Self-Concordant Analysis of Frank-Wolfe Algorithms

Pavel Dvurechensky[1 2 3] Petr Ostroukhov[4] Kamil Safin[4] Shimrit Shtern[5] Mathias Staudigl[6]

Abstract

Projection-free optimization via different variants of the Frank-Wolfe (FW), a.k.a. Conditional Gradient method has become one of the cornerstones in optimization for machine learning since in many cases the linear minimization oracle is much cheaper to implement than projections and some sparsity needs to be preserved. In a number of applications, e.g. Poisson inverse problems or quantum state tomography, the loss is given by a self-concordant (SC) function having unbounded curvature, implying absence of theoretical guarantees for the existing FW methods. We use the theory of SC functions to provide a new adaptive step size for FW methods and prove global convergence rate $O(1/k)$ after k iterations. If the problem admits a stronger local linear minimization oracle, we construct a novel FW method with linear convergence rate for SC functions.

1. Introduction

Statistical analysis using (generalized) self-concordant (SC) functions as a loss function is gaining increasing attention in the machine learning community (Bach, 2010; Owen, 2013; Ostrovskii & Bach, 2018; Marteau-Ferey et al., 2019). This class of loss functions allows to obtain faster statistical rates akin to least-squares. At the same time, the minimization of empirical risk in this setting is a challenging optimization problem in high dimensions. The reason is that for large-scale problems usually a First-order method (FOM) is the method of choice. Yet, self-concordant functions are usually not strongly convex and do not have Lipschitz continuous gradients. Hence, unless we can verify a relative smoothness condition (Bauschke et al., 2017; Lu et al., 2018), the cornerstone assumptions for FOM do not apply when minimizing SC functions. Further, many machine learning problems involve some sparsity constraint usually admitting a Linear minimization oracle (LMO) (Jaggi, 2013), meaning that for a reasonable computational price one can minimize a linear function over the feasible set; Examples include ℓ_1 regularization, Spectrahedron constraints (Hazan, 2008), and ordered weighted ℓ_1 regularization (Zeng & Figueiredo, 2014). In such settings an attractive method of choice is the Frank-Wolfe (FW) method (Frank & Wolfe, 1956), also known as the Conditional Gradient (CG). The modern convergence analysis of FW, starting with (Jaggi, 2013), relies on bounded curvature of the objective function f. A sufficient condition for this is that the objective function has a Lipschitz continuous gradient over the domain of interest. In general, SC functions have unbounded curvature and are not strongly convex. In fact, to the best of our knowledge, there is no existing convergence guarantee for FW when minimizing a general SC function. Given the plethora of examples of SC loss functions, and the importance of FW in machine learning, we focus on the question of convergence and complexity of FW for minimizing a SC function f over a set $\mathcal{X}$ with LMO.

Our Contributions. In this paper we demonstrate that FW indeed works when minimizing a SC function over a compact convex domain. However, since the standard arguments for FW are not applicable, we have to develop some new ideas. Section 3 constructs new adaptive variants of the FW method, whose main innovation is the construction of new step-size policies ensuring global convergence and $\mathcal{O}(1/k)$ convergence rates. One of the keys to our results is to develop step-sizes which ensure monotonicity of the method, leading to Lipschitz smoothness and strong convexity of the objective on a level set. While the Lipschitz smoothness on a level set is used in the analysis, the algorithms do not require knowledge of the Lipschitz constant but rather rely only on basic properties of SC functions. Section 4 gives a *linear* convergence result for FW, under the assumption that we have access to the strong convexity parameter on the level set and a Local Linear minimization oracle (LLOO), first constructed in Garber & Hazan (2016).

Previous work. Despite its great scalability properties,

[*]Equal contribution [1]Weierstrass Institute for Applied Analysis and Stochastics, Berlin, Germany. [2]National Research University Higher School of Economics, Moscow, Russia [3]Institute for Information Transmission Problems RAS, Moscow, Russia [4]Moscow Institute of Physics and Technology, Dolgoprudny, Russia [5]Grand Fellow, Technion - Israel Institute of Technology, Haifa, Israel. [6]Department of Data Science and Knowledge Engineering, Maastricht University, The Netherlands,. Correspondence to: Mathias Staudigl <mathias.staudigl@gmail.com>.

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

FW is plagued by slow convergence rates. Frank & Wolfe (1956) showed an $\mathcal{O}(1/k)$ convergence rate of the function values to the optimal value for quadratic programming problems. Later on Levitin & Polyak (1966) generalized this result to general functions and arbitrary compact convex sets, and Canon & Cullum (1968) proved that this rate is actually tight (see also Lan (2013)). However, a standard result in convex optimization states that projected gradient descent converges linearly to the minimizer of a strongly convex and Lipschitz (i.e. well-conditioned) smooth function. Subsequently, an important line of research emerged with the aim to accelerate FW. Guélat & Marcotte (1986) obtained linear convergence rates in well conditioned problems over polytopes under the a-priori assumption that the solution lies in the relative interior of the feasible set, and the rate of convergence explicitly depends on the distance the solution is away from the boundary (see also Epelman & Freund (2000); Beck & Teboulle (2004)). If no a-priori information on the location of the solution is available, there are essentially two known twists of the vanilla FW to boost the convergence rates. One twist is to modify the search directions via *corrective* or *away* search directions (Lacoste-Julien & Jaggi, 2015; Freund et al., 2017; Beck & Shtern, 2017; Gutman & Peña, 2020; Peña & Rodríguez, 2018). These methods require however more fine grained oracles which are not in general available for SC functions[1]. The alternative twist is to restrict the LMO in order to gain a more powerful local approximating model of the objective function. This last strategy has been used in several recent papers (Harchaoui et al., 2015; Garber & Hazan, 2016). A unified analysis for different settings was recently proposed in (Pedregosa et al., 2020), which inspired one of the variants of our methods.

None of these references explicitly studied the important case of SC minimization. Nesterov (2018b) requires the gradient of the objective function to be Hölder continuous. Implicitly it is assumed that the feasible set $\mathcal{X}$ is contained in the domain of the objective function f, an assumption we do not make, and is also not satisfied in important applications (e.g. $0 \in \mathcal{X}$ in the Poisson inverse problem below, but $0 \notin \operatorname{dom} f$). Specialized to solving a quadratic Poisson inverse problem in phase retrieval, Odor et al. (2016) provided a globally convergent FW method. They analyzed this problem via techniques developed in (Nesterov, 2018b) and with step-size specific to the application. We instead give a unified general analysis valid for *all* SC functions. Given the importance of SC minimization in machine learning, a large volume of work around proximal Newton and proximal gradient methods have been developed in a series of papers by (Dinh et al., 2013; Tran-Dinh et al., 2014; 2015). This paper complements this line of research by providing a FW analysis of SC optimization problems. Recently, (Liu et al., 2020) coupled a LMO with a second-order model of the SC objective function and obtained similar results to ours. Similar to (Liu et al., 2020), the recent paper (Carderera & Pokutta, 2020) studied a Newton-FW approach, but for well-conditioned functions.

2. Preliminaries

We consider the minimization problem

$$\min_{x\in\mathcal{X}} f(x), \tag{1}$$

where $\mathcal{X}$ is a compact convex set in $\mathbb{R}^n$ with nonempty interior. We assume that $\operatorname{dom} f \cap \mathcal{X} \neq \varnothing$ and the set $\mathcal{X}$ is represented by an LMO returning at a point x the target vector

$$s(x) \triangleq \arg\min_{s\in\mathcal{X}} \langle \nabla f(x), s\rangle. \tag{2}$$

In case of multiple solutions of (2) we assume that ties are broken by some arbitrary mechanism. The function f is assumed to be self-concordant (SC) (Nesterov & Nemirovskii, 1994) in the following sense: Let $f \in \mathbf{C}^3(\operatorname{dom} f)$ be a closed convex function with open domain $\operatorname{dom} f \triangleq \{x \in \mathbb{R}^n | f(x) < \infty\}$. For a fixed $x \in \operatorname{dom} f$ and direction $u \in \mathbb{R}^n$, define $\phi(x;t) \triangleq f(x+tu)$ for $t \in \operatorname{dom}\phi(x;\cdot)$.

Definition 2.1. A proper closed convex function $f : \mathbb{R}^n \to (-\infty, \infty]$ with open domain $\operatorname{dom} f$ is self-concordant (SC) with parameter $M > 0$ (i.e. $f \in \mathcal{F}_M$) iff

$$|\phi'''(x,0)| \leq M\phi''(x,0)^{3/2}. \tag{3}$$

SC functions were originally introduced in the context of interior-point methods (Nesterov & Nemirovskii, 1994), but recently have received significant attention in machine learning as many problems in supervised learning and empirical risk minimization involve loss functions which are SC. We give a (non-exhaustive) list of examples next. Note that, unlike classical settings, f in (1) is not assumed to be a self-concordant barrier for $\mathcal{X}$.

2.1. Examples

Poisson Inverse Problem. Consider the low-light imaging problem in signal processing, where the imaging data is collected by counting photons hitting a detector over time. In this setting, we wish to accurately reconstruct an image in low-light, which leads to noisy measurements due to low photon count. Assume that the data-generating process of the observations follows as Poisson distribution

$$p(y|Wx^*) = \prod_{i=1}^{m} \frac{(w_i^\top x^*)^{y_i}}{y_i!} \exp(-w_i^\top x^*),$$

[1] The away step version of FW needs a vertex oracle. Vertices need not be in the domain of a SC function (i.e. the origin in Example 2.1).

where $x^* \in \mathbb{R}^n$ is the true image, W is a linear operator that projects the scene onto observations, w_i is the i-th row of W and $y \in \mathbb{N}^m$ is the vector of observed photon counts. The maximum likelihood formulation of the Poisson inverse problem (Harmany et al., 2011) under sparsity constraints leads to the optimization problem

$$\min_{x \in \mathbb{R}^n_+ ; \|x\|_1 \leq M} \{ f(x) = \sum_{i=1}^m w_i^\top x - \sum_{i=1}^m y_i \ln(w_i^\top x) \}. \quad (4)$$

Setting $\varphi_i(t) \triangleq t - y_i \ln(t)$ for $t > 0$, we observe that $f(x) = \sum_{i=1}^m \varphi_i(w_i^\top \theta)$. Since each individual function φ_i is SC with parameter $M_{\varphi_i} \triangleq \frac{2}{\sqrt{y_i}}$, general rules of SC calculus shows that f is SC with domain $\bigcap_{i=1}^m \{x \in \mathbb{R}^n | w_i^\top x > 0\}$, and parameter $M \triangleq \max_{1 \leq i \leq m} \frac{2}{\sqrt{y_i}}$ (Nesterov (2018a), Thm. 5.1.1).

Learning Gaussian Markov random fields. We consider learning a Gaussian graphical random field of p nodes/variables from a data set $\{\phi_1, \ldots, \phi_N\}$ (Speed & Kiiveri, 1986; Dinh et al., 2013). Each random vector ϕ_j is an i.i.d realization from a p-dimensional Gaussian distribution with mean μ and covariance matrix Σ. Let $\Theta = \Sigma^{-1}$ be the precision matrix. To satisfy conditional dependencies between the random variables, Θ must have zero in Θ_{ij} if i and j are not connected in the underlying dependency graph. To learn the graphical model subject to sparsity constraints, we minimize the loss function

$$f(x) \triangleq -\ln\det(\mathrm{mat}(x)) + \mathrm{tr}(\hat{\Sigma}\,\mathrm{mat}(x))$$

over the ℓ_1-ball $\mathcal{X} \triangleq \{x \in \mathbb{R}^n | \,\mathrm{mat}(x) \succ 0, \|x\|_1 \leq R\}$, where $n = p^2$ and $\mathrm{mat}(x) \in \mathbb{R}^{p \times p}_{\mathrm{sym}}$ is the $p \times p$ symmetric matrix constructed from the $n = p^2$-dimensional vector x. f is SC with $M = 2$.

Logistic Regression. Consider the regularized logistic regression problem

$$f(x) = \frac{1}{N} \sum_{i=1}^N \ell(y_i(\langle \phi_i, x \rangle + \mu)) + \frac{\gamma}{2}\|x\|_2^2 \quad (5)$$

where $\ell(t) \triangleq \log(1 + e^{-t})$ is the logistic loss, μ is a given intercept, $y_i \in \{-1, 1\}$ and $\phi_i \in \mathbb{R}^n$ are given as input data for $i = 1, 2, \ldots, N$, and $\gamma > 0$ is a given regularization parameter. According to Prop. 5 in Sun & Tran-Dinh (2018), this functions is SC with parameter $M = \frac{1}{\sqrt{\gamma}} \max\{\|\phi_i\|_2 | 1 \leq i \leq n\}$. To promote sparsity, we minimize (5) over the ℓ_1-ball $\mathcal{X} = \{x \in \mathbb{R}^n | \|x\|_1 \leq R\}$. The resulting SC optimization problem (1) is the constrained formulation of the elastic-net regularization.

Portfolio Optimization. In this problem there are n assets with returns $r_t \in \mathbb{R}^n_+$ in period t of the investment horizon. The goal is to minimize the utility function $f(x)$ of the investor by choosing the weights of the assets in the portfolio. Our task is to design a portfolio x solving the problem

$$\min_x \left\{ f(x) = -\sum_{t=1}^T \ln(r_t^\top x) : x_i \geq 0, \sum_{i=1}^n x_i = 1 \right\}. \quad (6)$$

Note that this problem can be cast into an online optimization model (Hazan & Arora, 2006). It can be seen that $f \in \mathcal{F}_2$. We remark that this problem appears also in the classical universal prediction problem in information theory and online learning (Merhav & Feder, 1998).

Algorithm 1 Standard Frank-Wolfe method

```
Input: $x^0 \in \mathrm{dom}\, f \cap \mathcal{X}$ initial state, Step size policy
$(\alpha_k)_{k \geq 0}$ (either $\alpha_k = \frac{2}{k+2}$, or via line-search); $\varepsilon > 0$
tolerance level
for $k = 0, 1, \ldots$ do
  if $\mathsf{Gap}(x^k) > \varepsilon$ then
    Obtain $s^k = s(x^k)$
    Set $x^{k+1} = x^k + \alpha_k(s^k - x^k)$
  end if
end for
```

2.2. The Limits of Standard FW

In this section we explain why the standard analysis of FW does not work for SC functions. For this, we need to briefly recapitulate the main steps in the modern analysis initiated in Clarkson (2010), and later on refined by Hazan (2008) for matrix problems over a spactrahedron, and finally concluded in Jaggi (2013) for general convex compact domains.

The typical merit function employed in FW algorithms is the dual gap function

$$\mathsf{Gap}(x) \triangleq \max_{s \in \mathcal{X}} \langle \nabla f(x), x - s \rangle. \quad (7)$$

It is easy to see that $\mathsf{Gap}(x) \geq 0$ for all $x \in \mathcal{X}$, with equality if and only if x is a solution to (1). The convergence analysis of Algorithm 1 relies on the *curvature constant* (Jaggi, 2013)

$$C_f \triangleq \sup_{x, s \in \mathcal{X}, \gamma \in [0,1]} \frac{2}{\gamma^2} D_f(x + \gamma(s - x), x),$$

where we define the *Bregman divergence* of the smooth convex function f as

$$D_f(y, x) \triangleq f(y) - f(x) - \langle \nabla f(x), y - x \rangle$$

for all $x, y \in \mathrm{dom}\, f$. Under the assumption that $C_f < \infty$, Jaggi (2013) proved sublinear $O(1/k)$ rate of convergence in terms of the dual gap function $\mathsf{Gap}(\cdot)$. Unfortunately, minimizing a SC function over a compact set does not necessarily give us a finite curvature constant, as the following example illustrates.

Example 2.1. Consider the function $f(x,y)=-\ln(x)-\ln(y)$ considered over the set $\mathcal{X}=\{(x,y)\in\mathbb{R}^2_+|x+y=1\}$. This function is the standard SC barrier for the positive orthant (the log-barrier) and its Bregman divergence is easily calculated as

$$D_f(u,v)=\sum_{i=1}^{2}\left[-\ln\left(\frac{u_i}{v_i}\right)+\frac{u_i}{v_i}-1\right]$$

where $u=(u_1,u_2),v=(v_1,v_2)$. We see that $C_f=\infty$. Neither the function f, nor its gradient, is Lipschitz continuous over the set of interest. Moreover, if we start from $u^0=(1/4,3/4)$, using the standard step-size $\alpha_k=2/(k+2)$, we have $\alpha_0=1$ and $u^1=s(u^0)=(1,0)\notin\operatorname{dom} f$ and the method fails.

We point out that Example 2.1 is representative for the class of optimization problems of interest in this work since the logarithm is the prime example for a self-concordant function. It is thus clear that the standard analysis based on finite curvature estimates (or, as a particular case, Lipschitz continuous gradient) cannot be applied to analyze FW when applied to (1).

2.3. Basic Estimates

From Nesterov (2018a), Thm. 5.1.6, we know that if $\operatorname{dom} f$ contains no lines, then $\nabla^2 f(x)\succ 0$ for all $x\in\operatorname{dom} f$ (f is *non-degenerate*). We shall assume that f is non-degenerate. Define the local norm of $u\in\mathbb{R}^n$ at $x\in\operatorname{dom} f$ as $\|u\|_x\triangleq\sqrt{\langle\nabla^2 f(x)u,u\rangle}$. The dual norm at $x\in\operatorname{dom} f$ is defined as $\|u\|_x^*\triangleq\sqrt{\langle[\nabla^2 f(x)]^{-1}u,u\rangle}$. Given $f\in\mathcal{F}_M$, we define the distance-like function

$$\mathsf{d}(x,y)\triangleq\frac{M}{2}\|y-x\|_x,$$

as well as

$$\omega(t)\triangleq t-\ln(1+t),\text{ and }\omega_*(t)\triangleq -t-\ln(1-t).$$

It is not hard to verify that $\omega(t)\geq 0$ for all $t>-1$ and $\omega_*(t)\geq 0$ for every $t<1$; Moreover, ω and ω_* are increasing and strictly convex in $[0,\infty)$ and $[0,1)$, respectively. For all $x,\tilde{x}\in\operatorname{dom} f$ Thms. 5.1.8 and 5.1.9 in Nesterov (2018a) state that

$$f(\tilde{x})\geq f(x)+\langle\nabla f(x),\tilde{x}-x\rangle+\frac{4}{M^2}\omega\left(\mathsf{d}(x,\tilde{x})\right)\quad(8)$$

$$f(\tilde{x})\leq f(x)+\langle\nabla f(x),\tilde{x}-x\rangle+\frac{4}{M^2}\omega_*\left(\mathsf{d}(x,\tilde{x})\right)\quad(9)$$

where in the latter inequality we assume $\mathsf{d}(x,\tilde{x})<1$.
An attractive feature of SC functions, in particular from the point of view of FW algorithms (Jaggi, 2013), is that self-concordance is invariant under affine transformations and re-parametrizations of the domain. See Section A of the supplementary materials for a precise statement of this fact, and further properties of SC functions. Affine invariance allows us to cover a very broad class of composite convex optimization problems of the form

$$\min_{x\in\mathcal{X}}\{f(x)\triangleq g(\mathbf{E}x)+\langle q,x\rangle\},\quad(10)$$

where g is SC, and $\mathbf{E}\in\mathbb{R}^{m\times n}$, $q\in\mathbb{R}^n$. This formulation covers empirical risk-minimization with a convex norm-like regularization, as explained in Section 2.1, among many others.

We have the following existence result for solutions of problem (1). For a proof see Section A in the supplementary materials.

Proposition 2.2. *Suppose there exists $x\in\operatorname{dom} f\cap\mathcal{X}$ such that $\|\nabla f(x)\|_x^*\leq\frac{2}{M}$. Then (1) admits a unique solution.*

We note that there can be at most one solution to problem (1) because $\nabla^2 f\succ 0$ on $\operatorname{dom} f\cap\mathcal{X}$. Henceforth, we will assume that (1) admits a solution x^*.

3. FW works for Self-Concordant Functions

This section gives a high-level overview on the necessary modifications of FW in order to derive provably convergent schemes for minimizing an SC function over a compact convex set $\mathcal{X}$. We propose two new step-size policies, both derived from different approximating local models guaranteeing a version of the celebrated descent lemma. To explain our approach, let us define for all $x\in\operatorname{dom} f$

$$\mathsf{e}(x)\triangleq\mathsf{d}(x,s(x))=\frac{M}{2}\|s(x)-x\|_x.\quad(11)$$

The first step-size rule we analyze, called Variant 1 (V1), is based on the local sufficient decrease property (9). In particular, the step-size policy in V1 minimizes at each iteration a local model in the r.h.s. of (9), and reads explicitly as

$$\alpha_k^{V1}\triangleq\min\left\{1,\frac{\mathsf{Gap}(x^k)}{\mathsf{e}(x^k)(\mathsf{Gap}(x^k)+\frac{4}{M^2}\mathsf{e}(x^k))}\right\}.\quad(12)$$

Observe, that this step-size rule needs to compute the gap function and the local distance $\mathsf{e}(x^k)$. For evaluating this local distance, it is not necessary to evaluate the full Hessian matrix $\nabla^2 f(x)$, but rather only the matrix-vector product $\nabla^2 f(x)(s^k-x^k)$. In many situations of interests, this vector can be computed easily. A particular appealing instance where this applies is the generalized linear model $f(x)=\sum_{i=1}^N f_i(\langle a_i,x\rangle)$, each function f_i being SC. In this case, the Hessian matrix is a sum of rank-one matrices, Hessian-vector product amounts to vector-vector and vector-scalar products, and the cost of calculating $\mathsf{e}(x^k)$ is of the same order as the cost of calculating the gradient.

Variant 2 (V2) is based on a backtracking procedure in the spirit of Pedregosa et al. (2020). Specifically, V2 employs the quadratic model

$$Q(x^k, t, \mu) \triangleq f(x^k) - t\,\mathsf{Gap}(x^k) + \frac{t^2\mu}{2}\|s(x^k) - x^k\|_2^2 \tag{13}$$

If $\mu \geq \mathcal{L}_k$, where $\mathcal{L}_k$ is a local gradient Lipschitz constant estimate, $Q(x^k, t, \mu)$ is an upper approximation for the objective. This approximation has the advantage that its minimum over $t \in [0, 1]$ can be computed in closed form, yielding the step size

$$\alpha_k^{V2} \triangleq \min\left\{1, \frac{\mathsf{Gap}(x^k)}{\mathcal{L}_k\|s(x^k) - x^k\|_2^2}\right\}. \tag{14}$$

The quality of this quadratic approximation will depend on the local Lipschitz estimate $\mathcal{L}_k$. This parameter needs to be carefully selected to ensure convergence while keeping the number of function evaluations to a minimum. This is achieved through Algorithm 3, which mimics the backtracking procedure of Pedregosa et al. (2020).

Algorithm 2 Adaptive Frank-Wolfe method for SC functions

```
**Input:** $x^0 \in \mathrm{dom}\, f \cap \mathcal{X}$ initial state, $\beta \in (0, 1)$, Tolerance $\varepsilon > 0$.
for $k = 1, \ldots$ do
  if $\mathsf{Gap}(x^k) > \varepsilon$ then
    Obtain $s^k = s(x^k)$ and set $v^k = s^k - x^k$.
    Obtain $\mathsf{e}^k = \frac{M}{2}\|v^k\|_{x^k}$
    if Variant 1: then
      Set $\mathtt{t}_k = \frac{\mathsf{Gap}(x^k)}{\mathsf{e}^k(\mathsf{Gap}(x^k)+\frac{4}{M^2}\mathsf{e}^k)}$
      Set $\alpha_k = \min\{1, \mathtt{t}_k\}$
    end if
    if Variant 2:  then
      Set $(\alpha_k, \mathcal{L}_k) = \mathtt{step}(f, v^k, x^k, \mathsf{Gap}(x^k), \mathcal{L}_{k-1})$
    end if
    Set $x^{k+1} = x^k + \alpha_k v^k$
  end if
end for
```

The backtracking performed in this subroutine defines a candidate step size α and checks whether the sufficient decrease condition

$$f(x^k + \alpha(s^k - x^k)) < Q(x^k, \alpha, \mu) \tag{15}$$

is satisfied. If not, then we increase the proposed Lipschitz estimate to $\gamma_u\mathcal{L}, \gamma_u > 1$, and repeat. As will be shown in the analysis of this method, this described backtracking procedure is guaranteed to stop in finite steps.

3.1. Analysis of Variant 1

Section B in the supplementary materials contains complete proofs of all the derivations and claims made in this section.

Algorithm 3 Function $\mathtt{step}(f, v, x, g, \mathcal{L})$

```
Choose $\gamma_u > 1, \gamma_d < 1$
Choose $\mu \in [\gamma_d\mathcal{L}, \mathcal{L}]$
$\alpha = \min\{\frac{g}{\mathcal{L}\|d\|_2^2}, 1\}$
if $f(x + \alpha v) > Q(x, \alpha, \mu)$ then
  $\mu = \gamma_u\mu$
  $\alpha = \min\{\frac{g}{\mu\|v\|_2^2}, 1\}$
end if
Return $\alpha, \mu$
```

For $x \in \mathrm{dom}\, f$, define the target vector $s(x)$ as in (2), and $\mathsf{Gap}(x)$ as in (7). Given $x \in \mathcal{X}$ and $t > 0$, set $x_t^+ \triangleq x + t(s(x) - x)$. Assume that $\mathsf{e}(x) \neq 0$. By construction,

$$\mathsf{d}(x, x_t^+) = \frac{tM}{2}\|s(x) - x\|_x = t\mathsf{e}(x) < 1,$$

iff $t < 1/\mathsf{e}(x)$. Choosing $t \in (0, 1/\mathsf{e}(x))$, we conclude from (9)

$$\begin{aligned} f(x_t^+) &\leq f(x) + \langle \nabla f(x), x_t^+ - x\rangle + \frac{4}{M^2}\omega_*(t\mathsf{e}(x)) \\ &= f(x) - t\,\mathsf{Gap}(x) + \frac{4}{M^2}\omega_*(t\mathsf{e}(x)) \end{aligned}$$

This shows that when minimizing an SC function, we can search for a step size α_k which minimizes the r.h.s. of the previous inequality by maximizing

$$\eta_x(t) \triangleq t\,\mathsf{Gap}(x) - \frac{4}{M^2}\omega_*(t\mathsf{e}(x)), \tag{16}$$

for $t \in (0, 1/\mathsf{e}(x))$. As shown in Section B.2 of the supplementary materials, this function attains a unique maximum at the value

$$\mathtt{t}(x) \triangleq \frac{\mathsf{Gap}(x)}{\mathsf{e}(x)(\mathsf{Gap}(x) + \frac{4}{M^2}\mathsf{e}(x))} \equiv \frac{\gamma(x)}{\mathsf{e}(x)}. \tag{17}$$

We then construct the step size sequence $(\alpha_k)_{k\geq 0}$ by setting $\alpha_k = \min\{1, \mathtt{t}(x^k)\}$. The convexity of $\mathcal{X}$ and the fact that $\alpha_k\mathsf{e}(x^k) < 1$ guarantee that $(x^k)_{k\geq 0} \subset \mathrm{dom}\, f \cap \mathcal{X}$. Thus, at each iteration, we reduce the objective function value by at least the quantity $\Delta_k \equiv \eta_{x^k}(\alpha_k)$, so that $f(x^{k+1}) \leq f(x^k) - \Delta_k < f(x^k)$. Hence, we see that the method induces a sequence of function values $(f(x^k))_{k\geq 0}$ which is monotonically decreasing by at least the amount Δ_k in each iteration. Equivalently,

$$(x^k)_{k\geq 0} \subset \mathcal{S}(x^0) \triangleq \{x \in \mathrm{dom}\, f \cap \mathcal{X} | f(x) \leq f(x^0)\}.$$

Furthermore, $\sum_{k\geq 0}\Delta_k < \infty$, and hence the sequence $(\Delta_k)_{k\geq 0}$ converges to 0, and

$$\min_{0\leq k<K}\Delta_k \leq \frac{1}{K}(f(x^0) - f^*).$$

It follows from Lemma B.1 in the supplementary materials that $\frac{4}{M^2}\omega(\mathsf{d}(x^*,x)) \leq f(x) - f^*$ for all $x \in \operatorname{dom} f$. Consequently,

$$\mathcal{S}(x^0) \subseteq \left\{x \in \operatorname{dom} f | \omega(\mathsf{d}(x^*,x)) \leq \frac{M^2}{4}(f(x^0) - f^*)\right\}.$$

Since $\omega(\cdot)$ is continuous and increasing, $\mathcal{S}(x^0)$ is a closed, bounded and convex set. Accordingly, defining by $\lambda_{\max}(\nabla^2 f(x))$ and $\lambda_{\min}(\nabla^2 f(x))$ the largest, respectively smallest, eigenvalue of the Hessian $\nabla^2 f(x)$, the numbers

$$L_{\nabla f} \triangleq \max_{x\in\mathcal{S}(x^0)} \lambda_{\max}(\nabla^2 f(x)), \text{ and}$$
$$\sigma_f \triangleq \min_{x\in\mathcal{S}(x^0)} \lambda_{\min}(\nabla^2 f(x)),$$

are well defined and finite. In particular, for all $x \in \mathcal{S}(x^0)$ we have a restricted strong convexity of the function f, in the sense that

$$f(x) - f(x^*) \geq \frac{\sigma_f}{6}\|x - x^*\|_2^2. \tag{18}$$

In terms of these quantities, we can bound the sequence $(\mathsf{e}^k)_{k\geq 0}$, defined as $\mathsf{e}^k \triangleq \mathsf{e}(x^k)$, as

$$\frac{M\sqrt{\sigma_f}}{2}\|s^k - x^k\|_2 \leq \mathsf{e}^k \leq \frac{M\sqrt{L_{\nabla f}}}{2}\|s^k - x^k\|_2. \tag{19}$$

In order to derive convergence rates, we need to lower bound the per-iteration decrease in the objective function. A detailed analysis, given in Section B of the supplementary materials, reveals that we can construct a useful minorizing sequence for the per-iteration function decrease $(\Delta_k)_{k\geq 0}$ as

$$\Delta_k \geq \min\{\mathtt{a}\,\mathsf{Gap}(x^k), \mathtt{b}\,\mathsf{Gap}(x^k)^2\}, \tag{20}$$

where $\mathtt{a} \triangleq \min\left\{\frac{1}{2}, \frac{2(1-\ln(2))}{M\sqrt{L_{\nabla f}}\operatorname{diam}(\mathcal{X})}\right\}$ and $\mathtt{b} \triangleq \frac{1-\ln(2)}{L_{\nabla f}\operatorname{diam}(\mathcal{X})^2}$. With the help of this lower bound, we are now able to establish the $\mathcal{O}(k^{-1})$ convergence rate in terms of the approximation error $h_k \triangleq f(x^k) - f^*$.

By convexity, we have $\mathsf{Gap}(x^k) \geq h_k$. Therefore, the lower bound for Δ_k in (20) can be estimated as $\Delta_k \geq \min\{\mathtt{a}h_k, \mathtt{b}h_k^2\}$. Hence,

$$h_{k+1} \leq h_k - \min\{\mathtt{a}h_k, \mathtt{b}h_k^2\} \qquad \forall k \geq 0. \tag{21}$$

Given this recursion, we can identify two phases characterizing the process $(h_k)_{k\geq 0}$. In Phase I, the approximation error h_k is at least $\mathtt{a}/\mathtt{b}$, and in Phase II the approximation error falls below this value. The cut-off value $\mathtt{a}/\mathtt{b}$ determines the nature of the recursion (21), and yields immediately an estimate for the iteration complexity of Variant 1 of Algorithm 2.

Theorem 3.1. *For all $\varepsilon > 0$, define the stopping time*

$$N_\varepsilon(x^0) \triangleq \inf\{k \geq 0 | h_k \leq \varepsilon\}. \tag{22}$$

Then,

$$N_\varepsilon(x^0) \leq \left\lceil \frac{1}{\mathtt{a}}\ln\left(\frac{h_0\mathtt{b}}{\mathtt{a}}\right)\right\rceil + \frac{L_{\nabla f}\operatorname{diam}(\mathcal{X})^2}{(1-\ln(2))\varepsilon}. \tag{23}$$

The proof is in Section B.1 of the supplementary materials.

3.2. Analysis of Variant 2

For the analysis of V2, we first need to establish well-posedness of the backtracking scheme $\mathtt{step}(f, v, x, g, \mathcal{L})$. Calling this routine at the position $x = x^k$ within the execution of Algorithm 2, we require the search direction $v^k \triangleq s^k - x^k$, where $s^k = s(x^k)$ is the target vector (2). Define

$$\gamma_k \triangleq \max\{t \geq 0 | x^k + t(s^k - x^k) \in \mathcal{S}(x^k)\} \tag{24}$$

as the largest step size guaranteeing feasibility and decrease of the objective function value. Hence, for all $t \in [0, \gamma_k]$, we have $f(x^k + t(s^k - x^k)) \leq f(x^k)$, and $x^k + t(s^k - x^k) \in \mathcal{X}$.

Lemma 3.2. *Assume that $x^k \in \mathcal{S}(x^0)$ for all $k \geq 0$. Then, for all $t \in [0, \gamma_k]$, it holds true that $x^k + t(s^k - x^k) \in \mathcal{S}(x^k)$, and*

$$\|\nabla f(x^k + t(s^k - x^k)) - \nabla f(x^k)\|_2 \leq L_{\nabla f} t\|s^k - x^k\|_2.$$

Using this estimate, a localized version of the celebrated descent Lemma reads as

$$f(x^k + t(s^k - x^k)) \leq Q(x^k, t, L_{\nabla f}),$$

for all $t \in [0, \gamma_k]$ for the quadratic model introduced in (13). A direct minimization strategy for constructing a feasible step size policy based on the majorizing quadratic model $t \mapsto Q(x^k, t, L_{\nabla f})$ would yield the step size

$$\tau_k(L_{\nabla f}) \triangleq \min\left\{1, \frac{\mathsf{Gap}(x^k)}{L_{\nabla f}\|s^k - x^k\|_2^2}\right\}. \tag{25}$$

The main problem with this approach is that it needs the global parameter $L_{\nabla f}$. In practice this quantity is often hard to obtain and, furthermore, frequently numerically quite large. This in turn renders the step size to be inevitably small, leading to bad performance of the method. Our solution strategy is thus to implement the adaptive backtracking procedure calling the function $\mathtt{step}(f, v^k, x^k, \mathsf{Gap}(x^k), \mathcal{L}_{k-1})$, which requires a local Lipschitz estimate $\mathcal{L}_{k-1}$, guaranteed to be smaller than $L_{\nabla f}$. This subroutine needs an initial estimate $\mathcal{L}_{-1}$ whose choice is explained in Section C of the supplementary materials.

Furthermore, Algorithm 3 comes with two hyperparameters $\gamma_d < 1 < \gamma_u$. Practically efficient values for these parameters are reported to be $\gamma_d = 0.9$ and $\gamma_u = 2$ (Pedregosa et al., 2020). Section C of the supplementary materials gives an upper bound estimate on the number of necessary function evaluations during a single execution of the backtracking procedure. The $\mathcal{O}(k^{-1})$ convergence rate of V2 involves the *dual objective function*

$$\psi(z) \triangleq -f^*(z) - H_{\mathcal{X}}(-z),$$

where f^* denotes the Fenchel conjugate of f and $H_{\mathcal{X}}(c) \triangleq \sup_{x\in\mathcal{X}}\langle x, c\rangle$ is the support function over $\mathcal{X}$. Note that ψ is concave and we have $f^* \triangleq \min_{x\in\mathcal{X}} f(x) = \max_u \psi(u)$. The following result is then similar to Theorem 3 in Pedregosa et al. (2020), and reports the $\mathcal{O}(k^{-1})$ convergence rate in terms of the sequence of approximation errors $h_k = f(x^k) - f^*$.

Theorem 3.3. *Let f be an SC function and $(x^k)_{k\geq 0}$ generated by Variant 2 of Algorithm 2. Then*

$$h_k \leq \frac{2\,\mathsf{Gap}(x^0)}{(k+1)(k+2)} + \frac{k\,\mathrm{diam}(\mathcal{X})^2}{(k+1)(k+2)}\bar{\mathcal{L}}_k \tag{26}$$

where $\bar{\mathcal{L}}_k \triangleq \frac{1}{k}\sum_{i=0}^{k-1}\mathcal{L}_i$ is the arithmetic mean over all Lipschitz estimates computed by Algorithm 3 during k executions of the main protocol.

Since $\mathcal{L}_k \leq L_{\nabla f}$, this implies $h_k = \mathcal{O}(k^{-1})$. The proof of this Theorem can be found in supplementary material Section C.3.

4. Inducing Linear Convergence

In this section we use the construction of a LLOO given in Garber & Hazan (2016) to deduce an accelerated version of the base FW Algorithm 2. Garber & Hazan (2016) give an explicit construction of such an oracle for *any* polytope of form $\mathcal{X} = \{x \in \mathbb{R}^n | Ax = a, Bx \leq b\}$, where $A, B \in \mathbb{R}^{m\times n}$ and $a, b \in \mathbb{R}^m$. In the particularly important case where $\mathcal{X}$ is a standard unit simplex, the LLOO can be constructed efficiently. In the supplementary material we describe the construction for this special geometry. Let $\mathbb{B}(x, r) \triangleq \{y \in \mathbb{R}^n | \|y - x\|_2 \leq r\}$ denote the Euclidean ball with radius $r > 0$ and center x.

Definition 4.1 (Garber & Hazan (2016), Def. 2.5)**.** A procedure $\mathcal{A}(x, r, c)$, where $x \in \mathcal{X}, r > 0, c \in \mathbb{R}^n$, is a LLOO with parameter $\rho \geq 1$ for the polytope $\mathcal{X}$ if $\mathcal{A}(x, r, c)$ returns a point $s \in \mathcal{X}$ such that for all $y \in \mathbb{B}(x, r) \cap \mathcal{X}$

$$\langle c, y\rangle \geq \langle c, s\rangle \text{ and } \|x - s\|_2 \leq \rho r. \tag{27}$$

Algorithm 4 LLOO-based convex optimization

```
Input: $\mathcal{A}(x,r,c)$-LLOO with parameter $\rho \geq 1$ for polytope $\mathcal{X}$, $f \in \mathcal{F}_M$. $\sigma_f > 0$ convexity parameter.
$x^0 \in \mathrm{dom}\, f \cap \mathcal{X}$, and let $h_0 = f(x^0) - f^*$, $c_0 = 1$.
for k=0 do
  Obtain $r_0 = \sqrt{\frac{6\,\mathsf{Gap}(x^0)}{\sigma_f}}$.
  Obtain $s^0 = \mathcal{A}(x^0, r_0, \nabla f(x^0))$
  Set $\alpha_0 = \frac{1}{1+\mathsf{e}^0}\min\{1, \frac{\mathsf{Gap}(x^0)}{\frac{4}{M^2}(\mathsf{e}^0)^2}\}$, where $\mathsf{e}^0 = \|s^0 - x^0\|_{x^0}$.
  Update $x^1 = x^0 + \alpha_0(s^0 - x^0)$
end for
for $k = 1, \dots$ do
  if $\mathsf{Gap}(x^k) > \varepsilon$ then
    Set $c_k = \exp\left(-\frac{1}{2}\sum_{i=0}^{k-1}\alpha_i\right)$
    Set $\alpha_k = \min\{\frac{c_k\,\mathsf{Gap}(x^0)}{\frac{4}{M^2}(\mathsf{e}^k)^2}, 1\}\frac{1}{1+\mathsf{e}^k}$
    Set $r_k = r_0 c_k$.
    Obtain $s^k = \mathcal{A}(x^k, r_k, \nabla f(x^k))$
    Set $x^{k+1} = x^k + \alpha_k(s^k - x^k)$
  end if
end for
```

Assuming the availability of a procedure $\mathcal{A}(x, r, c)$ for any point $x \in \mathcal{X}$, we run Algorithm 4. Our analysis of this Algorithm departs from eq. (9), saying that for $\alpha \in (0, 1/\mathsf{e}^k)$ we have

$$f(x^{k+1}) \leq f(x^k) + \alpha\langle\nabla f(x^k), s^k - x^k\rangle + \frac{4}{M^2}\omega_*(\alpha\mathsf{e}^k),$$

To exploit the power of the LLOO, we need to control the right-hand side of the previous display by bounding carefully the radius of a single step made by the algorithm. Via a delicate induction argument, based on estimates for SC functions and Garber & Hazan (2016), we obtain the announced linear convergence result.

Theorem 4.2. *Let $(x^k)_{k\geq 0}$ be generated by Algorithm 4. Then for all $k \geq 0$ we have $x^* \in \mathbb{B}(x^k, r_k)$ and*

$$h_k \leq \mathsf{Gap}(x^0)\exp\left(-\frac{1}{2}\sum_{i=0}^{k-1}\alpha_i\right) \tag{28}$$

In particular, defining $\bar{\alpha} \triangleq \min\{\frac{\sigma_f}{6L_{\nabla f}\rho^2}, 1\}\frac{1}{1+\sqrt{L_{\nabla f}\frac{M\,\mathrm{diam}(\mathcal{X})}{2}}}$, we see that $\alpha_k \geq \bar{\alpha}$, and therefore

$$h_k \leq \mathsf{Gap}(x^0)\exp(-k\bar{\alpha}/2).$$

The proof is in Section C of the supplementary materials.

5. Numerical Experiments

In the numerical experiments we tested the performance of Variant 1 (V1) and Variant 2 (V2) of Algorithm 2, and

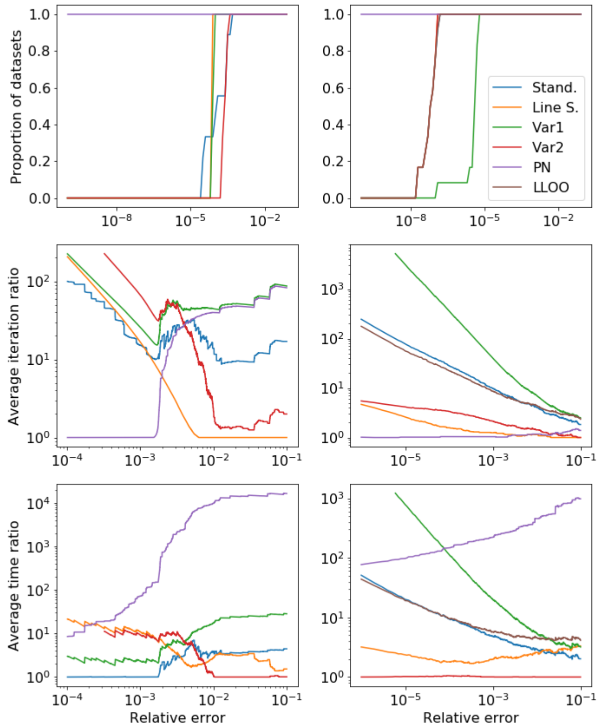

Figure 1. Performance of Algorithm 2 and 4 in the Poisson inverse problem (4) (left column) and in the Portfolio optimization problem (6) (right column). The top row reports the fraction of datasets for which a given relative error was achieved by an algorithm (Statistic $\rho_i(\varepsilon)$). The middle row presents the average ratio between number of iterations taken to reach a certain relative error to the method which reached this error in the minimal number of iterations (Statistic $\tilde{\rho}_i(\varepsilon)$). The bottom row presents the average time ratio between each method and that of the fastest method in order to reach a certain relative error (Statistic $\hat{\rho}_i(\varepsilon)$).

compared them with the performance of Frank-Wolfe with standard step-size of $\frac{2}{k+2}$ (stand.), and step-size determined by exact line-search (Line-S.). As a further benchmark, the self-concordant Proximal-Newton (PN) of Tran-Dinh et al. (2015), as implemented in the SCOPT package[2], is included. For the portfolio optimization problem, Algorithm 4 is also implemented. All codes are written in Python 3, with packages for scientific computing NumPy 1.18.1 and SciPy 1.4.1. The experiments were conducted on a PC with Intel Core i5-7500 3.4GHzs, with a total of 16GB RAM. In both experiments the Frank-Wolfe based methods have been terminated after 50,000 iterations. Because of its higher computational complexity, we decided to stop PN after 1,000 iterations. Each algorithm was stopped early if the optimality gap in a given iteration was lower than $1e-10$. LLOO was only implemented for the portfolio selection problem, using the local linear oracle given in (Garber & Hazan, 2016) for the simplex, as described in Section F in the supplementary materials.[3]

For the portfolio optimization problem we used synthetic data, as in Section 6.4 of (Sun & Tran-Dinh, 2018). The details of the data generating process are as follows. We generate matrix R with given price ratios as: $R := \text{ones}(n,p) + N(0, 0.1)$, which allows the closing price to vary about 10% between two consecutive periods. We used different sizes of matrix R: $(n,p) = (1000, 800), (1000, 1200)$, and $(1000, 1500)$ with 4 samples for each size. Hence, there are totally 12 datasets. The right column of Figure 1 displays the performance of all Algorithms developed in this paper when averaged over all 12 samples. The detailed results for individual samples are reported in Section E of the supplementary materials.

For the Poisson inverse problem we used the datasets a1a-a9a from the LIBSVM library (Chang & Lin, 2011). Averaged performance of the FW algorithms over these 9 datasets is displayed in the left column of Figure 1. The detailed results for each individual data set are reported in Section E in the supplementary materials.

The comparison between all the algorithms is made by the construction of versions of performance profiles, following Dolan & Moré (2002). In order to present the result, we first estimate f^* by the maximal lower bound on the function value achieved by any of the algorithms, and compute the relative error attained by each of the methods at iteration k. More precisely, given the set of methods $\mathcal{S}$ and test problems $\mathcal{P}$, denote by F_{ij} the function value attained by method $i \in \mathcal{S}$ on problem $j \in \mathcal{P}$. If $(x_{ij}^k)_k$ denotes the sequence produced by method i on problem j, we define the relative error as $r_{ij}^k = \frac{f(x_{ij}^k)-\min\{F_{sj}|s\in\mathcal{S}\}}{\min\{F_{sj}|s\in\mathcal{S}\}}$. Now, for all methods $i \in \mathcal{S}$ and any relative error ϵ, The top row of Figure 1 shows the proportion of datasets that achieves a relative error of at most ϵ, that is $\rho_i(\epsilon) = \frac{1}{|\mathcal{P}|}|\{j \in \mathcal{P}|\exists k,\ r_{ij}^k \leq \varepsilon\}|$. We are also interested in comparing iteration complexity and CPU time between the methods. For that purpose, we define $N_{ij}(\varepsilon) = \min\{k \geq 0 | r_{ij}^k \leq \varepsilon\}$ as the first iteration in which method $i \in \mathcal{S}$ achieves a relative error ε on problem $j \in \mathcal{P}$. Analogously, $T_{ij}(\varepsilon)$ measures the minimal CPU time in which method $i \in \mathcal{S}$ to achieves a relative error ε on problem $j \in \mathcal{P}$. Based on these statistics, the second row of Figure 1, we use the following average iteration ratio $\tilde{\rho}_i(\varepsilon) = \frac{1}{|\mathcal{P}|}\sum_{j\in\mathcal{P}} \frac{N_{ij}(\varepsilon)}{\min\{N_{sj}(\varepsilon)|s\in\mathcal{S}\}}$ for comparing the iteration complexity of all the methods. In the third row of Figure 1, we display the average time ratio $\hat{\rho}_i(\varepsilon) = \frac{1}{|\mathcal{P}|}\sum_{j\in\mathcal{P}} \frac{T_{ij}(\varepsilon)}{\min\{T_{sj}(\varepsilon)|s\in\mathcal{S}\}}$ for comparing the computational time of all the methods. In both cases, as the average ratio

[2] https://www.epfl.ch/labs/lions/technology/scopt/

[3] The codes are accessible via https://github.com/kamil-safin/SCFW.

is closer to 1 the performance of the method is closer to the best performance.

As expected, the top row of Figure 1 show that PN obtains a lower relative error than any of the Frank-Wolfe based methods. However, for the Frank-Wolfe methods, none of the step-sizes choices has a clear advantage over the other in obtaining lower relative error values in both examples. Moreover, in the portfolio, while PN has the lowest iteration complexity, Variant 2 in fact achieves the best times for all values of relative error, followed by line-search, with LLOO matching the performance of the standard step-size. We remark that the standard step-size policy has no theoretical convergence guarantees, as the problem has no finite curvature constant (nor has a Lipschitz continuous gradient over the unit simplex).
In the Poisson inverse problem displayed in the left column of Figure 1, Variant 2 obtains the best times for higher values of relative errors. The standard $2/(k+2)$ step size rule seems to be a good alternative here. We also implemented the convergent version with the step size $2/(k+3)$ derived in Odor et al. (2016) and did not observe any significant difference in practice with the standard step-size policy. It is remarkable that the worst-case step-size policy V1 is competitive with the problem-specific $2/(k+3)$ policy.

6. Conclusion

FW is a much appraised FOM for minimizing smooth convex functions over convex compact sets. The main merit of this method is the relative simplicity of its implementation and projection-free iterations. This yields great scalability properties making it a very attractive method for large-scale optimization. The price to pay for iteration-simplicity are, in general, slow convergence rates, and some sort of bounded curvature assumption. In this work, we show that for SC functions, which are neither strongly convex, nor have bounded curvature, we can obtain a novel step-size policy which, when coupled with local linear minimization oracles, features linear convergence rates. Under more standard assumptions of the feasible set given by LMO, we provide two novel step-size policies which lead to standard sublinear convergence rate for minimizing general SC functions by FW algorithm. In the future we plan to extend the results to the class of generalized self-concordant functions, as recently defined in (Sun & Tran-Dinh, 2018). Other directions of interest for future research are inertial methods, and stochastic optimization. We will address these problems in the near future. An interesting avenue could be to incorporate away steps. This seems to be challenging since vanilla away step is not trivially to implement as the function might not be bounded and the away step may lead to a point outside of the domain of the objective.

Acknowledgements

The authors sincerely thank Shoham Sabach for his contribution in the early stages of this project, including his part in developing the basic ideas used in this paper. We would also like to thank Quoc Tran-Dinh for sharing MATLAB codes. This research is supported by the COST Action CA16228 "European Network for Game Theory".

References

Bach, F. Self-concordant analysis for logistic regression. *Electron. J. Statist.*, 4:384–414, 2010. doi: 10.1214/09-EJS521. URL https://projecteuclid.org:443/euclid.ejs/1271941980.

Bauschke, H. H., Bolte, J., and Teboulle, M. A descent lemma beyond lipschitz gradient continuity: First-order methods revisited and applications. *Mathematics of Operations Research*, 42(2):330–348, 2017/09/12 2017. doi: 10.1287/moor.2016.0817. URL https://doi.org/10.1287/moor.2016.0817.

Beck, A. and Shtern, S. Linearly convergent away-step conditional gradient for non-strongly convex functions. *Mathematical Programming*, 164(1):1–27, 2017. doi: 10.1007/s10107-016-1069-4. URL https://doi.org/10.1007/s10107-016-1069-4.

Beck, A. and Teboulle, M. A conditional gradient method with linear rate of convergence for solving convex linear systems. *Mathematical Methods of Operations Research*, 59(2):235–247, 2004.

Canon, M. D. and Cullum, C. D. A tight upper bound on the rate of convergence of frank-wolfe algorithm. *SIAM Journal on Control*, 6(4):509–516, 2020/01/07 1968. doi: 10.1137/0306032. URL https://doi.org/10.1137/0306032.

Carderera, A. and Pokutta, S. Second-order conditional gradient sliding. *arXiv:2002.08907*, 2020.

Chang, C.-C. and Lin, C.-J. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3), May 2011. ISSN 2157-6904. doi: 10.1145/1961189.1961199. URL https://doi.org/10.1145/1961189.1961199.

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

Dinh, Q. T., Kyrillidis, A., and Cevher, V. A proximal newton framework for composite minimization: Graph learning without cholesky decompositions and matrix inversions. In *International Conference on Machine Learning*, pp. 271–279, 2013.

Dolan, E. D. and Moré, J. J. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002. ISSN 1436-4646. URL https://doi.org/10.1007/s101070100263.

Epelman, M. and Freund, R. M. Condition number complexity of an elementary algorithm for computing a reliable solution of a conic linear system. *Mathematical Programming*, 88(3):451–485, 2000. doi: 10.1007/s101070000136. URL https://doi.org/10.1007/s101070000136.

Frank, M. and Wolfe, P. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 2019/09/05 1956. doi: 10.1002/nav.3800030109. URL https://doi.org/10.1002/nav.3800030109.

Freund, R. M., Grigas, P., and Mazumder, R. An extended frank–wolfe method with "in-face"directions, and its application to low-rank matrix completion. *SIAM Journal on Optimization*, 27(1):319–346, 2020/02/05 2017. doi: 10.1137/15M104726X. URL https://doi.org/10.1137/15M104726X.

Garber, D. and Hazan, E. A linearly convergent variant of the conditional gradient algorithm under strong convexity, with applications to online and stochastic optimization. *SIAM Journal on Optimization*, 26(3):1493–1528, 2020/01/01 2016. doi: 10.1137/140985366. URL https://doi.org/10.1137/140985366.

Guélat, J. and Marcotte, P. Some comments on wolfe's 'away step'. *Mathematical Programming*, 35(1):110–119, 1986. doi: 10.1007/BF01589445. URL https://doi.org/10.1007/BF01589445.

Gutman, D. and Peña, J. F. The condition number of a function relative to a set. *Mathematical Programming*, 2020. doi: 10.1007/s10107-020-01510-. URL https://doi.org/10.1007/s10107-020-01510-4.

Harchaoui, Z., Juditsky, A., and Nemirovski, A. Conditional gradient algorithms for norm-regularized smooth convex optimization. *Mathematical Programming*, 152(1):75–112, 2015. doi: 10.1007/s10107-014-0778-9. URL https://doi.org/10.1007/s10107-014-0778-9.

Harmany, Z. T., Marcia, R. F., and Willett, R. M. This is spiral-tap: Sparse poisson intensity reconstruction algorithms—theory and practice. *IEEE Transactions on Image Processing*, 21(3):1084–1096, 2011.

Hazan, E. Sparse approximate solutions to semidefinite programs. In Laber, E. S., Bornstein, C., Nogueira, L. T., and Faria, L. (eds.), *LATIN 2008: Theoretical Informatics*, pp. 306–316, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-78773-0.

Hazan, E. and Arora, S. *Efficient algorithms for online convex optimization and their applications*. Princeton University Princeton, 2006. ISBN 0542747340.

Jaggi, M. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning*, pp. 427–435, 2013.

Lacoste-Julien, S. and Jaggi, M. On the global linear convergence of frank-wolfe optimization variants. In *Advances in Neural Information Processing Systems*, pp. 496–504, 2015.

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

Levitin, E. S. and Polyak, B. T. Constrained minimization methods. *USSR Computational Mathematics and Mathematical Physics*, 6(5):1–50, 1966. doi: https://doi.org/10.1016/0041-5553(66)90114-5. URL http://www.sciencedirect.com/science/article/pii/0041555366901145.

Liu, D., Cevher, V., and Tran-Dinh, Q. A newton frank-wolfe method for constrained self-concordant minimization. *arXiv:2002.07003*, 2020.

Lu, H., Freund, R., and Nesterov, Y. Relatively smooth convex optimization by first-order methods, and applications. *SIAM Journal on Optimization*, 28(1):333–354, 2019/03/21 2018. doi: 10.1137/16M1099546. URL https://doi.org/10.1137/16M1099546.

Marteau-Ferey, U., Ostrovskii, D., Bach, F., and Rudi, A. Beyond least-squares: Fast rates for regularized empirical risk minimization through self-concordance. *arXiv preprint arXiv:1902.03046*, 2019.

Merhav, N. and Feder, M. Universal prediction. *IEEE Transactions on Information Theory*, 44(6):2124–2147, 1998.

Nesterov, Y. *Lectures on Convex Optimization*, volume 137 of *Springer Optimization and Its Applications*. Springer International Publishing, 2018a.

Nesterov, Y. Complexity bounds for primal-dual methods minimizing the model of objective function. *Mathematical Programming*, 171(1):311–330, 2018b. doi: 10.1007/s10107-017-1188-6. URL https://doi.org/10.1007/s10107-017-1188-6.

Nesterov, Y. and Nemirovskii, A. *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, 2016/08/20 1994. ISBN 978-0-89871-319-0. doi: doi:10.1137/1.9781611970791. URL http://dx.doi.org/10.1137/1.9781611970791.

Odor, G., Li, Y.-H., Yurtsever, A., Hsieh, Y.-P., Tran-Dinh, Q., El Halabi, M., and Cevher, V. Frank-wolfe works for non-lipschitz continuous gradient objectives: Scalable poisson phase retrieval. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6230–6234, 2016.

Ostrovskii, D. and Bach, F. Finite-sample analysis of m-estimators using self-concordance. *arXiv preprint arXiv:1810.06838*, 2018.

Owen, A. B. Self-concordance for empirical likelihood. *Canadian Journal of Statistics*, 41(3):387–397, 2020/02/05 2013. doi: 10.1002/cjs.11183. URL https://doi.org/10.1002/cjs.11183.

Pedregosa, F., Negiar, G., Askari, A., and Jaggi, M. Linearly convergent frank-wolfe without line-search. In Chiappa, S. and Calandra, R. (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 1–10, Online, 26–28 Aug 2020. PMLR. URL http://proceedings.mlr.press/v108/pedregosa20a.html.

Peña, J. and Rodríguez, D. Polytope conditioning and linear convergence of the frank–wolfe algorithm. *Mathematics of Operations Research*, 44(1):1–18, 2020/01/07 2018. doi: 10.1287/moor.2017.0910. URL https://doi.org/10.1287/moor.2017.0910.

Speed, T. P. and Kiiveri, H. T. Gaussian markov distributions over finite graphs. *The Annals of Statistics*, pp. 138–150, 1986.

Sun, T. and Tran-Dinh, Q. Generalized self-concordant functions: a recipe for newton-type methods. *Mathematical Programming*, 2018. doi: 10.1007/s10107-018-1282-4. URL https://doi.org/10.1007/s10107-018-1282-4.

Tran-Dinh, Q., Kyrillidis, A., and Cevher, V. An inexact proximal path-following algorithm for constrained convex minimization. *SIAM Journal on Optimization*, 24 (4):1718–1745, 2014. doi: 10.1137/130944539. URL https://doi.org/10.1137/130944539.

Tran-Dinh, Q., Kyrillidis, A., and Cevher, V. Composite self-concordant minimization. *The Journal of Machine Learning Research*, 16(1):371–416, 2015.

Zeng, X. and Figueiredo, M. A. Decreasing weighted sorted ℓ_1 regularization. *IEEE Signal Processing Letters*, 21 (10):1240–1244, 2014.