Conditional Gradient Methods for Stochastically Constrained Convex Minimization

Maria-Luiza Vladarean [1] Ahmet Alacaoglu [1] Ya-Ping Hsieh [1] Volkan Cevher [1]

Abstract

We propose two novel conditional gradient-based methods for solving structured stochastic convex optimization problems with a large number of linear constraints. Instances of this template naturally arise from SDP-relaxations of combinatorial problems, which involve a number of constraints that is polynomial in the problem dimension. The most important feature of our framework is that only a subset of the constraints is processed at each iteration, thus gaining a computational advantage over prior works that require full passes. Our algorithms rely on variance reduction and smoothing used in conjunction with conditional gradient steps, and are accompanied by rigorous convergence guarantees. Preliminary numerical experiments are provided for illustrating the practical performance of the methods.

1. Introduction

We study the following optimization template:

$$\min_{x\in\mathcal{X}} \; f(x) := \mathbb{E}\left[f(x,\xi)\right]$$
$$A(\xi)x \in b(\xi) \text{ almost surely,} \tag{1}$$

where $f(x,\xi) : \mathbb{R}^d \to \mathbb{R}$ are random convex functions with L_f-Lipschitz gradient, $\mathcal{X}$ is a convex and compact set of $\mathbb{R}^d$, $A(\xi)$ is an $m \times d$ matrix-valued random variable, and $b(\xi)$ is a closed and projectable random convex set in $\mathbb{R}^m$.

Stochastically constrained convex optimization problems have recently gained interest in the machine learning community, as they provide a convenient and powerful framework for handling instances subject to a large, or even infinite number of constraints. For example, convex feasibility and optimal control problems have variables lying in a possibly infinite intersection of stochastic, projectable constraint sets, and hence are tackled through this lens by Patrascu & Necoara (2017). Xu (2018) also studies the minimization of a stochastic objective controlled by a very large number of stochastic functional constraints, with application to stochastic linear programming. Finally, put forth by Fercoq et al. (2019), extensions to situations where the number of constraints is unknown (e.g. online settings) can be modeled by a template highly similar to (1), thus addressing important applications such as online portfolio optimization.

In this paper, we are interested in a class of applications which can benefit from being cast under template (1), namely semidefinite programs (SDPs) with a large number of linear constraints, such as arise in combinatorial optimization. A prominent example in machine learning is the k-means clustering problem, whose SDP relaxation comprises $\mathcal{O}\left(d^2\right)$ linear constraints where d is the number of data samples (Peng & Wei, 2007). Maximum a posteriori estimation (Huang et al., 2014), quadratic assignment (Burer & Monteiro, 2005), k-nearest neighbor classification (Weinberger & Saul, 2009) and Sparsest cut (Arora et al., 2009) are other relevant SDP instances with linear constraints of order $\mathcal{O}\left(d^2\right)$ or $\mathcal{O}\left(d^3\right)$. Coupled with large input dimensions, such SDPs become problematic for most existing methods, due to the high cost of processing the constraints in-full during optimization.

In contrast, casting such SDPs into (1) suggests a simple solution: treat the linear constraints stochastically by only accessing a random subset at each iteration, then solve (1) using cheap gradient methods. However, the bottleneck in executing this idea is that existing methods require the constraint $\mathcal{X}$ to posses an efficient projection oracle, whereas projecting onto the semidefinite cone amounts to full singular value decompositions, an operation that is prohibitively expensive even when the problem dimension is moderate. We hence ask:

Does a scalable method exist for solving (1) when the set $\mathcal{X}$ does not have an efficient projection oracle?

The present work resolves the above challenge in the pos-

[1]École Polytechnique Fédérale de Lausanne, Switzerland. Correspondence to: Maria-Luiza Vladarean <maria-luiza.vladarean@epfl.ch>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

itive. To this end, we borrow tools from the conditional gradient methods (CGM) (Frank & Wolfe, 1956; Jaggi, 2013), which rely on the generally cheaper *linear minimization oracles* (lmo), rather than their projection counterparts. In particular, as the Lanczos method enables an efficient lmo computation for the spectrahedron (Arora et al., 2005), CGMs have already been proposed for solving SDPs (Jaggi, 2013; Garber & Hazan, 2016; Yurtsever et al., 2018; Locatello et al., 2019). However, none of these methods can handle the constraints stochastically.

In a nutshell, our approach relies on *homotopy smoothing* of the stochastic constraints in conjunction with CGM steps and a carefully chosen variance reduction procedure. Our analysis gives rise to two fully stochastic algorithms for solving problem (1) without projections onto $\mathcal{X}$. The first of the methods, H-SFW1, relies on a single sample (or fixed batch size) for computing the variance-reduced gradient and converges at a cost of $\mathcal{O}(\epsilon^{-6})$ lmo calls and $\mathcal{O}(\epsilon^{-6})$ stochastic first-order oracle (sfo) calls. The second, H-SPIDERFW, uses batches of increasing size under the SPIDER variance reduction scheme (Fang et al., 2018) and attains a theoretical complexity of $\mathcal{O}(\epsilon^{-2})$ lmo calls and $\mathcal{O}(\epsilon^{-4})$ sfo calls. The difference in convergence rates emphasizes the trade-off between between the computational cost per-iteration and the number of iterations required to reach the constrained optimum.

2. Related Work

The present work lies at the intersection of several lines of research, whose relevant literature we describe in the following sections.

Proximal Methods for Almost Sure Constraints. Problems of similar formulation to (1) have been addressed in prior literature under the assumption of an efficient projection oracle over $\mathcal{X}$. Works such as (Patrascu & Necoara, 2017; Xu, 2018; Fercoq et al., 2019) solve these problems via stochastic proximal methods and attain a complexity of $\mathcal{O}(\epsilon^{-2})$ sfo calls, which is known to be optimal even for unconstrained stochastic optimization. In particular, Patrascu & Necoara (2017) study convex constrained optimization, where the constraints are expressed as a (possibly infinite) intersection of stochastic, closed, convex and projectable sets X_ξ. Problem (1) can be partly cast to this template, with $A(\xi)X \in b(\xi)$ being the homologue of X_ξ. However, our additional set $\mathcal{X}$ does not allow for efficient projections, making this framework inapplicable.

Xu (2018) solves a convex constrained optimization problem over a convex set $\mathcal{X}$, subject to a large number of convex functional constraints $f_j,\ j = 1 \dots M$. The functions f_j are sampled uniformly at random during optimization, which corresponds to a finitely sampled instance of problem (1) for affine f_j. However, we meet again with the limiting condition that projections onto $\mathcal{X}$ are computationally expensive in our setting.

Finally, Fercoq et al. (2019) study convex problems subject to a possibly infinite number of almost sure linear inclusion constraints, a template which closely resembles ours. The limitation, however, lies in their inclusion of a proximal-friendly component in the objective used to perform stochastic proximal gradient steps. This assumption does not hold for our problem formulation.

Conditional Gradient Methods for Constrained Optimization. CGM was first proposed in the seminal work of Frank & Wolfe (1956) and its academic interest has witnessed a resurgence in the past decade. The advantage of CGMs lies in the low per-iteration cost of the lmo, alongside their ability to produce sparse solutions. In comparison to projection-based approaches, the lmo is cheaper to compute for several important domains, amongst which the spectrahedron, polytopes emerging from combinatorial optimization, and ℓ_p norm-induced balls (Garber, 2016). Consequently, CG-type methods have been studied under varying assumptions in (Hazan, 2008; Clarkson, 2010; Hazan & Kale, 2012; Jaggi, 2013; Lan, 2013; Balasubramanian & Ghadimi, 2018), and have been incorporated as cheaper subsolvers into algorithms which originally relied on projection oracles (Lan & Zhou, 2016; Liu et al., 2019).

CGMs have been further extended to the setting of convex composite minimization via the Augmented Lagrangian framework in (Gidel et al., 2018; Silveti-Falls et al., 2019; Yurtsever et al., 2019a). Most relevant to our work, CGM-based quadratic penalty methods have been studied for convex problems with constraints of the form $Ax - b \in \mathcal{K}$, where $\mathcal{K}$ is a closed, convex set (Yurtsever et al., 2018; Locatello et al., 2019). We compare our methods against the latter two in Section 4.5.

Variance Reduction. Stochastic variance reduction (VR) methods have gained popularity in recent years following their initial study by (Roux et al., 2012; Johnson & Zhang, 2013; Mahdavi et al., 2013). The VR technique relies on averaging schemes to reduce the variance inherent to stochastic gradients, with several different flavors having emerged in the past decade: SAG (Schmidt et al., 2017), SVRG (Johnson & Zhang, 2013), SAGA (Defazio et al., 2014), SVRRG++ (Allen-Zhu & Yuan, 2016), SARAH (Nguyen et al., 2017) and SPIDER (Fang et al., 2018). Such methods outperform the classical SGD under the finite sum model, a fact which led to their widespread use in large-scale applications and their further inclusion into other stochastic optimization algorithms (see for example (Xiao & Zhang, 2014; Hazan & Luo, 2016)).

Relevant to our setting, VR has been studied in the context of CGMs for convex minimization by (Mokhtari et al., 2018; Hazan & Luo, 2016; Locatello et al., 2019; Yurtsever et al., 2019b; Zhang et al., 2019). The sfo complexity of these methods varies depending on the VR scheme, with the best guarantee being of order $\mathcal{O}(\epsilon^{-2})$ (Zhang et al., 2019; Yurtsever et al., 2019b). For a thorough comparison of the complexities, we refer the reader to Section 6 of (Yurtsever et al., 2019b).

3. Preliminaries

Notation. We use $\|\cdot\|$ to express the Euclidean norm and $\langle\cdot,\ \cdot\rangle$ to denote the corresponding inner product. The distance between a point x and a set $\mathcal{X}$ is defined as $\operatorname{dist}(x,\mathcal{X}) := \inf_{y\in\mathcal{X}}\|y-x\|$. The indicator function of a set $\mathcal{X}$ is given by $\delta_{\mathcal{X}}(x) = 0,$ if $x\in\mathcal{X}$, and $\delta_{\mathcal{X}}(x) = +\infty$ otherwise. We denote by $\mathcal{D}_{\mathcal{X}} := \max_{(x,y)\in\mathcal{X}\times\mathcal{X}}\|x-y\|$ the diameter of a compact set $\mathcal{X}$.

For the probabilistic setting, we denote by ξ an element of our sample space and by $P(\xi)$ its probability measure. Unless stated otherwise, expectations will be taken with respect to ξ. We use $[n]$ to denote $\{1, 2, \dots n\}$.

Given a function $f : \mathbb{R}^d \to \mathbb{R}$ and $L > 0$, we say that f is L-smooth if ∇f is Lipschitz continuous, which is defined as $\|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|, \forall x, y \in \mathbb{R}^d$.

Following the same setup as in (Fercoq et al., 2019), the space of random variables used in this work is

$$\mathcal{H} = \left\{ y(\xi)_\xi \in \mathbb{R}^m \mid \ \xi \in \mathbb{R}^n,\ \mathbb{E}\left[\|y(\xi)_\xi\|^2\right] < +\infty \right\},$$

where the associated scalar product is given by $\langle x,\ z\rangle := \mathbb{E}\left[x(\xi)^T z(\xi)\right] = \int x(\xi)^T z(\xi) dP(\xi)$.

Smoothing. Nesterov (2005) proposes a technique for obtaining smooth approximations parametrized by β, of a non-smooth and convex function g. The resulting smoothed approximations take the following form:

$$g_\beta(x) = \max_y \langle y,\ x\rangle - g^*(y) - \frac{\beta}{2}\|y\|^2,$$

where $g^*(y) = \sup_z \langle z,\ y\rangle - g(z)$ is the Fenchel conjugate of g. Note that g_β is convex and $\frac{1}{\beta}$-smooth. The present work focuses on the case when $g(\cdot,\xi) = \delta_{b(\xi)}(\cdot)$. Smoothing the indicator function is studied in the context of proximal methods by Tran-Dinh et al. (2018); Fercoq et al. (2019) and for deterministic CGM by Yurtsever et al. (2018). Of particular note is that when $g(x) = \delta_{\mathcal{X}}(x)$, the smoothed function becomes $g_\beta(x) = \frac{1}{2\beta}\operatorname{dist}(x,\mathcal{X})^2$.

Optimality Conditions. We denote by x^* a solution to problem (1) and say that x is an ϵ-solution for (1) if it satisfies

$$\mathbb{E}\left[|f(x,\xi) - f(x^*)|\right] \le \epsilon, \quad \sqrt{\mathbb{E}\left[\operatorname{dist}(A(\xi)x, b(\xi))^2\right]} \le \epsilon. \tag{2}$$

Oracles. Our complexity results are given relative to the following oracles:

- **Stochastic first order oracle (sfo):** For a stochastic function $\mathbb{E}\left[f(\cdot,\xi)\right]$ with $\xi \sim P$, the sfo returns a pair $(f(x,\xi), \nabla f(x,\xi))$ where ξ is an i.i.d. sample from P (Nemirovsky & Yudin, 1983).
- **Incremental first order oracle (ifo):** For finite-sum problems, the ifo takes an index $i \in [n]$ and returns a pair $(f_i(x), \nabla f_i(x))$.
- **Linear minimization oracle (lmo):** The linear minimization oracle of set $\mathcal{X}$ is given by $\operatorname{lmo}_{\mathcal{X}}(y) = \arg\min_{x\in\mathcal{X}}\langle x,\ y\rangle$ and is assumed to be efficient to compute throughout this paper. This is the main projection-free oracle model for CGM-type methods.

4. Algorithms & Convergence

We now describe our proposed methods for solving (1), H-1SFW and H-SPIDER-FW, and provide their theoretical convergence guarantees.

4.1. Challenges and High-Level Ideas

Problem (1) can be rewritten equivalently as:

$$\min_{x\in\mathcal{X}} F(x) := \mathbb{E}\left[f(x,\xi) + \delta_{b(\xi)}(A(\xi)x)\right]. \tag{3}$$

Note that, in this form, our objective is non-smooth due to the indicator function. In order to leverage the conditional gradient framework, we *smooth* $\delta_{b(\xi)}(A(\xi)x)$ through the technique described in Section 3, thus obtaining a surrogate objective F_β. For notational simplicity, we refer to the smoothed stochastic indicator as:

$$g_\beta(A(\xi)x) = \frac{1}{2\beta}\operatorname{dist}(A(\xi)x, b(\xi))^2. \tag{4}$$

The minimization problem in terms of the smoothed objective thus becomes:

$$\min_{x\in\mathcal{X}} F_\beta(x) := \mathbb{E}\left[f(x,\xi) + g_\beta(A(\xi)x)\right], \tag{5}$$

with $\lim_{\beta\to 0} F_\beta(x) = F(x)$. A natural idea is to optimize smooth approximations F_β which are progressively more accurate representations of F. To this end, we apply *conditional gradient* steps in conjunction with decreasing the smoothness parameter β, practically emulating a homotopy transformation. As the iterations unfold our algorithms in fact approach the optimum of the original objective $F(x)$, as stated theoretically in Sections 4.3.2 and 4.4.2.

However, the aforementioned idea faces a technical challenge: decreasing the smoothing parameter β impacts the variance of the stochastic gradients $\nabla_x g_\beta(A(\xi)x)$, which increases proportionally. This issue has previously been signaled in the work of (Fercoq et al., 2019), where the authors address a similar setting using stochastic proximal gradient steps. Here, the problem is further aggravated by the use of lmo calls over $\mathcal{X}$, as it is well-known that CGMs are sensitive to non-vanishing gradient noise (Mokhtari et al., 2018).

Our solution is to simply perform VR on the stochastic gradients and theoretically establish a rate for $\beta \to 0$ in order to counteract the exploding variance. Precisely, we show how two different VR schemes can be successfully used within the homotopy framework:

- H-1SFW uses one stochastic sample to update a gradient estimator at every iteration, following the technique introduced in (Mokhtari et al., 2018). Depending on computational resources, the single-sample model can be extended to a fixed batch size with the same convergence guarantees.
- H-SPIDER-FW uses stochastic mini-batches of increasing size to compute the gradient estimator, using the technique proposed in (Fang et al., 2018).

The theoretical results characterizing our algorithms are presented in sections refsec:h1sfw and 4.4. First, we state the rate at which the β-dependent stochastic gradient noise vanishes under each VR scheme in lemmas 4.1 and 4.2. The main convergence theorems 4.1 and 4.2 then describe the performance of our algorithms in terms of the quantity $\mathbb{E}[S_{\beta_k}(x_k,\xi)] := \mathbb{E}[F_{\beta_k}(x_k,\xi) - f(x^*)]$, called the *smoothed gap*. Finally, in corollaries 4.1 and 4.2 we translate the aforementioned results into guarantees over the objective residual and constraint feasibility. All proofs are deferred to the appendix due to lack of space.

4.2. Technical Assumptions

Assumption 4.1. *The stochastic functions $f(\cdot,\xi)$ are convex and L_f-smooth. This further implies that $f(x)$ is L_f-smooth.*

Assumption 4.2. *The stochastic gradients $\nabla f(x,\xi)$ are unbiased and have a uniform variance bound σ_f^2. Formally,*

$$\begin{aligned}&\mathbb{E}[\nabla f(x,\xi)] = \nabla f(x)\\ &\mathbb{E}\left[\|\nabla f(x,\xi) - \nabla f(x)\|^2\right] \le \sigma_f^2 < +\infty. \end{aligned} \tag{6}$$

Algorithm 1 H-1SFW

```
Input: $x_1 \in \mathcal{X}, \beta_0 > 0, P(\xi)$
for $k = 1, 2, \ldots,$ do
    Set $\rho_k$, $\beta_k$ and $\gamma_k$; sample $\xi_k \sim P(\xi)$
    $d_k = (1-\rho_k)d_{k-1} + \rho_k \nabla_x F_{\beta_k}(x_k, \xi_k)$
    $w_k = \text{lmo}_{\mathcal{X}}(d_k)$
    $x_{k+1} = x_k + \gamma_k(w_k - x_k).$
end for
```

Assumption 4.3. *The domain $\mathcal{X}$ is convex and compact, with diameter $\mathcal{D}_{\mathcal{X}}$.*

Assumption 4.4. *Slater's condition holds for problem (3). Specifically, letting $G : \mathcal{H} \to \mathbb{R} \cup \{\infty\}$, $G(Ax) := \mathbb{E}\left[\delta_{b(\xi)}(A(\xi)x)\right]$, with the linear operator $A : \mathbb{R}^d \to \mathcal{H}$ defined as $(Ax)(\xi) := A(\xi)x$, $\forall x$, we require that*

$$0 \in \operatorname{sri}(\operatorname{dom}(G) - A\operatorname{dom}(f)),$$

where sri *is the strong relative interior of the set (Bauschke et al.).*

Assumption 4.5. *The spectral norm of the stochastic linear operator $A(\xi)$ is uniformly bounded by a constant L_A:*

$$L_A := \sup_\xi \|A(\xi)\|^2 < +\infty.$$

This assumption is also made in (Fercoq et al., 2019).

4.3. H(omotopy)-1SFW

We now describe our first algorithm which relies on the VR scheme proposed in (Mokhtari et al., 2018), and whose advantage lies in a simple update rule and single-loop structure.

4.3.1. Gradient Estimator Model

We denote the gradient estimator by d_k, and remark that it is biased with respect to the true gradient $\nabla F_\beta(x_k)$ and exhibits a vanishing variance. This scheme achieves VR while conveniently considering only one stochastic constraint at a time. The estimator update rule is given by

$$d_k = (1-\rho_k)d_{k-1} + \rho_k \nabla F_{\beta_k}(x_k, \xi_k),$$

where $\nabla F_{\beta_k}(x_k,\xi_k) = \nabla f(x_k,\xi_k) + \nabla g_{\beta_k}(A(\xi_k)X_k)$, and ρ_k is a decaying convex combination parameter. The proposed method is provided via pseudocode in Algorithm 1.

4.3.2. Convergence Results

Before stating the results, we remark that Lemma 4.1 is the counterpart of Lemma 1 in (Mokhtari et al., 2018) and its proof follows a similar route, up to bounding β-dependent quantities. It is worth noting that in our case, handling the stochastic linear inclusion constraints results in a rate surcharge factor of $\mathcal{O}\left(k^{1/3}\right)$.

Lemma 4.1. *Let $\rho_k = \frac{3}{(k+5)^{2/3}}$, $\gamma_k = \frac{2}{k+1}$, $\beta_k = \frac{\beta_0}{(k+1)^{1/6}}$, $\beta_0 > 0$ in Algorithm 1. Then, for all k,*

$$\mathbb{E}\left[\|\nabla F_{\beta_k}(x_k) - d_k\|^2\right] \leq \frac{C_1}{(k+5)^{1/3}},$$

where $C_1 = \max\left\{6^{1/3}\|\nabla F_{\beta_0}(x_0) - d_0\|^2,\right.$
$$\left. 2\left[18\sigma_f^2 + 112L_f^2\mathcal{D}_\mathcal{X}^2 + \frac{522L_A^2\mathcal{D}_\mathcal{X}^2}{\beta_0^2}\right]\right\}$$

Theorem 4.1. *Consider Algorithm 1 with parameters $\rho_k = \frac{3}{(k+5)^{2/3}}$, $\gamma_k = \frac{2}{k+1}$, $\beta_k = \frac{\beta_0}{(k+1)^{1/6}}$, $\beta_0 > 0$ (identical to Lemma 4.1). Then, for all k,*

$$\mathbb{E}\left[S_{\beta_k}(x_{k+1})\right] \leq \frac{C_2}{k^{1/6}},$$

where $C_2 = \max\left\{S_0(x_1),\ b = 2\mathcal{D}_\mathcal{X}\sqrt{C_1} + 2\mathcal{D}_\mathcal{X}^2\left(L_f + \frac{L_A}{\beta_0}\right)\right\}$ *and C_1 is defined in Lemma 4.1.*

Corollary 4.1. *The expected convergence in terms of objective suboptimality and feasibility of Algorithm 1 is, respectively,*

$$\begin{aligned}
\mathbb{E}\left[|f(x_k,\xi) - f(x^*)|\right] \quad &\in \mathcal{O}\left(k^{-1/6}\right)\\
\sqrt{\mathbb{E}\left[\text{dist}(A(\xi)x_k, b(\xi))^2\right]} \quad &\in \mathcal{O}\left(k^{-1/6}\right).
\end{aligned}$$

Consequently, the oracle complexity is $\#(sfo) \in \mathcal{O}\left(\epsilon^{-6}\right)$ and $\#(lmo) \in \mathcal{O}\left(\epsilon^{-6}\right)$.

4.4. H(omotopy)-SPIDER-FW

Our second algorithm presents a more complex VR scheme, which improves on the complexity of H-1SFW. The method relies on the SPIDER estimator originally proposed under the framework of Normalized Gradient Descent in (Fang et al., 2018) and further studied for CGMs in (Yurtsever et al., 2019b). Different from Section 4.3.2, the results that follow distinguish two scenarios: the first is customary to VR methods such as SVRG (Johnson & Zhang, 2013) or SARAH (Nguyen et al., 2017) and assumes a finite-sum form of f; the second, different from most other VR schemes, caters to objectives of the form $f(x) = \mathbb{E}\left[f(x,\xi)\right]$ where $\xi \sim P(\xi)$, and can handle a potentially infinite number of stochastic functions of (1).

Algorithm 2 H-SPIDER-FW

```
**Input:** $\bar{x}_1 \in \mathcal{X}, \beta_0 > 0, P(\xi)$
**for** $t = 1, 2, \ldots, T$ **do**
  $x_{t,1} = \bar{x}_t$
  Compute $\gamma_{t,1}, \beta_{t,1}, K_t$; sample $\xi_{\mathcal{Q}_t} \overset{\text{i.i.d}}{\sim} P(\xi)$
  $v_{t,1} = \tilde{\nabla} F_{\beta_{t,1}}(x_{t,1}, \xi_{\mathcal{Q}_t})$
  $w_{t,1} \in \text{lmo}_\mathcal{X}(v_{t,1})$
  $x_{t,2} = x_{t,1} + \gamma_{t,1}(w_{t,1} - x_{t,1})$
  **for** $k = 2, \ldots, K_t$ **do**
    Compute $\gamma_{t,k}, \beta_{t,k}$; sample $\xi_{\mathcal{S}_{t,k}} \overset{\text{i.i.d}}{\sim} P(\xi)$
    $v_{t,k} = v_{t,k-1} - \tilde{\nabla} F_{\beta_{t,k-1}}(x_{t,k-1}, \xi_{\mathcal{S}_{t,k}})$
                 $+ \tilde{\nabla} F_{\beta_{t,k}}(x_{t,k}, \xi_{\mathcal{S}_{t,k}})$
    $w_{t,k} \in \text{lmo}_\mathcal{X}(v_{t,k})$
    $x_{t,k+1} = x_{t,k} + \gamma_{t,k}(w_{t,k} - x_{t,k})$
  **end for**
  Set $\bar{x}_{t+1} = x_{t,K_t+1}$
**end for**
```

4.4.1. Gradient Estimator Model

We denote the SPIDER gradient estimator by $v_{t,k}$, and remark that it is also biased relative to $\nabla F_{\beta_k}(x_k)$ and exhibits a vanishing variance. This scheme achieves VR through the use of increasing-size mini-batches. The estimator update rule is given by

$$\begin{aligned}
v_{t,k} = v_{t,k-1} &- \tilde{\nabla} F_{\beta_{t,k-1}}(x_{t,k-1}, \xi_{\mathcal{S}_{t,k}})\\
&+ \tilde{\nabla} F_{\beta_{t,k}}(x_{t,k}, \xi_{\mathcal{S}_{t,k}}),
\end{aligned} \tag{7}$$

where $\tilde{\nabla} F_{\beta_{t,k}}(x_{t,k}, \xi_{\mathcal{S}_{t,k}}) = \tilde{\nabla} f(x_k, \xi_{\mathcal{S}_{t,k}}) + \tilde{\nabla} g_{\beta_{t,k}}(A(\xi_{\mathcal{S}_{t,k}})x_{t,k})$ defines the averaged gradient over a mini-batch of size $|\mathcal{S}_{t,k}|$.

The double indexing used in (7) hints at the double-loop structure of the algorithm, a format similar to most VR-based methods. The method is structured similarly to SPIDER-FW from (Yurtsever et al., 2019b), and proceeds in two steps: the outer loop computes an accurate gradient estimator and sets the batch size for the inner iterations. The inner-loop then iteratively 'refreshes' this gradient according to (7) and performs homotopy steps on β using a theoretically-determined schedule. The proposed method is provided via pseudocode in Algorithm 2.

4.4.2. Convergence Results

Again, we remark that Lemma 4.2 is the counterpart of Lemma 4, Appendix C in (Yurtsever et al., 2019b). However in this case, our proof takes a different, more tedious route, as the latter result does not accommodate homotopy steps.

In comparison, the bound we obtain depends linearly on the total iteration count, whereas the lemma of (Yurtsever et al., 2019b) depends only on the outer loop counter K_t.

Lemma 4.2 (Estimator variance for finite-sum problems)**.** *Consider Algorithm 2, and let ξ be finitely sampled from set $[n]$, $\xi_{\mathcal{Q}_t} = [n]$ and $\xi_{\mathcal{S}_{t,k}}$, such that $|\mathcal{S}_{t,k}| = K_t = 2^{t-1}$. Also, let $\gamma_{t,k} = \frac{2}{K_t+k}$, $\beta_{t,k} = \frac{\beta_0}{\sqrt{K_t+k}}$, $\beta_0 > 0$. Then, for a fixed t and for all $k \le K_t$,*

$$\mathbb{E}\left[\|\nabla F_{\beta_{t,k}}(x_{t,k}) - v_{t,k}\|^2\right] \le \frac{C_1}{K_t + k},$$

where $C_1 = 2\mathcal{D}_{\mathcal{X}}^2\left(8L_f^2 + \frac{98L_A^2}{\beta_0^2}\right)$.

Lemma 4.3 (Estimator variance for general expectation problems)**.** *Consider Algorithm 2 and let $\xi \sim P(\xi)$ and $\xi_{\mathcal{Q}_t}$ such that $|\mathcal{Q}_t| = \lceil\frac{2K_t}{\beta_{t,1}^2}\rceil$. Also, let $\xi_{\mathcal{S}_{t,k}}$, such that $|\mathcal{S}_{t,k}| = K_t = 2^{t-1}$, $\gamma_{t,k} = \frac{2}{K_t+k}$, $\beta_{t,k} = \frac{\beta_0}{\sqrt{K_t+k}}$, $\beta_0 > 0$. Then, for a fixed t and for all $k \le K_t$,*

$$\mathbb{E}\left[\|\nabla F_{\beta_{t,k}}(x_{t,k}) - v_{t,k}\|^2\right] \le \frac{C_2}{K_t + k},$$

where $C_2 = 16L_f^2\mathcal{D}_{\mathcal{X}}^2 + 2L_A^2\mathcal{D}_{\mathcal{X}}^2\left(\frac{98}{\beta_0^2}+1\right) + 2\beta_0^2\sigma_f^2$.

Theorem 4.2. *Consider Algorithm 2 with parameters $\gamma_{t,k} = \frac{2}{K_t+k}$, $\beta_{t,k} = \frac{\beta 0}{\sqrt{K_t+k}}$, $\beta_0 > 0$, and $\xi_{\mathcal{S}_{t,k}}$, such that $|\mathcal{S}_{t,k}| = K_t = 2^{t-1}$. Then,*

- *For ξ be finitely sampled from set $[n]$, $\xi_{\mathcal{Q}_t} = [n]$ and $\forall t \in \mathbb{N}$, $1 \le k \le 2^{t-1}$,*

$$\mathbb{E}\left[S_{\beta_{t,k}}(x_{t,k+1})\right] \le \frac{C_3}{\sqrt{K_t + k + 1}},$$

$$\text{where } C_3 = \max\Bigg\{S_{\beta_{1,0}}(x_{1,1}),\; 2\mathcal{D}_{\mathcal{X}}^2 L_f + 2\mathcal{D}_{\mathcal{X}}^2\sqrt{16L_f^2 + \frac{196L_A^2}{\beta_0^2}} + \frac{2\mathcal{D}_{\mathcal{X}}^2 L_A}{\beta_0}\Bigg\};$$

- *For $\xi \sim P(\xi)$, $\xi_{\mathcal{Q}_t}$ such that $|\mathcal{Q}_t| = \lceil\frac{2K_t}{\beta_{t,1}^2}\rceil$ and $\forall t \in \mathbb{N}$, $1 \le k \le 2^{t-1}$,*

$$\mathbb{E}\left[S_{\beta_{t,k}}(x_{t,k+1})\right] \le \frac{C_4}{\sqrt{K_t + k + 1}},$$

$$\text{where } C_4 = \max\Bigg\{S_{\beta_{1,0}}(x_{1,1}),\; 2\mathcal{D}_{\mathcal{X}}^2 L_f + \frac{2\mathcal{D}_{\mathcal{X}}^2 L_A}{\beta_0} + 2\mathcal{D}_{\mathcal{X}}\sqrt{16L_f^2\mathcal{D}_{\mathcal{X}}^2 + 2L_A^2\mathcal{D}_{\mathcal{X}}^2\left(\frac{98}{\beta_0^2}+1\right) + 2\beta_0^2\sigma_f^2}\Bigg\}.$$

Corollary 4.2. *The expected convergence in terms of objective suboptimality and feasibility of Algorithm 2 is, respectively,*

$$\mathbb{E}\left[|f(x_{t,k}) - f(x^*)|\right] \in \mathcal{O}\left((K_t+k)^{-1/2}\right)$$
$$\sqrt{\mathbb{E}\left[\text{dist}(A(\xi)x_{t,k}, b(\xi))^2\right]} \in \mathcal{O}\left((K_t+k)^{-1/2}\right)$$

for both the finite-sum and the general expectation setting, up to constants. Consequently, the oracle complexities are given by $\#(ifo) \in \mathcal{O}\left(n\log_2(\epsilon^-2) + \epsilon^{-4}\right)$ and $\#(lmo) \in \mathcal{O}\left(\epsilon^{-2}\right)$ for the finite-sum setting, and by $\#(sfo) \in \mathcal{O}\left(\epsilon^{-4}\right)$ and $\#(lmo) \in \mathcal{O}\left(\epsilon^{-2}\right)$ for the more general expectation setting.

4.5. Discussion

Rate Degradation in the Absence of Projection Oracles. Compared to proximal methods for solving (1), our algorithms require $\mathcal{O}(\epsilon^{-2})$ times more sfo calls to reach an ϵ-solution. This is well-known for CG-based methods: for instance, solving a fully deterministic version of (1) using the Augmented Lagrangian framework has a gradient complexity of $\mathcal{O}(\epsilon^{-1})$ (Xu, 2017), whereas the best known complexity for CG-based algorithms is $\mathcal{O}(\epsilon^{-2})$ (Yurtsever et al., 2018).

Comparison with SHCGM (Locatello et al., 2019). The state-of-the-art for solving (1) is the half-stochastic method SHCGM (Locatello et al., 2019), in which stochasticity is restricted to the objective function f, while the constraints are processed deterministically. This algorithm attains an $\mathcal{O}(\epsilon^{-3})$ sfo complexity and an $\mathcal{O}(\epsilon^{-3})$ lmo complexity, by resorting to the same VR scheme as H-1SFW applied only to $f(x, \xi)$. Since SHCGM handles the constraints deterministically, it does not face the challenge of exploding variance as $\beta \to 0$.

Our analysis shows that handling the β-dependence of the gradient noise comes at the price of H-1SFW being $\mathcal{O}(\epsilon^{-3})$ times more expensive in terms of both oracles. In contrast, owing to a more powerful variance-reduction scheme, H-SPIDER-FW attains only an $\mathcal{O}(\epsilon)$-times worse sfo complexity, while improving by an $\mathcal{O}(\epsilon)$ factor in terms of the lmo complexity. Given that an lmo call is generally more expensive than that of an sfo, we have in fact *improved* the complexity over the state-of-the-art, while being the first to process linear constraints stochastically. Moreover, we note that the lmo complexity of H-SPIDER-FW is on the same order as its fully deterministic counterpart, the HCGM (Yurtsever et al., 2018).

The Role of VR. The choice of VR technique dictates the worst-case convergence guarantees of our methods, a fact which is apparent from the discrepancy between the variance bounds of Lemmas 4.1 and 4.2- 4.3, respectively: $\mathcal{O}(k^{-1/3})$ for d_k vs. $\mathcal{O}(k^{-1})$ for $v_{t,k}$. This signals the existence of a trade-off: a more intricate way of handling stochastic penalty-type constraints can ensure the better convergence guarantees of H-SPIDER-FW, while a simpler VR scheme comes at the cost of the rather pessimistic ones of H-1SFW. Fortunately, as shown in the Section 5, the simple H-1SFW greatly outperforms its worst-case guarantees.

5. Numerical Experiments

For demonstrating the empirical efficiency of our algorithms, we apply them to three problem instances: synthetically-generated SDPs, the K-means clustering SDP relaxation and the Sparsest Cut-associated SDP.

Evaluation Metrics: Our experiments subscribe to a finite-sum template, where we define $f(x) := \sum_{i=1}^{n_1} f_i(x)$ and $g_\beta(Ax) = \sum_{i=1}^{n_2} g_{i,\beta}(A_i^T x)$. The objective convergence is recorded as $|f(x) - f^\star|$, with $f^\star := f(x^*)$. Due to imperfect feasibility, the value of $f(x)$ can overshoot $f^\star$, since the constrained optimum is not the global one. This usually appears as the increase of $|f(x) - f^\star|$ immediately after a significant drop when the quantity $f(x) - f^\star$ becomes negative; then the decreasing trend restarts, as the objective and constraints re-balance. Such a phenomenon is common for homotopy-based methods, see for instance (Yurtsever et al., 2018). Lastly, the feasibility is recorded as $\|Ax - b\|$.

Baseline: To the best of our knowledge, the HCGM (Yurtsever et al., 2018) and the SHCGM (Locatello et al., 2019) are the only algorithms which tackle SDPs under the conditional gradient framework. The latter represents the empirical state-of-the-art and we choose it as the baseline for our experiments.

5.1. Synthetic SDP Problems

This proof-of concept experiment aims to show the performance of our fully stochastic methods, given a fixed problem dimension and an increasing set of constraints. We consider the synthetic SDP:

$$\begin{aligned} \min_{\substack{X \in \mathbb{S}_+^d \\ \mathrm{tr}(X) \le \frac{1}{d}}} \quad & \langle C, X \rangle \\ \textbf{subject to} \quad & \mathrm{tr}(A_i X) = b_i, i = 1 \dots n \end{aligned}$$

where the entries of A_i and C are generated from $\mathcal{U}(0,1)$, and $b_i = \langle A_i, X^* \rangle$ for a fixed X^*. We perform uniform sampling on the pairs (A_i, b_i) for computing their stochastic gradients in our algorithms. We fix the dimension to be $d = 20$ and vary the size of constraints with $n =$ `5e2` and `5e3`.

For a fair comparison, we sweep the parameter β_0 for the three algorithms in the range [`1e-7`, `1e1`]. We settle for `1e-7`, `1e-7` and `1e-5` for SHCGM, H-1SFW and H-SPIDER-FW, respectively. For H-1SFW and SHCGM, we choose the batch size to be 1% of the data.

Figure 1 illustrates the outcome of the experiments, where we observe a clear improvement of the stochastic algorithms over the baseline with a stable margin throughout the test cases.

Interestingly, H-1SFW exhibits strong empirical performance on the synthetic data, much better than its theoretical worst-case bound. A possible explanation is that the entries of C and A_i are generated from a "benign" distribution and *concentrate* around its mean (Ledoux, 2001). In such scenarios, even a small subset of constraints allows for effective variance reduction. For comparison, we provide an additional set of results for synthetic SDPs generated from a less well-behaved distribution in Appendix A.2. Nevertheless, we observe the same good performance of H-1SFW even with real data, in the next sections.

Regarding H-SPIDER-FW, we observe that the suboptimality and feasibility decrease at the rate $k^{-\frac{1}{2}}$ and $k^{-\frac{3}{4}}$, respectively, which is better than the worst-case bounds in Theorem 4.2.

5.2. The K-means Clustering Relaxation

We consider the unsupervised learning task of partitioning d data points into k clusters. We adopt the SDP formulation in (Peng & Wei, 2007), which amounts to solving:

$$\begin{aligned} \min_{X \in \mathcal{X}} \quad & \langle C, X \rangle \\ \textbf{subject to} \quad & X\vec{1} = \vec{1}, \\ & X_{i,j} \ge 0, \quad 1 \le i, j \le d. \end{aligned} \tag{8}$$

Here, $C \in \mathbb{R}^{d \times d}$ is the Euclidean distance matrix of the d data points, $\mathcal{X} = \{X \in \mathbb{R}^{d \times d} : X \succeq 0, \ \mathrm{tr}(X) \le k\}$, $\vec{1}$ is the all 1's vector. Notice that the number of linear constraints in (8) is $\mathcal{O}(d^2)$.

In order to compare against existing work, we adopt the MNIST dataset ($k = 10$) (LeCun & Cortes, 2010) with $d = 10^3$ samples and perform data preprocessing as in (Mixon et al., 2016). The very same setup appeared in several works (Mixon et al., 2016; Yurtsever et al., 2018; Locatello et al., 2019), with SHCGM (Locatello et al., 2019) showing the best practical performance.

We perform parameter sweeping on $\beta_0 \in$ [`1e-7`, `1e2`] for H-1SFW and H-SPIDER-FW, and settle for `5e-2` and `6e0`, respectively. For SHCGM, we adopt the same hyperparameter as in (Locatello et al., 2019). The batchsize for H-1SFW and SHCGM is set to 5%.

The comparison of our algorithms against SHCGM is reported in Figure 2. H-1SFW and H-SPIDER-FW converge

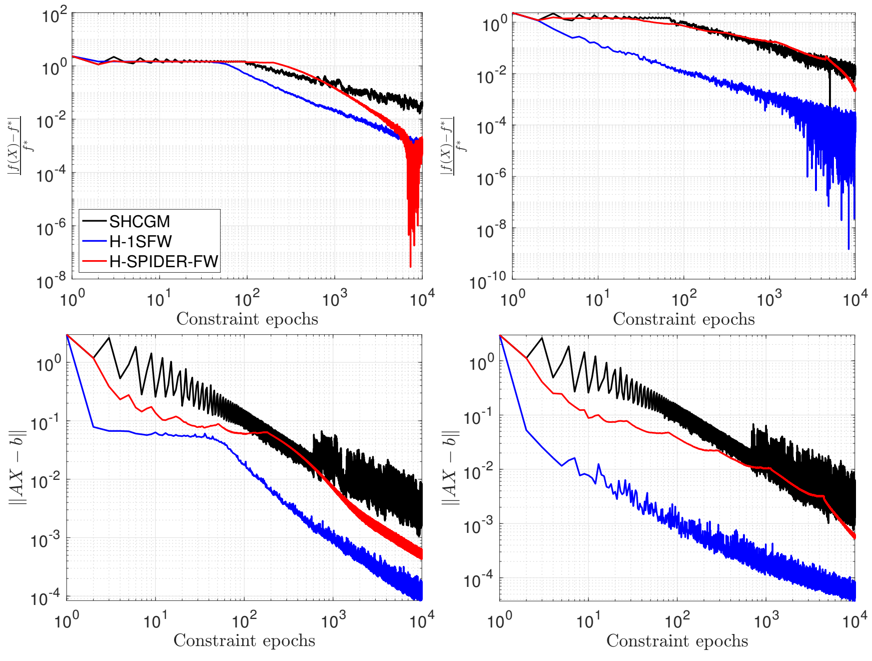

Figure 1. Synthetic SDPs, with each column showing the convergence in objective suboptimality (top) and in feasibility (bottom) for a specific problem. The left hand-side column corresponds to a problem with `5e2` constraints, while the right hand-side one to a problem with `5e3` constraints.

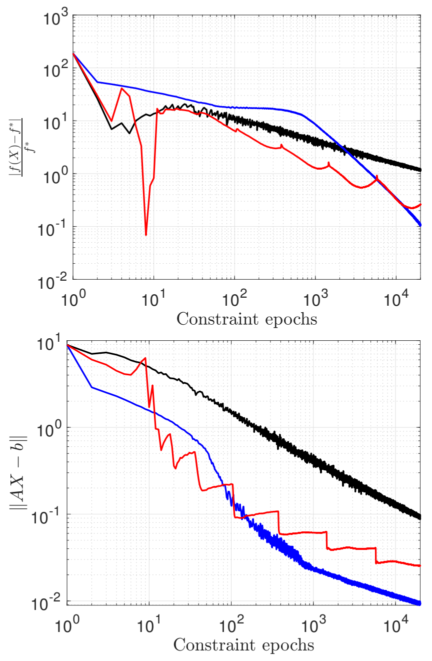

Figure 2. The K-means SDP relaxation, with convergence in objective suboptimality (top) and in feasibility (bottom).

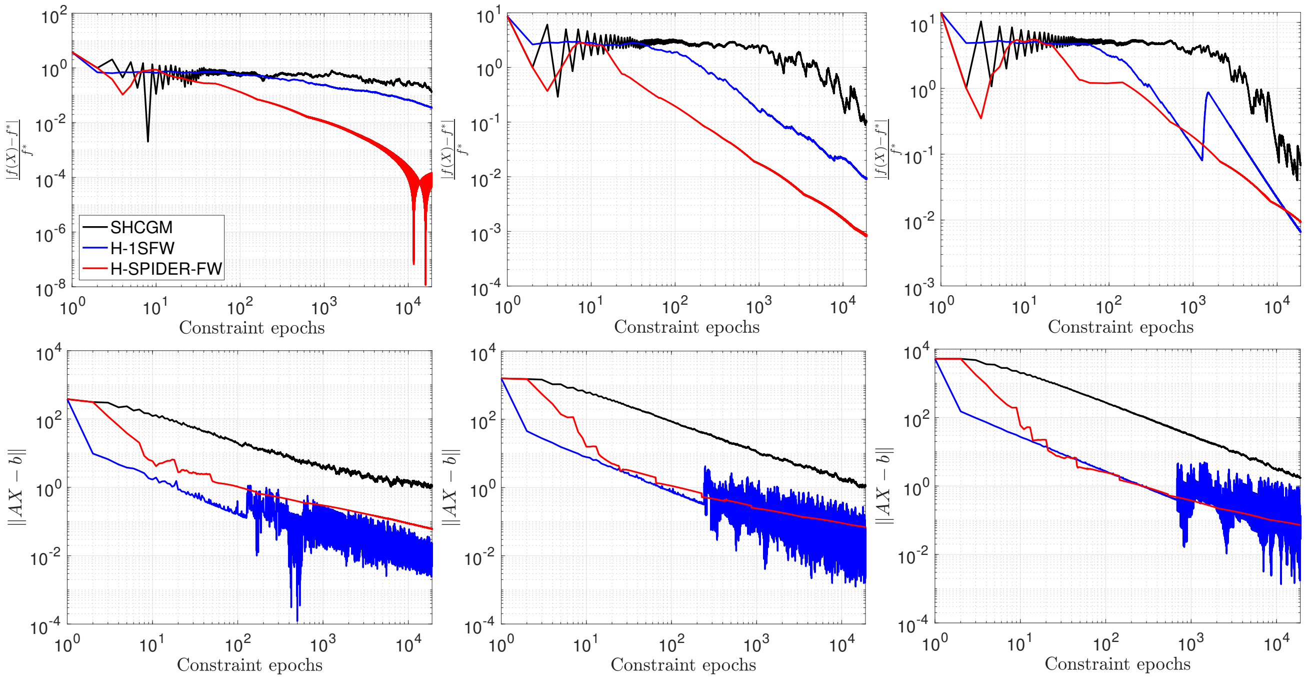

Figure 3. The Sparsest Cut-associated SDP relaxation, where each column shows the convergence in objective suboptimality (top) and feasibility (bottom) for a specific problem. From left to right, the results correspond to graphs *mammalia-primate-association-13*, *insecta-ant-colony1-day37* and *insecta-ant-colony4-day10*, sorted by increasing size.

Table 1. Details of the Network Repository (Rossi & Ahmed, 2015) graphs used in the experiments.

Graph name	$\|V\|$	$\|E\|$	Avg. node degree	Max. node degree	USC SDP dimension	USC SDP # constraints
mammalia-primate-association-13	25	181	14	19	$X \in \mathbb{R}^{25\times25}$	$\sim$ 6.90e3
insecta-ant-colony1-day37	55	1k	42	53	$X \in \mathbb{R}^{55\times55}$	$\sim$ 7.87e4
insecta-ant-colony4-day10	102	4k	79	99	$X \in \mathbb{R}^{102\times102}$	$\sim$ 5.15e5

at a comparable rate, with both clearly overtaking the baseline with regards to objective suboptimality and feasibility convergence.

5.3. Computing an ℓ_2^2 Embedding for the Uniform Sparsest Cut Problem

The Uniform Sparsest Cut problem (USC) aims to find a bipartition $(S, \bar{S})$ of the nodes of a graph $G = (V, E)$, $|V| = d$, which minimizes the quantity

$$\frac{E(S,\bar{S})}{|S||\bar{S}|},$$

where $E(S, \bar{S})$ is the number of edges connecting S and $\bar{S}$. This problem is of broad interest, with applications in areas such as VLSI layout design, topological design of communication networks and image segmentation, to name a few. Relevant to machine learning, it appears as a subproblem in hierarchical clustering algorithms (Dasgupta, 2016; Chatziafratis et al., 2018).

Computing such a bipartition is NP-hard and intense research has gone into designing efficient approximation algorithms for this problem. In the seminal work of Arora et al. (2009) an $\mathcal{O}\left(\sqrt{\log d}\right)$ approximation algorithm is proposed for solving USC, which relies on finding a *well-spread* ℓ_2^2 geometric representation of G where each node $i \in V$ is mapped to a vector v_i in $\mathbb{R}^d$. In this experimental section we focus on solving the SDP that computes this geometric embedding, as its high number of triangle inequality constraints ($\mathcal{O}\left(d^3\right)$) makes it a suitable candidate for our framework. The canonical formulation of the SDP is given below (for the original formulation, see Appendix A.3).

$$\begin{aligned} \min_{X\in\mathcal{X}} \quad & \langle L, X\rangle \\ \text{subject to} \quad & d\,\mathrm{Tr}(X) - \mathrm{Tr}(\mathbf{1}_{d\times d}X) = \frac{d^2}{2} \\ & X_{i,j} + X_{j,k} - X_{i,k} - X_{j,j} \le 0, \ \ \forall\, i,j,k \in V \end{aligned}$$

Here, L represents the Laplacian of G, $\mathcal{X} = \{X \in \mathbb{R}^{d\times d} : X \succeq 0,\ \mathrm{tr}(X) \le d\}$ and $X_{i,j} = \langle v_i,\ v_j\rangle$ gives the geometric embedding of the nodes. We run our algorithms on three graphs of different sizes from the Network Repository dataset (Rossi & Ahmed, 2015), whose details are summarized in Table 1. Note the cubic dependence of the number of constraints relative to the number of nodes. We perform parameter sweeping on $\beta_0 \in [1\mathrm{e}{-5}, 1\mathrm{e}5]$ using the smallest graph, *mammalia-primate-association-13*, and keep the same parameters for all the experiments. The values of β_0 for SHCGM, H-1SFW and H-SPIDER-FW are 1e2, 1e−2 and 1e1 respectively, and the batch size for both H-1SFW and SHCGM is set to 5%.

Figure 3 depicts the outcomes of the experiments, with both our algorithms consistently outperforming SHCGM and H-SPIDER-FW attaining the fastest convergence. A possible explanation is that, given the much larger number of constraints relative to the problem dimension ($\mathcal{O}\left(n^3\right)$ v.s $\mathcal{O}\left(n^2\right)$), H-SPIDER-FW's increasing mini-batches readily reach an adequate balance between feasibility enforcement and objective minimization.

Acknowledgements

The authors are grateful to Mehmet Fatih Sahin and Alp Yurtsever for the helpful discussions throughout the development of this paper.

This work was partially supported by the Swiss National Science Foundation (SNSF) under grant number 200021_178865 / 1; the Army Research Office under Grant Number W911NF-19-1-0404; the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no 725594 - time-data).

References

Allen-Zhu, Z. and Yuan, Y. Improved svrg for non-strongly-convex or sum-of-non-convex objectives. In *International conference on machine learning*, pp. 1080–1089, 2016.

Arora, S., Hazan, E., and Kale, S. Fast algorithms for approximate semidefinite programming using the multiplicative weights update method. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, pp. 339–348. IEEE, 2005.

Arora, S., Rao, S., and Vazirani, U. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM (JACM)*, 56(2):5, 2009.

Balasubramanian, K. and Ghadimi, S. Zeroth-order (non)-convex stochastic optimization via conditional gradient and gradient updates. In *Advances in Neural Information Processing Systems*, pp. 3455–3464, 2018.

Bauschke, H. H., Combettes, P. L., et al. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer.

Burer, S. and Monteiro, R. D. Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming*, 103(3):427–444, 2005.

Chatziafratis, V., Niazadeh, R., and Charikar, M. Hierarchical clustering with structural constraints. *arXiv preprint arXiv:1805.09476*, 2018.

Clarkson, K. L. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Transactions on Algorithms (TALG)*, 6(4):63, 2010.

Dasgupta, S. A cost function for similarity-based hierarchical clustering. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pp. 118–127, 2016.

Defazio, A., Bach, F., and Lacoste-Julien, S. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pp. 1646–1654, 2014.

Fang, C., Li, C. J., Lin, Z., and Zhang, T. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In *Advances in Neural Information Processing Systems*, pp. 689–699, 2018.

Fercoq, O., Alacaoglu, A., Necoara, I., and Cevher, V. Almost surely constrained convex optimization. *arXiv preprint arXiv:1902.00126*, 2019.

Frank, M. and Wolfe, P. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3: 95–110, 1956. doi: 10.1002/nav.3800030109. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800030109.

Garber, D. *Projection-free Algorithms for Convex Optimization and Online Learning*. PhD thesis, Technion-Israel Institute of Technology, Faculty of Industrial and , 2016.

Garber, D. and Hazan, E. Sublinear time algorithms for approximate semidefinite programming. *Mathematical Programming*, 158(1-2):329–361, 2016.

Gidel, G., Pedregosa, F., and Lacoste-Julien, S. Frank-wolfe splitting via augmented lagrangian method. In *International Conference on Artificial Intelligence and Statistics*, pp. 1456–1465, 2018.

Hazan, E. Sparse approximate solutions to semidefinite programs. In *Latin American symposium on theoretical informatics*, pp. 306–316. Springer, 2008.

Hazan, E. and Luo, H. Variance-reduced and projection-free stochastic optimization. In *International Conference on Machine Learning*, pp. 1263–1271, 2016.

Hazan, E. E. and Kale, S. Projection-free online learning. In *29th International Conference on Machine Learning, ICML 2012*, pp. 521–528, 2012.

Huang, Q., Chen, Y., and Guibas, L. Scalable semidefinite relaxation for maximum a posterior estimation. In *International Conference on Machine Learning*, pp. 64–72, 2014.

Jaggi, M. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In Dasgupta, S. and McAllester, D. (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 427–435, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL http://proceedings.mlr.press/v28/jaggi13.html.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pp. 315–323, 2013.

Lan, G. The complexity of large-scale convex programming under a linear optimization oracle. *arXiv preprint arXiv:1309.5550*, 2013.

Lan, G. and Zhou, Y. Conditional gradient sliding for convex optimization. *SIAM Journal on Optimization*, 26(2):1379–1409, 2016.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

Ledoux, M. *The concentration of measure phenomenon*. Number 89. American Mathematical Soc., 2001.

Liu, Y.-F., Liu, X., and Ma, S. On the nonergodic convergence rate of an inexact augmented lagrangian framework for composite convex programming. *Mathematics of Operations Research*, 44(2):632–650, 2019.

Locatello, F., Yurtsever, A., Fercoq, O., and Cevher, V. Stochastic conditional gradient method for composite convex minimization. *arXiv preprint arXiv:1901.10348*, 2019.

Mahdavi, M., Zhang, L., and Jin, R. Mixed optimization for smooth functions. In *Advances in neural information processing systems*, pp. 674–682, 2013.

Mixon, D. G., Villar, S., and Ward, R. Clustering subgaussian mixtures by semidefinite programming. *arXiv preprint arXiv:1602.06612*, 2016.

Mokhtari, A., Hassani, H., and Karbasi, A. Stochastic conditional gradient methods: From convex minimization to submodular maximization. *arXiv preprint arXiv:1804.09554*, 2018.

Nemirovsky, A. S. and Yudin, D. B. Problem complexity and method efficiency in optimization. 1983.

Nesterov, Y. Smooth minimization of non-smooth functions. *Mathematical programming*, 103(1):127–152, 2005.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takáč, M. Sarah: A novel method for machine learning problems using stochastic recursive gradient. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2613–2621. JMLR. org, 2017.

Patrascu, A. and Necoara, I. Nonasymptotic convergence of stochastic proximal point methods for constrained convex optimization. *Journal of Machine Learning Research*, 18: 198–1, 2017.

Peng, J. and Wei, Y. Approximating k-means-type clustering via semidefinite programming. *SIAM journal on optimization*, 18(1):186–205, 2007.

Rossi, R. A. and Ahmed, N. K. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015. URL http://networkrepository.com.

Roux, N. L., Schmidt, M., and Bach, F. R. A stochastic gradient method with an exponential convergence _rate for finite training sets. In *Advances in neural information processing systems*, pp. 2663–2671, 2012.

Schmidt, M., Le Roux, N., and Bach, F. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112, 2017.

Silveti-Falls, A., Molinari, C., and Fadili, J. Generalized conditional gradient with augmented lagrangian for composite minimization. *arXiv preprint arXiv:1901.01287*, 2019.

Tran-Dinh, Q., Fercoq, O., and Cevher, V. A smooth primal-dual optimization framework for nonsmooth composite convex minimization. *SIAM Journal on Optimization*, 28 (1):96–134, 2018.

Weinberger, K. Q. and Saul, L. K. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(2), 2009.

Xiao, L. and Zhang, T. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

Xu, Y. Accelerated first-order primal-dual proximal methods for linearly constrained composite convex programming. *SIAM Journal on Optimization*, 27(3):1459–1484, 2017.

Xu, Y. Primal-dual stochastic gradient method for convex programs with many functional constraints. *arXiv preprint arXiv:1802.02724*, 2018.

Yurtsever, A., Fercoq, O., Locatello, F., and Cevher, V. A conditional gradient framework for composite convex minimization with applications to semidefinite programming. In *35th International Conference on Machine Learning (ICML)*, pp. 5727–5736. PMLR, 2018.

Yurtsever, A., Fercoq, O., and Cevher, V. A conditional-gradient-based augmented lagrangian framework. In *International Conference on Machine Learning*, pp. 7272–7281, 2019a.

Yurtsever, A., Sra, S., and Cevher, V. Conditional gradient methods via stochastic path-integrated differential estimator. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7282–7291, Long Beach, California, USA, 09–15 Jun 2019b. PMLR. URL http://proceedings.mlr.press/v97/yurtsever19b.html.

Zhang, M., Shen, Z., Mokhtari, A., Hassani, H., and Karbasi, A. One sample stochastic frank-wolfe. *arXiv preprint arXiv:1910.04322*, 2019.