LSVRG, we set the length of an epoch to n. For SOREL, we set $T_k = m_k = n$. Moreover, we set batch size to 64 for all algorithms with mini-batching.

For SOREL, we follow the parameter values given in Theorem 1. In particular, we set $\theta_k = \frac{k}{k+1}$ and $\tau_k = \frac{20n}{k+1}$ in all experiments. Therefore, there are only two parameters α and η_k left to tune. We set $\eta_k = \frac{C(k+1)}{n}$ and choose C from $\{1\times10^{-2}, 2\times10^{-2}, 4\times10^{-2}, 1\times10^{-1}, 2\times10^{-1}, 4\times10^{-1}, 1\times10^{0}, 2\times10^{0}, 4\times10^{0}, 1\times10^{1}\}$, with two orders of magnitude higher numbers used in law, since the Lipschitz constant G is hard to estimate. We use grid search to select α and C, with the selection criteria being the same as the previous paragraph. We apply stochastic gradient descent to solve (5) instead of proximal stochastic gradient descent.

Experimental Environment. We run all experiments on a laptop with 16.0 GB RAM and Intel i7-1360P 2.20 GHz CPU. All algorithms are implemented in Python 3.8.

B.1 Experimental Details on Linear Regression

Dataset. We use the same dataset as that used in Mehta et al. (2024), as previously described.

Objectives. We use the least square loss in this experiment. For spectral risks, we adopt three types: ESRM ($\rho = 2$), Extremile ($r = 2.5$), and CVaR ($\alpha = 0.5$).

Evaluation. We set random seeds $s \in \{1, 2, 3, 4, 5\}$ as the seeds for the random algorithms. We compare the suboptimality versus passes (the number of samples divided by n) and runtime. The suboptimality is defined as

$$\text{Suboptimality}(\boldsymbol{w}_k) = \frac{R_{\boldsymbol{\sigma}}(\boldsymbol{w}_k) + g(\boldsymbol{w}_k) - R_{\boldsymbol{\sigma}}(\boldsymbol{w}^\star) - g(\boldsymbol{w}^\star)}{R_{\boldsymbol{\sigma}}(\boldsymbol{w}_0) + g(\boldsymbol{w}_0) - R_{\boldsymbol{\sigma}}(\boldsymbol{w}^\star) - g(\boldsymbol{w}^\star)},$$

where $\boldsymbol{w}^\star$ is calculated by L-BFGS (Nocedal & Wright, 1999).

B.2 Experimental Details on Fair Machine Learning

Dataset. We use two datasets, `acs` (Ding et al., 2021) and `law` (Wightman, 1998), which are for classification and regression tasks, respectively. For `acs`, we randomly selected $10,000$ sample points from the California data. We use data from four states Connecticut, Hawaii, West Virginia and Florida in Ding et al. (2021) as the test dataset to explore the out-of-distribution performance of models trained with spectral risks, with each state having 36287, 14400, 18066, and 202160 sample points, respectively.

Objectives. For the regression and classification tasks, we use the least squares loss and the binary logistic loss, respectively. For `acs`, we set the spectral risks to CVaR ($\alpha = 0.75$), ESRM ($\rho = 1.75$) and Extremile ($r = 2.1$). For `law`, we set the spectral risks to CVaR ($\alpha = 0.05$), ESRM ($\rho = 20$) and Extremile ($r = 10$).

Evaluation. We fix the number of training passes at 100. We split the training set and test set in a 4:1 ratio and used five-fold cross-validation to report the average results on the test set. For each training and test set split, we set random seeds $s \in \{1, 2, 3\}$ as the seeds for the random algorithms. All algorithms are implemented using mini-batching. We set race as the sensitive feature. For `acs`, the sensitive feature includes Black and White. For `law`, the sensitive feature includes non-White and White. For the task of exploring models' out-of-domain performance, we directly use the models obtained from the `acs` experiments as the models trained on California dataset. Then, we test these models on all data points from the other four states.

B.3 Experimental Details on Out-of-Distribution Generalization

Dataset. `amazon` (Ni et al., 2019) is for the multi-class classification task. We use the preprocessed data in Mehta et al. (2024). They fine-tuned a BERT model on 10, 000 held-out examples and applied PCA to the deep representations produced by BERT. The training set and test set each contain 10,000 samples. #features in Table 3 refers to the total dimension of the parameter vectors for all 5 classes.

Objectives. We use a linear model and the multinomial logistic loss. In `amazon`, we set the spectral risks to CVaR ($\alpha = 0.75$) and Extremile ($r = 2.0$).

Evaluation. We set random seeds $s \in \{1, 2, 3, 4, 5\}$ as the seeds for the random algorithms. We evaluate the worst group classification error (Sagawa et al., 2020) on the test set. Each group is classified based on the true labels. We fix the number of passes during training to 500 and report the average worst group classification error of the last ten passes. All algorithms are implemented using mini-batching.

B.4 Additional Experimental Results

Algorithms with mini-batching. In Figure 5, we present results of the algorithms with mini-batching for tasks in Section 5.1. Mini-batching has a significant improvement on the convergence rate of all the algorithms. Similar to what is shown in Figure 2, SGD, LSVEG and Prospect fail to converge to the true optimal points, especially in the first two datasets. SOREL converges to the optimal solutions in all settings, and achieves the best or competitive results, in terms of sample complexity, runtime, or both, except in the setting of CVaR and `power` dataset. Still SOREL performs competitively for the suboptimality of 10^{-7} in this setting.

Training curves in fair machine learning. Figure 6 shows the training curves for the task in Section 5.2. The training curves are extended to illustrate convergence. SOREL is able to achieve low suboptimality in the shortest amount of time.

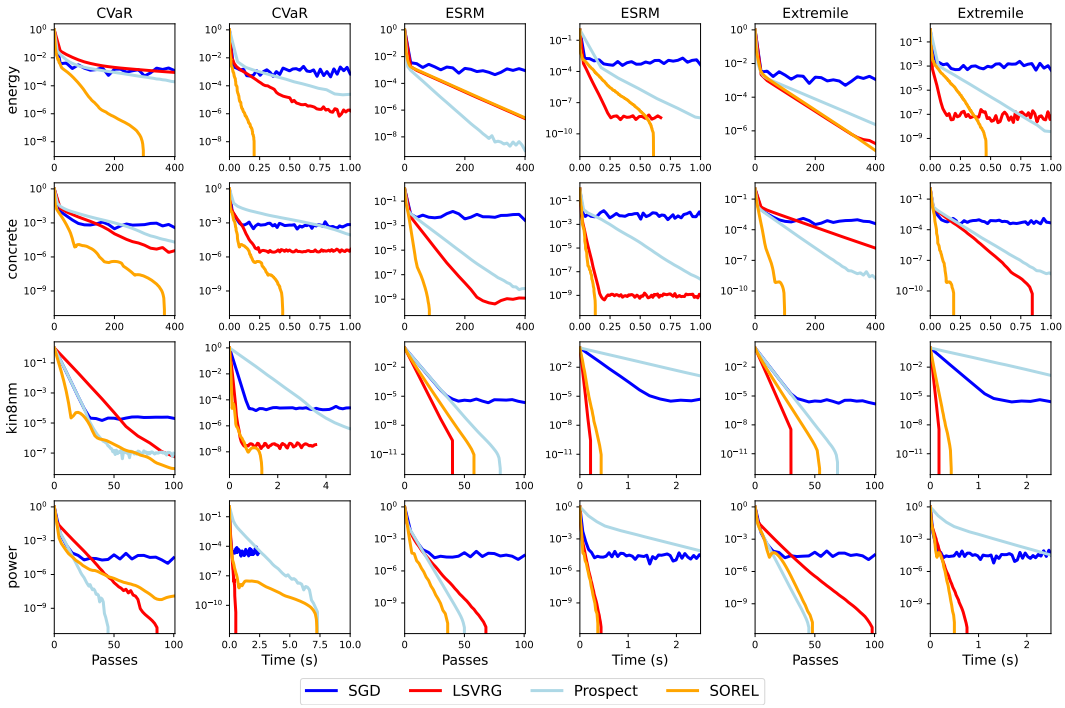

Figure 5: Suboptimality of spectral risks for different algorithms **with mini-batching**. The x-axis represents the effective number of samples used by the algorithm divided by n (odd columns) or CPU time (even columns).

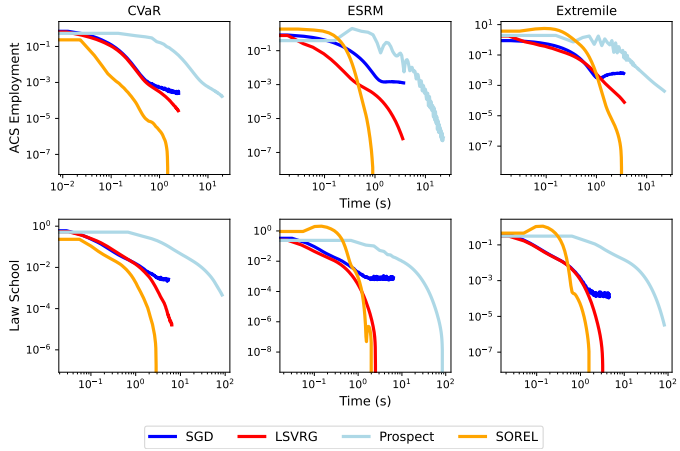

Figure 6: Suboptimality of spectral risks for different algorithms on fairness benchmarks. The x-axis represents the CPU time.

C EXAMPLE

To illustrate the necessity of stabilizing the trajectory of the primal variable in Section 3.1, we provide a toy example. For simplicity, we consider a one-dimensional problem

$$\min_{w\in\mathbb{R}} \sigma_1 \ell_{[1]}(w) + \sigma_2 \ell_{[2]}(w), \tag{23}$$

where $\sigma_1 = 0, \sigma_2 = 1$ and $\ell_1 = \frac{1}{2}(w-1)^2, \ell_2(w) = \frac{1}{2}(w+1)^2$. We use the following deterministic method, similar to Algorithm 1.

Example 1 *For any $0 < \alpha < 2$, suppose we solve Problem (23) using Algorithm 2 and T is sufficiently large. In that case, the iterative sequence $\{w_k\}$ can not converge to the optimal solution for any initial point w_0.*

Algorithm 2 Simplified Algorithm for Solving the Example Problem.

1: **for** $k = 0, 1, \ldots$ **do**
2: Update $\{\lambda_{k+1,1}, \lambda_{k+1,2}\} = \{\sigma_1, \sigma_2\}$ if $\ell_1(w_k) \geq \ell_2(w_k)$ else $\{\sigma_2, \sigma_1\}$. Set $w_k^0 = w_k$.
3: **for** $t = 0, 1, \ldots, T-1$ **do**
4: Compute the gradient $g^t = \lambda_{k+1,1}\nabla\ell_1(w_k^t) + \lambda_{k+1,2}\nabla\ell_2(w_k^t)$.
5: Update $w^{t+1} = w^t - \alpha g^t$.
6: **end for**
7: Set $w_{k+1} = w_k^T$.
8: **end for**

Without loss of generality, we assume $w_0 > 0$, in which case $\boldsymbol{\lambda}_1 = [0, 1]^\top$. We solve $\min_{w_1 \in \mathbb{R}} \frac{1}{2}(w_1 + 1)^2$ through sufficient steps of gradient descent to obtain $w_1 = -1$. At this point, $\boldsymbol{\lambda}_2 = [1, 0]^\top$. By iterating this process, w_k always oscillates between -1 and 1, unable to converge to $w^\star = 0$. If $w_0 = 0$, we set $\sigma_1 = 1$ and $\sigma_2 = 0$, reaching the same conclusion. A similar conclusion can be extended to stochastic methods in the expectation sense.

We know that $\ell_1(w^\star) = \ell_2(w^\star)$ at the optimal point $w^\star = 0$. Clearly, the iterative sequence of the algorithm oscillates at $w^\star$ and cannot converge to the optimal solution. Although Mehta et al. (2022; 2024) employ a similar approach to update $\boldsymbol{\lambda}$ for subgradient estimations, they consider the smoothed spectral risk by adding a strongly concave term with respect to $\boldsymbol{\lambda}$. However, for the original spectral risk minimization problems, updating $\boldsymbol{\lambda}$ with their method results in discontinuities, thereby lacking convergence guarantees.

D SOREL WITH MINI-BATCHING

In this section, we present the results of SOREL with mini-batching. To apply mini-batching, We only need to change Line 13 of Algorithm 1 to: Sample a mini-batch $b_t \subset \{1, \ldots, n\}$ without replacement, and change Line 14 to

$$\boldsymbol{d}_{k,t} = \frac{1}{b}\sum_{i \in b_t} \left[n\lambda_{i,k+1}\nabla\ell_i\left(\boldsymbol{w}_{k,t}\right) - n\lambda_{i,k+1}\nabla\ell_i(\bar{\boldsymbol{w}})\right] + \bar{\boldsymbol{g}},$$

where $b = |b_t|$ is the mini-batch size.

We first present the main result of SOREL with mini-batching.

Corollary 2 *Use the same conditions in Therorem 1. Additionally, set the step-size* $\alpha = \frac{b(n-1)}{5L(n-b)}$, $m_k = \frac{400L(n-b)}{(k+5)\mu b(n-1)} + 8$ *and* $T_k = O(m_k \log \frac{1}{\delta_k})$. *Then we obtain an output* $\boldsymbol{w}_K$ *of SOREL with mini-batching such that* $\mathbb{E}\|\boldsymbol{w}_k - \boldsymbol{w}^\star\|^2 \leq \epsilon$ *in a total sample complexity of* $O\left(\frac{n^{\frac{3}{2}}G}{\mu\sqrt{\epsilon}} \log \frac{\sqrt{n}}{G\sqrt{\epsilon}} + \frac{L(n-b)}{\mu(n-1)} \log \frac{\sqrt{n}}{G\sqrt{\epsilon}} \log \frac{\sqrt{n}G}{\mu\sqrt{\epsilon}}\right)$.

D.1 PROOF OF COROLLARY 2

We first discuss the inner loop in Lines 6-17 of Algorithm 1. This is an extension of SVRG (Johnson & Zhang, 2013; Xiao & Zhang, 2014). To illustrate more clearly, we consider the problem

$$\min_{\boldsymbol{w}} P(\boldsymbol{w}) := F(\boldsymbol{w}) + h(\boldsymbol{w}),$$

where $F(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^n f_i(\boldsymbol{w})$, each f_i is convex and L-smooth, and h is μ-strongly convex. We rewrite Lines 6-17 of Algorithm 1 to Algorithm 3. Note that, by setting $\bar{\boldsymbol{w}}_0 = \boldsymbol{w}_k$, $m = m_k$, $F(\boldsymbol{w}) = \boldsymbol{\lambda}_{k+1}^\top \ell(\boldsymbol{w})$, and $h(\boldsymbol{w}) = g(\boldsymbol{w}) + \frac{1}{2\tau_k}\|\boldsymbol{w} - \boldsymbol{w}_k\|^2$, Algorithm 3 is the same as Lines 6-17 of Algorithm 1.

Assumption 2 *Each* $f_i : \mathbb{R}^d \to \mathbb{R}$ *is convex and* L*-smooth.* $h : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$ *is proper, lower semicontinuous and* μ_h*-strongly convex.*

The following two results are adopted from Xiao & Zhang (2014), which will be used in the proof of the main result.

Algorithm 3 Simplified Inner Loop of Algorithm 1 with Mini-batching.

Input: initial $\bar{\boldsymbol{w}}_0$, the learning rate α, mini-batch size b, and the inner-loop length m.
for $s = 0, 1, \ldots$ **do**
  $\bar{\boldsymbol{w}} = \bar{\boldsymbol{w}}_{s-1}$, $\bar{\boldsymbol{g}} = \nabla F(\bar{\boldsymbol{w}})$.
  $\boldsymbol{w}_0 = \bar{\boldsymbol{w}}$.
  for $t = 0, 1, \ldots, m-1$ **do**
    Sample $b_t \subset \{1, \ldots, n\}$ of size b uniformly at random without replacement.
    $\boldsymbol{d}_t = \frac{1}{b}\sum_{i \in b_t} [\nabla f_i(\boldsymbol{w}_t) - \nabla f_i(\bar{\boldsymbol{w}})] + \bar{\boldsymbol{g}}$.
    $\boldsymbol{w}_{t+1} = \mathrm{Prox}_{\alpha h}\{\boldsymbol{w}_t - \alpha \boldsymbol{d}_t\}$.
  end for
  $\bar{\boldsymbol{w}}_s = \frac{1}{m}\sum_{t=1}^{m} \boldsymbol{w}_t$.
end for

Lemma 6 (Xiao & Zhang, 2014, Lemma 1) *Suppose Assumption 2 holds, and let* $\boldsymbol{w}_\star = \arg\min_{\boldsymbol{w}} P(\boldsymbol{w})$. *Then for all* $\boldsymbol{w} \in \mathbb{R}^d$

$$\frac{1}{n}\|\nabla f_i(\boldsymbol{w}) - \nabla f_i(\boldsymbol{w}_\star)\|^2 \leq 2L\left(P(\boldsymbol{w}) - P(\boldsymbol{w}_\star)\right).$$

Lemma 7 (Xiao & Zhang, 2014, Lemma 3) *Suppose Assuption 2 holds, let* $\Delta_t = \boldsymbol{d}_t - \nabla F(\boldsymbol{w}_t)$ *and* $\boldsymbol{w}_\star = \arg\min_{\boldsymbol{w}} P(\boldsymbol{w})$. *Then*

$$\|\boldsymbol{w}_{t+1} - \boldsymbol{w}_\star\|^2 \leq \|\boldsymbol{w}_t - \boldsymbol{w}_\star\|^2 - 2\alpha\left[P\left(\boldsymbol{w}_{t+1}\right) - P\left(\boldsymbol{w}_\star\right)\right] - 2\alpha\Delta_t^\top\left(\boldsymbol{w}_{t+1} - \boldsymbol{w}_\star\right). \tag{24}$$

The following lemma bounds the variance of the stochastic gradient $\boldsymbol{d}_t$.

Lemma 8 *Let* $\mathbb{E}_t$ *be the conditional expectation given* $\boldsymbol{w}_t$ *and* $\boldsymbol{w}_\star = \arg\min_{\boldsymbol{w}} P(\boldsymbol{w})$. *We have*

$$\mathbb{E}_t\|\boldsymbol{d}_t - \nabla F(\boldsymbol{w}_t)\|^2 \leq \frac{2(n-b)L}{b(n-1)}\left(P(\boldsymbol{w}_t) - P(\boldsymbol{w}_\star) + P(\bar{\boldsymbol{w}}) - P(\boldsymbol{w}_\star)\right).$$

Proof: Define $\xi_i = \nabla f_i(\boldsymbol{w}_t) - \nabla f_i(\bar{\boldsymbol{w}})$.

$$\begin{aligned}
&\mathbb{E}_t\|\boldsymbol{d}_t - \nabla F(\boldsymbol{w}_t)\|^2 \\
&= \mathbb{E}_t\|\frac{1}{b}\sum_{i_t \in b_t}\left(\nabla f_{i_t}(\boldsymbol{w}_t) - \nabla f_{i_t}(\bar{\boldsymbol{w}})\right) + \frac{1}{n}\sum_{i=1}^{n}\left(\nabla f_i(\bar{\boldsymbol{w}}) - \nabla f_i(\boldsymbol{w}_t)\right)\|^2 \\
&= \mathbb{E}_t\|\frac{1}{b}\sum_{i_t \in b_t}\xi_{i_t}\|^2 - \|\frac{1}{n}\sum_{i=1}^{n}\xi_i\|^2 \\
&= \frac{1}{b^2}\mathbb{E}_t\left[\sum_{i_t \neq j_t \in b_t}\xi_{i_t}^\top\xi_{j_t} + \sum_{i_t \in b_t}\xi_{i_t}^\top\xi_{i_t}\right] - \frac{1}{n^2}\sum_{i,j=1}^{n}\xi_i^\top\xi_j \\
&= \frac{1}{b^2}\left(\frac{b(b-1)}{n(n-1)}\sum_{i \neq j}\xi_i^\top\xi_j + \frac{b}{n}\sum_{i=1}^{n}\xi_i^\top\xi_i\right) - \frac{1}{n^2}\sum_{i,j=1}^{n}\xi_i^\top\xi_j \\
&= \frac{1}{nb}\left(\frac{b-1}{n-1}\sum_{i,j=1}^{n}\xi_i^\top\xi_j + \left(1 - \frac{b-1}{n-1}\right)\sum_{i=1}^{n}\xi_i^\top\xi_i\right) - \frac{1}{n^2}\sum_{i,j=1}^{n}\xi_i^\top\xi_j \\
&= \frac{n-b}{nb(n-1)}\left(-\frac{1}{n}\sum_{i,j=1}^{n}\xi_i^\top\xi_j + \sum_{i=1}^{n}\xi_i^\top\xi_i\right),
\end{aligned}$$

where the second equation is due to $\mathbb{E}\|\xi - \mathbb{E}\xi\|^2 = \mathbb{E}\|\xi\|^2 - \|\mathbb{E}\xi\|^2$.

Note that $\frac{1}{n}\sum_{i,j=1}^{n}\xi_i^\top\xi_j = n\|\frac{1}{n}\sum_{i=1}^{n}\xi_i\|^2 \geq 0$. Combing the above result with Lemma 6 we obtain that

$$\begin{aligned}
\mathbb{E}_t\|\boldsymbol{d}_t - \nabla F(\boldsymbol{w}_t)\|^2 &\leq \frac{n-b}{bn(n-1)}\sum_{i=1}^{n}\|\nabla f_i(\boldsymbol{w}_t) - \nabla f_i(\bar{\boldsymbol{w}})\|^2 \\
&\leq \frac{n-b}{bn(n-1)}\sum_{i=1}^{n}\left(\|\nabla f_i(\boldsymbol{w}_t) - \nabla f_i(\boldsymbol{w}_\star)\|^2 + \|\nabla f_i(\bar{\boldsymbol{w}}) - \nabla f_i(\boldsymbol{w}_\star)\|^2\right) \\
&\leq \frac{2(n-b)L}{b(n-1)}\left(P(\boldsymbol{w}_t) - P(\boldsymbol{w}_\star) + P(\bar{\boldsymbol{w}}) - P(\boldsymbol{w}_\star)\right).
\end{aligned}$$

This completes the proof. □

Lemma 9 *Suppose Assumption 2 holds and* $2\alpha L\frac{n-b}{b(n-1)} < 1$. *Let* $\boldsymbol{w}_\star = \arg\min_{\boldsymbol{w}} P(\boldsymbol{w})$. *Then we obtain an output* $\bar{\boldsymbol{w}}_s$ *of Algorithm 3 such that*

$$\mathbb{E}P(\bar{\boldsymbol{w}}_s) - P(\boldsymbol{w}_\star) \leq \rho^s\left(P(\bar{\boldsymbol{w}}_0) - P(\boldsymbol{w}_\star)\right),$$

where $\rho = \frac{U}{D}$, $U = 2\alpha m\left(1 - 2\alpha L\frac{n-b}{b(n-1)}\right)$ *and* $D = \left(\frac{2}{\mu_h} + 4\alpha^2 L(m+1)\frac{n-b}{b(n-1)}\right)$.

Proof: We first consieder the s-th outer iteration of Algorithm 3. We have that $\bar{\boldsymbol{w}} = \bar{\boldsymbol{w}}_{s-1}$. We define $\widetilde{\boldsymbol{w}}_{t+1} = \mathrm{Prox}_{\alpha h}\left(\boldsymbol{w}_t - \alpha\nabla F(\boldsymbol{w}_t)\right)$, which is independent of the mini-batch b_t. First we bound the last term in (24):

$$\begin{aligned}
&-2\alpha\Delta_t^\top(\boldsymbol{w}_{t+1} - \boldsymbol{w}_\star) = -2\alpha\Delta_t^\top(\boldsymbol{w}_{t+1} - \widetilde{\boldsymbol{w}}_{t+1} + \widetilde{\boldsymbol{w}}_{t+1} - \boldsymbol{w}_\star) \\
&\leq 2\alpha\|\Delta_t\|\|\boldsymbol{w}_{t+1} - \widetilde{\boldsymbol{w}}_{t+1}\| - 2\alpha\Delta_t^\top(\widetilde{\boldsymbol{w}}_{t+1} - \boldsymbol{w}_\star) \\
&\leq 2\alpha\|\Delta_t\|\|\boldsymbol{w}_t - \alpha\boldsymbol{d}_t - \boldsymbol{w}_t + \alpha\nabla F(\boldsymbol{w}_t)\| - 2\alpha\Delta_t^\top(\widetilde{\boldsymbol{w}}_{t+1} - \boldsymbol{w}_\star) \\
&= 2\alpha^2\|\Delta_t\|^2 - 2\alpha\Delta_t^\top(\widetilde{\boldsymbol{w}}_{t+1} - \boldsymbol{w}_\star),
\end{aligned} \tag{25}$$

where in the second inequality we use the non-expansiveness of the projection operator. Note that $\mathbb{E}_t\Delta_t^\top(\widetilde{\boldsymbol{w}}_{t+1} - \boldsymbol{w}_\star) = \left(\mathbb{E}_t\Delta_t\right)^\top(\widetilde{\boldsymbol{w}}_{t+1} - \boldsymbol{w}_\star) = 0$. Taking the conditional expectation $\mathbb{E}_t$ on both sides of (25) and using Lemma 8 we obtain that

$$-2\alpha\mathbb{E}_t\Delta_t^\top(\boldsymbol{w}_{t+1} - \boldsymbol{w}_\star) \leq \frac{4\alpha^2(n-b)L}{b(n-1)}\left(P(\boldsymbol{w}_t) - P(\boldsymbol{w}_\star) + P(\bar{\boldsymbol{w}}_{s-1}) - P(\boldsymbol{w}_\star)\right).$$

Taking the conditional expectation $\mathbb{E}_t$ on both sides of (24) and plugging in the above result, we obtain that

$$\begin{aligned}
\mathbb{E}_t\|\boldsymbol{w}_{t+1} - \boldsymbol{w}_\star\|^2 \leq \|\boldsymbol{w}_t - \boldsymbol{w}_\star\|^2 &- 2\alpha(\mathbb{E}_t P(\boldsymbol{w}_{t+1}) - P(\boldsymbol{w}_\star)) \\
&+ \frac{4\alpha^2(n-b)L}{b(n-1)}\left(P(\boldsymbol{w}_t) - P(\boldsymbol{w}_\star) + P(\bar{\boldsymbol{w}}_{s-1}) - P(\boldsymbol{w}_\star)\right).
\end{aligned} \tag{26}$$

Taking the expectation on both sides of (26), using the law of total expectation and summing over $t = 0, \ldots, m-1$ we obtain that

$$\begin{aligned}
&\mathbb{E}\|\boldsymbol{w}_m - \boldsymbol{w}_\star\|^2 + 2\alpha(\mathbb{E}P(\boldsymbol{w}_m) - P(\boldsymbol{w}_\star)) + 2\alpha\left(1 - 2\alpha L\frac{n-b}{b(n-1)}\right)\sum_{t=1}^{m-1}(\mathbb{E}P(\boldsymbol{w}_t) - P(\boldsymbol{w}_\star)) \\
&\leq \|\boldsymbol{w}_0 - \boldsymbol{w}_\star\|^2 + 4\alpha^2 L\frac{n-b}{b(n-1)}\left(P(\boldsymbol{w}_0) - P(\boldsymbol{w}_\star)\right) + 4\alpha^2 Lm\frac{n-b}{b(n-1)}\left(P(\bar{\boldsymbol{w}}_{s-1}) - P(\boldsymbol{w}_\star)\right).
\end{aligned} \tag{27}$$

Since $\boldsymbol{w}_0 = \bar{\boldsymbol{w}}_{s-1}$ and $2\alpha \geq 2\alpha\left(1 - 2\alpha L\frac{n-b}{b(n-1)}\right)$ by the assumption, we obtain that

$$\begin{aligned}
&\mathbb{E}\|\boldsymbol{w}_m - \boldsymbol{w}_\star\|^2 + 2\alpha\left(1 - 2\alpha L\frac{n-b}{b(n-1)}\right)\sum_{t=1}^{m}(\mathbb{E}P(\boldsymbol{w}_t) - P(\boldsymbol{w}_\star)) \\
&\leq \|\bar{\boldsymbol{w}}_{s-1} - \boldsymbol{w}_\star\|^2 + 4\alpha^2 L(m+1)\frac{n-b}{b(n-1)}\left(P(\bar{\boldsymbol{w}}_{s-1}) - P(\boldsymbol{w}_\star)\right).
\end{aligned} \tag{28}$$

By the μ_h-strong convexity of P and the definition of $\bar{\boldsymbol{w}}_s$, we obtain that

$$2\alpha m\left(1-2\alpha L\frac{n-b}{b(n-1)}\right)\left(\mathbb{E}P(\bar{\boldsymbol{w}}_s)-P(\boldsymbol{w}_\star)\right)$$
$$\leq\left(\frac{2}{\mu_h}+4\alpha^2L(m+1)\frac{n-b}{b(n-1)}\right)\left(P(\bar{\boldsymbol{w}}_{s-1})-P(\boldsymbol{w}_\star)\right).$$

Finally, by applying the above inequality recursively, we get the desired result. □

Corollary 3 *Let $\boldsymbol{w}_\star=\arg\min_{\boldsymbol{w}}P(\boldsymbol{w})$. With the same conditions as Lemma 9, setting the step size $\alpha=\frac{b(n-1)}{5L(n-b)}$ and the loop length $m=\frac{100L(n-b)}{\mu_h b(n-1)}+8$, we obtain an output $\boldsymbol{w}_s$ of Algorithm 3 such that $\mathbb{E}P(\boldsymbol{w}_s)-P(\boldsymbol{w}_\star)\leq\epsilon$ in a total sample complexity of $O\left(\left(n+\frac{(n-b)L}{(n-1)\mu}\right)\log\frac{1}{\epsilon}\right)$.*

Proof: Through simple calculations, we can obtain that $\rho=\frac{3}{4}$. Thus Algorithm 3 has geometric convergence. We need $s\geq\log\frac{4}{3}\log\frac{P(\boldsymbol{w}_0)-P(\boldsymbol{w}_\star)}{\epsilon}$ to obtain an ϵ-optimal solution in expectation. The total sample complexity is $s(n+bm)=O\left(\left(n+\frac{(n-b)L}{(n-1)\mu}\right)\log\frac{1}{\epsilon}\right)$. □

Now we are ready to prove the main result for SOREL with mini-batching based on Theorem 1.

Proof of Corollary 2

Proof: By Corollary 3, we get a δ_k approximate solution of the k-th outer loop of SOREL with mini-batching with the sample complexity of $O\left(\left(n+\frac{(n-b)L}{(k+1)(n-1)\mu}\right)\log\delta_k^{-1}\right)$. Similar to the proof of Corollary 1, through simple calculations we obtain the total sample complexity of $O\left(\frac{n^{\frac{3}{2}}G}{\mu\sqrt{\epsilon}}\log\frac{\sqrt{n}}{G\sqrt{\epsilon}}+\frac{L(n-b)}{\mu(n-1)}\log\frac{\sqrt{n}}{G\sqrt{\epsilon}}\log\frac{\sqrt{n}G}{\mu\sqrt{\epsilon}}\right)$.

□

E Experiments with Error Bars

In this section, we present the error bars of experiments in Section 5.

E.1 Linear Regression

Figure 7 presents the results of using different algorithms with minibatching, as described in Section 5.1. For each algorithm, we set five random seeds, as detailed in Appendix B, and report the mean training curves with standard deviations. Since our plots are in log scale, we only keep the upper error bar to make the plots easier to read. The performances of each algorithm in Figure 7 are consistent with those in Figure 5. SOREL achieves the best or competitive results in terms of sample complexity, runtime, or both.

E.2 Fair Machine Learning

Table 4 presents the mean fairness metrics (and standard deviations in parentheses) in Section 5.2. Note that the standard deviations are large in the `acs` dataset. We attribute this to the use of 5-fold cross-validation. The standard deviations primarily arise from differences in data across folds, instead of the randomness of stochastic algorithms. Indeed, the fairness metrics obtained using ERM (solved by L-BFGS) as the loss function also exhibit large standard deviations. We follow the 5-fold cross-validation approach as recommended by Williamson & Menon (2019).

Tables 5, 6, and 7 respectively show the fairness metrics (and standard deviations in parentheses) on each fold of the data when CVaR, ESRM, and Extremile are used as the loss functions. We observe that the standard deviations of each algorithm (except SGD) are small on each fold. Thus the standard deviations of the fairness metrics in Table 4 arise from differences in the data across folds. In the `acs` dataset, SOREL still achieves the lowest fairness metrics on each fold of the data in most cases.

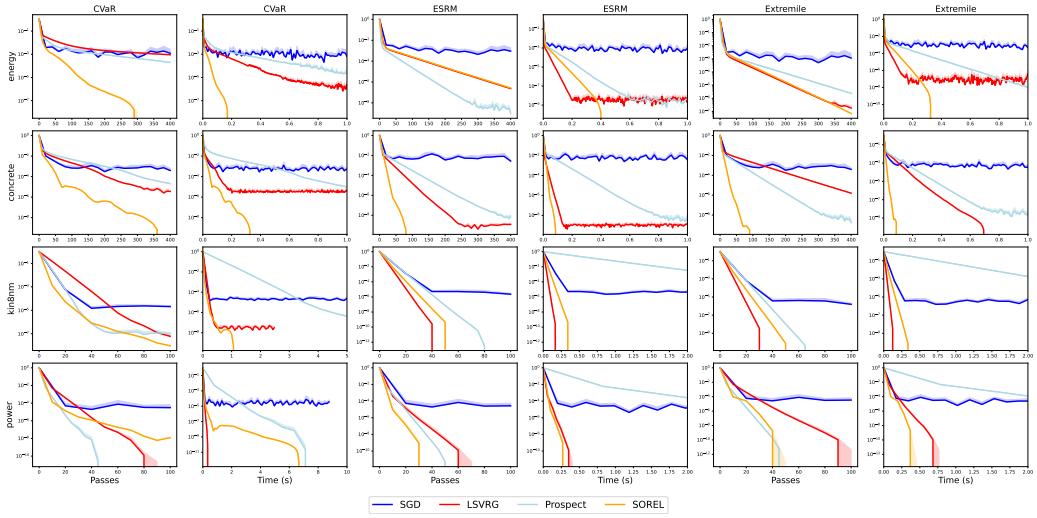

Figure 7: Suboptimality of spectral risks for different algorithms **with mini-batching**.

In the `acs` dataset, the fairness metrics for the folds 4 and 5 are negative. This is because we use the same spectral risk measure for all folds. Given the differences between folds, the spectral risk we used may be too aggressive for the folds 4 and 5. In practice, the parameters of the spectral risk could be tuned for each fold. However, as our primary focus is on the performances of each optimizer during training, this approach is beyond the scope of this paper.

Tables 8, 9 and 10 respectively show the mean suboptimality of algorithms on each fold of the data when CVaR, ESRM, and Extremile are used as the loss functions. SOREL achieves the lowest suboptimality in all settings, except for the Extremile setting in fold 2, `acs` dataset.

Table 4: The mean fairnes metrics of different algorithms on `acs` and `law` (and standard deviations in parentheses). Values closer to 0 indicate better fairness.

Datasets	`acs`				`law`			
ERM	0.02092 (0.04767)				0.05188 (0.00200)			
	SGD	LSVRG	Prospect	SOREL	SGD	LSVRG	Prospect	SOREL
CVaR	0.00645 (0.03539)	0.00816 (0.03464)	0.00634 (0.03494)	0.00551 (0.03555)	0.04019 (0.00177)	0.03896 (0.00163)	0.03893 (0.00172)	0.03890 (0.00176)
ESRM	0.02469 (0.04823)	0.01842 (0.04479)	0.01840 (0.04629)	0.01770 (0.04557)	0.04184 (0.00180)	0.04122 (0.00172)	0.04123 (0.00173)	0.04123 (0.00173)
Extremile	0.00424 (0.02970)	0.00377 (0.02899)	0.00237 (0.02888)	0.00130 (0.03006)	0.04416 (0.00171)	0.04377 (0.00166)	0.04380 (0.00167)	0.04381 (0.00167)

Table 5: The mean fairnes metrics of different algorithms on `acs` and `law` (and standard deviations in parentheses) for CVaR.

Datasets	`acs`				`law`			
	SGD	LSVRG	Prospect	SOREL	SGD	LSVRG	Prospect	SOREL
Fold 1	0.04199 (0.00095)	0.04144 (0.00095)	0.04144 (0.00095)	0.04089 (0.00000)	0.03724 (0.00105)	0.03643 (0.00004)	0.03614 (0.00016)	0.03597 (0.00000)
Fold 2	0.04071 (0.00210)	0.04132 (0.00000)	0.03950 (0.00000)	0.03950 (0.00000)	0.04221 (0.00038)	0.04119 (0.00000)	0.04131 (0.00016)	0.04116 (0.00000)
Fold 3	0.00324 (0.00184)	0.00875 (0.00000)	0.00324 (0.00000)	0.00324 (0.00000)	0.04008 (0.00032)	0.03834 (0.00003)	0.03867 (0.00005)	0.03855 (0.00000)
Fold 4	-0.00262 (0.00105)	-0.00141 (0.00000)	-0.00141 (0.00000)	-0.00323 (0.00000)	0.04053 (0.00023)	0.03981 (0.00003)	0.03928 (0.00013)	0.03925 (0.00000)
Fold 5	-0.05108 (0.00000)	-0.04928 (0.00000)	-0.05108 (0.00000)	-0.05287 (0.00000)	0.04088 (0.00053)	0.03904 (0.00008)	0.03926 (0.00011)	0.03957 (0.00000)

Table 6: The mean fairnes metrics of different algorithms on `acs` and `law` (and standard deviations in parentheses) for ESRM.

Datasets	acs				law			
	SGD	LSVRG	Prospect	SOREL	SGD	LSVRG	Prospect	SOREL
Fold 1	0.05054 (0.02043)	0.02530 ((0.00000)	0.02695 (0.00000)	0.02530 (0.00000)	0.03899 (0.00065)	0.03830 (0.00000)	0.03830 (0.00000)	0.03830 (0.00000)
Fold 2	0.09799 (0.00105)	0.09496 (0.00000)	0.09678 (0.00000)	0.09496 (0.00000)	0.04377 (0.00020)	0.04335 (0.00000)	0.04338 (0.00000)	0.04338 (0.00000)
Fold 3	0.02040 (0.00106)	0.01794 (0.00000)	0.01794 (0.00000)	0.01794 (0.00000)	0.04170 (0.00095)	0.04107 (0.00000)	0.04109 (0.00000)	0.04109 (0.00000)
Fold 4	-0.02011 (0.00000)	-0.02011 (0.00000)	-0.02011 (0.00000)	-0.02011 (0.00000)	0.04209 (0.00104)	0.04135 (0.00000)	0.04136 (0.00000)	0.04136 (0.00000)
Fold 5	-0.02539 (0.00103)	-0.02599 (0.00000)	-0.02957 (0.00000)	-0.02957 (0.00000)	0.04266 (0.00112)	0.04203 (0.00000)	0.04204 (0.00000)	0.04204 (0.00000)

Table 7: The mean fairnes metrics of different algorithms on `acs` and `law` (and standard deviations in parentheses) for Extremile.

Datasets	acs				law			
	SGD	LSVRG	Prospect	SOREL	SGD	LSVRG	Prospect	SOREL
Fold 1	0.01100 (0.00095)	0.01045 (0.00000)	0.00715 (0.00000)	0.00715 (0.00000)	0.04137 (0.00024)	0.04103 (0.00000)	0.04105 (0.00000)	0.04105 (0.00000)
Fold 2	0.04739 (0.00105)	0.04496 (0.00000)	0.04496 (0.00000)	0.04314 (0.00000)	0.04626 (0.00015)	0.04592 (0.00000)	0.04598 (0.00000)	0.04598 (0.00000)
Fold 3	0.00753 (0.00106)	0.00875 (0.00000)	0.00692 (0.00000)	0.00692 (0.00000)	0.04400 (0.00053)	0.04357 (0.00000)	0.04364 (0.00000)	0.04365 (0.00000)
Fold 4	-0.00262, (0.00105)	-0.00323 (0.00000)	-0.00504 (0.00000)	-0.00323 (0.00000)	0.04417 (0.00056)	0.04370 (0.00000)	0.04370 (0.00000)	0.04370 (0.00000)
Fold 5	-0.04211 (0.00000)	-0.04211 (0.00000)	-0.04211 (0.00000)	-0.04749 (0.00000)	0.04499 (0.00053)	0.04465 (0.00000)	0.04465 (0.00000)	0.04465 (0.00000)

E.3 Out-of-Distribution Generalization

Figure 8 shows the training curves and worst group classification errors with standard deviations for the experiments in Section 5.3. Table 11 shows the worst group classification errors (and standard deviations in parentheses) in Figure 8. SOREL achieves the lowest worst group classification error in both settings. SOREL is also the only algorithm that can converge to the true optimal solution under the CVaR setting, as stated in Section 5.3.

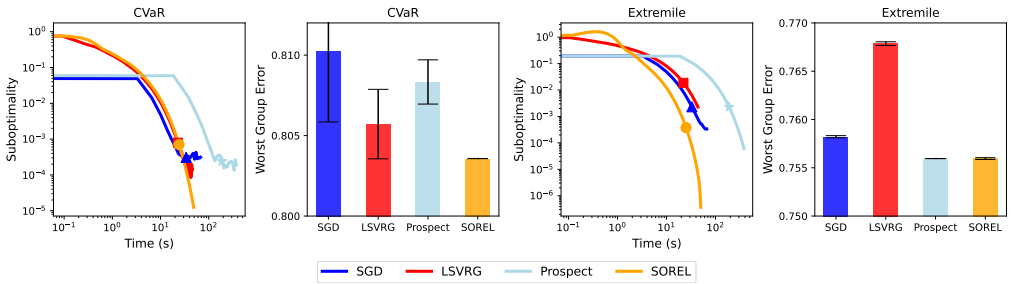

Figure 8: Training curves and worst group classification errors of different algorithms on the `amazon` dataset.

F Additional Experiments on Nonconvex Objectives

In this section, we empirically explore the performance of SOREL in optimizing nonlinear models. We train a two-layer neural network with ReLU activation function and set the hidden layer's dimension equal to the feature dimension of the input data. We use the regression task from Section 5.1.

Table 8: The mean Suboptimality of different algorithms on `acs` and `law` (and standard deviations in parentheses) for CVaR.

Datasets	acs				law			
	SGD	LSVRG	Prospect	SOREL	SGD	LSVRG	Prospect	SOREL
Fold 1	3.82e-04 (7.42e-05)	5.00e-04 (1.27e-06)	1.48e-04 (2.83e-05)	9.90e-07 (1.86e-08)	5.28e-03 (1.02e-03)	1.02e-02 (1.89e-06)	1.72e-03 (1.04e-05)	9.75e-05 (2.00e-07)
Fold 2	3.33e-04 (8.88e-05)	2.91e-04 (4.70e-06)	6.00e-05 (6.14e-06)	7.84e-07 (7.25e-08)	3.93e-03 (5.75e-04)	6.04e-03 (1.65e-05)	5.37e-04 (1.25e-05)	2.27e-05 (6.49e-08)
Fold 3	3.58e-04 (1.13e-04)	6.81e-04 (1.32e-06)	1.95e-04 (1.30e-05)	4.90e-06 (1.89e-08)	2.14e-03 (6.97e-04)	3.44e-03 (9.74e-06)	2.79e-04 (9.04e-06)	3.06e-07 (3.85e-08)
Fold 4	4.62e-04 (1.02e-05)	7.84e-04 (1.86e-06)	2.45e-04 (2.39e-06)	1.72e-06 (1.32e-08)	3.44e-03 (1.31e-04)	5.02e-03 (1.57e-05)	2.97e-04 (2.44e-05)	4.12e-06 (1.10e-08)
Fold 5	6.10e-04 (1.48e-05)	1.36e-03 (6.01e-07)	4.14e-04 (2.92e-06)	3.14e-06 (4.20e-08)	4.22e-03 (9.23e-05)	4.66e-03 (1.15e-05)	5.25e-04 (1.82e-05)	2.30e-05 (3.91e-08)

Table 9: The mean Suboptimality of different algorithms on `acs` and `law` (and standard deviations in parentheses) for ESRM.

Datasets	acs				law			
	SGD	LSVRG	Prospect	SOREL	SGD	LSVRG	Prospect	SOREL
Fold 1	1.02e-03 (3.63e-06)	1.15e-04 (2.75e-07)	2.07e-06 (1.04e-06)	-5.14e-08 (9.61e-12)	6.16e-04 (2.26e-05)	2.46e-04 (9.54e-08)	7.83e-07 (1.77e-09)	8.47e-08 (2.52e-10)
Fold 2	9.66e-04 (9.40e-05)	5.79e-05 (3.55e-07)	1.95e-05 (8.86e-07)	-1.23e-08 (1.04e-11)	5.53e-04 (1.19e-04)	9.69e-05 (8.61e-08)	-3.89e-08 (1.75e-09)	-2.73e-07 (2.21e-10)
Fold 3	1.25e-03 (5.74e-05)	2.62e-04 (8.65e-07)	1.68e-07 (1.25e-07)	-3.46e-08 (4.11e-12)	9.67e-04 (5.48e-04)	6.24e-05 (2.94e-08)	-3.25e-07 (5.19e-10)	-4.25e-07 (2.26e-10)
Fold 4	1.57e-03 (6.84e-05)	4.31e-04 (1.05e-06)	4.72e-08 (9.27e-09)	-8.08e-09 (6.04e-12)	8.70e-04 (4.06e-04)	8.52e-05 (3.28e-08)	1.87e-07 (1.14e-09)	3.83e-08 (5.34e-11)
Fold 5	2.54e-03 (8.78e-05)	7.98e-04 (1.44e-06)	1.43e-07 (2.76e-08)	-1.16e-08 (9.90e-12)	7.92e-04 (5.58e-04)	7.53e-05 (1.07e-07)	1.53e-07 (5.01e-10)	8.38e-09 (4.75e-10)

Thus, the loss function and the model can be written as

$$\ell(z) = \frac{1}{2}(z-y)^2$$

and

$$z = W_2\left(\text{ReLU}(W_1 x + b_1)\right) + b_2,$$

where $x \in \mathbb{R}^d, y \in \mathbb{R}$ are the feature and label, $W_1 \in \mathbb{R}^{d\times d}$, $b_1 \in \mathbb{R}^d$, $W_2 \in \mathbb{R}^{1\times d}$ and $b_2 \in \mathbb{R}$ are trainable parameters. The experimental setup is identical to that in Section 5.1.

Figure 9 shows the training curves using three spectral risk measures on the `energy` and `concrete` datasets. Note that none of the four algorithms have theoretical guarantees in the non-convex setting. However, SOREL achieves the optimal or near-optimal results across various settings. On the `energy` dataset, SOREL achieves the lowest losses, significantly outperforming LSVRG and Prospect. On the `concrete` dataset, SOREL also achieves slightly lower loss values compared to LSVRG and Prospect. Table 12 reports the mean losses and standard deviations over the last ten passes. We observe that SOREL achieves the lowest mean losses, demonstrating the effectiveness in optimizing non-convex functions, even though theoretical guarantees are not available.

G Further Discussion on the Complexity in Corollary 1

In Section 4, we discussed the optimality of the complexity in Corollary 1 with respect to ϵ. In this section, we further discuss the dependence of the complexity in the Corollary 1 on n. The sample complexity with respect to n and ϵ in Corollary 1 is $\widetilde{O}(n^{3/2}/\sqrt{\epsilon})$. Since SOREL requires computing the projection onto the permutahedron $\Pi_{\boldsymbol{\sigma}}$ in each outer iteration, which takes $O(n \log n)$ time, the total time complexity of SOREL includes an additional $O(Kn \log n)$ term, where $K = O\left(\frac{\sqrt{n}G}{\mu\sqrt{\epsilon}}\right)$

Table 10: The mean Suboptimality of different algorithms on `acs` and `law` (and standard deviations in parentheses) for Extremile.

Datasets	acs				law			
	SGD	LSVRG	Prospect	SOREL	SGD	LSVRG	Prospect	SOREL
Fold 1	5.34e-03 (1.13e-04)	6.04e-03 (4.08e-06)	2.49e-04 (4.61e-06)	1.03e-04 (1.84e-07)	1.38e-04 (7.31e-05)	9.60e-04 (1.73e-07)	1.46e-05 (9.97e-09)	-1.96e-07 (3.29e-12)
Fold 2	6.09e-03 (5.68e-04)	5.23e-03 (8.57e-07)	9.23e-05 (9.98e-07)	9.36e-05 (2.51e-07)	1.41e-04 (8.16e-05)	6.08e-04 (1.81e-07)	7.32e-06 (1.07e-08)	-4.77e-07 (5.09e-12)
Fold 3	4.36e-03 (1.74e-04)	7.66e-03 (2.76e-06)	5.18e-04 (2.92e-06)	7.98e-05 (3.52e-07)	1.95e-04 (1.22e-04)	4.30e-04 (1.58e-07)	5.47e-06 (7.48e-09)	-3.42e-07 (1.50e-12)
Fold 4	4.31e-03 (3.03e-04)	9.84e-03 (2.84e-06)	9.22e-04 (3.13e-06)	1.49e-04 (6.40e-07)	1.99e-04 (1.29e-04)	6.01e-04 (1.34e-07)	7.97e-06 (1.68e-08)	-9.44e-10 (2.04e-13)
Fold 5	5.46e-03 (4.21e-04)	1.07e-02 (5.52e-06)	1.56e-03 (7.60e-06)	1.43e-04 (7.27e-07)	1.43e-04 (1.50e-04)	4.71e-04 (3.69e-07)	5.40e-06 (7.82e-09)	-4.31e-08 (3.60e-13)

Table 11: Worst group classification errors of different algorithms (and standard deviations in parentheses) on the `amazon` dataset.

Spectral Risks	SGD	LSVRG	Prospect	SOREL
CVaR	0.8102 (0.0044)	0.8057 (0.0022)	0.8083 (0.0014)	0.8036 (0.0000)
Extremile	0.7582 (0.0017)	0.7679 (0.0000)	0.7560 (0.0000)	0.7560 (0.0000)

is given in the proof of Corollary 1. Therefore, the total complexity of SOREL with respect to n and ϵ is $\widetilde{O}(n^{3/2}/\sqrt{\epsilon})$.

Here, we discuss the complexity of baselines in Section 5 with respect to n and ϵ. SGD has been shown to fail to converge to the optimal solution of the spectral risk minimization problem, while LSVRG only guarantees convergence for $\nu \geq \Omega\left(nG^2/\mu\right)$ (Mehta et al., 2022). For Prospect, by setting $\nu = O(\epsilon)$, we obtain its sample complexity with respect to n and ϵ as $\widetilde{O}(n^2/\epsilon^2)$ or $\widetilde{O}(n/\epsilon^3)$ (depending on the size of ϵ). Moreover, Prospect requires computing the projection onto the permutahedron $\Pi_{\boldsymbol{\sigma}}$ at each step with a cost of $O(n \log n)$ time, which results in its total time complexity $\widetilde{O}(n^2/\epsilon^3)$ or $\widetilde{O}(n^3/\epsilon^2)$. Therefore, SOREL also has an advantage in terms of the total time complexity with respect to n. This is consistent with the experimental results in Section 5, where SOREL significantly outperforms Prospect in terms of runtime.

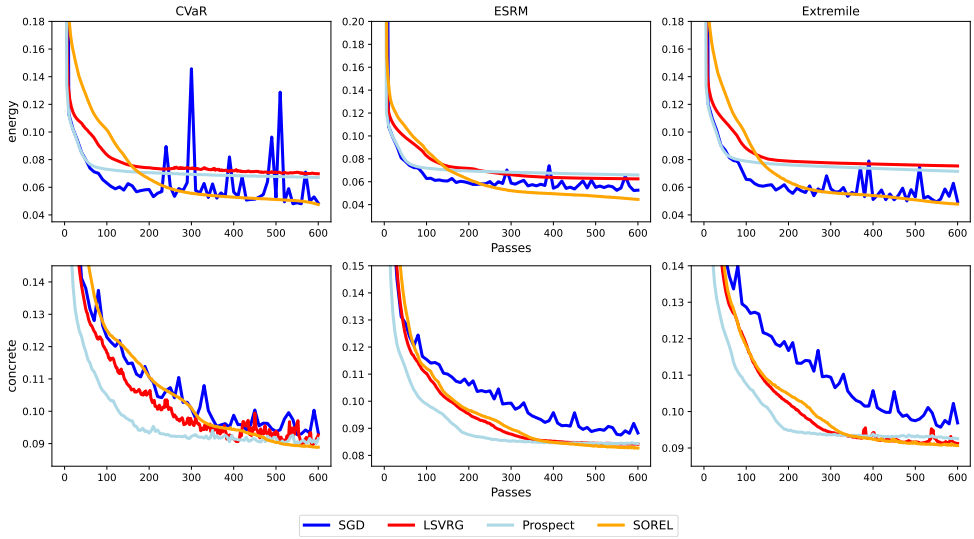

Figure 9: Results of training two-layer neural networks using different algorithms.

Table 12: The mean function values over the last ten passes (and standard deviations in parentheses).

Datasets	energy				concrete			
	SGD	LSVRG	Prospect	SOREL	SGD	LSVRG	Prospect	SOREL
CVaR	0.05533 (0.02004)	0.06986 (0.00027)	0.06715 (0.00024)	0.04787 (0.00045)	0.09573 (0.00299)	0.09149 (0.00276)	0.09099 (0.00188)	0.08888 (0.00104)
ESRM	0.05463 (0.01268)	0.06253 (0.00882)	0.06594 (0.00006)	0.04464 (0.00020)	0.08942 (0.00253)	0.08417 (0.00006)	0.08443 (0.00039)	0.08275 (0.00013)
Extremile	0.05153 (0.00329)	0.07544 (0.00005)	0.07155 (0.00005)	0.04792 (0.00005)	0.09832 (0.00325)	0.09132 (0.00012)	0.09261 (0.00097)	0.09073 (0.00077)