Provable Constrained Stochastic Convex Optimization with XOR-Projected Gradient Descent

Fan Ding
Yijie Wang
Jianzhu Ma
Yexiang Xue

1Computer Science Dept., Purdue University, West Lafayette, Indiana, USA
2Computer Science Dept., Indiana University, Bloomington, Indiana, USA
3Institute for Artificial Intelligence, Peking University, Beijing, China

Abstract

Provably solving stochastic convex optimization problems with constraints is essential for various problems in science, business, and statistics. Recently proposed XOR-Stochastic Gradient Descent (XOR-SGD) provides a convergence rate guarantee solving the constraints-free version of the problem by leveraging XOR-Sampling. However, the task becomes more difficult when additional equality and inequality constraints are needed to be satisfied. Here we propose XOR-PGD, a novel algorithm based on Projected Gradient Descent (PGD) coupled with the XOR sampler, which is guaranteed to solve the constrained stochastic convex optimization problem still in linear convergence rate by choosing proper step size. We show on both synthetic stochastic inventory management and real-world road network design problems that the rate of constraints satisfaction of the solutions optimized by XOR-PGD is 10% more than the competing approaches in very large searching space. The improved XOR-PGD algorithm is demonstrated to be more accurate and efficient than both XOR-SGD and SGD coupled with MCMC based samplers. It is also shown to be more scalable with respect to the number of samples and processor cores via experiments with large dimensions.

1 INTRODUCTION

Stochastic Convex Optimization problem is of great importance given its wide applicability in finance, control, robotics, management science, operational research, ecology, and conservation [Sodomka et al., 2007; Ziukov, 2016; Gomes et al., 2019]. Advancements made to address this problem have ramifications in many domains. Mathematically, it minimizes an objective in expectation across multiple probabilistic scenarios under uncertainty.

\[
\min_x \mathbb{E}_{\theta \sim P_{\theta}} f(x, \theta), \\
s.t. \ x \in C := \{\forall i, h_i(x) = 0; \forall j, g_j(x) \leq 0\}.
\]

where each \( f(x, \theta) \) is a convex function with respect to \( x \), \( C \) is the set of constraints, \( g_j(x) \) are convex functions and \( h_i(x) \) are linear. Variable \( \theta \) is sampled from distribution \( P_{\theta} \), which is represented as a Markov random field (MRF) in this paper. We can see that this problem is highly intractable (\#P-hard) [Ding and Xue, 2021]. Despite the intractability of computing the expectation over a general probability distribution, a common operator in probabilistic inference, the problem is harder to solve because of the existence of the constraints set \( C \). A recently proposed method, XOR-SGD [Ding and Xue, 2021] first harness XOR-Sampler [Ermon et al., 2013a] to solve the constraints-free version of this task with guaranteed convergence rate using Stochastic Gradient Descent (SGD). However, they cannot provide such a bound once the solutions are in a constrained space which is more common in the real world. In this paper we consider the setting of the constrained stochastic convex optimization and first provide an algorithm, XOR-Projected Gradient Descent (XOR-PGD), that has a provably convergence rate towards the optimal value. The key of XOR-PGD is to first draw a representative set of samples from \( P_{\theta} \) which yield an accurate estimation of the gradient direction, and then adjust the estimation by projecting the variable \( x \) to the constraints space \( C \). Notice that even if PGD is a common method in traditional constrained optimization, it is not trivial to extend it to a stochastic setting where unbiased estimation is unavailable.

Like XOR-SGD, our XOR-PGD also leverages XOR-Sampling with a constant approximation guarantee, which reduces the sampling problem into queries of NP oracles via hashing and projection. Our key contribution is the extension of classical convergence analysis of XOR-SGD on constrained stochastic convex optimization problems, where we show that a constant multiplicative bound on the expectation of the gradient direction is sufficient to bound the final
result against the true optimum. Our theoretic contribution also does not depend on the unbiasedness of the gradients, which was a necessary condition in previous analysis.

On learning probabilistic graphical models, stochastic optimization is related to the Marginal Maximum-a-posterior (MMAP) problem [Xue et al., 2016, Liu and Ihler, 2013, Marinescu et al., 2014, Maua and de Campos, 2012, Marinescu et al., 2015, Domke, 2013]. These problems can be formulated as (albeit non-convex) stochastic optimization problems. Convergence analysis of gradient descent has been studied for objectives with or without constraints [Wang et al., 2013, Dubey et al., 2016, Agarwal et al., 2017, Lee et al., 2015, Ruder, 2016, Jin et al., 2017, Ge et al., 2015, Duchi et al., 2011, Hinton et al., 2012, Kingma and Ba, 2014, Duchi et al., 2018, Allen-Zhu, 2017, 2018]. The constrained setting is related to a convex-concave saddle point minimax optimization problem [Mokhtari et al., 2020, Xie et al., 2020, Wang and Li, 2020] where primal-dual methods are often used [Du and Hu, 2019, Hamedani and Aybat, 2018]. A fruitful line of Stochastic alternating direction method of multipliers (ADMM) [Ouyang et al., 2013, Zheng and Kwok, 2016, Liu et al., 2017] has been proposed to deal with constrained optimization under uncertainty, yet this uncertainty is represented by a uniform distribution where the unbiased estimation is available. Similar idea of stochastic convex optimization is also proposed by Ding et al. [2021] for machine learning, where they leverage a XOR sampler to estimate the partition function in the learning process of an energy-based model. However, none of them offer a theoretical guarantee of convergence rate in the setting of constrained stochastic convex optimization.

Experimental results reveal that XOR-PGD is effective in optimizing constrained convex stochastic functions. XOR-PGD outperforms competing solvers XOR-SGD and those running SGD with either MCMC, BP or BPChain samplers on both the constrained stochastic inventory management problem and the constrained stochastic network design problems on real-world data under various conditions in not only accuracy and speed, but also the rate of constraints satisfaction. In particular, 90% solutions obtained by XOR-PGD satisfy the constraints set C even when the searching space is very large in the stochastic inventory management problem, approximately 10% more that of competing methods. Besides, the improved XOR-PGD algorithm converges faster than XOR-SGD by accessing 20 less XOR samples in each iteration and is able to find better solutions in the stochastic network design problem. See the experiments section for more details.

Notations For function $f : \mathbb{R}^d \rightarrow \mathbb{R}$, we call it $L$-smooth if for all $x, y$ in the convex domain $dom f$, $f(y) \leq f(x) + \nabla f(x)^T(y - x) + \frac{L}{2}||y - x||^2$. Denote $f^+(x)$ as the positive part of function $f(x)$. In other words, $f^+(x) = \max\{f(x), 0\}$. $f^-(x)$ is defined similarly. For a random vector $x$, we define $\mathbb{E}[|x|]$ as the element-wise expectation and the total variation $\text{Var}(x) = \mathbb{E}[||x||^2] - ||\mathbb{E}[x]||^2$ where $|| \cdot ||^2$ is the square of $l_2$ norm.

## 2 PRELIMINARIES

### 2.1 PROBABILISTIC DISTRIBUTION

The probability distribution $Pr(\theta)$ can be defined in various different forms. We consider $Pr(\theta)$ as a graphical model specified as a factor graph with $N = |V|$ discrete random variables $\theta_i \in \Theta_i$, $i \in V$ where $\Theta_i = \{0, 1\}$. The global random vector $\theta = (\theta_1, \theta_2, \ldots, \theta_N)$ takes value in the cartes-

### 2.2 XOR-SGD

XOR-SGD outperforms SGD with MCMC based samplers in that it has a constant bound on the probability of drawn samples and further guarantee a constant bound between the expectation of the distribution formed by samples and the expectation of true distribution, without requiring exponentially large number of samples. These samples are drawn by XOR-Sampling [Ermon et al., 2013b], a recently proposed sampling scheme with a constant approximation guarantee, which reduces the sampling problem into queries of NP oracles. XOR-SGD requires accessing NP-oracle queries at each iteration. Specifically, XOR-SGD is guaranteed to converge to a solution that is within a vanishing constant away from the true optimum in linear number of iterations, which is shown as Theorem 1

**Theorem 1.** [Ding and Xue, 2021] Let $\rho, \kappa$ be the constant approximation factor as $\rho \leq \kappa$, function $f(x, \theta) : \mathbb{R}^d \times \{0, 1\}^n \rightarrow \mathbb{R}$ be a $L$-smooth convex function w.r.t. $x$. Denote $\text{OPT} = \min_{x, \theta} \mathbb{E}_{\theta \sim \text{Pr}(\theta)} f(x, \theta)$ as the global optimum. Let $\sigma^2 = \max_{x, \theta} \{\text{Var}(\nabla_x f(x, \theta))\}$ and $
abla x^2 = \max_{x, \theta} \{||\nabla_x f(x, \theta)||^2\}$. For any $1 \leq \rho \leq \sqrt{2}$, step size $t = \frac{2\rho^n}{L \rho^2}$ and sample size $N \geq 1, \frac{\rho}{\kappa}$ is the output of XOR-SGD and $\text{obj} = \mathbb{E}_q[f(\pi_K, \theta)]$ is the objective
**Theorem 2.** Let \( f : \mathbb{R}^d \to \mathbb{R} \) be a \( \lambda \)-smooth and \( \mu \)-strongly convex function and \( x^* = \arg \min_{x \in C} f(x) \), where \( C \) is a convex constraint set of \( x \). In iteration \( k \), \( g_k \) is the estimated gradient, i.e., \( x_{k+1} = P_C(x_k - t_k g_k) \) where \( \text{Var}(g_k) \leq \sigma^2 \) and \( P_C(x) := \arg \min_{y \in C} \frac{1}{2} \|x - y\|^2 \) is the projection of \( x \) onto \( C \). If \( t_k = \frac{\mu}{c^2 \lambda} \) and there exists \( 1 \leq c \leq \sqrt{2} \) s.t. \( \frac{1}{c} |\nabla f(x_k)|^2 \leq \mathbb{E}[g_k^2] \leq c |\nabla f(x_k)|^2 \) and \( c |\nabla f(x_k)|^2 \leq \mathbb{E}[g_k^2] \leq \frac{1}{c} |\nabla f(x_k)|^2 \), then the convergence rate of the function value at \( x_k \). We have:

\[
E_{x_k}[f(x_k)] - f(x^*) \leq \frac{p}{2tK} \|x_0 - x^*\|^2 + \frac{t(\sigma^2 + c^2)}{N}.
\] (3)

Theorem 1 states that in expectation, the difference in terms of the objective function values between the output of XOR-SGD algorithm \( x_k \) and the true optimum \( f(x^*) \) is bounded by a term that scales inversely proportional to the number of SGD iterations \( K \) and a tail term \( O(\sigma^2 + c^2) \). Although hard to compute, both \( \sigma^2 \) and \( c^2 \) are from the input and do not depend on the algorithm. However, although the authors provide a constrained version of XOR-SGD in their paper, they are not able to give a theoretical guarantee of convergence rate of this algorithm under the constrained setting, which is known to be more intractable due to the extra constraints on the solution space.

### 3 XOR-PROJECTED GRADIENT DESCENT

In this section we propose XOR-PGD, a new method to solve the stochastic convex optimization problems with extra constraints. XOR-PGD converges to solutions that are at most a constant away from the true optimum in linear number of iterations. The detailed procedure of XOR-PGD is shown in Algorithm [8]. To approximate the gradient \( \nabla_x f(x_k, \theta) \) at step \( k \), XOR-PGD draws \( N \) samples \( \theta_1, \ldots, \theta_N \) from \( P_{\mathcal{R}}(\theta) \) using XOR-Sampling. Because XOR-Sampling has a failure rate, XOR-PGD repeatedly call XOR-Sampling until all \( N \) samples are obtained successfully. XOR-PGD uses the empirical mean \( f_k = \frac{1}{N} \sum_{i=1}^{N} \nabla_x f(x_k, \theta_i) \) as an approximation for \( \nabla_x f(x_k, \theta) \).

Then, by setting the step size \( t_k \) as \( \frac{\sigma^2 + c^2}{\mu^2} \), we update \( x_k \) by first a minus of \( t_k f_k \) and then a projection \( P_C(x) \) to project it to to constraints space \( C \). Here we define \( P_C(x) \) in Equation [4]

\[
P_C(x) := \arg \min_{y \in C} \frac{1}{2} \|x - y\|^2
\] (4)

Using this projection, we can prove that the output of XOR-PGD in expectation converges to the true optimum within a small constant distance at a linear speed w.r.t. the number of iterations \( K \).

**Algorithm 1:** XOR-PGD

**Input:** \( f(x, \theta), \mu, w(\theta), K, N, \beta, \delta, P, \alpha, C \)

1. Initialize \( x_1 \) for function \( f(x, \theta) \)
2. for \( k = 1 \) to \( K - 1 \) do
3. \( i \leftarrow 1 \)
4. while \( i \leq N \) do
5. \( s \leftarrow \text{XOR-Sampling}(w(\theta), \beta, \delta, P, \alpha) \)
6. if \( s \neq \text{Failure} \) then
7. \( \theta_i \leftarrow s \)
8. \( i \leftarrow i + 1 \)
9. end
10. end
11. Compute \( f_k = \frac{1}{N} \sum_{i=1}^{N} \nabla_x f(x_k, \theta_i) \)
12. Compute \( t_k = \frac{\mu^2}{\sigma^2 + c^2} \)
13. Update \( x_{k+1} \leftarrow \arg \min_{y \in C} \frac{1}{2} \|x_k - t_k f_k - y\|^2 \)
14. end
15. \( x_K \leftarrow x_K = \frac{1}{K} \sum_{k=1}^{K} x_k \)
16. return \( x_K \)

**Algorithm 2:** Improved XOR-PGD

1. ... as in Algorithm 1, except
2. replacing line 12 by: \( t_k = \frac{\mu}{\mu + (k+1)} \)
3. replacing line 15 by: \( x_K \leftarrow x_K = \frac{2 \sum_{k=1}^{K} x_k}{K+1} \)

**XOR-PGD** is \( O\left(\frac{\log K}{K}\right)\):

\[
E[f(x_K)] - f(x^*) \leq \frac{B}{2\mu K}(1 + \log(K)),
\] (5)

where \( x_K = \frac{1}{K} \sum_{k=1}^{K} x_k \).

**Proof.** (Theorem 2) Because orthogonal projections contract distances, we have

\[
\|x_{k+1} - x^*\|^2 = \|x_k - t_k g_k - x^*\|^2
= \|x_k - x^*\|^2 + t_k^2 \|g_k\|^2 - 2t_k \langle g_k, x_k - x^* \rangle
\]

Taking expectation on both side, we have

\[
E[\|x_{k+1} - x^*\|^2] \leq E[\|x_k - x^*\|^2] + t_k^2 E[\|g_k\|^2]
- 2t_k \langle E[g_k], x_k - x^* \rangle
\]

From Lemma 1 in [Ding and Xue 2021], we know

\[
- \langle E[g_k], x_k - x^* \rangle \leq -\frac{1}{c} \langle \nabla f(x_k), x_k - x^* \rangle.
\]

Then we have

\[
E[\|x_{k+1} - x^*\|^2] \leq E[\|x_k - x^*\|^2] + t_k^2 E[\|g_k\|^2]
- 2t_k \frac{1}{c} \langle \nabla f(x_k), x_k - x^* \rangle
\]

\[
\leq E[\|x_k - x^*\|^2] + t_k^2 E[\|g_k\|^2]
- 2t_k \frac{1}{c} \|f(x_k) - f(x^*)\| + \frac{\mu}{2} \|x_k - x^*\|^2
\]
The last inequality is because $f$ is $\mu$-strongly convex. After some rearranging and taking expectation on both sides, we have
\[
\mathbb{E}[f(x_k)] - f(x^*) \leq \frac{ct_k}{2} \mathbb{E}[[\|g_k\|^2]] + \frac{c - t_k \mu}{2t_k} \mathbb{E}[[x_k - x^*]_2^2] - \frac{2t_k}{c} \mathbb{E}[[x_{k+1} - x^*]_2^2].
\]
We know $\mathbb{E}[\|g_k\|^2] = \mathbb{E}[\|g_k\|^2] + \text{Var}(g_k)$, and from Lemma 1 in [Ding and Xue, 2021] we have
\[
\mathbb{E}[\|g_k\|^2] \leq c(\mathbb{E}[\|\nabla f(x_k)\|]\mathbb{E}[\|g_k\|]) \leq cL\mathbb{E}[\|g_k\|]
\]
Therefore, $\mathbb{E}[\|g_k\|^2] \leq c^2L^2$ is bounded above and $\mathbb{E}[\|g_k\|^2] \leq c^2L^2 + \sigma^2$. Apply this bound, we have
\[
\mathbb{E}[f(x_k)] - f(x^*) \leq \frac{ct_k}{2} (c^2L^2 + \sigma^2) + \frac{c - t_k \mu}{2t_k} \mathbb{E}[[x_k - x^*]_2^2] - \frac{2t_k}{c} \mathbb{E}[[x_{k+1} - x^*]_2^2]
\]
With $t_k = \frac{\mu}{2t_k}$, the above inequality becomes
\[
\mathbb{E}[f(x_k)] - f(x^*) \leq \frac{c^2(\mathbb{E}[[\nabla f(x_k)\|\mathbb{E}[\|g_k\|])]}{2t_k} + \frac{\mu(k-1)}{2} \mathbb{E}[[x_k - x^*]_2^2] - \frac{t_k}{2} \mathbb{E}[[x_{k+1} - x^*]_2^2]
\]
Summing the above equations for $k = 1, \ldots, K$, we get
\[
\mathbb{E}[\frac{1}{K} \sum_{k=1}^{K} f(x_k)] - f(x^*) \leq \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}[f(x_k)] - f(x^*)
\]
\[
\leq \frac{B}{2\mu K} K \sum_{k=1}^{K} [1 + \frac{\mu}{2K} 0 - K \mathbb{E}[[x_{K+1} - x^*]_2^2]
\]
\[
\leq \frac{B}{2\mu K} (1 + \log(K))
\]
where $B = c^2(\mathbb{E}[[\nabla f(x_k)\|\mathbb{E}[\|g_k\|])]}{2t_k}$. The first inequality is obtained using the convexity of $f$; the second inequality is obtained from a telescoping sum. Let $\bar{x}_K = \frac{1}{K} \sum_{k=1}^{K} x_k$, the above inequality can be written as
\[
\mathbb{E}[f(\bar{x}_K)] - f(x^*) \leq \frac{B}{2\mu K} (1 + \log(K))
\]
Therefore, the convergence rate is $O\left(\frac{\log(K)}{K}\right)$. This completes the proof.

Theorem 3 states that by choosing the step size $t_k$ inverse proportional to the step $k$, i.e., $t_k = \frac{\mu}{2\mu K}$, our XOR-PGD can converge to the optimal value in $O\left(\frac{\log K}{K}\right)$. In practice, we can increase the sample size from 1 to $N$ to further reduce the variance. By replacing the objective $f(x)$ in Theorem 2 with $f_{\theta}(x, \theta)$ and noticing $\text{Var}(\overline{g}_k) = \text{Var}(\nabla f(x, \theta))/N$ due to the sample size $N$, we have the following Theorem 5.

**Theorem 3.** Let $\rho, \kappa$ be the constant approximation factor as in [Ding and Xue, 2021], function $f(x, \theta) : \mathbb{R}^d \times \{0, 1\}^N \rightarrow \mathbb{R}$ be a $L$-smooth and $\mu$-strongly convex function w.r.t. $x$. Denote $\text{OPT} = \min_{x \in C} \mathbb{E}_{\theta \sim \mathcal{P}(\theta)} f(x, \theta)$ as the global optimum in constraints set $C$. Let $\sigma^2 = \text{max}_x \mathbb{E} \left[ \text{Var}(\nabla f(x, \theta)) \right]$ and $\epsilon^2 = \text{max}_x \mathbb{E} \left[ \mathbb{E} \left[ \text{Var}(\nabla f(x, \theta)) \right] \right]$. For any $1 \leq \rho \kappa \leq \sqrt{2}$, step size $t \leq \frac{\mu}{\rho \kappa}$ and sample size $N \geq 1$, $\overline{x}_K = \frac{1}{K} \sum_{k=1}^{K} x_k$ is the output of XOR-PGD (Algorithm 1) and $\text{obj} = \mathbb{E}_{\theta} [f(\overline{x}_K, \theta)]$ is the objective function value at $x_K$. We have:

\[
\mathbb{E}_{\theta}[\text{obj}] - \text{OPT} \leq \left( \frac{\rho \kappa L^2 \rho \kappa}{2 \mu} + \frac{\rho \kappa \sigma^2 + \epsilon^2}{2 \mu N} \right) \left( 1 + \log(K) \right)
\]

**Proof:** (Theorem 5) Since we use $N$ samples at each iteration, we have $\overline{x}_K = \frac{1}{N} \sum_{i=1}^{N} \overline{g}_k$ and $\mathbb{E}[\overline{g}_k] = \mathbb{E}[g_k]$. In each iteration $k$ we can adjust the parameters in XOR-Sampling to make the tail $c_0\rho \kappa$ zero, then for each sample $g_k$ we can obtain from Theorem 2 that

\[
\frac{1}{\rho \kappa} \mathbb{E}_{\theta} [\nabla f(x_k, \theta)]^+ \leq \mathbb{E}[g_k^+] \leq \rho \kappa \mathbb{E}_{\theta} [\nabla f(x_k, \theta)]^+.
\]

\[
\rho \kappa \mathbb{E}_{\theta} [\nabla f(x_k, \theta)]^- \leq \mathbb{E}[g_k^-] \leq \frac{1}{\rho \kappa} \mathbb{E}_{\theta} [\nabla f(x_k, \theta)]^-.
\]

The variance of each sample $g_k$ can also be bounded by
\[
\text{Var}(g_k) = \mathbb{E}_{\theta \sim \mathcal{P}(\theta)} [\|\nabla f(x_k, \theta')\|^2] - \mathbb{E}_{\theta \sim \mathcal{P}(\theta)} [\|\nabla f(x_k, \theta')\|^2]_2
\]
\[
\leq \rho \kappa \mathbb{E}_{\theta \sim \mathcal{P}(\theta)} [\|\nabla f(x_k, \theta')\|^2]_2
\]
\[
= \rho \kappa \mathbb{E}_{\theta \sim \mathcal{P}(\theta)} [\text{Var}(\nabla f(x_k, \theta)) + \mathbb{E}[\text{Var}(\nabla f(x_k, \theta))]]_2
\]
\[
\leq \rho \kappa (\sigma^2 + \epsilon^2).
\]

Denote $\overline{g}_k^+ = \text{max} \{\overline{g}_k, 0\}$ and $\overline{g}_k^- = \text{min} \{\overline{g}_k, 0\}$. Clearly, $g_k^+ \geq 0$ and $g_k^- \leq 0$. Moreover, for a given dimension, either $g_k^+ = 0$ for that dimension or $g_k^- = 0$. Evaluating $\overline{g}_k^+$ dimension by dimension, we can see that $\overline{g}_k^+ = \frac{1}{\rho \kappa} \sum_{i=1}^{N} g_i^+$ and $\overline{g}_k^- = \frac{1}{\rho \kappa} \sum_{i=1}^{N} g_i^-$. Combined with Equation [14] and [15] we know

\[
\frac{1}{\rho \kappa} \mathbb{E}_{\theta} [\nabla f(x_k, \theta)]^+ \leq \mathbb{E}[\overline{g}_k^+] \leq \rho \kappa \mathbb{E}_{\theta} [\nabla f(x_k, \theta)]^+.
\]

\[
\rho \kappa \mathbb{E}_{\theta} [\nabla f(x_k, \theta)]^- \leq \mathbb{E}[\overline{g}_k^-] \leq \frac{1}{\rho \kappa} \mathbb{E}_{\theta} [\nabla f(x_k, \theta)]^-.
\]

Because $\mathbb{E}[\overline{g}_k] = \mathbb{E}[g_k^+]$, we also have

\[
\text{Var}(\overline{g}_k) = \frac{1}{N^2} \text{Var}(\sum_{i=1}^{N} g_i^+) = \frac{\text{Var}(g_i^+)}{N}.
\]

Then the variance of $\overline{g}_k$ can be bounded as

\[
\text{Var}(\overline{g}_k) \leq \frac{\rho \kappa (\sigma^2 + \epsilon^2)}{N}.
\]
Therefore, we can then apply Theorem 5 to get the result in equation 5.

\[
\mathbb{E}_{\pi_N}[\mathbb{E}_\theta[f(\bar{x}_N, \theta)]] - \mathbb{E}_\theta[f(x^*, \theta)] \\
\leq \frac{\rho^1\kappa^4L^2 + \rho^2\kappa^2\max_k\{\text{Var}(\bar{y}_k)\}}{2\mu K}(1 + \log(K)) \\
\leq \frac{\rho^1\kappa^4L^2 + \rho^2\kappa^2\sigma^2 + \varepsilon^2}{2\mu K}(1 + \log(K))
\]

which can also be written as

\[
\mathbb{E}_{\pi_N}[\text{obj}] - \text{OPT} \\
\leq \left(\frac{\rho^1\kappa^4L^2}{2\mu} + \frac{\rho\kappa(\sigma^2 + \varepsilon^2)}{2\mu N}\right) \frac{1 + \log(K)}{K}.
\]

This completes the proof. \(\square\)

In Theorem 5, we can see the difference to the optimum is inversely proportional to both \(K\) and \(N\) to some extent. To tighten the bound with fixed number of iterations \(K\), we can either conduct more XOR-Sampling scheme leading to smaller \(\rho\), or generate more samples at each iteration to reduce the variance term.

### 3.1 IMPROVED XOR-PGD

Although we prove that the proposed XOR-PGD has a theoretical guarantee on the convergence rate, this convergence rate of \(O\left(\frac{\log K}{K}\right)\) is not rather surprising. Therefore, we further improve the algorithm by selecting a more reasonable step size. By selecting \(t_k = \frac{2\rho c}{\mu(k+1)}\) as in Algorithm 2 and compute the last output from a weighted average \(x_{k+1} = \frac{2\kappa}{\mu(k+1)}x_k\), we develop an improved algorithm of XOR-PGD named (I)XOR-PGD in Algorithm 2. We prove that this (I)XOR-PGD is able to further accelerate the convergence rate to \(O\left(\frac{1}{K}\right)\).

**Theorem 4.** Let \(f : \mathbb{R}^d \to \mathbb{R}\) be a \(L\)-smooth and \(\mu\)-strongly convex function and \(x^* = \arg\min_{x \in C} f(x)\), where \(C\) is a convex set of \(x\). In iteration \(k\), \(g_k\) is the estimated gradient, i.e., \(x_{k+1} = \mathbb{P}_C(x_k - t_kg_k)\) where \(\text{Var}(g_k) \leq \sigma^2\) and \(\mathbb{P}_C(x) := \arg\min_{y \in C} \frac{1}{2}\|x - y\|^2\) is the projection of \(x\) onto \(C\). If \(t_k = \frac{2\rho c}{\mu(k+1)}\) and there exists \(1 \leq c \leq \sqrt{2}\) such that

\[
\frac{1}{c}\|\nabla f(x_k)\|^2 \leq \mathbb{E}[g_k^2] \leq c\|\nabla f(x_k)\|^2 + c\|\nabla f(x_k)\|^+ \leq \mathbb{E}[g_k^+] \leq \frac{1}{c}\|\nabla f(x_k)\|^-,\text{ then the convergence rate of the Improved XOR-PGD algorithm is } O\left(\frac{1}{K}\right):
\]

\[
\mathbb{E}\left[f(\bar{x}_K)\right] - f(x^*) \leq \frac{2B}{\mu(K+1)},
\]

where \(\bar{x}_K = \frac{2}{K(K+1)}\sum_{k=1}^K kx_k\).

**Proof.** With \(t_k = \frac{2\rho c}{\mu(k+1)}\) and multiplying \(6\) by \(k+1\), we have

\[
k(\mathbb{E}[f(x_k)] - f(x^*)) \leq \frac{kB}{\mu(k+1)} \frac{k}{4}(k(k-1))\mathbb{E}[\|x_k - x^*\|^2] - \frac{k}{4}(k(k+1))\mathbb{E}[\|x_{k+1} - x^*\|^2]) \\
\leq \frac{B}{\mu} + \frac{\mu}{4}(k(k-1))\mathbb{E}[\|x_k - x^*\|^2] \\
- \frac{\mu}{4}(k(k+1))\mathbb{E}[\|x_{k+1} - x^*\|^2])
\]

Summing from \(k = 1\) to \(k = K\), we have

\[
\sum_{k=1}^K k(\mathbb{E}[f(x_k)] - f(x^*)) \\
\leq \frac{KB}{\mu} + \frac{\mu}{4}((0 - K(K+1))\mathbb{E}[\|x_{k+1} - x^*\|^2])
\]

Then, we have

\[
\mathbb{E}\left[f\left(\frac{2}{K(K+1)}\sum_{k=1}^K kx_k\right)\right] - f(x^*) + \frac{\mu}{2}\mathbb{E}[\|x_{k+1} - x^*\|^2] \\
\leq \frac{2B}{\mu(K+1)},
\]

which implies

\[
\mathbb{E}\left[f(\bar{x}_K)\right] - f(x^*) \leq \frac{2B}{\mu(K+1)},
\]

where \(\bar{x}_K = \frac{2}{K(K+1)}\sum_{k=1}^K kx_k\). Here we prove the \(O(1/K)\) convergence rate. \(\square\)

Similarly, we can get the final convergence rate of the Improved XOR-PGD as in Algorithm 2 by increasing the sample size from 1 to \(N\).

**Corollary 1.** (Main) Let \(\rho, \kappa\) be as before, function \(f(x, \theta) : \mathbb{R}^d \times \{0, 1\}^n \to \mathbb{R}\) be a \(L\)-Lipschitz and \(\mu\)-strongly convex function w.r.t. \(x\). Denote \(\text{OPT} = \min_{x \in C} \mathbb{E}_\theta[f(x, \theta)]\) as the global optimum. Let \(\sigma^2 = \max_x \{\text{Var}(\nabla_x f(x, \theta))\}\) and \(\varepsilon^2 = \max_x \{\|\nabla_x f(x, \theta)\|^2\}\). For any \(1 \leq \rho\kappa \leq \sqrt{2}\), step size \(t \leq \frac{2\rho\kappa}{\mu(K+1)}\) and sample size \(N \geq 1\), \(\bar{x}_K = \frac{2}{K(K+1)}\sum_{k=1}^K kx_k\) is the output of Improved XOR-PGD (algorithm 2) and \(\text{obj} = \mathbb{E}_\theta[f(\bar{x}_K, \theta)]\) is the objective function value at \(\bar{x}_K\). We have:

\[
\mathbb{E}_{\bar{x}_K}[\text{obj}] - \text{OPT} \\
\leq \left(\frac{\rho^1\kappa^4L^2}{\mu} + \frac{\rho\kappa(\sigma^2 + \varepsilon^2)}{\mu N}\right) \frac{1}{K+1}.
\]

We skip the proof since it is very similar to the proof of Theorem 5. It should be noticed that although hard to compute, \(\sigma^2\) and \(\varepsilon^2\) are from the input which do not depend on the algorithm. In addition, it should be noticed that the convergence rate of our algorithm is determined by the approximation constant \(\rho\kappa\) from XOR-sampling. The smaller this
constant is, the lower the success rate of XOR-Sampling has. By setting proper parameter values, we can get $\rho \kappa = \sqrt{2}$. As a consequence, we can collect $N$ samples successfully by running XOR-Sampling around $40N$ times. The time complexity can be further reduced via parallel sampling. In addition, the sampling procedure is independent of the optimization step since $Pr(\theta)$ does not depend on $x$.

4 EXPERIMENTS

We evaluate our (Improved) XOR-PGD algorithm on the same benchmark of XOR-SGD [Ding and Xue 2021], the inventory management [Ziukov 2016, Shapiro and Philpott 2007]) and the network design problems (Sheldon et al. 2012, Wu et al. 2017, 2016). For comparison, we consider XOR-SGD, and also those that use SGD methods with Gibbs Sampling, Belief Propagation (BP) (Yedidia et al. 2001, Murphy et al. 2013), or Belief Propagation Chain (BPC) (Fan and Xue 2020). Similar to the setting in [Ding and Xue 2021], for each setting of both applications, to produce a sample, we let Gibbs sampling have 100 burn in samples, and then draws one sample every 200 steps. We fix the number of iteration steps of both BP and BPC as 20, which is enough for belief propagation to converge. We allow SGD with Gibbs sampling, BP and BPC to draw more samples than both XOR-PGD and XOR-SGD for a fair comparison. For both applications, we use MRF as probabilistic models for $Pr(\theta)$. All experiments were conducted using single core architectures on Intel Xeon Gold 6126 2.60GHz machines with 96GB RAM and a wall-time limit of 10 hours. We use IBM ILOG Cplex 12.71 as the solver of NP oracle to produce each XOR sample for both XOR-PGD and XOR-SGD. Notice that the projection step in XOR-PGD is also solved by Cplex. Once a solution $x$ is generated by either algorithm, we use an exact weighted counter ACE [Barton et al. 2016] to evaluate $\mathbb{E}_{d \sim P_{r}(d)} f(x, \theta)$ exactly. All objective values reported are from ACE. Since we run each algorithm on one single core with a wall-time limit of 10 hours for a fair comparison thus not all algorithms can complete all iterations, we report the best results found by each algorithm within the time limit.

4.1 STOCHASTIC INVENTORY MANAGEMENT

We first investigate our algorithm on the stochastic inventory management problem studied in [Shapiro and Philpott 2007]. Assuming there are $n$ materials. The demand of material $i$ is $d_i$. Let $d = (d_1, \ldots, d_n)^T$ be the demand vector. The manager stocks $x_i$ amount of material $i$ at the beginning of the season. Each unit of material $i$ takes storage space $w_i$, and the total amount of pre-order is limited by the available storage space $X$. At the end of the production season, demand $d$ will be revealed to the manager. We assume the cost of ordering the $i$-th material is $c_i$ per unit. If the demand $d_i > x_i$, then a back order is needed, of which one unit costs $b_i \geq c_i$. Overall, the cost for back order is $b_i(d_i - x_i)$ if $d_i > x_i$, and is zero otherwise. On the other hand, if $d_i < x_i$, then a holding cost of $h_i$ per unit is incurred, leading to an additional total cost $h_i(x_i - d_i)$. Summing it up, the cost for material $i$ is $G_i = c_i x_i + b_i (d_i - x_i)^+ + h_i (x_i - d_i)^+$ where $[a]^+$ denotes the maximum of $a$ and 0. Then, the total cost will be $G(x, d) = \sum_{i=1}^{n} G_i$. The manager want to minimize his operational cost, which translates to this problem:

$$\min_{x \geq 0} \mathbb{E}_{d \sim Pr(d)} [G(x, d)], \text{ s.t. } w^T x \leq X. \quad (9)$$

where $G(x, d)$ is convex w.r.t. $x$. We run the experiments varying the number of materials $n$, the storage limit, and the number of samples we use in XOR-PGD and other methods. Parameters and experimental settings are the same as in [Ding and Xue 2021]. In terms of the parameters in XOR-Sampling we fix $P = 100$, $b = 7$, $t = 0.01$ and the others the same as in [Ermon et al. 2013] to guarantee $\rho \kappa = \sqrt{2}$. Learning rate $t$ is 0.1 at first and divided by 10 after 50 iterations, then further divided by 10 after 100 iterations. $\eta$ is 10 at first and divided by 10 after 50 iterations, then further divided by 10 after 100 iterations. The total number of both $K$ and $M$ are set to be 200. However, since we run each algorithm on one single core with a wall-time limit of 10 hours for a fair comparison, not all algorithms can complete all iterations. The plots are based on the best results found by each algorithm within the time limit.

Table 1 shows that our algorithm outperforms other methods varying the number of materials on the percentage reduction of the objective values of the solutions. In math form, for example for Gibbs Sampling, the metric is $(\text{obj}(\text{Gibbs}) - \text{obj}((I)\text{XOR-PGD}))/\text{obj}(\text{Gibbs})$ (metrics for other approaches are analogous). The objective optimized by (I)XOR-PGD is on average 2 percent better than XOR-SGD and 10% more than that optimized by other baselines. In addition, Table 2 shows that the rate of constraints satisfaction of the solutions from either XOR-PGD or (I) XOR-PGD is 10% more than that of XOR-SGD even in very large searching space. The first row in Figure 1 shows the objective values of solutions varying the storage limit, and the objective values varying the number of samples with 100% storage limit and 50 different materials. We can see from the left figure that with the storage limit increasing, (I)XOR-PGD is always better than all baselines, and from the right figure that (I)XOR-PGD found better solutions with less samples than XOR-SGD. Since Ding and Xue [2021] already reported that XOR-SGD is faster than the other competing approaches, here by comparing efficient sample size with XOR-SGD we conclude that (I)XOR-PGD is faster than all the other methods.
We evaluate our algorithms on the Flood Preparation problem, where results clearly show that (I)XOR-PGD outperforms other methods both in efficiency and in the quality of solutions. On all of the four different type of networks (I)XOR-PGD saves on average > 3% against XOR-SGD and more than 9% against the other competing methods. In addition, we would like to emphasize that (I)XOR-PGD with 40 samples already outperforms XOR-SGD with 60 samples in the right figure while finds better solutions at the same time, and even naive XOR-PGD finds better solutions than XOR-SGD given the same sample size. The largest budget size $B$ is 1000. Total number of SGD iterations is 2000, while not all algorithms can complete all 2000 iterations within the time limit of 10 hours. Parameters in XORSampling are set to be the same as in Ding and Xue [2021].

The results are similar to those for the inventory management problem, where results clearly show that (I)XOR-PGD outperforms other methods both in efficiency and in the quality of solutions. On all of the four different type of networks (I)XOR-PGD saves on average > 3% against XOR-SGD and more than 9% against the other competing methods. In addition, we would like to emphasize that (I)XOR-PGD with 40 samples already outperforms XOR-SGD with 60 samples in the right figure while finds better solutions at the same time, and even naive XOR-PGD finds better solutions than XOR-SGD given the same sample size.

The left figure in Figure 3 shows the percentage of savings between SGD with other sampling methods and XOR-SGD among all of the 4 different networks, while the middle and the right figures show the averaged commuting time with regard to different budget sizes and different number of samples, respectively. For the left and the middle figures, we let XOR-SGD take 100 samples in each iteration while SGD with other methods take 10,000. We can see from the left figure that objective optimized by XOR-SGD is at least 5% better than that optimized by other methods for all the 4 different networks. In addition, from the middle and the right figures we know that with the increase of either budget size or the number of samples, our method can find consistently better solutions than the compared methods. In particular,
from the right figure we can see even 40 samples in each iteration are enough for XOR-SGD to compete with the result from Gibbs with 20,000 samples. Meanwhile, XOR-SGD also runs faster than the compared method under this situation. In this experiment, XOR-SGD with 40 samples take 1 minutes 40 seconds per SGD iteration, while SGD with 20,000 Gibbs samples need 2.5 minutes per iteration. Since sampling time of both BP and BPChain is no shorter than Gibbs Sampling, we thus conclude that XOR-SGD outperforms other methods both in efficiency and in the quality of solutions found.

5 CONCLUSION

We proposed XOR-PGD, a provable algorithm to attack constrained convex stochastic optimization problems, which are crucial for many decision-making applications with uncertainty. We showed theoretically that our algorithm has a linear convergence rate to the global optimum by choosing proper step sizes. Empirically, we demonstrated the superior performance of XOR-PGD on both the stochastic inventory management and the stochastic network design problems. In particular, 90% solutions obtained by XOR-PGD satisfy the constraints set even when the searching space is very large in the inventory management problem, approximately 10% more that of competing methods. Besides, XOR-PGD converges faster than XOR-SGD by accessing 20% less XOR samples in each iteration and is able to find better solutions in the stochastic network design problem. Overall, our paper demonstrates the power of integrating cutting-edge computer science technology with real-world problems. Our paper will also stimulate further academic progress in stochastic optimization, constrained optimization, probabilistic inference with hashing and randomization, and non-convex optimizations with insights from real-world applications. Future work includes tightening the constant bound and accelerating the convergence rate. We will keep active to investigate if our approach can motivate new algorithms for non-convex stochastic optimization problems.
References

Naman Agarwal, Zeyuan Allen-Zhu, Brian Bullins, Elad Hazan, and Tengyu Ma. Finding approximate local minima faster than gradient descent. In Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, pages 1195–1199, 2017.

Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. The Journal of Machine Learning Research, 18(1):8194–8244, 2017.

Zeyuan Allen-Zhu. Natasha 2: Faster non-convex optimization than sgd. In Advances in neural information processing systems, pages 2675–2686, 2018.

John P Barton, Eleonora De Leonardis, Alice Coucke, and Simona Cocco. Ace: adaptive cluster expansion for maximum entropy graphical model inference. Bioinformatics, 32(20):3089–3097, 2016.

Fan Ding and Yexiang Xue. Xor-sgd: Provable convex stochastic optimization for decision-making. In Uncertainty in Artificial Intelligence. UAI, 2021.

Fan Ding, Jianzhu Ma, Jinbo Xu, and Yexiang Xue. Xorc: Linearly convergent constrained structure generation. In International conference on machine learning. ICML, 2021.

Justin Domke. Learning graphical model parameters with approximate marginal inference. IEEE transactions on pattern analysis and machine intelligence, 35(10):2454–2467, 2013.

Simon S Du and Wei Hu. Linear convergence of the primal-dual gradient method for convex-concave saddle point problems without strong convexity. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 196–205. PMLR, 2019.

Kumar Avinava Dubey, Sashank J Reddi, Sinead A Williamson, Barnabas Poczos, Alexander J Smola, and Eric P Xing. Variance reduction in stochastic gradient langevin dynamics. In NIPS, pages 1154–1162, 2016.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. Journal of machine learning research, 12(Jul): 2121–2159, 2011.

John Duchi, Feng Ruan, and Chulhee Yun. Minimax bounds on stochastic batched convex optimization. In Conference On Learning Theory, pages 3065–3162, 2018.

Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In Proceedings of the 30th ICML, 2013a.

Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Embed and project: Discrete sampling with universal hashing. In Advances in Neural Information Processing Systems (NIPS), 2013b.

Ding Fan and Yexiang Xue. Contrastive divergence learning with chained belief propagation. In International Conference on Probabilistic Graphical Models, 2020.

Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In Conference on Learning Theory, pages 797–842, 2015.

Arpita Ghosh, Stephen Boyd, and Amin Saberi. Minimizing effective resistance of a graph. SIAM review, 50(1):37–66, 2008.

Carla Gomes, Thomas Dietterich, Christopher Barrett, Jon Conrad, Bistra Dilkina, Stefano Ermon, Fei Fang, Andrew Farnsworth, Alan Fern, Xiaoli Fern, et al. Computational sustainability: Computing for a better world and a sustainable future. Communications of the ACM, 62(9):56–65, 2019.

Erfan Yazdandoost Hamedani and Necdet Serhat Aybat. A primal-dual algorithm with line search for general convex-concave saddle point problems. arXiv preprint arXiv:1803.01401, 2018.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. 2012.

Robert M. Inman, Brent L. Brock, Kristine H. Inman, Shawn S. Sartorius, Bryan C. Aber, Brian Giddings, Steven L. Cain, Mark L. Orme, Jay A. Fredrick, Bob J. Oakleaf, Kurt Alt, Eric A. Odell, and Guillaume Chapron. Developing priorities for metapopulation conservation at the landscape scale: Wolverines in the western united states. 2013.

Chi Jin, Praneeth Netrapalli, and Michael I Jordan. Accelerated gradient descent escapes saddle points faster than gradient descent. arXiv preprint arXiv:1711.10456, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

Jason D Lee, Qihang Lin, Tengyu Ma, and Tianbao Yang. Distributed stochastic variance reduced gradient methods and a lower bound for communication complexity. arXiv preprint arXiv:1507.07595, 2015.

Qiang Liu and Alexander T. Ihler. Variational algorithms for marginal MAP. Journal of Machine Learning Research, 14, 2013.
Yuanyuan Liu, Fanhua Shang, and James Cheng. Accelerated variance reduced stochastic admm. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 31, 2017.

Radu Marinescu, Rina Dechter, and Alexander T. Ihler. AND/OR search for marginal MAP. In Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, UAI, 2014.

Radu Marinescu, Rina Dechter, and Alexander Ihler. Pushing forward marginal map with best-first search. In Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI), 2015.

Denis Deratani Mauá and Cassio Polpo de Campos. Anytime marginal MAP inference. In Proceedings of the 29th ICML, 2012.

Meredith L McClure, Andrew J. Hansen, and Robert M. Inman. Connecting models to movements: testing connectivity model predictions against empirical migration and dispersal data. Landscape Ecology, 31:1419–1432, 2016.

Aryan Mokhtari, Asuman E Ozdaglar, and Sarath Pattathil. Convergence rate of o(1/k) for optimistic gradient and extragradient methods in smooth convex-concave saddle point problems. SIAM Journal on Optimization, 30(4):3230–3251, 2020.

Kevin Murphy, Yair Weiss, and Michael I Jordan. Loopy belief propagation for approximate inference: An empirical study. arXiv preprint arXiv:1301.6725, 2013.

Hua Ouyang, Niao He, Long Tran, and Alexander Gray. Stochastic alternating direction method of multipliers. In International Conference on Machine Learning, pages 80–88. PMLR, 2013.

Sebastian Ruder. An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747, 2016.

Alexander Shapiro and Andy Philpott. A tutorial on stochastic programming. 2007.

Daniel Sheldon, Bistra Dilkina, Adam N Elmachtoub, Ryan Finseth, Ashish Sabharwal, Jon Conrad, Carla P Gomez, David Shmoys, William Allen, Ole Amundsen, et al. Maximizing the spread of cascades using network design. arXiv preprint arXiv:1203.3514, 2012.

Eric Sodomka, John Collins, and Maria Gini. Efficient statistical methods for evaluating trading agent performance. 2007.

Chong Wang, Xi Chen, Alexander J Smola, and Eric P Xing. Variance reduction for stochastic gradient optimization. In Advances in Neural Information Processing Systems, pages 181–189, 2013.

Yuanhao Wang and Jian Li. Improved algorithms for convex-concave minimax optimization. arXiv preprint arXiv:2006.06359, 2020.

Xiaoqian Wu, Daniel R Sheldon, and Shlomo Zilberstein. Optimizing resilience in large scale networks. In Proceedings of the 30th Conference of AAAI, 2016.

Xiaoqian Wu, Yexiang Xue, Bart Selman, and Carla P. Gomez. Xor-sampling for network design with correlated stochastic events. In Proceedings of the 26th IJCAI, pages 4640–4647, 2017.

Guangzeng Xie, Luo Luo, Yijiang Lian, and Zhihua Zhang. Lower complexity bounds for finite-sum convex-concave minimax optimization problems. In International Conference on Machine Learning, pages 10504–10513. PMLR, 2020.

Yexiang Xue, Zhiyuan Li, Stefano Ermon, Carla P. Gomes, and Bart Selman. Solving marginal map problems with np oracles and parity constraints. In Proceedings of the 29th Annual Conference on NIPS, 2016.

Jonathan S Yedidia, William T Freeman, and Yair Weiss. Generalized belief propagation. In Advances in neural information processing systems, pages 689–695, 2001.

Shuai Zheng and James T Kwok. Fast-and-light stochastic admm. In IJCAI, pages 2407–2613, 2016.

Serhii Ziukov. A literature review on models of inventory management under uncertainty. 2016.
APPENDIX

A XOR-SAMPLING FOR THE WEIGHTED CASE

The text here provides a synopsis for the approach in Ermon et al. [2013b]. We still encourage the readers to read the original text for a better explanation. Let \( w(\theta), p(\theta) \) and \( Z \) as defined before, the high-level idea of XOR-Sampling is to first discretize \( w(\theta) \) to \( w'(\theta) \) as in Definition 1 followed by embedding the weighted \( w'(\theta) \) to the unweighted space \( \Delta_w \). Finally, XOR-sampling uses counting based on hashing and randomization to sample uniformly from \( \Delta_w \).

**Definition 1.** Assume \( w(\theta) \) has both upper and lower bound, namely, \( M = \max_\theta w(\theta) \) and \( m = \min_\theta w(\theta) \). Let \( b \geq 1, \epsilon > 0, r = 2^b/(2^b - 1) \) and \( l = [\log_\epsilon(2^n/\epsilon)] \). Partition the configurations into the following weighted based disjoint buckets: \( B_i = \{ \theta | w(\theta) \in (\frac{M}{r}, \frac{M}{r^i}] \}, i = 0, \ldots, l - 1 \) and \( B_l = \{ \theta | w(\theta) \in (0, \frac{M}{r^l}) \} \). The discretized weight function \( w' : \{0, 1\}^n \rightarrow \mathbb{R}^+ \) is defined as follows: \( w'(\theta) = \frac{M}{r^i} \), if \( \theta \in B_i, i = 0, \ldots, l - 1 \) and \( w'(\theta) = 0 \) if \( \theta \in B_l \). This leads to the corresponding discretized probability distribution \( p'(\theta) = w'(\theta)/Z' \) where \( Z' \) is the normalization constant of \( w'(\theta) \).

For the weighted case, the goal of XOR-sampling is to guarantee that the probability of sampling one \( \theta \) is proportional to the unnormalized density (up to a multiplicative constant). By Definition 1 we obtain a distribution \( p'(x) \) which satisfying \( \frac{1}{\rho} p(x) \leq p'(x) \leq \rho p(x) \) where \( \rho = \frac{r^2}{\epsilon} \). Then, XOR-sampling implements a horizontal slice technique to transform a weighted problem into an unweighted one. For the easiness of illustration, we denote \( M' = \max_\theta w'(\theta) \) and \( m' \) as the smallest non-zero value of \( w'(\theta) \). Then consider the simple case where \( b = 1 \) and \( r = 2 \), where we have \( M' = 2^{l-1}m' \). Let \( \delta = (\delta_0, \ldots, \delta_{l-2})^T \in \{0, 1\}^{l-1} \) be a binary vector of length \( l - 1 \), XOR-sampling samples \( \{(\theta, \delta) : w'(\theta) \leq 2^{l+1}m' \Rightarrow \delta'_0 = 0\} \).

If we sample \( (\theta, \delta) \) uniformly at random from \( \Delta_w \) and then only return \( \theta \), it can be proved that the probability of sampling \( \theta \) from \( w'(\theta) \) is proportional to \( m'2^{l-1} \) when \( w(\theta) \) is sandwiched between \( m'2^{l-1} \) and \( m'2^l \). Therefore, this technique leads to the constant approximation guarantee of XOR-Sampling. The precise statement of the guarantee is in Theorem 2. For general case of \( b \) and \( r \), please refer to Ermon et al. [2013b].

Setting \( \epsilon \eta_0 \) to Zero In Definition 1 we can make \( b \) larger and \( \epsilon \) smaller enough, then there will be a possibly large but finite value of \( l \) such that \( \frac{M}{r^l} \) is smaller than \( m \), which leads \( B_l \) to be empty and \( \epsilon \eta_0 \) to be zero.

B PROOFS

B.1 PROOF OF LEMMA 1

We define two functions \( g_k^+ = \max\{g_k, 0\} \) and \( g_k^- = \min\{g_k, 0\} \) where \( 0 \) is a vector of all 0 which has the same dimension as \( g_k \). We have \( g_k = g_k^+ + g_k^- \). We define both \( \nabla f(x_k)^+ \) and \( \nabla f(x_k)^- \) in the similar way. Then Lemma 1 gives the new bounds of two terms assuming the constant bound on the gradient, which are essential to the proof of convergence rate. The proof of Lemma 1 is as follows:

Proof. (Lemma 1) Since we have the constant bound that

\[
\frac{1}{c} \nabla f(x_k)^+ \leq \mathbb{E}[g_k^+] \leq c \nabla f(x_k)^+. \quad (11)
\]

\[
\frac{1}{c} \nabla f(x_k)^- \leq \mathbb{E}[g_k^-] \leq \frac{1}{c} \nabla f(x_k)^-. \quad (12)
\]

and because of \( g_k^+ \geq 0 \) and \( g_k^- \leq 0 \) we can obtain

\[
\frac{1}{c} \mathbb{E}[g_k^+] \leq \frac{1}{c} \mathbb{E}[g_k^+], \quad \mathbb{E}[g_k^-] \leq \frac{1}{c} \mathbb{E}[g_k^-].
\]

which exactly means

\[
\frac{1}{c} \mathbb{E}[g_k] \leq \frac{1}{c} \mathbb{E}[g_k], \quad \mathbb{E}[g_k] \leq \mathbb{E}[g_k] \leq c \mathbb{E}[g_k].
\]

To prove the second inequality, we need to take advantage of the convexity of \( f \). Denote \( \{x_k - x^*\}^+ = \max\{x_k - x^*, 0\} \) and \( \{x_k - x^*\}^- = \min\{x_k - x^*, 0\} \), we know \( x_k - x^* = \{x_k - x^*\}^+ + \{x_k - x^*\}^- \). In addition, because \( f \) is convex, the index set of non-zero entries of \( \{x_k - x^*\}^+ \) and \( \nabla f(x_k)^+ \) is the same. The index set of non-zero entries of \( \{x_k - x^*\}^- \) and \( \nabla f(x_k)^- \) is also the same. In addition, because of Equation 11 and 12, the index set of non-zero entries of \( \mathbb{E}[g_k^+] \) (\( \mathbb{E}[g_k] \)) is the same with \( \nabla f(x_k)^+ \) (\( \nabla f(x_k)^- \)).

Combining these facts with Equations 11 and 12 we have

\[
\frac{1}{c} \mathbb{E}[g_k^+], \quad \frac{1}{c} \mathbb{E}[g_k^-] \leq \frac{1}{c} \mathbb{E}[g_k], \quad \frac{1}{c} \mathbb{E}[g_k] \leq \frac{1}{c} \mathbb{E}[g_k].
\]

Combining these two equations, we have

\[
\frac{1}{c} \mathbb{E}[g_k], \quad \frac{1}{c} \mathbb{E}[g_k] \leq \frac{1}{c} \mathbb{E}[g_k], \quad \frac{1}{c} \mathbb{E}[g_k] \leq \frac{1}{c} \mathbb{E}[g_k].
\]

This completes the proof. □
B.2 PROOF OF THEOREM 4

Theorem 5. (Main) Let \( b, \epsilon, \delta, \alpha, \rho, \kappa \) and \( B_t \) be as in section A in appendix, function \( f(x, \theta) : \mathbb{R}^d \to \mathbb{R} \) be a \( L \)-smooth convex function w.r.t. \( x \). Denote \( OPT = \min_{x} \mathbb{E}_{\theta \sim \mathcal{P}(\theta)} f(x, \theta) \) as the global optimum. Let \( \sigma^2 = \max_{x} \{ \mathbb{V} \mathbb{a} \mathbb{r}(\nabla_x f(x, \theta)) \} \) and \( \varepsilon^2 = \max_{x} \{ \| \nabla_x f(x, \theta) \|_2 \}^2 \). For any \( 1 \leq \rho \kappa \leq \sqrt{2} \) step size \( t \leq \frac{L_{\rho \kappa}^2}{\rho \kappa} \) and sample size \( N \geq 1 \), \( \overline{x}^{\mathcal{F}} \) is the output of XOR-SGD and \( \text{obj} = \mathbb{E}_{\theta} [f(\overline{x}^{\mathcal{F}}, \theta)] \) is the objective function value at \( \overline{x}^{\mathcal{F}} \). We have:

\[
\mathbb{E}_{\overline{x}^{\mathcal{F}}} [\text{obj}] - \text{OPT} \leq \frac{\rho \kappa \| x_0 - x^* \|_2^2 + t(\sigma^2 + \varepsilon^2)}{2tK}.
\]

Proof. (Theorem 4) Since we use \( N \) samples at each iteration, we have \( \overline{x}_{\rho \kappa} = \frac{1}{N} \sum_{i=1}^{N} g_k \) and \( \mathbb{E}[\overline{x}_{\rho \kappa}] = \mathbb{E}[g_k] \). In each iteration \( k \) we can adjust the parameters in XOR-Sampling to make the tail \( \epsilon_{\overline{n}_{\rho \kappa}} \) zero, then for each sample \( g_k \) we can obtain from Theorem 2 that

\[
\left\| \frac{1}{\rho \kappa} \mathbb{E}_\theta [\nabla f(x_k, \theta)]^+ \right\|_2 \leq \mathbb{E}[g_k^+] \leq \rho \kappa \mathbb{E}_\theta [\nabla f(x_k, \theta)]^+.
\]

\[
\rho \kappa \mathbb{E}_\theta [\nabla f(x_k, \theta)]^- \leq \mathbb{E}[g_k^-] \leq \frac{1}{\rho \kappa} \mathbb{E}_\theta [\nabla f(x_k, \theta)]^-.
\]

The variance of each sample \( g_k \) can also be bounded by

\[
\text{Var}(g_k) = \mathbb{E}_{\bm{\theta} \sim \mathcal{P}(\bm{\theta})} [\| \nabla f(x_k, \theta)^2 ] - \left\| \mathbb{E}_{\bm{\theta} \sim \mathcal{P}(\bm{\theta})} [\nabla f(x_k, \theta)]^2 \right\|^2,
\]

\[
\leq \rho \kappa \mathbb{E}_{\bm{\theta} \sim \mathcal{P}(\bm{\theta})} [\| \nabla f(x_k, \theta)]^2 \right\|^2,
\]

\[
= \rho \kappa (\text{Var}(\nabla f(x_k, \theta)) + \left\| \mathbb{E}_{\bm{\theta} \sim \mathcal{P}(\bm{\theta})} [\nabla f(x_k, \theta)]^2 \right\|^2),
\]

\[
\leq \rho \kappa (\sigma^2 + \varepsilon^2).
\]

Denote \( \overline{g}_k^+ = \max\{g_k, 0\} \) and \( \overline{g}_k^- = \min\{g_k, 0\} \). Clearly, \( g_k^+ \geq 0 \) and \( g_k^- \leq 0 \). Moreover, for a given dimension, either \( g_k^+ = 0 \) for that dimension or \( g_k^- = 0 \). Evaluating \( \overline{g}_k \) by dimension by dimension, we can see that \( \overline{g}_k^+ = \frac{1}{N} \sum_{i=1}^{N} g_k^+ \) and \( \overline{g}_k^- = \frac{1}{N} \sum_{i=1}^{N} g_k^- \). Combined with Equations 14 and 15 we know

\[
\frac{1}{\rho \kappa} \mathbb{E}_\theta [\nabla f(x_k, \theta)]^+ \leq \mathbb{E}[\overline{g}_k^+] \leq \rho \kappa \mathbb{E}_\theta [\nabla f(x_k, \theta)]^+.
\]

\[
\rho \kappa \mathbb{E}_\theta [\nabla f(x_k, \theta)]^- \leq \mathbb{E}[\overline{g}_k^-] \leq \frac{1}{\rho \kappa} \mathbb{E}_\theta [\nabla f(x_k, \theta)]^-.
\]

Because \( \mathbb{E}[\overline{g}_k] = \mathbb{E}[g_k] \), we also have

\[
\text{Var}(\overline{g}_k) = \frac{1}{N^2} \sum_{i=1}^{N} \text{Var}(g_k^i) = \frac{\text{Var}(g_k)}{N}.
\]

Then the variance of \( \overline{g}_k \) can be bounded as

\[
\text{Var}(\overline{g}_k) \leq \frac{\rho \kappa (\sigma^2 + \varepsilon^2)}{N}.
\]

Therefore, we can then apply Theorem 3 to get the result in equation 5.

\[
\mathbb{E}_{\overline{x}^{\mathcal{F}}} [\mathbb{E}_{\theta} [f(\overline{x}^{\mathcal{F}}, \theta)]] - \mathbb{E}_{\theta} [f(x^*, \theta)] \leq \frac{\rho \kappa \| x_0 - x^* \|_2^2 + \left( \max_{\mathcal{F}} \{ \text{Var}(\overline{g}_k) \} \right)}{2tK} + \rho \kappa \frac{\| x_0 - x^* \|_2^2 + t(\sigma^2 + \varepsilon^2)}{N}.
\]

which can also be written as

\[
\mathbb{E}_{\overline{x}^{\mathcal{F}}} [\text{obj}] - \text{OPT} \leq \frac{\rho \kappa \| x_0 - x^* \|_2^2 + t(\sigma^2 + \varepsilon^2)}{2tK} + \frac{\rho \kappa (\sigma^2 + \varepsilon^2)}{N}.
\]

This completes the proof. □

C EXPERIMENTS

We evaluate our XOR-SGD algorithm on the inventory management problem [Zukov 2016, Shapiro and Philpott 2007] and the network design problems [Sheldon et al. 2012, Wu et al. 2017 2016]. For each setting of both applications, to produce a sample, Gibbs sampling first takes 100 steps to burn in, and then draws samples every 30 steps. We fix the iteration step of both BP and BPChain as 20, which is enough for BP to converge. We allow SGD with Gibbs sampling, BP and BPChain to draw more samples than XOR-SGD for a fair comparison. All experiments were conducted using single core architectures on Intel Xeon Gold 6126 2.60GHz machines with 96GB RAM and a wall-time limit of 10 hours. For both applications, we use MRF as probabilistic models for \( P(\theta) \), which can be seen in the next section. For a fair comparison, once a solution \( x \) is generated by either algorithm, we use an exact weighted counter ACE [Barton et al. 2016] to evaluate \( \mathbb{E}_{\theta \sim \mathcal{P}(\theta)} f(x, \theta) \) exactly. All objective values reported here are from ACE.

C.1 SETTINGS OF STOCHASTIC INVENTORY MANAGEMENT

Taking into account of the storage constraint, the original problem is equivalent to the following problem:

\[
\min \max_{x \geq 0, \mu \geq 0} \mathbb{E}_{d \sim P(r(d))} [G(x, d)] + \mu (w^T x - X).
\]

For inventory management problem, we assume each \( d_i \) can take two different values, one corresponding to the high demand one corresponding to the low demand. Then, we introduce a new vector \( \theta \) where \( \theta_i = 1 \) means \( d_i \) is the high value while \( \theta_i = 0 \) otherwise. In the experiment we range \( n \) from 10 to 100 increased by a step size of 10 and draw 10 instances for each setting. Under each setting, we draw every \( c_i \) uniformly from \((0, 5)\), \( h_i \) uniformly from \((0, 10)\), sample \( s_i \) uniformly drawn from \((0, 10)\) and let \( b_i = c_i + s_i \). The two values of each \( d_i \) are also uniformly drawn from \((0, 10)\). We
model $P_r(\theta)$ as a MRF with several cliques. The variables in each clique are highly correlated with each other. For a problem with $n$ products, we draw the number of cliques uniformly from $[n, 2n]$. The domain size of each clique $\phi_\alpha$ is chosen from the range of $[1, 6]$ at random. The potential function of a clique involving $l$ variables is in the form of a table of size $2^l$. The $i$-th entry of this table, denoted as $v_i$, is modeled as $v_i = v_{i1} + v_{i2}v_{i3}$, where $v_{i1}$ is uniformly drawn from $(0, 1)$, $v_3$ uniformly from $(10, 1000)$ and binary variable $v_{i2}$ uniformly randomly drawn from $\{0, 1\}$. Each storage requirement $w_i$ is drawn from $(0, 10)$ uniformly at random. The largest storage limit $X$ is set to be $5n$. We also evaluate our method given different percentages of the largest storage limit, which is shown in Figure 2 (middle). In the SGD algorithm, $x$ is initialized with the absolute value of a Gaussian random variable from $\mathcal{N}(5, 3)$ to ensure it is non-negative.

C.2 SETTINGS OF STOCHASTIC NETWORK DESIGN

The task in equation 8 is equivalent to solving the following problem:

$$\min_{\Delta g \geq 0} \max_{\mu \geq 0} \mathbb{E}_{\theta \sim P_r(\theta)}[C(g + \Delta g, \theta)] + \mu(\sum_{e \in E} c_e \Delta g_e - B).$$

(18)

Because of the convexity of $C(g + \Delta g, \theta)$ and strong duality, both problems have the same optimal solution.

We test our algorithm on a real-world problem, the so-called Flood Preparation problem for the emergency medical services (EMS) on road networks [Wu et al. 2016]. The problem setup, including the graph structure and the definition of $P_r(\theta)$, are the same as that in [Wu et al. 2016]. The original network is unweighted, hence we set the initial conductance value for each edge as 1. $c_e$ is initialized uniformly from the range $(0, 10)$. The largest budget size $B$ is 1000. We evaluate our method varying the percentage allowed of the largest budget size, which is shown in Figure 3 (middle). In the experiment, each entry of $\Delta g$ is initialized with the absolute value of a Gaussian random variable from $\mathcal{N}(0, 1)$. Total number of SGD iterations is 2000, while not all algorithms can complete all 2000 iterations within the time limit of 10 hours. The experimental results reported in the plots are based on the best solutions found by each algorithm within the time limit. Learning rate $t$ is 1 at first and divided by 10 after 20 iterations, further by 10 after 100 iterations. Parameters in XOR-Sampling are set to be the same as in the inventory management problem.