Mathematical Programming (2023) 198:255–323
https://doi.org/10.1007/s10107-022-01771-1

FULL LENGTH PAPER

Series A

Generalized self-concordant analysis of Frank–Wolfe algorithms

Pavel Dvurechensky[1] · Kamil Safin[2] · Shimrit Shtern[3] · Mathias Staudigl[4]

Received: 2 October 2020 / Accepted: 5 January 2022 / Published online: 29 January 2022

Abstract

Projection-free optimization via different variants of the Frank–Wolfe method has become one of the cornerstones of large scale optimization for machine learning and computational statistics. Numerous applications within these fields involve the minimization of functions with self-concordance like properties. Such generalized self-concordant functions do not necessarily feature a Lipschitz continuous gradient, nor are they strongly convex, making them a challenging class of functions for first-order methods. Indeed, in a number of applications, such as inverse covariance estimation or distance-weighted discrimination problems in binary classification, the loss is given by a generalized self-concordant function having potentially unbounded curvature. For such problems projection-free minimization methods have no theoretical convergence guarantee. This paper closes this apparent gap in the literature by developing provably convergent Frank–Wolfe algorithms with standard $\mathcal{O}(1/k)$ convergence rate guarantees. Based on these new insights, we show how these sublinearly convergent methods can be accelerated to yield linearly convergent projection-free

✉ Mathias Staudigl
m.staudigl@maastrichtuniversity.nl

Pavel Dvurechensky
Pavel.Dvurechensky@wias-berlin.de

Kamil Safin
kamil.safin@phystech.edu

Shimrit Shtern
shimrits@technion.ac.il

1 Weierstrass Institute for Applied Analysis and Stochastics, Mohrenstr. 39, 10117 Berlin, Germany

2 Moscow Institute of Physics and Technology, Dolgoprudny, Russia

3 Faculty of Industrial Engineering and Management, Technion - Israel Institute of Technology, 3200003 Technion City, Haifa, Israel

4 Department of Data Science and Knowledge Engineering, Maastricht University, P.O. Box 616, 6200 MD Maastricht, The Netherlands

methods, by either relying on the availability of a local liner minimization oracle, or a suitable modification of the away-step Frank–Wolfe method.

Keywords Generalized self-concordant functions · Frank–Wolfe algorithm · Convex programming

Mathematics Subject Classification 90C25 · 65K05 · 90C06

1 Introduction

Statistical analysis using generalized self-concordant (GSC) functions as a loss function is gaining increasing attention in the machine learning community [2,45,46,50]. Beyond machine learning, GSC loss functions are also used in image analysis [44] and quantum state tomography [33]. This class of loss functions allows to obtain faster statistical rates similar to least-squares [37]. At the same time, the minimization of empirical risk in this setting is a challenging optimization problem in high dimensions. Thus, without knowledge of specific structure, interior point, or other polynomial time methods, are unappealing. Moreover, large-scale optimization models in machine learning often depend on noisy data and thus high-accuracy solutions are not really needed, or obtainable. All these features make simple optimization algorithms, with low implementation costs, the preferred methods of choice. In this paper we focus on projection-free methods which rely on the availability of a Linear Minimization Oracle (LMO). Such algorithms are known as Conditional Gradient (CG) or Frank–Wolfe (FW) methods. These classes of gradient-based algorithms belong to the oldest convex optimization tools, and their origins can be traced back to [22,32]. For a given convex compact set $\mathcal{X} \subset \mathbb{R}^n$, and a convex objective function f, FW methods solve the smooth convex optimization problem

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

by sequential calls of a LMO, returning at point x the target state

$$s(x) \in \arg\min_{d \in \mathcal{X}} \langle \nabla f(x), d \rangle. \tag{1}$$

The selection $s(x)$ is determined via some pre-defined tie breaking rule, whose specific form is of no importance for the moment. Computing this target state is the only computational bottleneck of the method. Progress of the algorithm is monitored via a merit function. The standard merit function in this setting is the *Frank–Wolfe (dual) gap*

$$\mathsf{Gap}(x) \triangleq \max_{s \in \mathcal{X}} \langle \nabla f(x), x - s \rangle. \tag{2}$$

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 (P). The vanilla implementation of FW (Algorithm 1) aims to reduce the gap function by sequentially solving linear minimization subproblems to obtain

Algorithm 1: `FW-Standard` and `FW-Line Search`

Input: $x^0 \in \operatorname{dom} f \cap \mathcal{X}$ initial state; $\varepsilon > 0$ tolerance level
for $k = 1, \ldots$ **do**
  if $\mathsf{Gap}(x^k) > \varepsilon$ **then**
    Obtain $s^k = s(x^k)$
    Chose $\alpha_k = \frac{2}{k+2}$ (`FW-Standard`), or via exact line search (`FW-Line Search`)

$$\alpha_k = \operatorname{argmin}_{t \in [0,1]} f((1-t)x^k + ts^k). \tag{3}$$

    Update $x^{k+1} = x^k + \alpha_k(s^k - x^k)$.
  end if
end for

the target point $s(x)$. As always, the general performance of an algorithm depends heavily on the availability of practical step-size policies $\{\alpha_k\}_{k\in\mathbb{N}}$. Two popular choices are either $\alpha_k = \frac{2}{k+2}$ (`FW-Standard`), or an exact line-search (`FW-Line Search`). Under either choice, the algorithm exhibits an $\mathcal{O}(1/k)$ rate of convergence for solving (P) in case where f is convex and either possess a Lipschitz continuous gradient, or a bounded curvature constant. The latter concept is a slight weakening of the classical Lipschitz gradient assumption, and is the key quantity in the modern analysis of FW due to Jaggi [28]. The *curvature constant* is defined as

$$\kappa_f \triangleq \sup_{x,s\in\mathcal{X},t\in[0,1]} \frac{2}{t^2}\left[f(x + t(s-x)) - f(x) - t\langle \nabla f(x), s - x\rangle\right].$$

Assuming that $\kappa_f < \infty$, [28] estimated the iteration complexity of Algorithm 1 to be $\mathcal{O}(1)\frac{\kappa_f \operatorname{diam}(\mathcal{X})}{\varepsilon}$. This iteration complexity is in fact optimal [30], even when f is strongly convex. This is quite surprising, since gradient methods are known to display linear convergence on *well-conditioned* optimization problems, i.e. when the objective function is strongly convex with a Lipschitz continuous gradient [41].

Frank–Wolfe for ill-conditioned functions. In this paper we are interested in functions which are possibly *ill-conditioned*: f is neither assumed to be globally strongly convex, nor to posses a Lipschitz continuous gradient over the feasible set. Recently, many empirical risk minimization problems have been identified to be ill-conditioned, or at least nearly so [36,37,45]. This explains why the study of algorithms for this challenging class of problems received a lot of attention recently. The role of self-concordance-like properties of loss functions has been clarified in the influential seminal work by Bach [2]. Since then, numerous papers at the intersection between statistics, machine learning and optimization, exploited the self-concordance like behavior of typical statistical loss function to improve existing statistical rate estimates [37,45,46], or to improve the practical performance and theoretical guarantees of optimization algorithms [8,16,19,20,51–53]. Besides applications in statistics, generalized self-concordant functions are of some importance in scientific computing. [54] construct self-concordant barriers for a class of polytopes arising naturally in

combinatorial optimization. [50] show that the well-known matrix balancing problem minimizes a GSC function. We believe that our results are going to be useful in such problems as well.
The main difficulties one faces in minimizing functions with self-concordance like properties can be easily illustrated with a basic, in some sense minimal, example:

Example 1 Consider the function $f(x, y) = -\ln(x) - \ln(y)$ where $x, y > 0$ satisfy $x + y = 1$. This function is the standard self-concordant barrier for the positive orthant (the log-barrier) and thus $(2, 3)$-generalized self-concordant (see Definition 1). Its Bregman divergence is easily calculated as

$$D_f(u, v) = \sum_{i=1}^{2} \left[-\ln\left(\frac{u_i}{v_i}\right) + \frac{u_i}{v_i} - 1 \right] \quad u = (u_1, u_2), v = (v_1, v_2).$$

Neither the function f, nor its gradient, is Lipschitz continuous over the set of interest. In particular the curvature constant is unbounded, i.e. $\kappa_f = \infty$. Moreover, if we start from $u^0 = (1/4, 3/4)$ and apply the standard $2/(k+2)$-step size policy, then $\alpha_0 = 1$, which leads to $u^1 = s(u^0) = (1, 0) \notin \operatorname{dom} f$. Clearly, the standard method fails. ♦

The logarithm is one of the canonical members of (generalized) self-concordant functions, and thus the above example is quite representative for the class of optimization problems of interest in this paper. It is therefore clear that the standard analysis of [28], and all subsequent investigations relying on estimates of the Lipschitz constant of the gradient or the curvature, cannot be applied straightforwardly to the problem of minimizing a GSC function via projection-free methods.

1.1 Related literature

The development of FW methods for ill-conditioned problems has received quite some attention recently. [40] requires the gradient of the objective function to be Hölder continuous and similar results for this setting are obtained in [7,49]. Implicitly it is assumed that $\mathcal{X} \subseteq \operatorname{dom} f$. This would also not be satisfied in important GSC minimization problems, and hence we do not impose it (e.g. $0 \in \mathcal{X}$, but $0 \notin \operatorname{dom} f$ in the Covariance Estimation problem in Sect. 6.4). Specialized to solving a quadratic Poisson inverse problem in phase retrieval, [44] provided a globally convergent FW method using the convex and *self-concordant (SC)* reformulation, based on the PhaseLift approach [9]. They constructed a provably convergent FW variant using a new step size policy derived from estimate sequence techniques [3,39], in order to match the proof technique of [40].

Very recently, two other FW-methods for ill-conditioned problems appeared. [34] employed a FW-subroutine for computing the Newton step in a proximal Newton framework for minimizing self-concordant (SC)-functions over a convex compact set. After the first submission of this work, Professor Robert M. Freund sent us the preprint [57], in which the SC-FW method from our previous conference paper [17] is refined to

minimize a logarithmically homogeneous barrier [42] over a convex compact set. They also propose new stepsizes for FW for minimizing functions with Hölder continuous gradient. None of these recent contributions develop FW methods for the much larger class of GSC-functions, nor do they consider linearly convergent variants.

Linearly convergent Frank–Wolfe methods Given their slow convergence, it is clear that the application of projection-free methods can only be interesting if projections onto the feasible set are computationally expensive. Various previous papers worked out conditions under which the iteration complexity of projection-free methods can be potentially improved. [25] obtained linear convergence rates in well conditioned problems 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 of the solution from the boundary (see also [5,21]). 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 [23,25,26,48,55]. The Away-Step Frank Wolfe (ASFW) method can remove weight from "bad" atoms in the active set. These *drop steps* have the potential to circumvent the well-known zig-zagging phenomenon of FW when the solution lies on the boundary of the feasible set. When the feasible set $\mathcal{X}$ is a polytope, [29] derived linear convergence rates for ASFW using the "pyramidal width constant" in the well-conditioned optimization case. Unfortunately, the pyramidal width is the optimal value of a complicated combinatorial optimization problem, whose value is unknown even on simple sets such as the unit simplex. [4] improved their construction by replacing the pyramidal width with a much more tractable gradient bound condition, involving the "vertex-facet distance". In many instances, including the unit simplex, the ℓ_1-ball and the ℓ_∞-ball, the vertex-facet distance can be computed (see Section 3.4 in [4]). In this paper we develop a corresponding away-step FW variant for the minimization of a GSC function (Algorithm 8 (`ASFWGSC`)), extending [4] to ill-conditioned problems.

While we were working on the revision of this paper, Professor Sebastian Pokutta shared with us the recent preprint [10], where a monotone modification of `FW-Standard` applied to GSC-minimization problems is proposed. They derive a $\mathcal{O}(1/k)$ convergence rate guarantee for minimizing GSC functions. Moreover, they exhibit a linearly convergent variant using away-steps. These results have been achieved independently from our work, and they give a nice complementary view on our away-step variant `ASFWGSC`. The basic difference between our analysis and [10] is that we exploit the vertex-facet distance instead of the pyramidal width. As already said, this gives explicit and efficiently computable error bounds for some important geometries, and thus allows for a more in-depth complexity assessment.

The alternative twist to obtain linear convergence is to change the design of the LMO [24,27,30] via a well-calibrated localization procedure. Extending the work by Garber and Hazan [24], we construct another linearly convergent FW-variant based on *local* linear minimization oracles (Algorithm 7, `FWLLOO`).

1.2 Main contributions and outline of the paper

In this paper, we demonstrate that projection-free methods extend to a large class of potentially ill-conditioned convex programming problems, featuring self-concordant like properties. Our main contributions can be succinctly summarized as follows:

(i) *Ill-Conditioned problems* We construct a set of globally convergent projection-free methods for minimizing generalized self-concordant functions over convex compact domains.
(ii) *Detailed Complexity analysis* Algorithms with sublinear and linear convergence rate guarantees are derived.
(iii) *Adaptivity* We develop new backtracking variants in order to come up with new step size policies which are adaptive with respect to local estimates of the gradient's Lipschitz constant, or basic parameters related to the self-concordance properties of the objective function. The construction of these backtracking schemes fully exploits the basic properties of GSC-functions. Specifically, Algorithm 3 (`LBTFWGSC`) builds on a standard quadratic upper model over which a local search for the Lipschtiz modulus of the gradient, restricted to level sets, can be performed. This local search method is inspired by [47], but our convergence proof is much simpler and direct. Our second backtracking variant (Algorithm 5, `MBTFWGSC`) performs a local search for the generalized self-concordance constant. To the best of our knowledge this is the first algorithm which adaptively adjusts the self-concordance parameters on-the-fly. We thus present three new sublinearly converging FW-variants which are all adaptive, and share the standard *sublinear* $\mathcal{O}(1/\varepsilon)$ complexity bound which is proved in Sect. 4. On top of that, we derive two new linearly converging schemes, either building on the availability of Local Linear Optimization Oracle (LLOO) (Algorithm 7 (`FWLLOO`)), or suitably defined Away-Steps (Algorithm 8 (`ASFWGSC`)).
(iv) *Detailed Numerical experiments* We test the performance of our method on a set of challenging test problems, spanning all possible GSC parameters over which our algorithms are provably convergent.

This paper builds on, and significantly extends, our conference paper [17]. This previous work exclusively focused on the minimization of standard self-concordant functions. The extension to generalized self-concordant functions requires some careful additional steps and a detailed case-by-case analysis that are not simple corollaries of [17]. On top of that, in this paper we derive two completely new projection-free algorithms, and new proofs of existing algorithms we already introduced in our first publication. In light of these contributions, this paper significantly extends the results reported in [17].

Outline Section 2 contains necessary definitions and properties for the class of GSC functions in a self-contained way. Our algorithmic analysis starts in Sect. 3 where a new FW variant with an analytic step-size rule is presented (Algorithm 2, `FWGSC`). This algorithm can be seen as the basic template from which the other methods are subsequently derived. Section 4 presents the convergence analysis for the three sublinearly convergent variants presented in Sect. 3. Section 5 presents the

two linearly convergent variants and their convergence analysis. Section 6 reports results from extensive numerical experiments using the proposed algorithms and their comparison with the baselines. Section 7 concludes the paper.

Notation Given a proper, closed, and convex function $f : \mathbb{R}^n \to (-\infty, \infty]$, we denote by $\operatorname{dom} f \triangleq \{x \in \mathbb{R}^n | f(x) < \infty\}$ the (effective) domain of f. For a set X, we define the indicator function $\delta_X(x) = \infty$ if $x \notin X$, and $\delta_X(x) = 0$ otherwise. We use $\mathbf{C}^k(\operatorname{dom} f)$ to denote the class of functions $f : \mathbb{R}^n \to (-\infty, \infty]$ which are k-times continuously differentiable on their effective domain. We denote by ∇f the gradient map, and $\nabla^2 f$ the Hessian map.

Let $\mathbb{R}_+$ and $\mathbb{R}_{++}$ denote the set of nonnegative, and positive real numbers, respectively. We use $\mathbb{S}^n \triangleq \{x \in \mathbb{R}^{n\times n} | x^\top = x\}$ the set of symmetric matrices, and $\mathbb{S}^n_+, \mathbb{S}^n_{++}$ to denote the set of symmetric positive semi-definite and positive definite matrices, respectively. Given $Q \in \mathbb{S}^n_{++}$ we define the weighted inner product $\langle u, v\rangle_Q \triangleq \langle Qu, v\rangle$ for $u, v \in \mathbb{R}^n$, and the corresponding norm $\|u\|_Q \triangleq \sqrt{\langle u, u\rangle_Q}$. The associated dual norm is $\|v\|^*_Q \triangleq \sqrt{\langle v, v\rangle_{Q^{-1}}}$. For $Q \in \mathbb{S}^n$, we let $\lambda_{\min}(Q)$ and $\lambda_{\max}(Q)$ denote the smallest and largest eigenvalues of the matrix Q, respectively.

2 Generalized self-concordant functions

Following [50], we briefly introduce the basic properties of the class of GSC functions. Let $\varphi : \mathbb{R} \to \mathbb{R}$ be a three-times continuously differentiable function on $\operatorname{dom} \varphi$. Recall that φ is convex if and only if $\varphi''(t) \geq 0$ for all $t \in \operatorname{dom} \varphi$.

Definition 1 [50] Let $\varphi \in \mathbf{C}^3(\operatorname{dom} \varphi)$ be a convex function with $\operatorname{dom} \varphi$ open. Given $\nu > 0$ and $M_\varphi > 0$ some constants, we call φ (M_φ, ν) *generalized self-concordant* (GSC) if

$$|\varphi'''(t)| \leq M_\phi \varphi''(t)^{\frac{\nu}{2}} \qquad \forall t \in \operatorname{dom} \varphi. \tag{4}$$

If $\varphi(t) = \frac{a}{2}t^2 + bt + c$ for any constant $a \geq 0$ we get a $(0, \nu)$-generalized self-concordant function. Hence, any convex quadratic function is GSC for any $\nu > 0$. Standard one-dimensional examples are summarized in Table 1 (based on [50]).

This definition generalizes to multivariate functions by requiring GSC along every straight line. Specifically, let $f : \mathbb{R}^n \to (-\infty, +\infty]$ be a closed convex, lower

Table 1 Examples of univariate GSC functions (based on [50])

Function name	Form of $\varphi(t)$	ν	M_φ	$\operatorname{dom} \varphi$	Lipschitz smooth
Burg entropy	$-\ln(t)$	3	2	$(0, \infty)$	No
Logistic	$\ln(1 + e^{-t})$	2	1	$(-\infty, \infty)$	Yes
Exponential	e^{-t}	2	1	$(-\infty, \infty)$	No
Negative power	$t^{-q}, q > 0$	$\frac{2(q+3)}{q+2}$	$\frac{q+2}{q+\sqrt[2]{q(q+1)}}$	$(0, \infty)$	No
Arcsine distribution	$\frac{1}{\sqrt{1-t^2}}$	$\frac{14}{5}$	< 3.25	$(-1, 1)$	No

semi-continuous function with effective domain dom f which is an open nonempty subset of $\mathbb{R}^n$. For $x \in \text{dom } f$ and $u, v \in \mathbb{R}^n$, define the real-valued function $\varphi(t) := \langle \nabla^2 f(x + tv)u, u\rangle$. For $t \in \text{dom}\,\varphi$, one sees that $\phi'(t) = \langle D^3 f(x + tv)[v]u, u\rangle$, where $D^3 f(x)[v]$ denotes the third-derivative tensor at (x, v), viewed as a bilinear mapping $\mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$. The Hessian of the function f defines a semi-norm $\|u\|_x \triangleq \sqrt{\langle u, u\rangle_{\nabla^2 f(x)}}$ for all $x \in \text{dom } f$, with dual norm $\|a\|_x^* \triangleq \sup_{d\in\mathbb{R}^n}\{2\langle d, a\rangle - \|d\|_x^2\}$. If $\nabla^2 f(x) \in \mathbb{S}^n_{++}$ then $\|\cdot\|_x$ is a true norm, and $\|d\|_x^* = \sqrt{\langle d, d\rangle_{[\nabla^2 f(x)]^{-1}}}$.

Definition 2 [50] A closed convex function $f \in \mathbf{C}^3(\text{dom } f)$, with dom f open, is called (M_f, ν) generalized self-concordant of the order $\nu \in [2, 3]$ and with constant $M_f \geq 0$, if for all $x \in \text{dom } f$

$$|\langle D^3 f(x)[v]u, u\rangle| \leq M_f \|u\|_x^2 \|v\|_x^{\nu-2} \|v\|_2^{3-\nu} \qquad \forall u, v \in \mathbb{R}^n. \tag{5}$$

We denote this class of functions as $\mathcal{F}_{M_f,\nu}$.

In the extreme case $\nu = 2$ we recover the definition $|\langle D^3 f(x)[v]u, u\rangle| \leq M_f \|u\|_x^2 \|v\|_2$, which is the generalized self-concordance definition proposed by Bach [2]. If $\nu = 3$ and $u = v$ the definition becomes $|\langle D^3 f(x)[u]u, u\rangle| \leq M_f \|u\|_x^3$, which is the standard self-concordance definition due to [42].
Given $\nu \in [2, 3]$ and $f \in \mathcal{F}_{M_f,\nu}$, we define the distance-like function

$$\mathsf{d}_\nu(x, y) \triangleq \begin{cases} M_f \|y - x\|_2 & \text{if } \nu = 2, \\ \frac{\nu-2}{2} M_f \|y - x\|_2^{3-\nu} \cdot \|y - x\|_x^{\nu-2} & \text{if } \nu \in (2, 3], \end{cases} \tag{6}$$

and the *Dikin Ellipsoid*

$$\mathcal{W}(x; r) \triangleq \{y \in \mathbb{R}^n : \mathsf{d}_\nu(x, y) < r\} \quad \forall (x, r) \in \text{dom } f \times \mathbb{R}. \tag{7}$$

Since $f \in \mathcal{F}_{M_f,\nu}$ are closed convex functions with open domain, it follows that they are *barrier functions* for dom f: Along any sequence $\{x_n\}_{n\in\mathbb{N}} \subset \text{dom } f$ with dist $(x_n, \text{bd}(\text{dom } f)) \to 0$ we have $f(x_n) \to \infty$. This fact allows us to use the Dikin Ellipsoid as a safeguard region within which we can perturb the current position x without falling off dom f.

Lemma 1 ([50], Prop. 7) *Let $f \in \mathcal{F}_{M_f,\nu}$ with $\nu \in (2, 3]$. We have $\mathcal{W}(x; 1) \subset \text{dom } f$ for all $x \in \text{dom } f$.*

The inclusion $\mathcal{W}(x; 1) \subset \text{dom } f$ for $\nu \in (2, 3]$ is a generalization of a well-known classical property of self-concordant functions [42]. It gains relevance for the case $\nu > 2$, since when $\nu = 2$, we have dom $f = \mathbb{R}^n$, making the statement trivial.
The next Lemma gives a-priori local bounds on the function values.

Lemma 2 ([50], Prop. 10) *Let $x, y \in \text{dom } f$ for $f \in \mathcal{F}_{M_f,\nu}$ and $\nu \in [2, 3]$. Then*

$$f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle + \omega_\nu(-\mathsf{d}_\nu(x, y))\|y - x\|_x^2, \text{ and} \tag{8}$$

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \omega_\nu(d_\nu(x, y))\|y - x\|_x^2, \tag{9}$$

where, if $\nu > 2$, the right-hand side of (9) holds if and only if $d_\nu(x, y) < 1$. Here $\omega_\nu(\cdot)$ is defined as

$$\omega_\nu(t) \triangleq \begin{cases} \frac{1}{t^2}(e^t - t - 1) & \text{if } \nu = 2, \\ \frac{-t - \ln(1-t)}{t^2} & \text{if } \nu = 3, \\ \left(\frac{\nu-2}{4-\nu}\right)\frac{1}{t}\left[\frac{\nu-2}{2(3-\nu)t}((1-t)^{\frac{2(3-\nu)}{2-\nu}} - 1) - 1\right] & \text{if } \nu \in (2, 3). \end{cases} \tag{10}$$

The function $\omega_\nu(\cdot)$ is strictly convex and one can check that $\omega_\nu(t) \geq 0$ for all $t \in \mathrm{dom}(\omega_\nu)$. These bounds on the function values can be seen as local versions of the standard approximations valid for strongly convex functions, respectively for functions with a Lipschitz continuous gradient (see e.g. [41], Def. 2.1.3 and Lemma 1.2.3). In particular, the upper bound (9) corresponds to a local version of the celebrated *descent lemma*, a fundamental tool in the analysis of first-order methods [18]. To emphasize this analogy, we will also refer to (9) as the *GSC-descent lemma*.

3 Frank–Wolfe works for generalized self-concordant functions

In this section we describe three provably convergent modifications of Algorithm 1, displaying sublinear convergence rates.

3.1 Preliminaries

Assumption 1 The following assumptions shall be in place throughout this paper:

- The function f in (P) belongs to the class $\mathcal{F}_{M_f,\nu}$ with $\nu \in [2, 3]$.
- The solution set $\mathcal{X}^*$ of (P) is nonempty, with $x^* \in \mathcal{X}^*$ representing a solution and $f^* = f(x^*)$ the corresponding objective function value.
- $\mathcal{X}$ is convex compact and the search direction (1) can be computed efficiently and accurately.
- $\nabla^2 f$ is continuous and positive definite on $\mathcal{X} \cap \mathrm{dom}\, f$.

Define the *Frank–Wolfe search direction* as

$$v_{\mathrm{FW}}(x) \triangleq s(x) - x. \tag{11}$$

We also declare the functions $\epsilon(x) \triangleq \|v_{\mathrm{FW}}(x)\|_x$ and $\beta(x) \triangleq \|v_{\mathrm{FW}}(x)\|_2$ for all $x \in \mathrm{dom}\, f$.

3.2 A Frank–Wolfe method with analytical step-size

Our first Frank–Wolfe method (Algorithm 2, `FWGSC`) for minimizing generalized self-concordant functions builds on a new adaptive step-size rule, which we derive from

Algorithm 2: FWGSC

Input: $x^0 \in \operatorname{dom} f \cap \mathcal{X}$ initial state, $\varepsilon > 0$ error tolerance, and $f \in \mathcal{F}_{M,\nu}$.
for $k = 0, \ldots$ **do**
  if $\mathsf{Gap}(x^k) > \varepsilon$ **then**
    Obtain $s^k = s(x^k)$ from (1) and define $v^k = v_{\mathrm{FW}}(x^k)$;
    Obtain $\alpha_k = \alpha_v(x^k)$ from (15);
    Set $x^{k+1} = x^k + \alpha_k v^k$
  end if
end for

a judicious application of the GSC-descent Lemma (9). An attractive feature of this new step size policy is that it is available in analytical form, which allows us to do away with any globalization strategy (e.g. line search). This has significant practical impact when function evaluations are expensive.
Given $x \in \mathcal{X}$, set $x_t^+ \triangleq x + t v_{\mathrm{FW}}(x)$, and assume that $\mathsf{e}(x) \neq 0$. Moving from the current position x to the point x_t^+, we know that $\mathsf{d}_\nu(x, x_t^+) = t M_f \delta_\nu(x)$, where

$$\delta_\nu(x) \triangleq \begin{cases} \beta(x) & \text{if } \nu = 2, \\ \frac{\nu-2}{2}\beta(x)^{3-\nu}\mathsf{e}(x)^{\nu-2} & \text{if } \nu > 2. \end{cases} \tag{12}$$

Choosing $t \in (0, \frac{1}{M_f \delta_\nu(x)})$, the GSC-descent lemma (9) gives us the upper bound

$$\begin{aligned} f(x_t^+) &\leq f(x) + \langle \nabla f(x), x_t^+ - x\rangle + \omega_\nu(\mathsf{d}_\nu(x, x_t^+))\|x_t^+ - x\|_x^2 \\ &= f(x) + \langle \nabla f(x), x_t^+ - x\rangle + \omega_\nu\left(t M_f \delta_\nu(x)\right) t^2 \mathsf{e}(x)^2 \\ &= f(x) - t\,\mathsf{Gap}(x) + \omega_\nu\left(t M_f \delta_\nu(x)\right) t^2 \mathsf{e}(x)^2. \end{aligned}$$

For $x \in \operatorname{dom} f \cap \mathcal{X}$, define $\eta_{x,M,\nu} : \mathbb{R}_+ \to (-\infty, +\infty]$ by

$$\eta_{x,M,\nu}(t) \triangleq \mathsf{Gap}(x)\left[t - \omega_\nu\left(t M \delta_\nu(x)\right) t^2 \frac{\mathsf{e}(x)^2}{\mathsf{Gap}(x)}\right]. \tag{13}$$

Note that $\eta_{x,M,\nu}(t)$ is strictly concave on $\operatorname{dom}(\eta_{x,M,\nu}) \subseteq [0, \frac{1}{M\delta_\nu(x)}]$. This leads to the per-iteration change in the objective function value as

$$f(x_t^+) - f(x) \leq -\eta_{x,M_f,\nu}(t) \qquad \forall t \in (0, \frac{1}{M_f \delta_\nu(x)}).$$

Since $\eta_{x,M_f,\nu}(t) > 0$ for $t \in (0, \frac{1}{M_f\delta_\nu(x)})$, we are ensured that we make progress in reducing the objective function value when choosing a step size within the indicated range. Given the triple (x, M, ν), we search for a value t such that the per-iteration decrease is as big as possible. Hence, we aim to find $t \geq 0$ which solves the concave maximization problem

$$\sup_{t \geq 0} \eta_{x,M,\nu}(t). \tag{14}$$

Call $\mathtt{t}_{M,\nu}(x)$ a solution of this program. Since we have to stay within the feasible set, we cannot simply use the number $\mathtt{t}_{M,\nu}(x)$ as our step size as it might lead to an infeasible point. Consequently, we propose the truncated step-size

$$\alpha_{M,\nu}(x) \triangleq \min\left\{1, \mathtt{t}_{M,\nu}(x)\right\} \quad \forall x \in \operatorname{dom} f. \tag{15}$$

In Sect. 4 we show that this step-size policy guarantees feasibility and a sufficient decrease.

Remark 1 We emphasize that the basic step-size rule is derived by identifying a suitable local majorizing model $f(x) - \eta_{x,M_f,\nu}(t)$. Minimization with respect to t aligns the model as close as possible to the effective progress we are making in reducing the objective function value. This upper model holds for all GSC functions with the same characteristic parameter (M_f, ν). Thus, our derived step size strategy is universally applicable to all functions within the class $\mathcal{F}_{M_f,\nu}$. Therefore, akin to [50,52], the derived adaptive step size policy can be regarded as an optimal choice in the analytic worst-case sense.

3.3 Backtracking Frank–Wolfe variants

Algorithm `FWGSC` comes with several drawbacks. First, it relies on the minimization of a universal upper model derived from the GSC-descent Lemma. This over-estimation strategy leads to a worst-case performance estimate, relying on various state-dependent quantities, such as the local norm $\in(x^k)$, and the GSC parameters (M_f, ν). Evaluating the local norm requires the computation of the matrix-vector product between the Hessian $\nabla^2 f(x^k)$, and the FW search direction $v_{\mathrm{FW}}(x^k)$.[1] The GSC parameter M_f is a global quantity, relating the second and third derivative over the entire domain of the function f. Additionally, it restricts the interval of admissible step sizes $(0, \frac{1}{M_f \delta_\nu(x)})$. Consequently, a local search for this parameter could lead to larger step-sizes, which may improve the performance. Motivated by these facts, this section presents two backtracking variants of the basic Frank–Wolfe method. Both methods are based on the assumption that we can easily answer the question whether a given candidate search point x belongs to the domain of the function f, or not.

Assumption 2 (Domain Oracle) Given a point x, it is easy to decide if $x \in \operatorname{dom} f$, or not.

Remark 2 For many problems such domain oracles are easy to construct. As a concrete example, consider the problem of minimizing the log-barrier function over a compact domain in $\mathbb{R}^n_+$, which is a standard routine in interior-point methods (e.g. the computation of the *analytic center*). For this problem, a simple domain oracle is a single pass through all the coordinates of the vector x and checking if each entry is positive. The complexity of such an oracle is linear in the number of variables.

[1] In fact, evaluating the local norm requires the Hessian matrix $\nabla^2 f(x)$, and thus `FWGSC` is actually second-order method. At the same time, no inversion of the Hessian is needed. For instance, the matrix-vector product can be efficiently computed when the objective belongs to the class of generalized linear models, where the Hessian is given as a sum of rank 1 matrices.

Algorithm 3: FWGSC with backtracking over the Lipschitz parameter (LBTFWGSC)

Input: $x^0 \in \operatorname{dom} f \cap \mathcal{X}$ initial state, $f \in \mathcal{F}_{M,\nu}$, $\mathcal{L}_{-1} > 0$ initial Lipschitz estimate, $\gamma_u > 1$, $\gamma_d < 1$ fixed scaling parameters for the backtracking routine.
for $k = 0, \ldots$ **do**
  if $\mathsf{Gap}(x^k) > \varepsilon$ **then**
    Obtain $s^k = s(x^k)$ and set $v^k = v_{\mathrm{FW}}(x^k)$
    Obtain $(\alpha_k, \mathcal{L}_k) = \mathtt{step}_L(f, v^k, x^k, \mathcal{L}_{k-1})$
    Update $x^{k+1} = x^k + \alpha_k v^k$
  end if
end for

Algorithm 4: Function $\mathtt{step}_L(f, v, x, \mathcal{L})$

Choose $\tilde{L} \in [\gamma_d \mathcal{L}, \mathcal{L}]$
$\alpha = \min\{1, \frac{\mathsf{Gap}(x)}{\tilde{L}\|v\|_2^2}\}$
if $x + \alpha v \notin \operatorname{dom} f$ or $f(x + \alpha v) > Q_L(x, \alpha, \tilde{L})$ **then**
  $\tilde{L} \leftarrow \gamma_u \tilde{L}$
  $\alpha \leftarrow \min\{\frac{\mathsf{Gap}(x)}{\tilde{L}\|v\|_2^2}, 1\}$
end if
Return $\alpha, \tilde{L}$

3.3.1 Backtracking over the Lipschitz constant

Our first backtracking variant of FWGSC preforms a local search over the Lipschitz modulus of the gradient over level sets. From the previous analysis we know that under an appropriate choice of the stepsize the algorithm is guaranteed to stay in the level set on which the function has Lipschitz gradient. Thus, we can appropriately modify and apply the Backtracking FW algorithm proposed in [47] for functions with Lipschitz-continuous gradient. The main difference is that we additionally check that the step is feasible w.r.t. dom f. Also our proof is both simpler and much more direct. We also remark that our algorithm is not only applicable to generalized self-concordant minimization, but also for other settings with locally Lipschitz gradient.

Consider the quadratic model

$$Q_L(x, t, \mathcal{L}) \triangleq f(x) - t\,\mathsf{Gap}(x) + \frac{t^2\mathcal{L}}{2}\|v_{\mathrm{FW}}(x)\|_2^2 = f(x) - t\,\mathsf{Gap}(x) + \frac{t^2\mathcal{L}}{2}\beta(x)^2, \tag{16}$$

where $x \in \mathcal{X}$ is the current position of the algorithm, and $t, \mathcal{L} > 0$ are parameters. From the complexity analysis of FWGSC, we know that there exists a range of step-size parameters $t > 0$ that guarantee decrease in the objective function value. Denote by $\mathcal{S}(x) \triangleq \{x' \in \mathcal{X} | f(x') \le f(x)\}$, and set $\gamma_k \triangleq \sup\{t > 0 | x^k + t(s^k - x^k) \in \mathcal{S}(x^k)\}$ as well as $L_k \triangleq \max_{x \in \mathcal{S}(x^k)} \lambda_{\max}^2(\nabla^2 f(x))$. Then, for all $t \in [0, \gamma_k]$, it holds true that $f(x^k + t(s^k - x^k)) \le f(x^k)$. Therefore, by the mean-value-theorem

$$\|\nabla f(x^k + t(s^k - x^k)) - \nabla f(x^k)\| \le L_k t \|s^k - x^k\|_2 \qquad \forall t \in (0, \gamma_k).$$

Algorithm 5: FWGSC with backtracking over the GSC parameter M_f (MBTFWGSC)

Input: $x^0 \in \operatorname{dom} f \cap \mathcal{X}$ initial state, $f \in \mathcal{F}_{M_f,\nu}$, $\mu_{-1} > 0$ initial GSC parameter. $\gamma_u > 1$, $\gamma_d < 1$ fixed scaling parameters for the backtracking routine.
for $k = 0, \ldots$ **do**
  if $\mathsf{Gap}(x^k) > \varepsilon$ **then**
    Obtain $s^k = s(x^k)$ and set $v^k = v_{\mathrm{FW}}(x^k)$
    Obtain $(\alpha_k, \mu_k) = \mathtt{step}_M(f, v^k, x^k, \mu_{k-1})$
    Update $x^{k+1} = x^k + \alpha_k v^k$
  end if
end for

Algorithm 6: Function $\mathtt{step}_M(f, v, x, \mu)$

Choose $\tilde{M} \in [\gamma_d \mu, \mu]$
$\alpha = \alpha_{\tilde{M},v}(x)$ defined in (15)
if $x + \alpha v \notin \operatorname{dom} f$ or $f(x + \alpha v) > Q_M(x, \alpha, \tilde{M})$ **then**
  $\tilde{M} \leftarrow \gamma_u \tilde{M}$
  $\alpha \leftarrow \alpha_{\tilde{M},v}(x)$
end if
Return α, $\tilde{M}$

Hence, for all $t \in (0, \gamma_k)$,

$$f(x^k + t(s^k - x^k)) - f(x^k) \leq -t\,\mathsf{Gap}(x^k) + \frac{L_k t^2}{2}\|s^k - x^k\|_2^2 = Q_L(x^k, t, L_k) - f(x^k), \tag{17}$$

The idea is to dispense with the computation of the local Lipschitz estimate L_k over the level set $\mathcal{S}(x^k)$, and replace it by the backtracking procedure $\mathtt{step}_L(f, v^k, x^k, \mathcal{L}_{k-1})$ (Algorithm 4) as an inner-loop within Algorithm 3 (LBTFWGSC). In particular, using Assumption 2, the implementation of LBTFWGSC does not require the evaluation of the Hessian matrix $\nabla^2 f(x^k)$, and simultaneously determines a step size which minimizes the quadratic model under the prevailing local Lipschitz estimate.

3.3.2 Backtracking over the GSC parameter M_f

Our second backtracking variant performs a local search for the GSC parameter M_f. Our goal is to construct a backtracking procedure for the constant M_f such that for a given candidate GSC parameter $\mu > 0$ and search point $x_t^+ = x + t v_{\mathrm{FW}}(x)$, we have feasibility: $x_t^+ \in \operatorname{dom} f$, and sufficient decrease:

$$f(x_t^+) \leq f(x) - t\,\mathsf{Gap}(x) + t^2 \mathsf{e}(x)^2 \omega_\nu(t\mu\delta_\nu(x)) \triangleq Q_M(x, t, \mu). \tag{18}$$

Optimizing the new upper model $Q_M(x, t, \mu)$ with respect to $t \geq 0$ yields a step-size $\mathtt{t}_{\mu,\nu}(x)$, whose definition is just like the maximizer in (14), but using the parameters (x, μ, ν) as input. This approach allows us to define a localized step-size, exploiting the analytic structure of the step-size policy associated with the base algorithm FWGSC.

The main merit of this backtracking method can be seen by revisiting the analytical step-size criterion attached with FWGSC, defined in eq. (15). Inspection of the definition of the function $\alpha_{M,\nu}(x)$ that a larger M cannot lead to a larger step size. Hence, a precise local estimate of the GSC parameter M opens up possibilities to make larger steps and thus improve the practical performance of the method. We will see in our numerical experiments in Sect. 6 that this claim has some substance in important machine learning problems.

4 Complexity analysis

4.1 Complexity analysis of FWGSC

Based on the preliminary discussion of Sect. 3.2, our strategy to determine the step-size policy is to first compute $\mathtt{t}_{M_f,\nu}(x)$, defined as the solution to program (14), and then clip the value accordingly. A technical analysis of the optimization problem (14), relegated to "Appendix B", yields the following explicit expression for $\mathtt{t}_{M_f,\nu}(x)$.

Proposition 1 *The unique solution to program* (14) *is given by*

$$\mathtt{t}_{M_f,\nu}(x) = \begin{cases} \frac{1}{M_f\delta_2(x)} \ln\left(1 + \frac{\mathit{Gap}(x) M_f \delta_2(x)}{\mathtt{e}(x)^2}\right) & \text{if } \nu = 2, \\ \frac{1}{M_f\delta_\nu(x)} \left[1 - \left(1 + \frac{M_f\delta_\nu(x)\,\mathit{Gap}(x)}{\mathtt{e}(x)^2} \frac{4-\nu}{\nu-2}\right)^{-\frac{\nu-2}{4-\nu}}\right] & \text{if } \nu \in (2,3), \\ \frac{\mathit{Gap}(x)}{M_f\delta_3(x)\,\mathit{Gap}(x) + \mathtt{e}(x)^2} & \text{if } \nu = 3. \end{cases} \tag{19}$$

where $\delta_\nu(x)$, $\nu \in [2,3]$, *is defined in eq.* (12).

Next we show that FWGSC is well-defined using the step size policy (15).

Proposition 2 *Let* $\{x^k\}_{k\geq 0}$ *be generated by* FWGSC *with step size policy* $\{\alpha_{M_f,\nu}(x^k)\}_{k\geq 0}$ *defined in* (15). *Then* $x^k \in \mathcal{X} \cap \operatorname{dom} f$ *for all* $k \geq 0$.

Proof The proof proceeds by induction. By assumption, $x^0 \in \operatorname{dom} f \cap \mathcal{X}$. To perform the induction step, assume that $x^k \in \mathcal{X} \cap \operatorname{dom} f$ for some $k \geq 0$. We consider two cases:

- If $\nu = 2$, then since $\alpha_{M_f,2}(x^k) \leq 1$, feasibility follows immediately from convexity of $\mathcal{X}$ (recall that $\operatorname{dom} f = \mathbb{R}^n$ in this case).
- If $\nu \in (2,3]$, then, whenever $x^k \in \mathcal{X}$, we deduce from (19) that $\mathtt{t}_{M_f,\nu}(x^k) M_f\delta_\nu(x^k) < 1$. If $\mathtt{t}_{M_f,\nu}(x^k) > 1$, then $\alpha_{M_f,\nu}(x^k) M_f\delta_\nu(x^k) = M_f\delta_\nu(x^k) < \mathtt{t}_{M_f,\nu}(x^k) M_f\delta_\nu(x^k) < 1$. The claim then follows thanks to Lemma 1.

□

In order to simplify the notation, let us introduce the sequences $\alpha_k \equiv \alpha_{M_f,\nu}(x^k)$ and $\Delta_k \equiv \eta_{x^k,M_f,\nu}(\alpha_{M_f,\nu}(x^k))$. Along the sequence $\{x^k\}_{k\geq 0}$, we have $\mathsf{d}_\nu(x^k, x^{k+1})$

$= M_f \alpha_k \delta_\nu(x^k) < 1$, and we know that we reduce the objective function value by at least the quantity $\Delta_k > 0$. Whence,

$$f(x^{k+1}) \leq f(x^k) - \Delta_k < f(x^k), \tag{20}$$

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

Lemma 3 *The set $\mathcal{S}(x^0)$ is compact.*

Proof $\mathcal{S}(x^0) \subseteq \mathcal{X}$ and therefore it is bounded. Moreover, since $x^0 \in \operatorname{dom} f \cap \mathcal{X}$, f is closed and convex and $\mathcal{X}$ is also closed. $\mathcal{S}(x^0)$ is closed as the intersection of two closed sets, and therefore compact. □

Accordingly, $\mathcal{S}(x^0) \subset \operatorname{dom}(f)$ and the numbers $L_{\nabla f} \triangleq \max_{x\in\mathcal{S}(x^0)} \lambda_{\max}(\nabla^2 f(x))$ and $\sigma_f \triangleq \min_{x\in\mathcal{S}(x^0)} \lambda_{\min}(\nabla^2 f(x))$ are well defined and finite. Furthermore, since the level set $\mathcal{S}(x^0)$ is compact, Assumption 1 guarantees $\nabla^2 f(x) \succ 0$ for all $x \in \mathcal{S}(x^0)$, and hence $\sigma_f > 0$. By [41, Thm.2.1.11], for any $x \in \mathcal{S}(x^0)$ it holds that

$$f(x) - f^* \geq \frac{\sigma_f}{2} \|x - x^*\|_2^2. \tag{21}$$

Proposition 3 below shows asymptotic convergence to a solution along subsequences. We omit the proof, as it follows from [17].

Proposition 3 *Suppose* Assumption 1 *holds. Then, the following assertions hold for* `FWGSC`*:*

(a) $\{f(x^k)\}_{k\geq 0}$ *is non-increasing;*
(b) $\sum_{k\geq 0} \Delta_k < \infty$*, and hence the sequence* $\{\Delta_k\}_{k\geq 0}$ *converges to 0;*
(c) *For all* $K \geq 1$ *we have* $\min_{0\leq k<K} \Delta_k \leq \frac{1}{K}(f(x^0) - f^*)$.

In order to assess the iteration complexity of `FWGSC`, we need a lower bound on the sequence $\{\Delta_k\}_{k\geq 0}$. We start with a bound at iterations satisfying $\mathtt{t}_{M_f,\nu}(x^k) > 1$.

Lemma 4 *If* $\mathtt{t}_{M_f,\nu}(x^k) > 1$*, we have* $\Delta_k \geq \frac{1}{2}\, \mathit{Gap}(x^k)$.

Proof See "Appendix C.1". □

Next, we turn to iterates for which $\mathtt{t}_{M_f,\nu}(x^k) \leq 1$. In this case, the per-iteration progress reads as $\Delta_k = \eta_{x^k,M_f,\nu}(\mathtt{t}_{M_f,\nu}(x^k))$, and enjoys the following lower bound:

Lemma 5 *If* $\mathtt{t}_{M_f,\nu}(x^k) \leq 1$*, we have*

$$\Delta_k \geq \tilde{\Delta}_k \triangleq \begin{cases} \frac{2\ln(2)-1}{\operatorname{diam}(\mathcal{X})} \min\left\{\frac{\mathit{Gap}(x^k)}{M_f}, \frac{\mathit{Gap}(x^k)^2}{\operatorname{diam}(\mathcal{X}) L_{\nabla f}}\right\} & \text{if } \nu = 2, \\ \frac{\tilde{\gamma}_\nu}{\operatorname{diam}(\mathcal{X})} \min\left\{\frac{\mathit{Gap}(x^k)}{(\frac{\nu}{2}-1) M_f L_{\nabla f}^{(\nu-2)/2}}, \frac{-1}{\mathtt{b}} \frac{\mathit{Gap}(x^k)^2}{L_{\nabla f} \operatorname{diam}(\mathcal{X})}\right\} & \text{if } \nu \in (2,3), \\ \frac{2(1-\ln(2))}{\sqrt{L_{\nabla f}} \operatorname{diam}(\mathcal{X})} \min\left\{\frac{\mathit{Gap}(x^k)}{M_f}, \frac{\mathit{Gap}(x^k)^2}{\sqrt{L_{\nabla f}} \operatorname{diam}(\mathcal{X})}\right\} & \text{if } \nu = 3. \end{cases} \tag{22}$$

where $\tilde{\gamma}_\nu \triangleq 1 + \frac{4-\nu}{2(3-\nu)}\left(1 - 2^{2(3-\nu)/(4-\nu)}\right)$ *and* $\mathtt{b} \triangleq \frac{2-\nu}{4-\nu}$.

Proof See "Appendix C.2". □

Remark 3 It can be checked that $\lim_{\nu \to 3} \tilde{\gamma}_\nu = 1 - \ln(2)$, so that the lower bound $\tilde{\Delta}_k$ is continuous in the parameter range $\nu \in (2, 3]$.

Combining Lemma 4 together with Lemma 5 and estimates summarized in "Appendix C.2", we get the next fundamental relation.

Proposition 4 *Suppose* Assumption 1 *holds. Let* $\{x^k\}_{k \geq 0}$ *be generated by* FWGSC. *Then, for all* $k \geq 0$, *we have*

$$\Delta_k \geq \min\{\mathtt{c}_1(M_f, \nu)\, \mathit{Gap}(x^k), \mathtt{c}_2(M_f, \nu)\, \mathit{Gap}(x^k)^2\},$$

where, for $(M, \nu) \in (0, \infty) \times [2, 3]$, *we define*

$$\mathtt{c}_1(M, \nu) \triangleq \begin{cases} \min\left\{\frac{1}{2}, \frac{2\ln(2)-1}{M \operatorname{diam}(\mathcal{X})}\right\} & \text{if } \nu = 2, \\ \min\left\{\frac{1}{2}, \frac{\tilde{\gamma}_\nu}{\operatorname{diam}(\mathcal{X})(\nu/2-1) M L_{\nabla f}^{(\nu-2)/2}}\right\} & \text{if } \nu \in (2, 3), \\ \min\left\{\frac{1}{2}, \frac{2(1-\ln 2)}{M \sqrt{L_{\nabla f}} \operatorname{diam}(\mathcal{X})}\right\} & \text{if } \nu = 3. \end{cases} \tag{23}$$

and

$$\mathtt{c}_2(M, \nu) \triangleq \begin{cases} \frac{2\ln(2)-1}{L_{\nabla f} \operatorname{diam}(\mathcal{X})^2} & \text{if } \nu = 2, \\ \frac{-1}{\mathtt{b}} \frac{\tilde{\gamma}_\nu}{\operatorname{diam}(\mathcal{X})^2 L_{\nabla f}} & \text{if } \nu \in (2, 3), \\ \frac{2(1-\ln 2)}{L_{\nabla f} \operatorname{diam}(\mathcal{X})^2} & \text{if } \nu = 3. \end{cases} \tag{24}$$

Proof We only illustrate the lower bound for the case $\nu = 2$. All other claims can be verified in exactly the same way. From Lemma 4, we know that $\Delta_k \geq \frac{1}{2} \mathsf{Gap}(x^k)$ whenever $\mathtt{t}_{M_f,2}(x^k) > 1$. Moreover, from Lemma 5 we have that $\mathtt{t}_{M_f,2}(x^k) \leq 1$, then

$$\Delta_k \geq \frac{2\ln 2 - 1}{\operatorname{diam}(\mathcal{X})} \min\left\{\frac{\mathsf{Gap}(x^k)}{M_f}, \frac{\mathsf{Gap}(x^k)^2}{\operatorname{diam}(\mathcal{X}) L_{\nabla f}}\right\}.$$

Consequently,

$$\Delta_k \geq \min\left\{\min\left\{\frac{1}{2}, \frac{2\ln(2) - 1}{M_f \operatorname{diam}(\mathcal{X})}\right\} \mathsf{Gap}(x^k), \frac{2\ln(2) - 1}{\operatorname{diam}(\mathcal{X})^2 L_{\nabla f}} \mathsf{Gap}(x^k)^2\right\}.$$

□

With the help of the lower bound in Proposition 4, we are now able to establish the $\mathcal{O}(1/\varepsilon)$ convergence rate in terms of the *approximation error* $h_k \triangleq f(x^k) - f^*$.

Theorem 1 *Suppose that* Assumption 1 *holds. Let* $\{x^k\}_{k\geq 0}$ *be generated by* FWGSC. *For* $x^0 \in \mathcal{X} \cap \operatorname{dom} f$ *and* $\varepsilon > 0$, *define* $N_\varepsilon(x^0) \triangleq \inf\{k \geq 0 | h_k \leq \varepsilon\}$. *Then, for all* $\varepsilon > 0$,

$$N_\varepsilon(x^0) \leq \frac{\ln\left(\frac{\mathrm{c}_1(M_f,\nu)}{h_0 \mathrm{c}_2(M_f,\nu)}\right)}{\ln(1-\mathrm{c}_1(M_f,\nu))} + \frac{1}{\mathrm{c}_2(M_f,\nu)\varepsilon}. \tag{25}$$

Proof To simplify the notation, let us set $\mathrm{c}_1 \equiv \mathrm{c}_1(M_f,\nu)$ and $\mathrm{c}_2 \equiv \mathrm{c}_2(M_f,\nu)$. By convexity, we have $\mathsf{Gap}(x^k) \geq h_k$. Therefore, Proposition 4 shows that $\Delta_k \geq \min\{\mathrm{c}_1 h_k, \mathrm{c}_2 h_k^2\}$. This implies

$$h_{k+1} \leq h_k - \min\{\mathrm{c}_1 h_k, \mathrm{c}_2 h_k^2\} \qquad \forall k \geq 0.$$

From this inequality we see that h_k is decreasing and there are two potential phases of convergence:

Phase I. $\mathrm{c}_1 h_k < \mathrm{c}_2 h_k^2$, which is equivalent to $h_k > \frac{\mathrm{c}_1}{\mathrm{c}_2}$.

Phase II. $\mathrm{c}_1 h_k \geq \mathrm{c}_2 h_k^2$, which is equivalent to $h_k \leq \frac{\mathrm{c}_1}{\mathrm{c}_2}$.

For fixed initial condition $x^0 \in \operatorname{dom} f \cap \mathcal{X}$, we can thus subdivide the time domain into the set $\mathcal{K}_1(x^0) \triangleq \{k \geq 0 | h_k > \frac{\mathrm{c}_1}{\mathrm{c}_2}\}$ (Phase I) and $\mathcal{K}_2(x^0) \triangleq \{k \geq 0 | h_k \leq \frac{\mathrm{c}_1}{\mathrm{c}_2}\}$ (Phase II). Since in Phase I $\{h_k\}_{k\in\mathcal{K}_1(x^0)}$ is decreasing and bounded from below by the positive constant $\mathrm{c}_1/\mathrm{c}_2$, the set $\mathcal{K}_1(x^0)$ is bounded. Let us set

$$T_1(x^0) \triangleq \inf\left\{k \geq 0 | h_k \leq \frac{\mathrm{c}_1}{\mathrm{c}_2}\right\}, \tag{26}$$

the first time at which the process $\{h_k\}_k$ enters Phase II. To get a worst-case estimate on this quantity, we assume without loss of generality that $0 \in \mathcal{K}_1(x^0)$, so that $\mathcal{K}_1(x^0) = \{0, 1, \ldots, T_1(x^0) - 1\}$. Then, using the definition of the Phase I, for all $k = 1, \ldots, T_1(x^0) - 1$ we have

$$\frac{\mathrm{c}_1}{\mathrm{c}_2} < h_k \leq h_{k-1} - \min\{\mathrm{c}_1 h_{k-1}, \mathrm{c}_2 h_{k-1}^2\} = h_{k-1} - \mathrm{c}_1 h_{k-1}.$$

Note that $\mathrm{c}_1 \leq 1/2$, so we make progressions like a geometric series, i.e. we have linear convergence in this phase. Hence, $h_k \leq (1-\mathrm{c}_1)^k h_0$ for all $k = 0, \ldots, T_1(x^0) - 1$. By the definition of the Phase I, $h_{T_1(x^0)-1} > \frac{\mathrm{c}_1}{\mathrm{c}_2}$, so we get $\frac{\mathrm{c}_1}{\mathrm{c}_2} \leq h_0(1-\mathrm{c}_1)^{T_1(x^0)-1}$ iff $(T_1(x^0)-1)\ln(1-\mathrm{c}_1) \geq \ln\left(\frac{\mathrm{c}_1}{h_0\mathrm{c}_2}\right)$. Hence,

$$T_1(x^0) \leq \lceil \frac{\ln\left(\frac{\mathrm{c}_1}{h_0\mathrm{c}_2}\right)}{\ln(1-\mathrm{c}_1)} \rceil + 1. \tag{27}$$

After these number of iterations, the process will enter Phase II, at which $h_k \leq \frac{\mathtt{c}_1}{\mathtt{c}_2}$ holds. Therefore, $h_k \geq h_{k+1} + \mathtt{c}_2 h_k^2$, or equivalently,

$$\frac{1}{h_{k+1}} \geq \frac{1}{h_k} + \mathtt{c}_2 \frac{h_k}{h_{k+1}} \geq \frac{1}{h_k} + \mathtt{c}_2. \tag{28}$$

Pick $N > T_1(x^0)$ an arbitrary integer. Summing (28) from $k = T_1(x^0)$ up to $k = N-1$, we arrive at

$$\frac{1}{h_N} \geq \frac{1}{h_{T_1(x^0)}} + \mathtt{c}_2(N - T_1(x^0) + 1).$$

By definition $h_{T_1(x^0)} \leq \frac{\mathtt{c}_1}{\mathtt{c}_2}$, so that for all $N > T_1(x^0)$, we see

$$\frac{1}{h_N} \geq \frac{\mathtt{c}_2}{\mathtt{c}_1} + \mathtt{c}_2(N - T_1(x^0) + 1).$$

Consequently,

$$h_N \leq \frac{1}{\frac{\mathtt{c}_2}{\mathtt{c}_1} + \mathtt{c}_2(N - T_1(x^0) + 1)} \leq \frac{1}{\mathtt{c}_2(N - T_1(x^0) + 1)}. \tag{29}$$

By definition of the stopping time $N_\varepsilon(x^0)$, it is true that $h_{N_\varepsilon(x^0)-1} > \varepsilon$. Consequently, evaluating (29) at $N = N_\varepsilon(x^0) - 1$, we obtain

$$\varepsilon \leq \frac{1}{\mathtt{c}_2(N_\varepsilon(x^0) - T_1(x^0))} \Leftrightarrow N_\varepsilon(x^0) \leq T_1(x^0) + \frac{1}{\mathtt{c}_2 \varepsilon}.$$

Combining this upper bound with (27) shows the claim. □

Remark 4 Combining the result of Theorem 1 and the definitions of the constants $\mathtt{c}_1(M, \nu)$ in (23) and $\mathtt{c}_2(M, \nu)$ in (24), we can see that, neglecting the logarithmic terms and using that $-\frac{1}{\ln(1-x)} \leq \frac{1}{x}$ for $x \in [0, 1]$, the iteration complexity of `FWGSC` can be bounded as

$$\max\left\{c_1, c_2 M_f L_{\nabla f}^{(\nu-2)/2} \operatorname{diam}(\mathcal{X})\right\} + \frac{c_3 L_{\nabla f} \operatorname{diam}(\mathcal{X})^2}{\varepsilon}, \tag{30}$$

where c_1, c_2, c_3 are numerical constants. The first term corresponds to Phase I where one observes the linear convergence, the second term corresponds to the Phase II with sublinear convergence. Interestingly, the second term has the same form as the standard complexity bound for FW methods. The only difference is that the global Lipschitz constant of the gradient is changed to the Lipschitz constant over the level set defined by the starting point.

4.2 Complexity analysis of backtracking versions

The complexity analysis of both backtracking-based algorithms (LBTFWGSC and MBTFWGSC) use similar ideas, which all essentially rest on the specific form of the employed upper model Q_L and Q_M, respectively. We will first derive a uniform bound on the per-iteration decrease of the objective function value, and then deduce the complexity analysis from Theorem 1. In both algorithms we use a generic bound on the backtracking parameter.

Lemma 6 *Let $\{\mathcal{L}_k\}_{k\in\mathbb{N}}$ be the sequence of Lipschitz estimates produced by the procedure $\texttt{step}_L(f, v^k, x^k, \mathcal{L}^{k-1})$ and $\{\mu_k\}_{k\in\mathbb{N}}$ the sequence of GSC-parameter estimates produced by $\texttt{step}_M(f, v^k, x^k, \mu^{k-1})$, respectively. We have $\mathcal{L}_k \leq \max\{\mathcal{L}_{-1}, \gamma_u L_{\nabla f}\}$ and $\mu^k \leq \max\{\mu_{-1}, \gamma_u M_f\}$.*

Proof We proof the statement only for the sequence $\{\mathcal{L}_k\}_k$. The claim for $\{\mu_k\}_{k\in\mathbb{N}}$ can be shown in the same way. By construction of the backtracking procedure, we know that if the sufficient decrease condition is evaluated successfully at the first run, then $\mathcal{L}_{k-1} \geq \mathcal{L}_k \geq \gamma_d \mathcal{L}_{k-1}$. If not, then it is clear that $\mathcal{L}_k \leq \gamma_d L_{\nabla f}$. Hence, for all $k \geq 0$, $\mathcal{L}_k \leq \max\{\gamma_d L_{\nabla f}, \mathcal{L}_{k-1}\}$. By backwards induction, it follows then $\mathcal{L}_k \leq \max\{\mathcal{L}_{-1}, \gamma_u L_{\nabla f}\}$. □

4.2.1 Analysis of LBTFWGSC

Calling Algorithm LBTFWGSC at position x^k generates a step size α_k and a local Lipschitz estimate $\mathcal{L}_k$ via $(\alpha_k, \mathcal{L}_k) = \texttt{step}_L(f, v_{\mathrm{FW}}(x^k), x^k, \mathcal{L}_{k-1})$. The thus produced new search point satisfies $x^{k+1} = x^k + \alpha_k v^k \in \operatorname{dom} f \cap \mathcal{X}$, and

$$f(x^{k+1}) \leq f(x^k) - \alpha_k \,\mathsf{Gap}(x^k) + \frac{\mathcal{L}_k \alpha_k^2}{2}\beta_k^2 \quad \text{where } \beta_k \equiv \beta(x^k).$$

The reported step size is $\alpha_k = \min\left\{1, \frac{\mathsf{Gap}(x^k)}{\mathcal{L}_k \beta_k^2}\right\}$. For each of these possible realizations of this step size, we will provide a lower bound of the achieved reduction in the objective function value.

Case 1 If $\alpha_k = 1$, then $\mathcal{L}_k \beta_k^2 \leq \mathsf{Gap}(x^k)$ and $x^{k+1} = x^k + v^k \in \operatorname{dom} f \cap \mathcal{X}$. Hence,

$$f(x^{k+1}) \leq f(x^k) - \mathsf{Gap}(x^k) + \frac{\mathcal{L}_k}{2}\beta_k^2 \leq f(x^k) - \frac{\mathsf{Gap}(x^k)}{2}.$$

Case 2 If $\alpha_k = \frac{\mathsf{Gap}(x^k)}{\mathcal{L}_k \beta_k^2}$, then

$$f(x^{k+1}) \leq f(x^k) - \frac{\mathsf{Gap}(x^k)^2}{2\mathcal{L}_k \beta_k^2}.$$

Since $\mathcal{L}_k \leq \max\{\gamma_u L_{\nabla f}, \mathcal{L}_{-1}\} \equiv \bar{L}$ (Lemma 6), we obtain the performance guarantee

$$f(x^k) - f(x^{k+1}) \geq \min\left\{\frac{\mathsf{Gap}(x^k)}{2}, \frac{\mathsf{Gap}(x^k)^2}{2\mathcal{L}_k\beta_k^2}\right\}$$
$$\geq \min\left\{\frac{\mathsf{Gap}(x^k)}{2}, \frac{\mathsf{Gap}(x^k)^2}{2\bar{L}\,\mathrm{diam}(\mathcal{X})^2}\right\}.$$

Set $\mathtt{c}_1 \equiv \frac{1}{2}$ and $\mathtt{c}_2 \equiv \frac{1}{2\bar{L}\,\mathrm{diam}(\mathcal{X})^2}$, it therefore follows that

$$f(x^k) - f(x^{k+1}) \geq \min\left\{\mathtt{c}_1\,\mathsf{Gap}(x^k), \mathtt{c}_2\,\mathsf{Gap}(x^k)^2\right\}.$$

In terms of the approximation error, this implies

$$h_k - h_{k+1} \geq \min\{\mathtt{c}_1 h_k, \mathtt{c}_2 h_k^2\}.$$

Thus, we can use a similar analysis as in the one in the proof of Theorem 1, and obtain the following $\mathcal{O}(1/\varepsilon)$ iteration complexity guarantee for method `LBTFWGSC`.

Theorem 2 *Suppose that* Assumptions 1 and 2 *hold. Let* $\{x^k\}_{k\geq 0}$ *be generated by* `LBTFWGSC`*. For* $x^0 \in \mathcal{X} \cap \mathrm{dom}\, f$ *and* $\varepsilon > 0$*, define* $N_\varepsilon(x^0) \triangleq \inf\{k \geq 0 | h_k \leq \varepsilon\}$*. Then, for all* $\varepsilon > 0$*,*

$$N_\varepsilon(x^0) \leq \frac{\ln(\bar{L}\,\mathrm{diam}(\mathcal{X})^2/h_0)}{\ln(1/2)} + \frac{2\bar{L}\,\mathrm{diam}(\mathcal{X})^2}{\varepsilon}, \tag{31}$$

where $\bar{L} = \max\{\gamma_u L_{\nabla f}, \mathcal{L}_{-1}\}$.

4.2.2 Analysis of `MBTFWGSC`

The complexity analysis of this algorithm is completely analogous to the one corresponding to Algorithm `LBTFWGSC`. The main difference between the two variants is the upper model employed in the local search. Calling `MBTFWGSC` at position x^k, generates the pair $(\alpha_k, \mu_k) = \mathtt{step}_M(f, v_{\mathrm{FW}}(x^k), x^k, \mu_{k-1})$ such that

$$f(x^{k+1}) \leq f(x^k) - \alpha_k\,\mathsf{Gap}(x^k) + \alpha_k^2 \mathtt{e}_k^2 \omega_\nu(\mu_k\alpha_k\delta_\nu(x^k)),$$

where $\mathtt{e}_k \equiv \mathtt{e}(x^k)$. The step size parameter α_k satisfies $\alpha_k = \min\{1, \mathtt{t}_{\mu_k,\nu}(x^k)\}$. We can thus apply Proposition 4 in order to obtain the recursion

$$h_{k+1} \leq h_k - \min\{\mathtt{c}_1(\mu_k, \nu)h_k, \mathtt{c}_2(\mu_k, \nu)h_k^2\},$$

involving the constants defined in (23) and (24). By construction of the backtracking step, we know that $\mu_k \leq \max\{\gamma_u M_f, \mu_{-1}\} \equiv \bar{M}$ (Lemma 6). Hence, after setting $\mathrm{c}_1 \equiv \mathrm{c}_1(\bar{M}, \nu)$, $\mathrm{c}_2 \equiv \mathrm{c}_2(\bar{M}, \nu)$, we arrive at

$$h_{k+1} \leq h_k - \min\{\mathrm{c}_1 h_k, \mathrm{c}_2 h_k^2\} \qquad \forall k \geq 0.$$

From here the complexity analysis proceeds as in Theorem 1. The only change that has to be made is to replace the expressions $\mathrm{c}_1(M_f, \nu)$ and $\mathrm{c}_2(M_f, \nu)$ by the numbers $\mathrm{c}_1(\bar{M}, \nu)$ and $\mathrm{c}_2(\bar{M}, \nu)$, respectively.

Theorem 3 *Suppose that* Assumption 1 and 2 *hold. Let* $\{x^k\}_{k\geq 0}$ *be generated by* `MBTFWGSC`. *For* $x^0 \in \mathcal{X} \cap \operatorname{dom} f$ *and* $\varepsilon > 0$, *define* $N_\varepsilon(x^0) \triangleq \inf\{k \geq 0 | h_k \leq \varepsilon\}$. *Then, for all* $\varepsilon > 0$,

$$N_\varepsilon(x^0) \leq \frac{\ln\left(\frac{\mathrm{c}_1(\bar{M},\nu)}{h_0 \mathrm{c}_2(\bar{M},\nu)}\right)}{\ln(1 - \mathrm{c}_1(\bar{M}, \nu))} + \frac{1}{\mathrm{c}_2(\bar{M}, \nu)\varepsilon}, \tag{32}$$

where $\bar{M} = \max\{\gamma_u M_f, \mu_{-1}\}$.

Note that a similar remark to Remark 4 can be made in this case.

Remark 5 While the proofs of Theorem 2 and Theorem 3 follow the same steps, the underlying models are different. `LBTFWGSC` is based on the observation that since the algorithm is monotone, it stays in the level set on which the objective function has Lipschitz-continuous gradient. This allows us to use the quadratic upper bound (16) to find the corresponding stepsize. On the contrary, `MBTFWGSC` is based on the upper bound (18) that is specific to generalized self-concordant functions. Moreover, these two different models lead to different stepsize definitions, different estimates for the per-iteration progress, and slightly different complexity results, yet with similar dependence on ε.

5 Linearly convergent variants of Frank–Wolfe for GSC functions

In the development of all our linearly convergent variants, we assume that the feasible set is a polytope described by a system of linear inequalities.

Assumption 3 The feasible set $\mathcal{X}$ admits the explicit representation

$$\mathcal{X} \triangleq \{x \in \mathbb{R}^n | \mathbf{B}x \leq b\}, \tag{33}$$

where $\mathbf{B} \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$.

Algorithm 7: FWLLOO

Input: $\mathcal{A}(x, r, c)$-LLOO with parameter $\rho \geq 1$ for polytope $\mathcal{X}$, $f \in \mathcal{F}_{M_f,\nu}$. $\sigma_f > 0$ convexity parameter.
$x^0 \in \operatorname{dom} f \cap \mathcal{X}$, and let $h_0 = f(x^0) - f^*$, and $c_0 = 1$.
$r_0 = \sqrt{\frac{2\,\mathsf{Gap}(x^0)}{\sigma_f}}$
for $k = 0, 1, \ldots$ **do**
  if $\mathsf{Gap}(x^k) > \varepsilon$ **then**
    Set $r_k^2 = r_0^2 c_k$;
    Obtain $u^k = u(x^k, r_k, \nabla f(x^k))$ by querying procedure $\mathcal{A}(x^k, r_k, \nabla f(x^k))$;
    Set $\alpha_k = \alpha_v(x^k)$ by evaluating (37);
    Set $x^{k+1} = x^k + \alpha_k(u^k - x^k)$;
    Set $c_{k+1} = c_k \exp(-\frac{1}{2}\alpha_k)$.
  end if
end for

5.1 Local linear minimization oracles

In this section we show how the local linear minimization oracle of [24] can be adapted to accelerate the convergence of FW-methods for minimizing GSC functions. In particular, we work out an analytic step-size criterion which guarantees linear convergence towards a solution of (P). The construction is a non-trivial modification of [24], as it exploits the local descent properties of GSC functions. In particular, we neither assume global Lipschitz continuity, nor strong convexity of the objective function. Instead, our working assumption in this section is the availability of a local linear minimization oracle, defined as follows:

Definition 3 ([24], Def. 2.5) A procedure $\mathcal{A}(x, r, c)$, where $x \in \mathcal{X}$, $r > 0$, $c \in \mathbb{R}^n$, is a Local Linear Optimization Oracle (LLOO) with parameter $\rho \geq 1$ for the polytope $\mathcal{X}$ if $\mathcal{A}(x, r, c)$ returns a point $u(x, r, c) = u \in \mathcal{X}$ such that

$$\forall y \in \mathbb{B}(x, r) \cap \mathcal{X} : \langle c, y\rangle \geq \langle c, u\rangle, \text{ and } \|x - u\|_2 \leq \rho r. \tag{34}$$

We refer to [24] for illustrative examples for oracles $\mathcal{A}(x, r, c)$. In particular, [24] provide an explicit construction of the LLOO for a simplex and for general polytopes. We further redefine the local norm as

$$\mathsf{e}(x) \triangleq \|u(x, r, \nabla f(x)) - x\|_x \qquad \forall x \in \operatorname{dom} f.$$

With an obvious abuse of notation, we also redefine

$$\delta_v(x) \triangleq \begin{cases} \|u(x, r, \nabla f(x)) - x\|_2 & \text{if } v = 2, \\ \frac{v-2}{2}\|u(x, r, \nabla f(x)) - x\|_2^{3-v}\|u(x, r, \nabla f(x)) - x\|_x^{v-2} & \text{if } v \in (2, 3]. \end{cases} \tag{35}$$

As in the previous sections, our goal is to come up with a step-size policy guaranteeing feasibility and a sufficient decrease. As will become clear in a moment, our construction relies on a careful analysis of the function

$$\psi_\nu(t) \triangleq t - \xi\omega_\nu(t\delta)t^2 \qquad t \in [0, 1/\delta),$$

where $\xi, \delta \geq 0$ are free parameters. This function is also used in the complexity analysis of `FWGSC`, and thoroughly discussed in "Appendix B". In particular, the analysis in "Appendix B" shows that $t \mapsto \psi_\nu(t)$ is concave, unimodal with $\psi_\nu(0) = 0$, increasing on the interval $[0, t_\nu^*)$ and decreasing on $[t_\nu^*, \infty)$, where the cut-off value t_ν^* is defined in eq. (62). Moreover, $\psi_\nu(t) \geq 0$ for $t \in [0, t_\nu^*]$. To facilitate the discussion, let us redefine this cut-off value in a way which emphasizes its dependence on structural parameters. We call

$$t_\nu^* = t_\nu^*(\delta, \xi) \triangleq \begin{cases} \frac{1}{\delta}\ln\left(1 + \frac{\delta}{\xi}\right) & \text{if } \nu = 2, \\ \frac{1}{\delta}\left[1 - \left(1 + \frac{\delta}{\xi}\frac{4-\nu}{\nu-2}\right)^{-\frac{\nu-2}{4-\nu}}\right] & \text{if } \nu \in (2, 3), \\ \frac{1}{\delta+\xi} & \text{if } \nu = 3. \end{cases} \tag{36}$$

We construct our step size policy iteratively. Suppose we are given the current iterate $x^k \in \operatorname{dom} f \cap \mathcal{X}$, produced by k sequential calls of `FWLLOO`, using a finite sequence $\{\alpha_i\}_{i=0}^{k-1}$ of step-sizes and search radii $\{r_i\}_{i=0}^{k-1}$. Set $c_k = \exp\left(-\sum_{i=0}^{k-1}\alpha_i\right)$. Call the LLOO to obtain the target state $u^k = u(x^k, r_k, \nabla f(x^k))$, using the updated search radius $r_k = r_0 c_k$. We define the next step size $\alpha_k = \alpha_\nu(x^k)$ by setting

$$\alpha_\nu(x^k) \triangleq \min\left\{1, t_\nu^*\left(M_f\delta_\nu(x^k), \frac{2\epsilon(x^k)^2}{\mathsf{Gap}(x^0)c_k}\right)\right\}. \tag{37}$$

Update the sequence of search points to $x^{k+1} = x^k + \alpha_k(u^k - x^k)$. By construction of $t_\nu^k \equiv t_\nu^*\left(M_f\delta_\nu(x^k), \frac{2\epsilon(x^k)^2}{\mathsf{Gap}(x^0)c_k}\right)$, this point lies in $\operatorname{dom} f \cap \mathcal{X}$. To see this, consider first the case in which $\alpha_k = 1 < t_\nu^k$. Then, $\mathsf{d}_\nu(x^{k+1}, x^k) = \alpha_k M_f\delta_\nu(x^k) = M_f\delta_\nu(x^k) < t_\nu^k M_f\delta_\nu(x^k) < 1$. On the other hand, if $\alpha_k = t_\nu^k$, then it follows from the definition of the involved quantities that $\mathsf{d}_\nu(x^{k+1}, x^k) = \alpha_k M_f\delta_\nu(x^k) < 1$.

Repeating this procedure iteratively yields a sequence $\{x^k\}_{k\in\mathbb{N}}$, whose performance guarantees in terms of the approximation error $h_k = f(x^k) - f^*$ are described in the Theorem below.

Theorem 4 *Suppose* Assumption 1 *holds. Let* $\{x^k\}_{k\geq 0}$ *be generated by* `FWLLOO`*. Then, for all* $k \geq 0$*, we have* $x^* \in \mathbb{B}(x^k, r_k)$ *and*

$$h_k \leq \mathsf{Gap}(x^0)\exp\left(-\frac{1}{2}\sum_{i=0}^{k-1}\alpha_i\right) \tag{38}$$

where the sequence $\{\alpha_k\}_k$ *is constructed as in* (37)*.*

Proof Let us define $\mathcal{P}(x^0) \triangleq \left\{x \in \mathcal{X} | f(x) \leq f^* + \mathsf{Gap}(x^0)\right\}$. We proceed by induction. For $k = 0$, we have $x^0 \in \mathrm{dom}\, f \cap \mathcal{X}$ by assumption and $x^0 \in \mathcal{P}(x^0)$ by definition. (21) gives

$$f(x^0) - f^* = h_0 \geq \frac{\sigma_f}{2}\|x^0 - x^*\|_2^2. \tag{39}$$

Let $u^0 \equiv u(x^0, r_0, \nabla f(x^0))$, $\delta_0 \equiv \delta_v(x^0)$, $\xi_0 = \frac{2\mathsf{e}(x^0)^2}{\mathsf{Gap}(x^0)}$ and $\alpha_0 = \alpha_v(x^0)$ obtained by evaluating (37) with the cut-off value $t_v^*(M_f\delta_0, \xi_0)$. Since $r_0 = \sqrt{\frac{2\,\mathsf{Gap}(x^0)}{\sigma_f}} \geq \sqrt{\frac{2h_0}{\sigma_f}}$, (39) implies that $x^* \in \mathbb{B}(x^0, r_0)$. The definition of the LLOO gives us

$$\langle \nabla f(x^0), u^0 - x^0\rangle \leq \langle \nabla f(x^0), x^* - x^0\rangle. \tag{40}$$

Set $x^1 = x^0 + \alpha_0(u^0 - x^0) \in \mathrm{dom}\, f \cap \mathcal{X}$. The GSC-descent lemma (9) gives then

$$\begin{aligned} f(x^1) &\leq f(x^0) + \alpha_0\langle \nabla f(x^0), u^0 - x^0\rangle + \alpha_0^2\mathsf{e}(x^0)^2\omega_v(\alpha_0 M_f\delta_0) \\ &\overset{(40)}{\leq} f(x^0) + \alpha_0\langle \nabla f(x^0), x^* - x^0\rangle + \alpha_0^2\mathsf{e}(x^0)^2\omega_v(\alpha_0 M_f\delta_0) \\ &\leq f(x^0) + \alpha_0(f^* - f(x^0)) + \alpha_0^2\mathsf{e}(x^0)^2\omega_v(\alpha_0 M_f\delta_0) \end{aligned}$$

Hence, writing the above in terms of the approximation error $h_k = f(x^k) - f^*$, we obtain

$$\begin{aligned} h_1 &\leq h_0(1 - \alpha_0) + \alpha_0^2\mathsf{e}(x^0)^2\omega_v(\alpha_0 M_f\delta_0) \\ &\leq (1 - \alpha_0)\,\mathsf{Gap}(x^0) + \alpha_0^2\mathsf{e}(x^0)^2\omega_v(\alpha_0 M_f\delta_0) \\ &= \left(1 - \frac{\alpha_0}{2}\right)\mathsf{Gap}(x^0) - \frac{\mathsf{Gap}(x^0)}{2}\left(\alpha_0 - \alpha_0^2\frac{2\mathsf{e}(x^0)^2}{\mathsf{Gap}(x^0)}\omega_v(\alpha_0 M_f\delta_0)\right). \end{aligned}$$

We see that the second summand in the right-hand side above is just the value of the function $\psi_v(\alpha_0)$, with the parameters $\delta = M_f\delta_0$ and $\xi = \xi_0 = \frac{2\mathsf{e}(x^0)^2}{\mathsf{Gap}(x^0)}$. Hence, by construction, the second summand is nonnegative, which gives us the bound

$$h_1 \leq (1 - \frac{\alpha_0}{2})\,\mathsf{Gap}(x^0) \leq \exp(-\alpha_0/2)\,\mathsf{Gap}(x^0).$$

To perform the induction step, assume that for some $k \geq 1$ it holds

$$h_k \leq \mathsf{Gap}(x^0)c_k,\ c_k \triangleq \exp\left(-\frac{1}{2}\sum_{i=0}^{k-1}\alpha_i\right). \tag{41}$$

Since $c_k \in (0, 1)$, we readily see that $x^k \in \mathcal{P}(x^0)$. Call $\delta_k = \delta_\nu(x^k)$ and $\xi_k = \frac{2\mathsf{e}(x^k)^2}{\mathsf{Gap}(x^0)c_k}$. (21) leads to

$$\|x^k - x^*\|_2^2 \leq \frac{2h_k}{\sigma_f} \leq \frac{2\,\mathsf{Gap}(x^0)}{\sigma_f} c_k = r_0^2 c_k \equiv r_k^2 \Rightarrow x^* \in \mathbb{B}(x^k, r_k). \tag{42}$$

Call the LLOO to obtain the target point $u^k = \mathcal{A}(x^k, r_k, \nabla f(x^k))$. Using the definition of the LLOO, (42) implies

$$\langle \nabla f(x^k), u^k - x^k \rangle \leq \langle \nabla f(x^k), x^* - x^k \rangle. \tag{43}$$

Define the step size $\alpha_k = \alpha_\nu(x^k)$, and declare the next search point $x^{k+1} = x^k + \alpha_k(u^k - x^k) \in \operatorname{dom} f \cap \mathcal{X}$. By the discussion preceeding the Theorem, it is clear that $x^{k+1} \in \mathcal{X} \cap \operatorname{dom} f$. Via the GSC-descent lemma and the induction hypothesis we arrive in exactly the same way as for the case $k = 0$ to the inequality

$$h_{k+1} \leq \left(1 - \frac{\alpha_k}{2}\right) \mathsf{Gap}(x^0)c_k - \frac{\mathsf{Gap}(x^0)c_k}{2} \left(\alpha_k - \alpha_k^2 \frac{2\mathsf{e}(x^k)^2}{\mathsf{Gap}(x^0)c_k} \omega_\nu(\alpha_k M_f \delta_k)\right).$$

The construction of the step size α_k ensures that the expression in the brackets on the right-hand-side is non-negative. Consequently, we obtain $h_{k+1} \leq (1 - \alpha_k/2)\,\mathsf{Gap}(x^0)c_k \leq \mathsf{Gap}(x^0)c_k \exp(-\alpha_k/2) = \mathsf{Gap}(x^0)c_{k+1}$, which finishes the induction proof. □

To obtain the final linear convergence rate, it remains to lower bound the step size sequence $\alpha_k = \alpha_\nu(x^k)$. Note that for all values $\nu \in [2, 3]$, $t_\nu^*(\delta, \xi)$ is an increasing function of $\frac{1}{\delta}$ and $\frac{\delta}{\xi}$. Thus, our next steps are to lower bound the values of the non-negative sequences $\{\frac{1}{M_f \delta_k}\}_k$ and $\{\frac{M_f \delta_k}{\xi_k}\}_k$, where $\delta_k = \delta_\nu(x^k)$ and $\xi_k = \frac{2\mathsf{e}(x^k)^2}{\mathsf{Gap}(x^0)c_k}$ for all $k \geq 0$. We have

$$\frac{1}{M_f \delta_k} = \begin{cases} \frac{1}{M_f \|u^k - x^k\|_2} & \text{if } \nu = 2, \\ \frac{1}{\frac{\nu-2}{2} M_f \|u^k - x^k\|_2^{3-\nu} \|u^k - x^k\|_{x^k}^{\nu-2}} & \text{if } \nu \in (2, 3]. \end{cases}$$

By definition of the LLOO, we have $\|u^k - x^k\|_2 \leq \min\{\rho r_k, \operatorname{diam}(\mathcal{X})\}$. Thus, if $\nu = 2$, we have

$$\frac{1}{M_f \delta_k} \geq \frac{1}{M_f \min\{\rho r_k, \operatorname{diam}(\mathcal{X})\}} \geq \frac{1}{M_f \rho r_k},$$

while if $\nu > 2$, we observe

$$\frac{1}{M_f \delta_k} \geq \frac{1}{\frac{\nu-2}{2} M_f \|u^k - x^k\|_2^{3-\nu} L_{\nabla f}^{\frac{\nu-2}{2}} \|u^k - x^k\|_2^{\nu-2}} = \frac{1}{\frac{\nu-2}{2} M_f L_{\nabla f}^{\frac{\nu-2}{2}} \|u^k - x^k\|_2}$$
$$\geq \frac{1}{\frac{\nu-2}{2} M_f L_{\nabla f}^{\frac{\nu-2}{2}} \min\{\rho r_k, \operatorname{diam}(\mathcal{X})\}} \geq \frac{1}{\frac{\nu-2}{2} M_f L_{\nabla f}^{\frac{\nu-2}{2}} \rho r_k}.$$

Furthermore, from the identity $\frac{2\,\mathsf{Gap}(x^0)c_k}{\sigma_f} = r_k^2$, we conclude $\mathsf{Gap}(x^0)c_k = \frac{\sigma_f r_k^2}{2}$. Hence,

$$\frac{M_f \delta_k}{\xi_k} = \frac{M_f \delta_\nu(x^k)\,\mathsf{Gap}(x^0)c_k}{2\mathsf{e}(x^k)^2} = \begin{cases} \frac{M_f \|u^k - x^k\|_2 \frac{\sigma_f r_k^2}{2}}{2\|u^k - x^k\|_{x^k}^2} & \text{if } \nu = 2, \\ \frac{\frac{\nu-2}{2} M_f \|u^k - x^k\|_2^{3-\nu} \mathsf{e}(x^k)^{\nu-2} \frac{\sigma_f r_k^2}{2}}{2\mathsf{e}(x^k)^2} & \text{if } \nu \in (2, 3]. \end{cases}$$

If $\nu = 2$, we see that

$$\frac{M_f \delta_k}{\xi_k} \geq \frac{M_f \|u^k - x^k\|_2 \sigma_f r_k^2}{4L_{\nabla f} \|u^k - x^k\|_2^2} = \frac{M_f \sigma_f r_k^2}{4L_{\nabla f} \|u^k - x^k\|_2} \geq \frac{M_f \sigma_f r_k^2}{4L_{\nabla f} \min\{\rho r_k, \operatorname{diam}(\mathcal{X})\}}$$
$$\geq \frac{M_f \sigma_f r_k}{4\rho L_{\nabla f}},$$

while if $\nu > 2$, we have in turn

$$\frac{M_f \delta_k}{\xi_k} = \frac{(\nu-2) M_f \|u^k - x^k\|_2^{3-\nu} \sigma_f r_k^2}{8\mathsf{e}(x^k)^{4-\nu}} \geq \frac{(\nu-2) M_f \|u^k - x^k\|_2^{3-\nu} \sigma_f r_k^2}{8 L_{\nabla f}^{\frac{4-\nu}{2}} \|u^k - x^k\|_2^{4-\nu}}$$
$$= \frac{(\nu-2) M_f \sigma_f r_k^2}{8 L_{\nabla f}^{\frac{4-\nu}{2}} \|u^k - x^k\|_2} \geq \frac{(\nu-2) M_f \sigma_f r_k^2}{8 L_{\nabla f}^{\frac{4-\nu}{2}} \min\{\rho r_k, \operatorname{diam}(\mathcal{X})\}}$$
$$\geq \frac{(\nu-2) M_f \sigma_f r_k}{8\rho L_{\nabla f}^{\frac{4-\nu}{2}}} = \frac{(\nu-2) M_f L_{\nabla f}^{\frac{\nu-2}{2}} \sigma_f r_k}{8\rho L_{\nabla f}}.$$

Denoting $\gamma_\nu = \frac{\nu-2}{2} M_f L_{\nabla f}^{\frac{\nu-2}{2}}$ for $\nu > 2$ and $\gamma_\nu = M_f$ for $\nu = 2$, and substituting these lower bounds to the expression for t_ν^*, we obtain

$$t_\nu^k \equiv t_\nu^*\left(M_f \delta_\nu(x^k), \frac{2\epsilon(x^k)^2}{\mathsf{Gap}(x^k)c_k}\right)$$
$$\geq \underline{t}_k \triangleq \begin{cases} \frac{1}{\gamma_\nu \rho r_k} \ln\left(1 + \frac{\gamma_\nu \sigma_f r_k}{4\rho L_{\nabla f}}\right) & \text{if } \nu = 2, \\ \frac{1}{\gamma_\nu \rho r_k}\left[1 - \left(1 + \frac{\gamma_\nu \sigma_f r_k}{4\rho L_{\nabla f}} \frac{4-\nu}{\nu-2}\right)^{-\frac{\nu-2}{4-\nu}}\right] & \text{if } \nu \in (2,3), \\ \frac{1}{\gamma_\nu \rho r_k} \frac{1}{1+\frac{4\rho L_{\nabla f}}{\gamma_\nu \sigma_f r_k}} & \text{if } \nu = 3. \end{cases}$$

For all $\nu \in [2,3]$, the minorizing sequence $\{\underline{t}_k\}_k$ has a limit $\frac{\sigma_f}{4\rho^2 L_{\nabla f}}$ as $r_k \to 0$. Moreover, as the search radii sequence $\{r_k\}_{k\in\mathbb{N}}$ is decreasing, basic calculus shows that the sequence $\{\underline{t}_k\}_k$ is monotonically increasing. Whence, we get a uniform lower bound of the cut-off values $\{t_\nu^k\}_k$ as

$$t_\nu^k \geq \underline{t} \triangleq \begin{cases} \frac{1}{\gamma_\nu \rho r_0} \ln\left(1 + \frac{\gamma_\nu \sigma_f r_0}{4\rho L_{\nabla f}}\right) & \text{if } \nu = 2, \\ \frac{1}{\gamma_\nu \rho r_0}\left[1 - \left(1 + \frac{\gamma_\nu \sigma_f r_0}{4\rho L_{\nabla f}} \frac{4-\nu}{\nu-2}\right)^{-\frac{\nu-2}{4-\nu}}\right] & \text{if } \nu \in (2,3) \\ \frac{1}{\gamma_\nu \rho r_0} \frac{1}{1+\frac{4\rho L_{\nabla f}}{\gamma_\nu \sigma_f r_0}} & \text{if } \nu = 3. \end{cases} \tag{44}$$

Corollary 1 *Suppose* Assumption 1 *holds. Algorithm* `FWLLOO` *guarantees linear convergence in terms of the approximation error:*

$$h_k \leq \mathit{Gap}(x^0)\exp(-k\bar{\alpha}/2) \qquad \forall k \geq 0,$$

where $\bar{\alpha} = \min\{\underline{t}, 1\}$ *with* $\underline{t}$ *defined in* (44).

Proof It is clear that $\alpha_k \geq \bar{\alpha} = \min\{\underline{t}, 1\}$ for all $k \geq 0$. Hence $\exp\left(-\frac{1}{2}\sum_{i=0}^{k-1}\alpha_i\right) \leq \exp(-k\bar{\alpha}/2)$, and the claim follows. □

The obtained bound can be quite conservative since we used a uniform bound for the sequence $\underline{t}_k$. At the same time, since r_k geometrically converges to 0 and for all $\nu \in [2,3]$, the minorizing sequence $\{\underline{t}_k\}_k$ has a limit $\frac{\sigma_f}{4\rho^2 L_{\nabla f}}$ as $r_k \to 0$, we may expect that after some burn-in phase, the sequence α_k can be bounded from below by $\frac{\sigma_f}{8\rho^2 L_{\nabla f}}$. This lower bound leads to the linear convergence as $h_k \leq \mathsf{Gap}(x^0)\exp(-k_0\bar{\alpha}/2))\exp(-(k-k_0)\frac{\sigma_f}{16\rho^2 L_{\nabla f}})$ for $k \geq k_0$, where the length of the burn-in phase k_0 is up to logarithmic factors equal to $\frac{1}{\bar{\alpha}}$. This corresponds to the iteration complexity

$$k_0 + \frac{16\rho^2 L_{\nabla f}}{\sigma_f} \ln \frac{\mathsf{Gap}(x^0)\exp(-k_0\bar{\alpha}/2))}{\varepsilon}.$$

Interestingly, the second term has the same form as the complexity bound for FW method under the LLOO proved in [24] with $\frac{\rho^2 L_{\nabla f}}{\sigma_f}$ playing the role of condition

number. The only difference is that the global Lipschitz constant of the gradient is changed to the Lipschitz constant over the level set defined by the starting point.

5.2 Away-step Frank–Wolfe (ASFW)

We start with some preparatory remarks. Recall that in this section Assumption 3 is in place. Hence, $\mathcal{X}$ is a polytope of the form (33). By compactness and the Krein-Milman theorem, we know that $\mathcal{X}$ is the convex hull of finitely many vertices (extreme points) $\mathcal{U} \triangleq \{u_1, \ldots, u_q\}$. Let $\Delta(\mathcal{U})$ denote the set of discrete measures $\mu \triangleq (\mu_u : u \in \mathcal{U})$ with $\mu_u \geq 0$ for all $u \in \mathcal{U}$ and $\sum_{u\in\mathcal{U}} \mu_u = 1, \mu_u \geq 0$. A measure $\mu^x \in \Delta(\mathcal{U})$ is a *vertex representation* of x if $x = \sum_{u\in\mathcal{U}} \mu^x_u u$. Given $\mu \in \Delta(\mathcal{U})$, we define $\text{supp}(\mu) \triangleq \{u \in \mathcal{U} | \mu_u > 0\}$ and the set of active vertices $\mathcal{U}(x) \triangleq \{u \in \mathcal{U} | u \in \text{supp}(\mu^x)\}$ of point $x \in \mathcal{X}$ under the *vertex representation* $\mu^x \in \Delta(\mathcal{U})$. We use $I(x) \triangleq \{i \in \{1, \ldots, m\} | \mathbf{B}_i x = b_i\}$ to denote the set of binding constraints at x. For a given set $V \subset \mathcal{U}$, we let $I(V) = \bigcap_{u\in V} I(u)$.
For the linear minimization oracle generating the target point $s(x)$, we invoke an explicit tie-breaking rule in the definition of the linear minimization oracle.

Assumption 4 The linear minimization procedure

$$s(x) \in \text{argmin}_{d\in\mathcal{X}} \langle \nabla f(x), d \rangle$$

returns a vertex solution, i.e. $s(x) \in \mathcal{U}$ for all $x \in \mathcal{X}$.

Remark 6 [4] refer to this as a *vertex linear oracle*.

ASFW needs also a target vertex which is as much aligned as possible with the same direction of the gradient vector at the current position x. Such a target vertex is defined as

$$u(x) \in \text{argmax}_{u\in\mathcal{U}(x)} \langle \nabla f(x), u \rangle \tag{45}$$

At each iteration, we assume that the iterate x^k is represented as a convex combination of active vertices $x^k = \sum_{u\in\mathcal{U}} \mu^k_u u$, where $\mu^k \in \Delta(\mathcal{U})$. In this case, the sets $U^k = \mathcal{U}(x^k)$ and the carrying measure $\mu^k = \mu^{x^k}$ provide a compact representation of x^k. The ASFW scheme updates the thus described representation (U^k, μ^k) via the *vertex representation updating* (VRU) scheme, as defined in [4]. A single iteration of ASFW can perform two different updating steps:

1. *Forward Step* This update is constructed in the same way as `FWGSC`.
2. *Away Step* This is a correction step in which the weight of a single vertex is reduced, or even nullified. Specifically, the away step regime builds on the following ideas: Let $x \in \mathcal{X}$ be the current position of the algorithm with vertex representation $x = \sum_{u\in\mathcal{U}} \mu^x_u u$. Pick $u(x)$ as in (45). Define the *away direction*

$$v_A(x) \triangleq x - u(x), \tag{46}$$

and apply the step size $t > 0$ to produce the new point

$$\begin{aligned} x_t^+ &= x + t v_{\mathrm{A}}(x) \\ &= \sum_{u \in \mathcal{U}(x) \setminus \{u(x)\}} (1+t)\mu_u^x u + \left(\mu_{u(x)}^x (1+t) - t\right) u(x). \end{aligned}$$

Choosing $t \equiv \bar{t}(x) \triangleq \frac{\mu_{u(x)}^x}{1-\mu_{u(x)}^x}$ eliminates the vertex $u = u(x)$ from the support of the current point x and leaves us with the new position $x^+ = x_{\bar{t}(x)}^+ = \sum_{u \in \mathcal{U}(x) \setminus \{u(x)\}} \frac{\mu_u^x}{1-\mu_{u(x)}^x} u$. This vertex removal is called a *drop step.*

For the complexity analysis of ASFWGSC, we introduce some convenient notation. Define the vector field $v : \mathcal{X} \to \mathbb{R}^n$ by

$$v(x) \triangleq \begin{cases} v_{\mathrm{FW}}(x) & \text{if a Forward Step is performed,} \\ v_{\mathrm{A}}(x) & \text{if an Away Step is performed.} \end{cases} \tag{47}$$

The modified gap function is

$$G(x) \triangleq -\langle \nabla f(x), v(x) \rangle = \max\{\langle \nabla f(x), x - s(x) \rangle, \langle \nabla f(x), u(x) - x \rangle\}. \tag{48}$$

Algorithm 8: ASFWGSC

$x^0 \in \operatorname{dom} f \cap \mathcal{U}$ where $\mu_u^1 = 0$ for all $u \in \mathcal{U} \setminus \{x^1\}$ and $U^1 = \{x^1\}$.
for $k = 0, 1, \ldots$ **do**
  Set $s^k = s(x^k)$, $u^k = u(x^k)$, and $v_{\mathrm{A}}(x^k) = x^k - u^k$, $v_{\mathrm{FW}}(x^k) = s^k - x^k$
  if $\langle \nabla f(x^k), s^k - x^k \rangle \leq \langle \nabla f(x^k), x^k - u^k \rangle$ **then**
    Set $v^k = v_{\mathrm{FW}}(x^k)$
  else
    Set $v^k = v_{\mathrm{A}}(x^k)$
  end if
  Set $\beta_k = \|v^k\|_2$, $\mathrm{e}_k = \|v^k\|_{x^k}$, $\bar{t}_k \equiv \bar{t}(x^k)$ defined in (49)
  Find $\alpha_k = \operatorname{argmin}_{t \in [0, \bar{t}_k]} t\langle \nabla f(x^k), v^k \rangle + t^2 \mathrm{e}_k^2 \omega_v(t M_f \delta_v(x^k))$
  Update $x^{k+1} = x^k + \alpha_k v^k$
  if $v^k = v_{\mathrm{FW}}(x^k)$ **then**
    Update $U^{k+1} = U^k \cup \{s^k\}$
  else
    if $v^k = v_{\mathrm{A}}(x^k)$ and $\alpha_k = \bar{t}_k$ **then**
      Update $U^{k+1} = U^k \setminus \{u^k\}$ and μ^{k+1} via the VRU of [4].
    else
      Update $U^{k+1} = U^k$
    end if
  end if
end for

One observes that $G(x) \geq 0$ for all $x \in \operatorname{dom} f \cap \mathcal{X}$. To construct a feasible method, we need to impose bounds on the step-size. To that end, define

$$\bar{t}(x) \triangleq \begin{cases} 1 & \text{if a Forward Step is performed,} \\ \frac{\mu_{u(x)}}{1-\mu_{u(x)}} & \text{if an Away Step is performed,} \end{cases} \tag{49}$$

where $\{\mu_u\}_{u \in \mathcal{U}} \in \Delta(\mathcal{U})$ is a given vertex representation of the current point x, and $u(x)$ is the target state identified under the away-step regime (45).
The construction of our step size policy is based on an optimization argument, similar to the one used in the construction of FWGSC. In order to avoid unnecessary repetitions, we thus only spell out the main steps.

Recall that if $\mathsf{d}_\nu(x, x + tv(x)) < 1$, then we can apply the generalized self-concordant descent lemma (9):

$$f(x + tv(x)) \leq f(x) + t\langle \nabla f(x), v(x)\rangle + t^2 \|v(x)\|_x^2 \omega_\nu(tM_f \delta_\nu(x)),$$

where $\delta_\nu(x)$ is defined as in (12), modulo the change $\beta(x) = \|v(x)\|_2$ and $\mathfrak{e}(x) = \|v(x)\|_x$. Using the modified gap function (48), this gives the upper model for the objective function

$$f(x + tv(x)) \leq f(x) - G(x)\left[t - t^2 \frac{\mathfrak{e}(x)^2}{G(x)} \omega_\nu(tM_f \delta_\nu(x))\right],$$

provided that $G(x) > 0$. This upper model is structurally equivalent to the one employed in the step-size analysis of FWGSC. Hence, to obtain an adaptive step-size rule in Algorithm 8, we solve the concave program

$$\max_{t \geq 0} \tilde{\eta}_{x,\nu}(t) \triangleq t - t^2 \frac{\mathfrak{e}(x)^2}{G(x)} \omega_\nu(tM_f \delta_\nu(x)). \tag{50}$$

As in Sect. 3.2, and with some deliberate abuse of notation, let us denote the unique solution to this maximization problem by $\mathfrak{t}_\nu(x)$ (dependence on M_f is suppressed here, since we consider this parameter as given and fixed in this regime). Building on the insights we gained from proving Proposition 1, we thus obtain the familiarly looking characterization of the unique maximizer of the concave program (50):

Theorem 5 *The unique solution to program* (50) *is given by*

$$\mathfrak{t}_\nu(x) = \begin{cases} \frac{1}{M_f \delta_2(x)} \ln\left(1 + \frac{G(x) M_f \delta_2(x)}{\mathfrak{e}(x)^2}\right) & \text{if } \nu = 2, \\ \frac{1}{M_f \delta_\nu(x)} \left[1 - \left(1 + \frac{M_f \delta_\nu(x) G(x)}{\mathfrak{e}(x)^2} \frac{4-\nu}{\nu-2}\right)^{-\frac{\nu-2}{4-\nu}}\right] & \text{if } \nu \in (2, 3), \\ \frac{G(x)}{M_f \delta_3(x) G(x) + \mathfrak{e}(x)^2} & \text{if } \nu = 3, \end{cases} \tag{51}$$

where $\delta_\nu(x)$ is defined in eq. (12)*, with $\beta(x) = \|v(x)\|_2$ and $\mathfrak{e}(x) = \|v(x)\|_x$ considering the vector field* (47)*.*

Analogously to Proposition 2, we see that when applying the step-size policy

$$\alpha_\nu(x) \triangleq \min\{\bar{t}(x), \mathtt{t}_\nu(x)\}, \tag{52}$$

we can guarantee that $x^k \in \mathcal{X}$ for all $k \geq 0$. Indeed, inspecting the expression (51) for each value $\nu \in [2, 3]$, it is easy to see that $M_f \delta_\nu(x) \mathtt{t}_\nu(x) < 1$. Hence, if $\bar{t}(x) \leq \mathtt{t}_\nu(x)$, it is immediate that $\bar{t}(x) M_f \delta_\nu(x) < 1$. Consequently, $x + \alpha_\nu(x) v(x) \in \mathcal{X} \cap \operatorname{dom} f$ for all $x \in \mathcal{X} \cap \operatorname{dom} f$. Therefore, the sequence generated by Algorithm 8 is always well defined. In terms of the thus constructed process $\{x^k\}_{k \geq 0}$, we can quantify the per-iteration progress $\Delta_k \equiv \tilde{\eta}_{x^k,\nu}(\alpha_k)$, setting $\alpha_k \equiv \alpha_\nu(x^k)$, via the following modified version of Lemma 5:

Lemma 7 *If $\mathtt{t}_\nu(x) \leq \bar{t}(x)$, we have*

$$\Delta_k \geq \tilde{\Delta}_k \triangleq \begin{cases} \frac{2\ln(2)-1}{\operatorname{diam}(\mathcal{X})} \min\left\{\frac{G(x^k)}{M_f}, \frac{G(x^k)^2}{\operatorname{diam}(\mathcal{X}) L_{\nabla f}}\right\} & \text{if } \nu = 2, \\ \frac{\tilde{\gamma}_\nu}{\operatorname{diam}(\mathcal{X})} \min\left\{\frac{G(x^k)}{(\frac{\nu}{2}-1) M_f L_{\nabla f}^{(\nu-2)/2}}, \frac{-1}{\mathfrak{b}} \frac{G(x^k)^2}{L_{\nabla f} \operatorname{diam}(\mathcal{X})}\right\} & \text{if } \nu \in (2, 3), \\ \frac{2(1-\ln(2))}{\sqrt{L_{\nabla f}} \operatorname{diam}(\mathcal{X})} \min\left\{\frac{G(x^k)}{M_f}, \frac{G(x^k)^2}{\sqrt{L_{\nabla f}} \operatorname{diam}(\mathcal{X})}\right\} & \text{if } \nu = 3, \end{cases} \tag{53}$$

where $\tilde{\gamma}_\nu \triangleq 1 + \frac{4-\nu}{2(3-\nu)} \left(1 - 2^{2(3-\nu)/(4-\nu)}\right)$ and $\mathfrak{b} \triangleq \frac{2-\nu}{4-\nu}$.

This means that at each iteration of Algorithm 8 in which $\alpha_k = \mathtt{t}_\nu(x^k)$, we succeed in reducing the objective function value by at least

$$f(x^{k+1}) \leq f(x^k) - \tilde{\Delta}_k.$$

To proceed further with the complexity analysis of `ASFWGSC`, we need the following technical angle condition, valid for polytope domains:

Lemma 8 (Corollary 3.1, [4]) *For any $x \in \mathcal{X} \setminus \mathcal{X}^*$ with support $\mathcal{U}(x)$, we have*

$$\max_{u \in \mathcal{U}(x), w \in \mathcal{U}} \langle \nabla f(x), u - w \rangle \geq \frac{\Omega_{\mathcal{X}}}{|\mathcal{U}(x)|} \max_{x^* \in \mathcal{X}^*} \frac{\langle \nabla f(x), x - x^* \rangle}{\|x - x^*\|}, \tag{54}$$

where

$$\zeta \triangleq \min_{u \in \mathcal{U}, i \in \{1,\ldots,m\} : b_i > (\mathbf{B}u)_i} (b_i - \mathbf{B}_i u), \ \varphi \triangleq \max_{i \in \{1,\ldots,m\} \setminus I(x)} \|\mathbf{B}_i\|, \text{ and } \Omega_{\mathcal{X}} \triangleq \frac{\zeta}{\varphi}.$$

To assess the overall iteration complexity of Algorithm 8 we consider separately the following cases:

(a) If the step size regime $\alpha_k = \mathtt{t}_\nu(x^k)$ applies, then from Proposition 4 we deduce that $f(x^{k+1}) - f(x^k) \leq -\Delta_k$, were

$$\Delta^k \geq \min\{\mathtt{c}_1(M_f, \nu) G(x^k), \mathtt{c}_2(M_f, \nu) G(x^k)^2\}.$$

The multiplicative constants $\mathtt{c}_1(M_f, \nu)$, $\mathtt{c}_2(M_f, \nu)$ are the ones defined in (23) and (24). Hence,

$$f(x^{k+1}) - f(x^k) \leq -\min\{\mathtt{c}_1(M_f, \nu)G(x^k), \mathtt{c}_2(M_f, \nu)G(x^k)^2\}.$$

(b) Else, we apply the step size $\alpha_k = \bar{t}_k$. Then, there are two cases to consider:

(b.i) If a Forward Step is applied, then we know that $\bar{t}_k = 1$. Since $1 < \mathtt{t}_\nu(x^k)$, we can apply Lemma 4, but now evaluating the function $\tilde{\eta}_{x,\nu}(t)$ at $t = 1$, to obtain the bound

$$\frac{\tilde{\eta}_{x^k,\nu}(\bar{t}_k)}{G(x^k)} \geq \frac{1}{2}.$$

This gives the per-iteration progress

$$f(x^{k+1}) - f(x^k) \leq -\frac{1}{2}G(x^k).$$

(b.ii) If an Away Step is applied, then we do not have a lower bound on $\bar{t}_k$. However, we know that $f(x^{k+1}) - f(x^*) \leq f(x^k) - f(x^*)$. As in [4], we know that such drop steps can happen at most half of the iterations.

Collecting these cases, we are ready to state and prove the main result of this section.

Theorem 6 *Let $\{x^k\}_{k\geq 0}$ be the trajectory generated by* Algorithm 8 *(ASFWGSC). Suppose that* Assumptions 1, 3 and 4 *are in place. Then, for all $k \geq 0$ we have*

$$h_k \leq (1-\theta)^{k/2} h_0 \leq \exp\left(-\theta\frac{k}{2}\right) h_0. \tag{55}$$

where $\theta \triangleq \min\left\{\frac{1}{2}, \frac{\mathtt{c}_1(M_f,\nu)\Omega}{2\,\mathrm{diam}(\mathcal{X})}, \frac{\mathtt{c}_2(M_f,\nu)\Omega^2\sigma_f}{8}\right\}$, $\Omega \equiv \frac{\Omega_{\mathcal{X}}}{|\mathcal{U}|}$.

Proof We say that iteration k is productive if it is either a Forward step or an Away step, which is not a drop step. Based on the estimates developed by inspecting thes cases (a) and (b.i) above, we see that at all productive steps we reduce the objective function value according to

$$f(x^{k+1}) - f(x^k) \leq -\min\left\{\min\{\frac{1}{2}, \mathtt{c}_1(M_f, \nu)\}G(x^k), c_2(M_f, \nu)G(x^k)^2\right\}.$$

We now develop a uniform bound for this decrease.

First, we recall that on the level set $\mathcal{S}(x^0)$, we have the strong convexity estimate

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

Using Lemma 8 and the definition of an Away-Step, we obtain the bound

$$\langle \nabla f(x^k), u^k - s^k \rangle \geq \frac{\Omega}{\|x^k - x^*\|} \langle \nabla f(x^k), x^k - x^* \rangle,$$

where $\Omega \equiv \frac{\Omega_{\mathcal{X}}}{|\mathcal{U}|} \leq \frac{\Omega_{\mathcal{X}}}{|\mathcal{U}(x^k)|}$. At the same time,

$$\begin{aligned}\langle \nabla f(x^k), u^k - s^k \rangle &= \langle \nabla f(x^k), u^k - x^k \rangle + \langle \nabla f(x^k), x^k - s^k \rangle \\ &\leq 2 \max \left\{ \langle \nabla f(x^k), u^k - x^k \rangle, \langle \nabla f(x^k), x^k - s^k \rangle \right\} \\ &= 2G(x^k).\end{aligned}$$

Consequently,

$$G(x^k) \geq \frac{1}{2} \langle \nabla f(x^k), u^k - s^k \rangle, \tag{56}$$

and

$$\begin{aligned}G(x^k) &\geq \frac{1}{2} \langle \nabla f(x^k), u^k - s^k \rangle \geq \frac{\Omega}{2\|x^k - x^*\|} \langle \nabla f(x^k), x^k - x^* \rangle \\ &\geq \frac{\Omega}{2\|x^k - x^*\|} (f(x^k) - f^*) \geq \frac{\Omega}{2 \operatorname{diam}(\mathcal{X})} (f(x^k) - f^*).\end{aligned}$$

Furthermore,

$$\begin{aligned}G(x^k)^2 &\geq \frac{\Omega^2}{4\|x^k - x^*\|^2} (f(x^k) - f^*)^2 \geq \frac{\Omega^2}{4} \frac{(f(x^k) - f^*)^2}{\frac{2}{\sigma_f}(f(x^k) - f^*)} \\ &= \frac{\Omega^2 \sigma_f}{8} (f(x^k) - f^*).\end{aligned}$$

Hence, in the cases (a) and (b.i), we can lower bound the per-iteration progress in terms of the approximation error $h_k = f(x^k) - f^*$ as

$$h_{k+1} - h_k \leq -\min \left\{ \frac{1}{2}, \frac{c_1(M_f, \nu)\Omega}{2 \operatorname{diam}(\mathcal{X})}, \frac{c_2(M_f, \nu)\Omega^2 \sigma_f}{8} \right\} h_k \equiv -\theta h_k.$$

Since we are making a full drop step in at most $k/2$ iterations (recall that we initialize the algorithm from a vertex), we conclude from this that

$$h_k \leq (1 - \theta)^{k/2} h_0 \leq \exp\left(-\theta \frac{k}{2}\right) h_0.$$

□

Remark 7 We would like to point out that Algorithm `ASFWGSC` does not need to know the constants σ_f, $L_{\nabla f}$ which may be hard to estimate. Moreover, the constants

in Lemma 8 are also used only in the analysis and are not required to run the algorithm. Compared to [10], our ASFW does not rely on the backtracking line search, but requires to evaluate the Hessian, yet without its inversion. Furthermore, our method does not involve the pyramidal width of the feasible set, which is in general extremely difficult to evaluate.

6 Numerical results

We provide four examples to compare our methods with existing methods in the literature. As competitors we take Algorithm 1, with its specific versions `FW-Standard` and `FW-Line Search`.[2] As further benchmarks, we implement the self-concordant Proximal-Newton (`PN`) and the Proximal-Gradient (`PG`) of [50,52], as available in the SCOPT package.[3] 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 Intel(R) Xeon(R) Gold 6254 CPU @ 3.10 GHz server with a total of 300 GB RAM and 72 threads, where each method was allowed to run on a maximum of two threads.

We ran all first order methods for a maximum 50,000 iterations and `PN`, which is more computationally expensive, for a maximum of 1000 iterations. `FW-Line Search` is run with a tolerance of 10^{-10}. In order to ensure that `FW-standard` generates feasible iterates for $\nu > 2$, we check if the next iterate is inside the domain; If not we replace the step-size by 0, as suggested in [10]. `PG` was only used in instances where $\nu = 3$ as this method has been developed for standard self-concordant functions only [52]. Within `PN` we use monotone FISTA [6], with at most 100 iterations and a tolerance of 10^{-5} to find the Newton direction. The step size used in `PG` is determined by the Barzilai-Borwein method [43] with a limit of 100 iterations, similar to [52]. `FWLLOO` was only implemented for experiments where the feasible set is the simplex, for which [24] provide an explicit LLOO, since the LLOO implementation for general polytopes suggested in [24] is non-trivial and involves calculating barycentric coordinates of the iterates.

Our comparison is made by the construction of performance profiles [15]. In order to present the result, we first estimate f^* by the best 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 $\mathcal{S}$, test problems $\mathcal{P}$ and initial points $\mathcal{I}$, denote by F_{ijl} the function value attained by method $j \in \mathcal{S}$ on problem $i \in \mathcal{P}$ starting from point $l \in \mathcal{I}$. We define the estimate of the optimal value of problem j by $f_j^* = \min\{F_{ijl} | j \in \mathcal{S}, l \in \mathcal{I}\}$. Denoting $\{x_{ijl}^k\}_k$ the sequence produced by method j on problem i starting from point l, we define the *relative error* as $r_{ijl}^k = \frac{f(x_{ijl}^k) - f_j^*}{f_j^*}$.

Now, for all methods $j \in \mathcal{S}$ and any relative error ε, we compute the proportion of data sets that achieve a relative error of at most ε (*successful instances*). We construct this statistic as follows: Let $\bar{N}_j$ denote the maximum allowed number of iterations for

[2] At the time of writing this paper, there did not exist a general convergence proof for FW-standard in the generalized self-concordance setting. In the meantime, in the related paper [10], a modified version of FW-standard admitting a sublinear convergence rate similar to the methods in this paper was developed.

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

method $j \in \mathcal{S}$ (i.e for first-order methods 50,000 and for PN 1000). Define $\mathcal{I}_{ij}(\varepsilon) \triangleq \{l \in \mathcal{I} : \exists k \leq \bar{N}_j, r_{ijl}^k \leq \varepsilon\}$. Then, the proportion of successful instances is

$$\rho_j(\varepsilon) \triangleq \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} |\mathcal{I}_{ij}(\varepsilon)| \quad \text{(average success ratio)}.$$

We are also interested in comparing the iteration complexity and CPU time. For that purpose, we define $N_{ijl}(\varepsilon) \triangleq \min\{0 \leq k \leq \bar{N}_j | r_{ijl}^k \leq \varepsilon\}$ as the first iteration in which method $j \in \mathcal{S}$ achieves a relative error ε on problem $i \in \mathcal{P}$ starting from point $l \in \mathcal{I}$. Analogously, $T_{ijl}(\varepsilon)$ measures the minimal CPU time in which method $j \in \mathcal{S}$ achieves a relative error ε on problem $i \in \mathcal{P}$ starting from point $l \in \mathcal{I}$. For comparing the iteration complexity and the CPU time across methods we construct the following statistics:

$$\tilde{\rho}_j(\varepsilon) \triangleq \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{1}{|\mathcal{I}_{ij(\varepsilon)}|} \sum_{l \in \mathcal{I}_{ij}(\epsilon)} \frac{N_{ijl}(\varepsilon)}{\min\{N_{isl}(\varepsilon) | s \in \mathcal{S}\}} \quad \text{(average iteration ratio)},$$

$$\hat{\rho}_j(\varepsilon) \triangleq \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{1}{|\mathcal{I}_{ij}(\varepsilon)|} \sum_{l \in \mathcal{I}_{ij}} \frac{T_{ijl}(\varepsilon)}{\min\{T_{isl}(\varepsilon) | s \in \mathcal{S}\}} \quad \text{(average time ratio)}.$$

Besides average performance, we also report the mean and standard deviation of $N_{ijl}(\varepsilon)$ and $T_{ijl}(\varepsilon)$ across starting points, for specific values of relative error ε for all tested methods and data sets.

6.1 Logistic regression

Starting with [2], the logistic regression problem has been the main motivation from the perspective of statistical theory to analyze self-concordant functions in detail. The objective function involved in this standard classification problem is given by

$$f(x) = \frac{1}{p} \sum_{i=1}^{p} \ln\left(1 + \exp\left(-y_i(\langle a_i, x\rangle + \mu)\right)\right) + \frac{\gamma}{2}\|x\|_2^2. \tag{57}$$

Here μ is a given intercept, $y_i \in \{-1, 1\}$ is the label attached to the i-th observation, and $a_i \in \mathbb{R}^n$ are predictors given as input data for $i = 1, 2, \ldots, p$. The regularization parameter $\gamma > 0$ is usually calibrated via cross-validation. The task is to learn a linear hypothesis $x \in \mathbb{R}^n$. According to [50], we can treat (57) as a $(M_f^{(3)}, 3)$-GSC function minimization problem with $M_f^{(3)} \triangleq \frac{1}{\sqrt{\gamma}} \max\{\|a_i\|_2 | 1 \leq i \leq p\}$. On the other hand, we can also consider it as a $(M_f^{(2)}, 2)$-GSC minimization problem with $M_f^{(2)} \triangleq \max\{\|a_i\|_2 | 1 \leq i \leq p\}$. It is important to observe that the regularization parameter $\gamma > 0$ affects the self-concordant parameter $M_f^{(3)}$ but not $M_f^{(2)}$. This gains relevance, since usually the regularization parameter is negatively correlated with the sample size p. Hence, for $p \gg 1$, the GSC constant M_f could differ by orders of

magnitude, which suggests considerable differences in the performance of numerical algorithms.

We consider the elastic net formulation of the logistic regression problems [58], by enforcing sparsity of the estimators via an added ℓ_1 penalty. The resulting optimization problem reads as

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t. } \|x\|_1 \leq R$$

This introduces another free parameter $R > 0$, which can be treated as another hyper-parameter just like γ.

We test our algorithms using $R = 10$, $\mu = 0$ and $\gamma = 1/p$, where a_i and y_i are based on data sets a1a–a9a from the LIBSVM library [12], where the predictors are normalized so that $\|a_i\| = 1$. Hence, $M_f^{(2)}/M_f^{(3)} = p^{-1/2}$. For each data set, the methods were ran for 10 randomly generated starting points, where each starting point was chosen as a random vertex of the ℓ_1 ball with radius 10.

We first compare the methods that are affected by the value of $\nu \in \{2, 3\}$ and $M_f \in \{M_f^{(2)}, M_f^{(3)}\}$, i.e. `FWGSC`, `MBTFWGSC`, `ASFWGSC`, and `PN`. We display the comparison of the average relative error over the starting points versus iteration and time for data set a9a in Fig. 1. Note that for this data set we have $p = 32,561$. It is apparent that the linearly convergent methods `ASFWGSC` and `PN` gain the most benefit from the lower M_f associated with the shift from $\nu = 3$ to $\nu = 2$, reducing both iteration complexity and time. Moreover, for `FWGSC` and `MBTFWGSC` the change of ν only seems to benefit the method in earlier iteration, but does not create any asymptotic speedup. Specifically, the benefit for `MBTFWGSC` is very small, probably since the backtracking procedure already takes advantage of the possible increase in the step-size that is partially responsible for the improved performance in the other methods. We observed the same behavior for all other data sets considered. Thus, we next compare these methods with $\nu = 2$ to the `MBTFWGSC`, `FW-standard`, `FW-Line Search`, and `PG` and display the performance of all tested methods using the aggregate statistics $\rho(\varepsilon)$, $\tilde{\rho}(\varepsilon)$, $\hat{\rho}(\varepsilon)$, in Fig. 2. Table 2 reports statistics for $N(\varepsilon)$ and $T(\varepsilon)$ for each individual data set. `PG` has the best performance in terms of time to reach a certain value of relative error, followed by `FW-standard` and `ASFWGSC`. `FW-standard` is slightly better for relative error higher than 10^{-5} but becomes

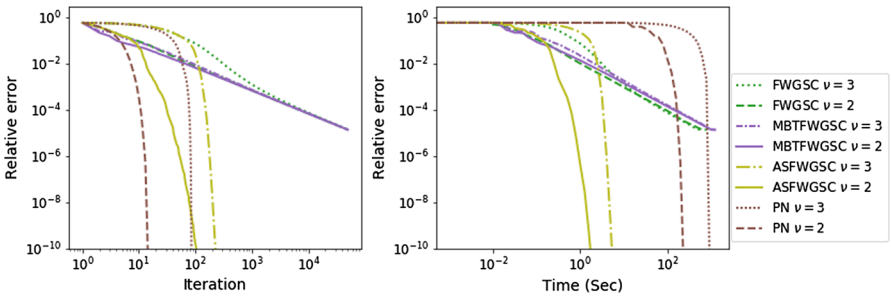
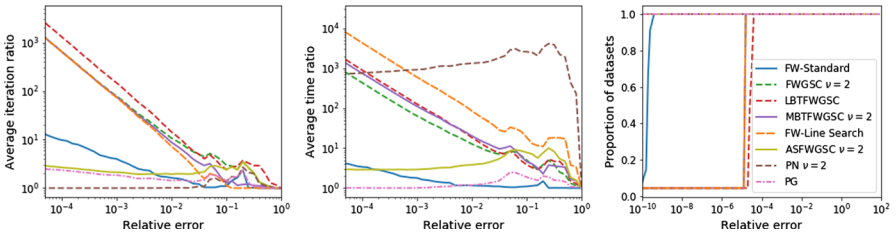


Fig. 1 Comparison between $\nu = 3$ and $\nu = 2$ for data set a9a



(a) Average iteration ratio $\tilde{\rho}(\varepsilon)$. **(b)** Average time ratio $\hat{\rho}(\varepsilon)$. **(c)** Average success ratio $\rho(\varepsilon)$.

Fig. 2 Performance profile for the logistic regression problem (57) obtained after averaging over 9 binary classification problems

inferior to ASFWGSC for lower error values. From this example we conclude that given a choice using the FW algorithms with lower parameter ν is preferable since the upper bound obtained by this choice is tighter. The fact that we want to use the GSC setting with $\nu = 2$ even when the problem can be formulated in the standard self concordant setting $\nu = 3$ provides motivation for developing methods and analyses for the GSC case. Moreover, the advantage of ASFWGSC over other FW methods is in its ability to achieve higher accuracy in lower iteration complexity, however, this may be hindered by its higher computational complexity per iteration.

6.2 Portfolio optimization with logarithmic utility

We study high-dimensional portfolio optimization problems with logarithmic utility [13]. In this problem there are n assets with returns $r_t \in \mathbb{R}^n_+$ in period t of the investment horizon. More precisely, r_t measures the return as the ratio between the closing price of the current day $R_{t,i}$ and the previous day $R_{t-1,i}$, i.e. $r_{t,i} = R_{t,i}/R_{t-1,i}$, $1 \leq i \leq n$. The utility function of the investor is given as

$$f(x) = -\sum_{t=1}^{p} \log(r_t^\top x).$$

Based on historical observations r_t, $t \in \{1, \ldots, p\}$, our task is to design a portfolio x solving the problem

$$\min_{x \in \mathbb{R}^n} f(x) \text{ s.t.: } x_i \geq 0, \sum_{i=1}^{n} x_i = 1. \tag{58}$$

Since f is the sum of n standard self-concordant functions, we know that $f \in \mathcal{F}_{2,3}$ with effective domain $\operatorname{dom} f = \{x \in \mathbb{R}^n | r_t^\top x > 0 \text{ for all } 1 \leq t \leq p\}$. We remark that this self-concordant minimization problem gains relevance when trying to find the best constant rebalanced portfolio. Moreover, this problem has connections to the universal prediction problem in information theory [38] and online portfolio optimization [11] for which the performance of problem (58) can serve as a benchmark.

For this example, computing a LLOO with $\rho = \sqrt{n}$ is simple and a complete description can be found in [24]. Therefore, we also ran algorithm FWLLOO, where

Table 2 Results for logistic regression problem (57)

Problem			FW-Standard			FWGSC $\nu = 2$			LBTFWGSC		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−04											
a1a	128	1605	93.7 (2.1)	0.03 (0.00)	7.39e−05 (1.57e−05)	6467.2 (2129.3)	2.64 (0.88)	9.96e−05 (1.19e−06)	16240.5 (5370.3)	9.31 (3.08)	9.99e−05 (3.67e−07)
a2a	128	2265	96.9 (12.0)	0.03 (0.00)	8.90e−05 (1.10e−05)	6095.9 (2007.5)	3.49 (1.16)	1.00e−04 (3.79e−08)	15927.6 (5269.2)	12.68 (4.23)	9.98e−05 (7.33e−07)
a3a	128	3185	98.7 (16.1)	0.05 (0.01)	9.54e−05 (3.65e−06)	6090.8 (2002.0)	5.20 (1.75)	9.99e−05 (3.08e−07)	16356.7 (5409.4)	18.21 (6.03)	9.99e−05 (3.40e−07)
a4a	128	4781	89.0 (12.9)	0.07 (0.01)	8.50e−05 (1.11e−05)	5982.7 (1968.3)	9.43 (3.33)	9.95e−05 (1.39e−06)	11324.8 (3735.1)	19.62 (6.48)	9.97e−05 (8.14e−07)
a5a	128	6414	117.2 (4.1)	0.14 (0.01)	6.61e−05 (1.34e−05)	6862.0 (25.6)	15.45 (0.43)	1.00e−04 (8.50e−09)	11166.4 (33.0)	26.73 (0.93)	1.00e−04 (5.59e−09)
a6a	128	11220	89.2 (4.9)	0.21 (0.01)	9.06e−05 (1.17e−05)	6670.6 (20.8)	28.69 (0.96)	1.00e−04 (3.49e−09)	12305.2 (256.8)	55.67 (1.32)	1.00e−04 (2.13e−09)
a7a	128	16100	97.0 (4.6)	0.32 (0.01)	8.43e−05 (3.80e−06)	6661.1 (21.5)	41.57 (0.74)	1.00e−04 (7.13e−09)	11313.9 (30.0)	71.98 (1.02)	1.00e−04 (7.00e−09)
a8a	128	22696	86.6 (6.3)	0.41 (0.03)	9.58e−05 (1.13e−06)	6698.6 (19.8)	61.31 (1.38)	1.00e−04 (2.78e−09)	11399.7 (62.9)	104.56 (1.18)	1.00e−04 (4.41e−09)
a9a	128	32561	87.0 (0.0)	0.61 (0.01)	6.69e−05 (8.92e−06)	6821.1 (18.3)	90.59 (1.72)	1.00e−04 (9.86e−09)	11036.1 (28.9)	143.32 (0.64)	1.00e−04 (5.57e−09)

Problem			MBTFWGSC $\nu = 2$			FW-Line Search			ASFWGSC $\nu = 2$		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−04											
a1a	128	1605	6432.3 (2120.3)	5.27 (1.75)	9.97e−05 (9.52e−07)	6404.5 (2113.0)	28.23 (9.56)	9.97e−05 (7.97e−07)	21.6 (2.7)	0.02 (0.01)	8.48e−05 (9.21e−06)
a2a	128	2265	6062.2 (1993.9)	6.88 (2.28)	1.00e−04 (5.87e−08)	6029.8 (1989.8)	38.64 (12.84)	9.99e−05 (2.87e−07)	23.8 (3.0)	0.02 (0.01)	7.32e−05 (8.93e−06)
a3a	128	3185	6065.6 (1954.0)	11.23 (3.65)	1.00e−04 (1.40e−08)	6026.0 (1984.1)	55.10 (18.37)	9.98e−05 (5.03e−07)	25.5 (2.1)	0.04 (0.01)	8.58e−05 (7.79e−06)
a4a	128	4781	5944.5 (1955.6)	15.83 (5.23)	1.00e−04 (5.48e−08)	5916.7 (1949.6)	83.60 (28.05)	9.99e−05 (2.53e−07)	28.2 (1.2)	0.07 (0.02)	8.86e−05 (7.53e−06)
a5a	128	6414	6845.2 (26.1)	25.43 (1.24)	1.00e−04 (8.30e−09)	6829.6 (25.7)	143.23 (13.10)	1.00e−04 (4.22e−09)	23.4 (6.5)	0.09 (0.03)	8.99e−05 (8.06e−06)
a6a	128	11220	6632.3 (21.4)	51.29 (2.24)	1.00e−04 (3.65e−09)	6604.8 (21.7)	271.42 (3.59)	1.00e−04 (1.34e−08)	25.0 (5.3)	0.19 (0.03)	8.34e−05 (9.55e−06)
a7a	128	16100	6623.4 (20.6)	69.14 (1.35)	1.00e−04 (4.59e−09)	6591.4 (19.7)	431.07 (6.12)	1.00e−04 (1.11e−08)	25.1 (7.8)	0.25 (0.09)	8.54e−05 (1.90e−05)

Table 2 continued

Problem			MBTFWGSC $\nu = 2$			FW-Line Search			ASFWGSC $\nu = 2$		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
a8a	128	22696	6661.2 (19.3)	105.23 (4.21)	1.00e−04 (1.26e−08)	6668.6 (20.4)	637.62 (7.40)	1.00e−04 (6.42e−09)	27.0 (6.6)	0.47 (0.11)	9.09e−05 (1.02e−05)
a9a	128	32561	6782.6 (17.0)	140.57 (3.18)	1.00e−04 (1.06e−08)	6749.1 (18.5)	941.31 (1.88)	1.00e−04 (8.88e−09)	30.2 (4.6)	0.56 (0.12)	6.10e−05 (1.65e−05)

Problem			PN $\nu = 2$			PG		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−04								
a1a	128	1605	9.0 (1.2)	4.11 (0.33)	5.33e−05 (2.41e−05)	22.6 (4.2)	**0.01 (0.00)**	7.09e−05 (2.90e−05)
a2a	128	2265	9.0 (1.1)	5.76 (0.59)	6.36e−05 (2.26e−05)	23.8 (2.6)	**0.02 (0.00)**	7.89e−05 (1.30e−05)
a3a	128	3185	9.7 (1.3)	8.35 (0.81)	2.55e−05 (1.92e−05)	23.8 (3.4)	**0.02 (0.00)**	8.19e−05 (1.46e−05)
a4a	128	4781	9.3 (1.3)	15.80 (2.36)	5.38e−05 (2.07e−05)	23.6 (4.8)	**0.03 (0.01)**	5.39e−05 (2.33e−05)
a5a	128	6414	10.1 (0.3)	27.04 (1.46)	3.64e−05 (7.11e−06)	20.4 (2.9)	**0.03 (0.01)**	7.64e−05 (1.84e−05)
a6a	128	11220	9.9 (0.5)	50.91 (3.33)	2.55e−05 (2.95e−05)	20.4 (2.0)	**0.06 (0.01)**	7.25e−05 (1.79e−05)
a7a	128	16100	10.1 (0.3)	76.51 (3.87)	1.25e−05 (2.83e−06)	22.1 (3.8)	**0.08 (0.02)**	5.55e−05 (2.75e−05)
a8a	128	22696	10.1 (0.3)	109.09 (3.48)	1.80e−05 (6.37e−06)	22.1 (3.0)	**0.11 (0.02)**	6.87e−05 (2.12e−05)
a9a	128	32561	10.1 (0.3)	165.20 (5.99)	3.02e−05 (4.13e−06)	21.1 (3.1)	**0.15 (0.03)**	7.83e−05 (1.90e−05)

Problem			FW-Standard			FWGSC $\nu = 2$			LBTFWGSC		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−06											
a1a	128	1605	846.9 (129.4)	0.21 (0.03)	9.18e−07 (3.40e−08)	*45019.0 (14946.0)	17.27 (5.74)	1.30e−05 (4.01e−06)	*45038.0 (14889.0)	25.55 (8.46)	3.27e−05 (1.06e−05)
a2a	128	2265	765.4 (141.6)	0.26 (0.05)	7.86e−07 (2.28e−07)	*45054.9 (14838.3)	25.55 (8.48)	1.23e−05 (3.75e−06)	*45822.4 (12535.8)	36.15 (9.94)	3.21e−05 (1.04e−05)
a3a	128	3185	836.9 (111.4)	0.41 (0.07)	9.34e−07 (6.11e−08)	*45020.8 (14940.6)	38.56 (12.94)	1.22e−05 (3.75e−06)	*45036.3 (14894.1)	49.78 (16.48)	3.31e−05 (1.07e−05)
a4a	128	4781	786.2 (119.0)	0.61 (0.10)	9.29e−07 (7.43e−08)	*45101.5 (14698.5)	69.02 (22.74)	1.20e−05 (3.67e−06)	*45030.5 (14911.5)	75.72 (25.18)	2.28e−05 (7.28e−06)

Table 2 continued

Problem			FW-Standard			FWGSC $\nu = 2$			LBTFWGSC		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
a5a	128	6414	787.2 (90.5)	0.91 (0.12)	9.63e−07 (5.37e−08)	*50001.0 (0.0)	114.93 (2.82)	1.37e−05 (9.74e−09)	*50001.0 (0.0)	116.70 (3.44)	2.24e−05 (2.00e−08)
a6a	128	11220	789.6 (46.1)	1.74 (0.11)	9.23e−07 (1.06e−07)	*50001.0 (0.0)	215.70 (3.56)	1.33e−05 (7.75e−09)	*50001.0 (0.0)	223.99 (0.88)	2.56e−05 (1.27e−07)
a7a	128	16100	803.1 (75.9)	2.62 (0.23)	8.90e−07 (3.91e−08)	*50001.0 (0.0)	311.24 (2.74)	1.33e−05 (7.61e−09)	*50001.0 (0.0)	315.61 (1.26)	2.27e−05 (1.55e−08)
a8a	128	22696	830.5 (55.8)	3.96 (0.28)	9.60e−07 (2.81e−08)	*50001.0 (0.0)	452.37 (4.36)	1.34e−05 (6.81e−09)	*50001.0 (0.0)	457.78 (2.56)	2.29e−05 (3.22e−08)
a9a	128	32561	747.0 (41.9)	5.21 (0.27)	9.59e−07 (3.10e−08)	*50001.0 (0.0)	625.00 (6.15)	1.36e−05 (6.46e−09)	*50001.0 (0.0)	648.40 (1.63)	2.22e−05 (1.52e−08)

Problem			MBTFWGSC $\nu = 2$			FW-Line Search			ASFWGSC $\nu = 2$		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−06											
a1a	128	1605	*45018.2 (14948.4)	32.02 (10.66)	1.30e−05 (4.01e−06)	*45017.6 (14950.2)	197.89 (67.20)	1.30e−05 (4.01e−06)	41.5 (2.6)	0.03 (0.01)	8.72e−07 (5.47e−08)
a2a	128	2265	*45350.1 (13952.7)	49.31 (15.24)	1.22e−05 (3.75e−06)	*45194.8 (14418.6)	290.94 (93.53)	1.22e−05 (3.75e−06)	41.9 (5.9)	0.04 (0.01)	8.52e−07 (9.17e−08)
a3a	128	3185	*45220.3 (14342.1)	72.77 (23.15)	1.22e−05 (3.74e−06)	*45157.0 (14532.0)	420.58 (136.98)	1.22e−05 (3.74e−06)	48.0 (4.2)	0.07 (0.01)	8.76e−07 (7.06e−08)
a4a	128	4781	*45254.7 (14238.9)	116.82 (37.04)	1.20e−05 (3.66e−06)	*45230.7 (14310.9)	636.09 (204.80)	1.20e−05 (3.66e−06)	53.3 (5.3)	0.11 (0.02)	8.47e−07 (1.32e−07)
a5a	128	6414	*50001.0 (0.0)	182.36 (6.11)	1.37e−05 (9.62e−09)	*50001.0 (0.0)	1045.25 (91.14)	1.37e−05 (5.91e−08)	43.3 (7.3)	0.16 (0.03)	8.50e−07 (9.53e−08)
a6a	128	11220	*50001.0 (0.0)	354.80 (13.93)	1.33e−05 (7.81e−09)	*50001.0 (0.0)	2048.42 (23.44)	1.33e−05 (8.16e−09)	46.7 (7.0)	0.33 (0.04)	9.15e−07 (6.69e−08)
a7a	128	16100	*50001.0 (0.0)	488.12 (6.14)	1.33e−05 (7.36e−09)	*50001.0 (0.0)	3262.60 (31.67)	1.33e−05 (7.13e−09)	47.9 (13.5)	0.49 (0.16)	8.77e−07 (9.21e−08)
a8a	128	22696	*50001.0 (0.0)	720.64 (10.06)	1.34e−05 (3.26e−08)	*50001.0 (0.0)	4774.39 (16.59)	1.34e−05 (7.11e−09)	44.1 (11.5)	0.76 (0.20)	8.93e−07 (8.30e−08)
a9a	128	32561	*50001.0 (0.0)	1041.81 (10.64)	1.36e−05 (6.40e−09)	*50001.0 (0.0)	6936.60 (26.73)	1.36e−05 (1.78e−08)	44.1 (8.0)	0.81 (0.18)	8.22e−07 (1.29e−07)

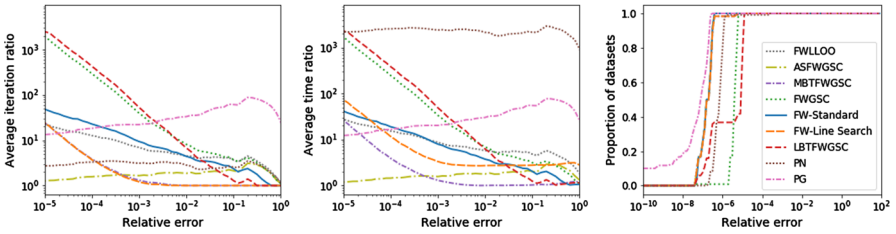
Problem			PN $\nu = 2$			PG		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−06								
a1a	128	1605	10.7 (1.3)	4.96 (0.32)	1.38e−07 (1.26e−07)	37.3 (3.8)	**0.02 (0.00)**	7.57e−07 (2.33e−07)
a2a	128	2265	11.2 (1.6)	7.24 (0.84)	3.73e−08 (4.76e−08)	37.4 (3.9)	**0.03 (0.00)**	6.08e−07 (3.10e−07)
a3a	128	3185	11.3 (1.6)	9.86 (1.06)	6.71e−08 (6.75e−08)	35.3 (5.0)	**0.03 (0.00)**	5.85e−07 (3.02e−07)

Table 2 continued

Problem			PN $\nu = 2$			PG		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error
a4a	128	4781	10.8 (1.3)	18.48 (2.28)	2.18e−07 (3.01e−07)	33.1 (2.9)	**0.04 (0.01)**	7.57e−07 (2.06e−07)
a5a	128	6414	11.1 (0.3)	29.85 (1.28)	6.62e−07 (2.44e−07)	32.6 (4.6)	**0.05 (0.01)**	5.63e−07 (2.88e−07)
a6a	128	11220	11.1 (0.3)	57.22 (2.90)	1.27e−07 (1.69e−07)	34.4 (3.5)	**0.09 (0.02)**	5.50e−07 (2.52e−07)
a7a	128	16100	11.1 (0.3)	84.39 (3.84)	1.82e−07 (7.37e−08)	34.2 (6.6)	**0.12 (0.03)**	4.99e−07 (3.34e−07)
a8a	128	22696	11.1 (0.3)	119.91 (3.69)	1.72e−07 (1.25e−07)	35.1 (6.0)	**0.17 (0.04)**	7.60e−07 (2.68e−07)
a9a	128	32561	12.0 (0.4)	196.76 (8.13)	8.46e−08 (2.08e−07)	33.0 (3.5)	**0.23 (0.03)**	7.97e−07 (1.74e−07)

Mean (standard deviation) across starting point realizations of number of iterations and CPU time in seconds to achieve a certain relative error or best relative error achieved by methods, as well as the relative error achieved at that iteration. We highlight in bold the best performance among all competitors

*Maximum iteration number was reached without obtaining the desired relative error for at least one of the starting points



(a) Average iteration ratio $\tilde{\rho}(\varepsilon)$. **(b)** Average time ratio $\hat{\rho}(\varepsilon)$. **(c)** Average success ratio $\rho(\varepsilon)$.

Fig. 3 Performance profile for the portfolio selection problem (58) obtained after averaging over 12 synthetically generated data sets

σ_f is evaluated by the lowest eigenvalue of the Hessian observed at the initial point. If due to numerical errors, this number is nonpositive, we take $\sigma_f = 10^{-10}$.

For conducting numerical experiments, we generated synthetic data, as in Section 6.4 in [50]. We generate a matrix $[r_{t,i}]_{1\le t\le p, 1\le i\le n} \in \mathbb{R}^{p\times n}$ with given price ratios as: $r_{t,i} = 1 + N(0, 0.1)$ for any $i \in \{1, \ldots, n\}$ and $t \in \{1, \ldots, p\}$, which allows the closing price to vary by about 10% between two consecutive periods. We used $(p, n) = (1000, 800)$, $(1000, 1200)$, and $(1000, 1500)$ with 4 samples for each size. Hence, there are 12 data sets in total. For each data set, all methods were initialized from 10 randomly chosen vertices of the unit simplex.

Figure 3 collects results on the average performance of our methods and Table 3 reports numerical values obtained for each individual data set. `MBTFWGSC` and `ASFWGSC` outperforms all other methods considered in terms of time to reach a certain relative error, including `PN` and `PG`. Moreover, the advantage of `ASFWGSC` becomes more significant as the relative error decreases. Interestingly the iteration complexity of `MBTFWGSC` is almost identical to `FW-Line Search` while having superior time complexity. Additionally, despite its theoretical linear convergence, `FWLLOO` has inferior performance to both `MBTFWGSC` and `ASFWGSC`, indicating that the use of the strong convexity parameter σ_f within the algorithm may be detrimental to its performance, since a small estimation for σ_f leads to a large convergence coefficient. We attribute the competitive performance of `MBTFWGSC` to two causes: (1) the adaptive choice of M allows `MBTFWGSC` to imitate the steps of `FW-Line Search` closely but has lower computational complexity per iteration, (2) `FW-Line Search` effectively has a linear convergence rate for these examples, possibly due to the location of the optimal solution. We note that while `ASFWGSC` is inferior to `MBTFWGSC` in some instances, its linear rate of convergence is independent of the problem instance.

6.3 Distance weighted discrimination

In the context of binary classification, an interesting modification of the classical support-vector machine is the distance weighted discrimination (DWD) problem, introduced in [35]. In that problem, the classification loss attains the form

$$f(x) = \frac{1}{n}\sum_{i=1}^{p}(a_i^\top w + \mu y_i + \xi_i)^{-q} + c^\top \xi,$$

Table 3 Results for portfolio selection problem (58)

Problem			FW-Standard			FWGSC			LBTFWGSC		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−03											
syn_1000_800_10_50	800	1000	48.0 (6.4)	0.11 (0.02)	9.29e−04 (5.31e−05)	200.0 (1.2)	0.46 (0.03)	9.97e−04 (1.89e−06)	295.3 (188.2)	0.77 (0.49)	9.90e−04 (1.35e−05)
syn_1000_800_10_50_1	800	1000	49.6 (6.9)	0.11 (0.02)	8.98e−04 (1.05e−04)	261.6 (1.6)	0.62 (0.01)	9.98e−04 (1.01e−06)	385.0 (244.8)	1.02 (0.65)	9.49e−04 (7.66e−05)
syn_1000_800_10_50_2	800	1000	48.0 (6.4)	0.11 (0.01)	9.29e−04 (6.75e−05)	266.2 (2.4)	0.62 (0.02)	9.98e−04 (1.23e−06)	462.3 (224.8)	1.19 (0.58)	9.63e−04 (7.23e−05)
syn_1000_800_10_50_3	800	1000	42.8 (11.4)	0.10 (0.03)	8.69e−04 (1.47e−04)	166.2 (50.5)	0.38 (0.12)	9.81e−04 (4.62e−05)	246.9 (192.8)	0.64 (0.50)	9.34e−04 (8.92e−05)
syn_1000_1200_10_50	1200	1000	47.2 (5.2)	0.17 (0.02)	9.46e−04 (3.05e−05)	98.5 (0.8)	0.36 (0.01)	9.90e−04 (2.90e−06)	68.6 (97.2)	0.26 (0.37)	6.69e−04 (2.15e−04)
syn_1000_1200_10_50_1	1200	1000	43.5 (7.1)	0.16 (0.03)	9.33e−04 (4.85e−05)	197.2 (1.3)	0.72 (0.04)	9.97e−04 (1.67e−06)	214.2 (188.3)	0.85 (0.74)	9.77e−04 (2.63e−05)
syn_1000_1200_10_50_2	1200	1000	43.4 (6.1)	0.16 (0.02)	9.26e−04 (7.07e−05)	126.7 (1.2)	0.46 (0.03)	9.95e−04 (3.82e−06)	78.7 (112.6)	0.31 (0.44)	5.87e−04 (2.68e−04)
syn_1000_1200_10_50_3	1200	1000	52.3 (7.6)	0.19 (0.03)	9.23e−04 (5.04e−05)	242.6 (1.6)	0.89 (0.04)	9.96e−04 (3.06e−06)	284.4 (230.7)	1.13 (0.91)	9.22e−04 (9.42e−05)
syn_1000_1500_10_50	1500	1000	47.0 (7.1)	0.21 (0.03)	9.17e−04 (9.28e−05)	218.8 (1.0)	1.00 (0.02)	9.97e−04 (2.25e−06)	319.6 (206.7)	1.56 (1.02)	9.21e−04 (1.23e−04)
syn_1000_1500_10_50_1	1500	1000	47.0 (7.1)	0.21 (0.03)	9.17e−04 (9.28e−05)	218.8 (1.0)	1.01 (0.01)	9.97e−04 (2.25e−06)	319.6 (206.7)	1.55 (1.00)	9.21e−04 (1.23e−04)
syn_1000_1500_10_50_2	1500	1000	48.0 (3.5)	0.22 (0.02)	9.52e−04 (4.41e−05)	248.1 (1.2)	1.14 (0.01)	9.98e−04 (1.11e−06)	456.0 (150.2)	2.23 (0.74)	9.83e−04 (4.82e−05)
syn_1000_1500_10_50_3	1500	1000	42.9 (6.4)	0.20 (0.03)	9.28e−04 (6.49e−05)	198.8 (1.5)	0.91 (0.02)	9.97e−04 (1.75e−06)	275.3 (175.7)	1.35 (0.86)	9.56e−04 (6.54e−05)

Problem			MBTFWGSC			FW-Line Search			ASFWGSC		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−03											
syn_1000_800_10_50	800	1000	4.0 (0.0)	**0.01 (0.00)**	2.63e−04 (0.00e+00)	4.0 (0.0)	0.03 (0.00)	2.81e−04 (6.05e−12)	8.8 (0.4)	0.02 (0.00)	3.05e−04 (3.96e−05)
syn_1000_800_10_50_1	800	1000	7.0 (0.0)	**0.02 (0.00)**	3.78e−04 (0.00e+00)	7.0 (0.0)	0.05 (0.00)	2.10e−04 (5.39e−12)	9.5 (0.5)	**0.02 (0.00)**	7.62e−04 (1.95e−04)
syn_1000_800_10_50_2	800	1000	29.8 (65.4)	0.08 (0.17)	8.13e−04 (6.20e−05)	26.8 (59.4)	0.19 (0.42)	9.07e−04 (3.07e−05)	10.6 (0.7)	**0.03 (0.00)**	7.00e−04 (1.71e−04)
syn_1000_800_10_50_3	800	1000	6.0 (0.0)	**0.02 (0.00)**	3.45e−04 (2.17e−04)	6.0 (0.0)	0.04 (0.00)	3.40e−04 (1.59e−04)	8.1 (0.5)	**0.02 (0.00)**	8.39e−04 (1.37e−04)
syn_1000_1200_10_50	1200	1000	5.0 (0.0)	**0.02 (0.00)**	4.78e−04 (0.00e+00)	5.0 (0.0)	0.06 (0.01)	4.80e−04 (6.86e−12)	6.2 (0.4)	0.03 (0.00)	2.67e−04 (6.25e−15)
syn_1000_1200_10_50_1	1200	1000	4.0 (0.0)	**0.02 (0.00)**	1.02e−04 (1.36e−20)	4.0 (0.0)	0.04 (0.00)	1.26e−04 (6.41e−12)	7.5 (1.0)	0.03 (0.01)	5.71e−04 (1.74e−04)

Table 3 continued

Problem			MBTFWGSC			FW-Line Search			ASFWGSC		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
syn_1000_1200_10_50_2	1200	1000	3.0 (0.0)	**0.01 (0.00)**	4.44e−05 (0.00e+00)	3.0 (0.0)	0.03 (0.00)	6.32e−05 (9.51e−12)	6.4 (0.7)	0.03 (0.00)	1.37e−04 (3.95e−05)
syn_1000_1200_10_50_3	1200	1000	25.1 (51.3)	0.10 (0.21)	4.24e−04 (1.92e−04)	15.5 (25.5)	0.22 (0.40)	3.81e−04 (2.03e−04)	10.0 (0.8)	**0.04 (0.00)**	7.40e−04 (1.21e−04)
syn_1000_1500_10_50	1500	1000	5.0 (0.0)	**0.03 (0.00)**	2.03e−04 (1.38e−05)	5.0 (0.0)	0.07 (0.01)	1.94e−04 (4.60e−12)	8.2 (0.6)	0.04 (0.00)	3.86e−04 (2.12e−04)
syn_1000_1500_10_50_1	1500	1000	5.0 (0.0)	**0.03 (0.00)**	2.03e−04 (1.38e−05)	5.0 (0.0)	0.07 (0.00)	1.94e−04 (4.60e−12)	8.2 (0.6)	0.04 (0.00)	3.86e−04 (2.12e−04)
syn_1000_1500_10_50_2	1500	1000	6.0 (0.0)	**0.03 (0.00)**	5.03e−04 (1.29e−05)	6.0 (0.0)	0.09 (0.01)	5.12e−04 (2.19e−05)	9.1 (0.3)	0.05 (0.00)	4.54e−04 (2.66e−04)
syn_1000_1500_10_50_3	1500	1000	4.0 (0.0)	**0.02 (0.00)**	1.88e−04 (3.06e−05)	4.0 (0.0)	0.06 (0.01)	2.01e−04 (7.09e−12)	8.3 (0.5)	0.04 (0.00)	3.96e−04 (3.03e−04)
Problem			FWLLOO			PN			PG		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−03											
syn_1000_800_10_50	800	1000	34.5 (0.9)	0.11 (0.01)	8.21e−04 (1.03e−04)	10.8 (0.6)	21.25 (1.34)	8.34e−04 (7.49e−05)	90.4 (14.3)	0.23 (0.04)	5.05e−04 (2.50e−04)
syn_1000_800_10_50_1	800	1000	43.8 (0.6)	0.15 (0.01)	8.48e−04 (8.55e−05)	12.9 (1.3)	26.68 (1.89)	6.51e−04 (9.17e−05)	85.0 (16.9)	0.22 (0.04)	4.45e−04 (2.74e−04)
syn_1000_800_10_50_2	800	1000	46.0 (1.5)	0.16 (0.01)	7.80e−04 (7.61e−05)	13.4 (1.4)	27.03 (3.01)	7.17e−04 (1.28e−04)	83.0 (16.7)	0.21 (0.04)	4.90e−04 (2.65e−04)
syn_1000_800_10_50_3	800	1000	28.5 (4.9)	0.10 (0.02)	7.90e−04 (1.89e−04)	15.2 (2.8)	30.89 (4.42)	6.31e−04 (1.38e−04)	84.8 (22.4)	0.21 (0.05)	6.51e−04 (3.05e−04)
syn_1000_1200_10_50	1200	1000	18.9 (2.0)	0.11 (0.01)	7.63e−04 (1.86e−04)	17.7 (1.3)	58.15 (5.09)	8.37e−04 (5.23e−05)	121.1 (15.3)	0.46 (0.08)	9.49e−04 (3.62e−05)
syn_1000_1200_10_50_1	1200	1000	34.6 (0.9)	0.19 (0.01)	8.92e−04 (8.89e−05)	15.1 (1.2)	50.65 (2.35)	7.37e−04 (7.81e−05)	111.7 (25.0)	0.44 (0.07)	4.28e−04 (2.92e−04)
syn_1000_1200_10_50_2	1200	1000	23.1 (2.0)	0.13 (0.01)	7.12e−04 (2.00e−04)	15.7 (1.8)	49.83 (7.33)	6.53e−04 (1.22e−04)	123.3 (22.7)	0.48 (0.10)	6.74e−04 (2.39e−04)
syn_1000_1200_10_50_3	1200	1000	41.0 (2.0)	0.23 (0.02)	8.82e−04 (9.21e−05)	17.3 (2.5)	53.01 (8.38)	7.71e−04 (7.70e−05)	115.9 (22.2)	0.44 (0.10)	5.81e−04 (2.03e−04)
syn_1000_1500_10_50	1500	1000	39.5 (2.6)	0.28 (0.04)	6.92e−04 (2.03e−04)	15.8 (0.6)	69.84 (2.33)	7.70e−04 (7.42e−05)	119.9 (19.7)	0.58 (0.09)	7.11e−04 (2.14e−04)

Table 3 continued

Problem			FWLLOO			PN			PG		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
syn_1000_1500_10_50_1	1500	1000	39.5 (2.6)	0.26 (0.02)	6.92e−04 (2.03e−04)	15.8 (0.6)	70.06 (1.71)	7.70e−04 (7.42e−05)	119.9 (19.7)	0.59 (0.10)	7.11e−04 (2.14e−04)
syn_1000_1500_10_50_2	1500	1000	42.8 (1.9)	0.29 (0.02)	8.66e−04 (9.92e−05)	16.3 (1.7)	68.44 (5.07)	7.33e−04 (6.59e−05)	112.3 (17.7)	0.55 (0.09)	3.42e−04 (2.26e−04)
syn_1000_1500_10_50_3	1500	1000	34.0 (1.8)	0.22 (0.02)	8.77e−04 (1.18e−04)	17.6 (2.5)	77.05 (7.43)	7.49e−04 (6.85e−05)	116.0 (23.3)	0.56 (0.10)	5.43e−04 (2.50e−04)

Problem			FW-Standard			FWGSC			LBTFWGSC		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−05											
syn_1000_800_10_50	800	1000	452.5 (59.3)	1.03 (0.14)	9.47e−06 (3.52e−07)	18897.1 (2.3)	44.78 (2.26)	1.00e−05 (2.32e−10)	30313.3 (19832.3)	78.30 (51.38)	9.33e−06 (1.03e−06)
syn_1000_800_10_50_1	800	1000	480.4 (73.6)	1.13 (0.17)	9.50e−06 (3.73e−07)	25331.4 (20.3)	60.64 (1.00)	1.00e−05 (7.21e−11)	*35008.2 (22901.9)	92.15 (60.28)	1.04e−05 (1.15e−06)
syn_1000_800_10_50_2	800	1000	462.0 (75.8)	1.06 (0.18)	9.39e−06 (7.19e−07)	25769.5 (3.6)	60.67 (0.89)	1.00e−05 (1.85e−10)	*40007.6 (19986.8)	101.75 (51.00)	1.11e−05 (1.53e−06)
syn_1000_800_10_50_3	800	1000	405.9 (124.3)	0.94 (0.29)	8.68e−06 (2.61e−06)	15578.2 (5180.1)	36.64 (12.18)	9.93e−06 (2.17e−07)	25597.8 (20866.3)	66.12 (53.91)	9.12e−06 (1.32e−06)
syn_1000_1200_10_50	1200	1000	459.0 (46.4)	1.66 (0.19)	9.69e−06 (1.49e−07)	9598.4 (3.7)	35.56 (0.66)	1.00e−05 (7.30e−10)	6806.6 (10380.5)	26.77 (40.83)	6.22e−06 (2.48e−06)
syn_1000_1200_10_50_1	1200	1000	425.4 (68.2)	1.57 (0.30)	9.69e−06 (1.89e−07)	18653.2 (1.7)	69.59 (2.96)	1.00e−05 (2.16e−10)	25043.1 (20431.4)	98.42 (80.30)	8.60e−06 (1.72e−06)
syn_1000_1200_10_50_2	1200	1000	432.1 (68.4)	1.59 (0.23)	9.68e−06 (1.64e−07)	11766.7 (1.6)	43.82 (2.03)	1.00e−05 (3.25e−10)	7985.2 (12183.8)	31.21 (47.63)	8.17e−06 (1.20e−06)
syn_1000_1200_10_50_3	1200	1000	507.9 (83.5)	1.85 (0.31)	9.53e−06 (4.32e−07)	23710.5 (3.6)	87.15 (3.89)	1.00e−05 (1.84e−10)	*30011.4 (24482.2)	115.69 (95.10)	9.39e−06 (1.46e−06)
syn_1000_1500_10_50	1500	1000	460.6 (56.9)	2.12 (0.27)	9.60e−06 (2.23e−07)	21226.1 (1.4)	98.69 (0.76)	1.00e−05 (1.52e−10)	34823.7 (22776.3)	170.37 (111.45)	9.75e−06 (6.04e−07)
syn_1000_1500_10_50_1	1500	1000	460.6 (56.9)	2.10 (0.26)	9.60e−06 (2.23e−07)	21226.1 (1.4)	98.46 (1.26)	1.00e−05 (1.52e−10)	34823.7 (22776.3)	169.87 (111.13)	9.75e−06 (6.04e−07)
syn_1000_1500_10_50_2	1500	1000	469.1 (61.2)	2.14 (0.29)	9.52e−06 (2.95e−07)	24412.3 (1.9)	113.32 (0.90)	1.00e−05 (9.46e−11)	*45005.1 (14987.7)	219.94 (73.27)	1.07e−05 (5.64e−07)
syn_1000_1500_10_50_3	1500	1000	422.6 (68.9)	1.93 (0.31)	9.50e−06 (4.06e−07)	18982.5 (2.1)	87.99 (1.08)	1.00e−05 (1.63e−10)	28388.0 (18573.2)	138.89 (90.88)	8.60e−06 (2.13e−06)

Problem			MBTFWGSC			FW-Line Search			ASFWGSC		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−05											
syn_1000_800_10_50	800	1000	9.0 (0.0)	**0.02 (0.00)**	6.54e−06 (8.47e−22)	8.0 (0.0)	0.06 (0.00)	8.95e−06 (5.74e−12)	12.2 (0.4)	0.03 (0.00)	7.45e−06 (1.77e−06)
syn_1000_800_10_50_1	800	1000	35.0 (0.0)	0.10 (0.00)	9.74e−06 (0.00e+00)	34.0 (0.0)	0.25 (0.02)	9.49e−06 (4.80e−12)	14.2 (0.9)	**0.04 (0.00)**	5.87e−06 (2.77e−06)

Table 3 continued

Problem			MBTFWGSC			FW-Line Search			ASFWGSC		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
syn_1000_800_10_50_2	800	1000	2585.8 (7709.4)	7.05 (21.02)	7.97e−06 (6.78e−07)	2579.9 (7703.7)	18.71 (55.86)	8.88e−06 (3.74e−07)	16.9 (1.4)	**0.04 (0.00)**	6.26e−06 (2.25e−06)
syn_1000_800_10_50_3	800	1000	18.8 (8.4)	0.05 (0.02)	9.48e−06 (4.83e−09)	19.6 (7.8)	0.14 (0.06)	5.98e−06 (1.24e−06)	12.1 (0.5)	**0.03 (0.00)**	2.99e−06 (1.33e−06)
syn_1000_1200_10_50	1200	1000	19.0 (0.0)	0.08 (0.00)	9.83e−06 (1.69e−21)	18.0 (0.0)	0.21 (0.02)	9.02e−06 (5.37e−12)	10.2 (0.4)	**0.04 (0.00)**	9.68e−06 (5.52e−15)
syn_1000_1200_10_50_1	1200	1000	8.0 (0.0)	**0.03 (0.00)**	9.32e−06 (0.00e+00)	8.0 (0.0)	0.09 (0.00)	8.49e−06 (5.78e−12)	10.2 (1.2)	0.04 (0.01)	4.93e−06 (2.27e−06)
syn_1000_1200_10_50_2	1200	1000	4.0 (0.0)	**0.02 (0.00)**	5.47e−06 (8.47e−22)	4.0 (0.0)	0.04 (0.00)	7.40e−06 (9.05e−12)	7.8 (0.9)	0.03 (0.00)	3.41e−06 (2.50e−06)
syn_1000_1200_10_50_3	1200	1000	2383.9 (7082.7)	9.94 (29.54)	9.75e−06 (8.47e−08)	2372.8 (7055.4)	30.31 (90.14)	9.72e−06 (9.34e−08)	16.7 (1.2)	**0.07 (0.01)**	5.99e−06 (1.45e−06)
syn_1000_1500_10_50	1500	1000	13.0 (0.0)	0.07 (0.00)	8.71e−06 (7.30e−08)	12.0 (0.0)	0.18 (0.02)	9.64e−06 (3.91e−12)	12.4 (1.0)	**0.06 (0.01)**	5.60e−06 (2.21e−06)
syn_1000_1500_10_50_1	1500	1000	13.0 (0.0)	0.07 (0.00)	8.71e−06 (7.30e−08)	12.0 (0.0)	0.17 (0.01)	9.64e−06 (3.91e−12)	12.4 (1.0)	**0.06 (0.01)**	5.60e−06 (2.21e−06)
syn_1000_1500_10_50_2	1500	1000	22.2 (1.0)	0.12 (0.01)	9.50e−06 (5.29e−08)	20.8 (1.5)	0.30 (0.04)	9.68e−06 (1.63e−07)	14.4 (0.9)	**0.07 (0.01)**	6.35e−06 (2.13e−06)
syn_1000_1500_10_50_3	1500	1000	6.0 (0.0)	**0.03 (0.00)**	6.71e−06 (1.35e−06)	6.0 (0.0)	0.09 (0.01)	6.38e−06 (6.48e−12)	12.1 (0.7)	0.06 (0.00)	5.54e−06 (3.05e−06)

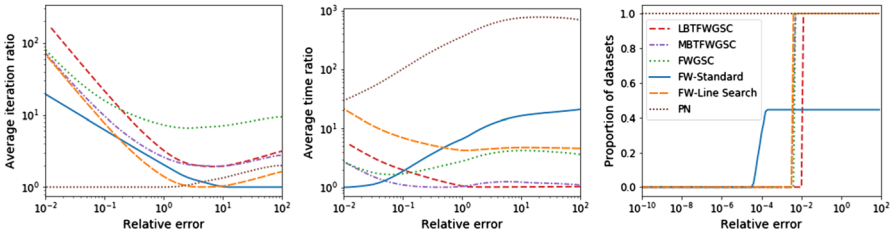
Problem			FWLLOO			PN			PG		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−05											
syn_1000_800_10_50	800	1000	212.0 (7.5)	0.66 (0.04)	8.90e−06 (7.08e−07)	*19.0 (3.7)	35.96 (2.20)	2.95e−05 (6.90e−05)	99.5 (15.3)	0.25 (0.05)	7.69e−06 (1.36e−06)
syn_1000_800_10_50_1	800	1000	288.8 (14.9)	1.02 (0.05)	9.07e−06 (1.09e−06)	19.1 (1.3)	41.09 (1.82)	7.75e−06 (1.75e−06)	90.9 (17.5)	0.23 (0.04)	7.87e−06 (1.82e−06)
syn_1000_800_10_50_2	800	1000	322.0 (17.8)	1.15 (0.08)	9.76e−06 (1.96e−07)	20.4 (1.4)	43.34 (3.37)	5.93e−06 (1.06e−06)	90.7 (15.4)	0.23 (0.04)	8.53e−06 (3.94e−07)
syn_1000_800_10_50_3	800	1000	167.9 (32.4)	0.58 (0.10)	8.62e−06 (1.06e−06)	21.6 (2.9)	46.04 (4.63)	7.32e−06 (1.78e−06)	98.6 (26.0)	0.25 (0.06)	8.60e−06 (8.25e−07)
syn_1000_1200_10_50	1200	1000	88.1 (9.1)	0.48 (0.06)	8.27e−06 (1.05e−06)	27.6 (1.5)	94.69 (8.22)	7.36e−06 (8.66e−07)	153.6 (16.3)	0.58 (0.09)	9.36e−06 (4.68e−07)
syn_1000_1200_10_50_1	1200	1000	218.3 (10.6)	1.15 (0.07)	9.30e−06 (4.20e−07)	25.9 (1.2)	89.03 (3.76)	7.79e−06 (6.18e−07)	120.3 (24.4)	0.48 (0.07)	8.22e−06 (1.30e−06)
syn_1000_1200_10_50_2	1200	1000	114.1 (5.8)	0.60 (0.05)	8.72e−06 (7.02e−07)	25.9 (1.8)	84.54 (9.72)	7.87e−06 (8.94e−07)	140.8 (22.0)	0.55 (0.11)	8.97e−06 (6.23e−07)

Table 3 continued

Problem			FWLLOO			PN			PG		
Name	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
syn_1000_1200_10_50_3	1200	1000	280.1 (18.5)	1.49 (0.10)	9.42e−06 (3.96e−07)	25.8 (2.5)	85.28 (10.08)	7.99e−06 (1.63e−06)	126.7 (22.4)	0.48 (0.10)	7.74e−06 (8.24e−07)
syn_1000_1500_10_50	1500	1000	266.2 (26.1)	1.71 (0.17)	9.37e−06 (4.00e−07)	30.0 (1.1)	130.79 (8.05)	7.70e−06 (4.81e−07)	133.6 (19.3)	0.64 (0.09)	8.22e−06 (1.04e−06)
syn_1000_1500_10_50_1	1500	1000	266.2 (26.1)	1.70 (0.15)	9.37e−06 (4.00e−07)	30.0 (1.1)	131.56 (7.27)	7.70e−06 (4.81e−07)	133.6 (19.3)	0.65 (0.09)	8.22e−06 (1.04e−06)
syn_1000_1500_10_50_2	1500	1000	287.3 (19.1)	1.86 (0.11)	9.06e−06 (6.59e−07)	27.5 (1.6)	119.17 (5.27)	8.00e−06 (7.97e−07)	120.9 (16.9)	0.59 (0.08)	7.98e−06 (1.56e−06)
syn_1000_1500_10_50_3	1500	1000	225.5 (15.9)	1.44 (0.14)	8.56e−06 (9.10e−07)	27.7 (2.5)	127.59 (7.08)	8.37e−06 (7.37e−07)	128.9 (25.4)	0.62 (0.11)	8.73e−06 (7.09e−07)

Mean (standard deviation) across starting point realizations of number of iterations and CPU time in seconds to achieve a certain relative error or best relative error achieved by methods, as well as the relative error achieved at that iteration. We highlight in bold the best performance among all competitors

*Maximum iteration number was reached without obtaining the desired relative error for at least one of the starting points



(a) Average iteration ratio $\tilde{\rho}(\varepsilon)$. **(b)** Average time ratio $\hat{\rho}(\varepsilon)$. **(c)** Average success ratio $\rho(\varepsilon)$.

Fig. 4 Performance profile for the DWD problem averaged over binary classification problems

over the convex compact set

$$\mathcal{X} = \{x = (w, \mu, \xi) | \ \|w\|^2 \leq 1, \mu \in [-u, u], \|\xi\|^2 \leq R, \xi \in \mathbb{R}^p_+\},$$

where $R > 0$ is a hyperparameter that has to be learned via cross-validation.

The parameter $q \geq 1$ calibrates the statistical loss function, and $(a_i, y_i) \in \mathbb{R}^d \times \{-1, 1\}$, $(i = 1, 2, \ldots, p)$ is the observed sample. The decision variable is decoded as $x = (w, \mu, \xi) \in \mathbb{R}^n$, where $n = d + 1 + p$, corresponding to a normal vector $w \in \mathbb{R}^d$, an intercept $\mu \in \mathbb{R}$ and a slack variable $\xi \in \mathbb{R}^p$. Since $\varphi(t) = t^{-q}$, $q \geq 1$ is generalized self-concordant with parameters $M_\varphi = \frac{q+2}{\sqrt[q+2]{q(q+1)}}$ and $\nu = \frac{2(q+3)}{q+2} \in (2, 3)$ (cf. Table 1) we get a GSC minimization problem over the compact set $\mathcal{X}$, with parameters:

$$\nu = \frac{2(q+3)}{q+2} \text{ and}$$
$$M_f = \frac{(q+2)n^{1/(q+2)}}{\sqrt[q+2]{q(q+1)}} \max\left\{\|(a_i^\top, y_i, e_i^\top)^\top\|_2^{q/(q+2)} : 1 \leq i \leq n\right\}.$$

The special case $q = 1$ corresponds to the loss function of [35], who solved this problem via a second-order cone reformulation. We test our algorithms using $q = 2$, and the observations a_i and y_i are based on data sets a1a–a9a from the LIBSVM library [12], where a_i are normalized. For each data set, the methods were ran 10 times, one for each randomly generated starting point of the structure $(0, 0, \xi)$ where ξ is sampled uniformly from its domain. The results presented are averages across these realizations. We set $c_i = 1$ for all $i = 1, \ldots, p$, $u = 5$, and $R = 10$.

`PG` cannot be applied to this problem, since $2 < \nu < 3$. We also do not apply `ASFWGSC`, since $\mathcal{X}$ is not a polyhedral set. Figure 4 collects results on the average performance of our methods and Table 4 shows the results obtained for each individual data set. Here we see that for all data sets and all starting points all FW based methods reach a minimal relative error 10^{-3}, with the exception of `standard-FW` which reaches a relative error of 10^{-4} for the smaller instances a1a–a4a but obtains a relative error higher than 10^2 for the larger instances a5a-a9a. The poor performance of `FW-Standard` on the largest instances is due to the monotonically decreasing step sizes and the fact that it requires very small step size in order to keep the iterates in the domain in the first few iterations. This highlights the drawbacks of using

Table 4 Results for distance weighted discrimination (DWD) problem

Problem			FW-Standard			FWGSC			LBTFWGSC		
Name	d	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−01											
a1a	128	1605	1040.7 (64.7)	**1.60 (0.59)**	9.98e−02 (1.18e−04)	3576.2 (9.2)	3.07 (0.01)	1.00e−01 (1.14e−05)	5803.1 (6.2)	4.59 (0.02)	1.00e−01 (4.25e−06)
a2a	128	2265	1269.4 (66.0)	4.03 (1.74)	9.99e−02 (6.32e−05)	3744.8 (8.3)	4.24 (0.03)	1.00e−01 (1.11e−05)	5822.8 (8.4)	5.82 (0.09)	1.00e−01 (2.73e−06)
a3a	128	3185	1505.6 (80.6)	7.06 (2.37)	9.99e−02 (8.28e−05)	3939.1 (6.3)	5.73 (0.04)	1.00e−01 (1.46e−05)	5896.3 (6.8)	7.61 (0.13)	1.00e−01 (1.50e−06)
a4a	128	4781	1855.6 (91.5)	19.98 (11.21)	9.99e−02 (5.79e−05)	4187.1 (6.9)	8.74 (0.11)	1.00e−01 (1.31e−05)	5923.0 (9.1)	10.73 (0.18)	1.00e−01 (4.56e−06)
a5a	128	6414	*50001.0 (0.0)	79.32 (2.06)	3.26e+07 (6.74e+07)	4367.9 (9.4)	16.07 (1.02)	1.00e−01 (1.13e−05)	6026.9 (11.1)	14.09 (0.08)	1.00e−01 (5.61e−06)
a6a	128	11220	*50001.0 (0.0)	152.16 (3.84)	3.56e+07 (6.65e+07)	4799.5 (7.4)	37.78 (2.44)	1.00e−01 (1.47e−05)	6094.4 (8.9)	42.70 (3.35)	1.00e−01 (1.28e−06)
a7a	128	16100	*50001.0 (0.0)	190.41 (7.69)	3.66e+07 (6.63e+07)	5105.2 (7.0)	56.17 (2.99)	1.00e−01 (1.34e−05)	6118.9 (10.3)	60.29 (2.40)	1.00e−01 (2.07e−06)
a8a	128	22696	*50001.0 (0.0)	293.74 (7.57)	7.68e+07 (1.23e+08)	5437.6 (5.4)	82.52 (3.34)	1.00e−01 (8.24e−06)	6152.5 (8.4)	83.56 (3.76)	1.00e−01 (5.44e−06)
a9a	128	32561	*50001.0 (0.0)	292.82 (3.70)	1.38e+08 (2.22e+08)	5798.7 (4.1)	82.73 (1.12)	1.00e−01 (1.25e−05)	6200.2 (6.7)	83.75 (3.00)	1.00e−01 (3.70e−06)

Problem			MBTFWGSC			FW-Line Search			PN		
Name	d	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−01											
a1a	128	1605	2479.3 (9.6)	2.54 (0.22)	1.00e−01 (1.33e−05)	2070.9 (3.8)	16.76 (0.04)	1.00e−01 (1.18e−05)	186.9 (8.6)	175.90 (10.11)	8.23e−02 (1.39e−02)
a2a	128	2265	2522.3 (7.2)	**3.21 (0.03)**	1.00e−01 (1.36e−05)	2081.2 (3.7)	21.49 (0.13)	1.00e−01 (1.80e−05)	206.6 (11.9)	240.96 (17.34)	8.27e−02 (1.21e−02)
a3a	128	3185	2579.3 (5.0)	**4.14 (0.04)**	1.00e−01 (1.22e−05)	2103.9 (3.5)	27.63 (0.14)	1.00e−01 (1.66e−05)	236.4 (13.9)	345.42 (17.70)	8.81e−02 (1.05e−02)
a4a	128	4781	2633.4 (5.9)	**5.90 (0.10)**	1.00e−01 (1.68e−05)	2119.7 (4.2)	39.43 (0.37)	1.00e−01 (1.04e−05)	270.6 (13.4)	529.22 (25.68)	7.99e−02 (9.10e−03)
a5a	128	6414	2670.9 (11.0)	**7.75 (0.04)**	1.00e−01 (9.60e−06)	2127.4 (5.8)	66.93 (2.13)	1.00e−01 (1.26e−05)	284.9 (24.5)	972.85 (75.90)	7.39e−02 (1.42e−02)
a6a	128	11220	2785.5 (4.2)	**23.51 (1.84)**	1.00e−01 (1.10e−05)	2170.0 (5.6)	134.99 (3.66)	1.00e−01 (1.52e−05)	330.3 (34.6)	2220.70 (184.15)	8.25e−02 (1.15e−02)

Table 4 continued

Problem			MBTFWGSC			FW-Line Search			PN		
Name	d	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
a7a	128	16100	2852.6 (5.9)	**34.10 (1.79)**	1.00e−01 (1.35e−05)	2195.0 (4.0)	159.07 (3.54)	1.00e−01 (1.45e−05)	364.2 (42.5)	3464.89 (282.62)	8.51e−02 (9.32e−03)
a8a	128	22696	2944.2 (10.8)	**49.00 (2.00)**	1.00e−01 (1.49e−05)	2231.9 (4.4)	269.81 (5.56)	1.00e−01 (1.31e−05)	386.3 (50.3)	5276.16 (407.81)	8.05e−02 (1.04e−02)
a9a	128	32561	3030.2 (8.2)	**47.45 (1.70)**	1.00e−01 (1.44e−05)	2262.8 (3.4)	247.10 (1.86)	1.00e−01 (1.48e−05)	417.2 (49.8)	4786.52 (394.76)	7.56e−02 (1.64e−02)

Problem			FW-Standard			FWGSC			LBTFWGSC		
Name	d	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−02											
a1a	128	1605	3294.5 (203.1)	**2.56 (0.63)**	9.99e−03 (3.51e−06)	21677.4 (9.2)	18.60 (0.06)	1.00e−02 (1.59e−07)	*50001.0 (0.0)	39.47 (0.15)	1.13e−02 (1.42e−06)
a2a	128	2265	4018.8 (208.7)	**5.48 (1.80)**	9.99e−03 (2.94e−06)	21837.4 (8.4)	24.84 (0.18)	1.00e−02 (1.68e−07)	*50001.0 (0.0)	50.57 (0.38)	1.13e−02 (1.89e−06)
a3a	128	3185	4763.0 (254.0)	**9.19 (2.46)**	1.00e−02 (1.91e−06)	22164.7 (6.4)	32.36 (0.23)	1.00e−02 (1.37e−07)	*50001.0 (0.0)	64.22 (0.67)	1.14e−02 (1.53e−06)
a4a	128	4781	5870.0 (290.0)	**23.70 (11.31)**	1.00e−02 (1.86e−06)	22420.5 (6.8)	47.08 (0.55)	1.00e−02 (1.26e−07)	*50001.0 (0.0)	90.70 (1.65)	1.14e−02 (2.04e−06)
a5a	128	6414	*50001.0 (0.0)	79.32 (2.06)	3.26e+07 (6.74e+07)	22556.9 (9.4)	82.36 (2.86)	1.00e−02 (1.26e−07)	*50001.0 (0.0)	117.38 (0.52)	1.15e−02 (2.58e−06)
a6a	128	11220	*50001.0 (0.0)	152.16 (3.84)	3.56e+07 (6.65e+07)	23203.8 (7.3)	180.84 (4.84)	1.00e−02 (1.33e−07)	*50001.0 (0.0)	342.18 (8.76)	1.15e−02 (2.06e−06)
a7a	128	16100	*50001.0 (0.0)	190.41 (7.69)	3.66e+07 (6.63e+07)	23591.6 (6.8)	255.31 (8.34)	1.00e−02 (1.34e−07)	*50001.0 (0.0)	476.41 (9.49)	1.15e−02 (2.38e−06)
a8a	128	22696	*50001.0 (0.0)	293.74 (7.57)	7.68e+07 (1.23e+08)	24132.5 (5.6)	375.03 (6.98)	1.00e−02 (1.41e−07)	*50001.0 (0.0)	673.87 (12.50)	1.15e−02 (1.95e−06)
a9a	128	32561	*50001.0 (0.0)	292.82 (3.70)	1.38e+08 (2.22e+08)	24647.0 (4.0)	352.49 (4.25)	1.00e−02 (1.13e−07)	*50001.0 (0.0)	655.19 (10.25)	1.15e−02 (1.53e−06)

Problem			MBTFWGSC			FW-Line Search			PN		
Name	d	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−02											
a1a	128	1605	20195.1 (9.6)	20.37 (1.08)	1.00e−02 (9.78e−08)	19594.0 (3.7)	159.40 (0.40)	1.00e−02 (1.33e−07)	189.6 (8.6)	177.89 (10.16)	3.40e−03 (2.03e−03)
a2a	128	2265	20197.9 (7.2)	25.78 (0.19)	1.00e−02 (1.72e−07)	19548.5 (3.8)	202.15 (1.21)	1.00e−02 (1.34e−07)	209.4 (11.9)	243.64 (17.38)	4.42e−03 (2.52e−03)
a3a	128	3185	20354.1 (4.8)	32.72 (0.27)	1.00e−02 (1.40e−07)	19653.1 (3.4)	262.66 (2.19)	1.00e−02 (1.45e−07)	239.4 (13.9)	349.12 (17.70)	5.38e−03 (2.14e−03)

Table 4 continued

Problem			MBTFWGSC			FW-Line Search			PN		
Name	d	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
a4a	128	4781	20371.3 (5.9)	45.80 (0.73)	1.00e−02 (1.71e−07)	19609.5 (4.3)	359.21 (1.68)	1.00e−02 (1.40e−07)	273.6 (13.4)	534.50 (25.68)	4.81e−03 (1.58e−03)
a5a	128	6414	20330.3 (10.6)	**58.95 (0.22)**	1.00e−02 (1.44e−07)	19521.2 (5.7)	613.87 (6.03)	1.00e−02 (1.41e−07)	287.6 (25.0)	980.10 (76.83)	5.71e−03 (2.21e−03)
a6a	128	11220	20584.6 (4.1)	**173.73 (4.92)**	1.00e−02 (7.18e−08)	19664.8 (5.8)	1236.31 (17.44)	1.00e−02 (1.66e−07)	333.4 (34.6)	2237.24 (182.66)	4.85e−03 (2.01e−03)
a7a	128	16100	20679.6 (5.9)	**244.99 (7.16)**	1.00e−02 (1.36e−07)	19689.6 (3.8)	1619.53 (20.03)	1.00e−02 (1.57e−07)	367.5 (42.7)	3488.93 (282.36)	4.54e−03 (2.12e−03)
a8a	128	22696	20922.2 (10.5)	**344.96 (6.52)**	1.00e−02 (1.53e−07)	19847.3 (4.3)	2396.21 (14.76)	1.00e−02 (1.42e−07)	389.2 (50.9)	5303.81 (408.01)	5.00e−03 (1.81e−03)
a9a	128	32561	21097.3 (8.3)	**325.08 (4.07)**	1.00e−02 (1.27e−07)	19934.5 (3.5)	2056.31 (14.62)	1.00e−02 (1.18e−07)	420.2 (50.5)	4815.02 (399.30)	4.16e−03 (2.55e−03)

Mean (standard deviation) across starting point realizations of number of iterations and CPU time in seconds to achieve a certain relative error or best relative error achieved by method after 50,000 iterations, as well as the relative error achieved at that iteration

*Maximum iteration number was reached without obtaining the desired relative error for at least one of the starting points

FW-Standard in the GSC setting: while it has a simple step size rule, the number of calls to the domain oracle required to find the step-size for which we stay in the domain may be very large, resulting in poor performance in practice. From the other methods, MBTFWGSC and FWGSC perform the best, with MBTFWGSC having a slight advantage for lower accuracy due to the use of a smaller M_f values. Again, we see how important the adaptive nature of MBTFWGSC is to improving FWGSC, especially in the early iterations, resulting in iteration complexity closer to FW-Line Search with lower per-iteration computational cost.

6.4 Inverse covariance estimation

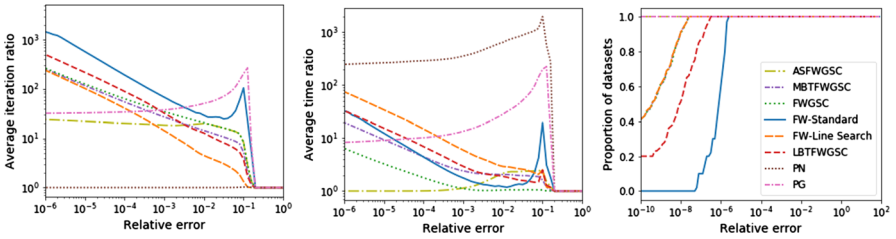
Undirected graphical models offer a way to describe and explain the relationships among a set of variables, a central element of multivariate data analysis. The principle of parsimony dictates that we should select the simplest graphical model that adequately explains the data. The typical approach to tackle this problem [52] is the following: Given a data set, we solve a maximum likelihood problem with an added low-rank penalty to make the resulting graph as sparse as possible. We consider learning a Gaussian graphical random field of p nodes/variables from a data set $\{\phi_1, \ldots, \phi_N\}$. Each random vector ϕ_j is an iid 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 graphical model. To learn the graphical model via an ℓ_1-regularization framework in its constrained formulation, we minimize the loss function

$$f(x) = -\log\det(\mathrm{mat}(x)) + \mathrm{tr}(\hat{\Sigma}\,\mathrm{mat}(x)) \tag{59}$$

over set of symmetric matrices with ℓ_1-ball restriction, that is $\mathcal{X} = \{x \in \mathbb{R}^n \mid \|x\|_1 \leq R, \mathrm{mat}(x) \in \mathbb{S}^n\}$ where $R = \lceil\sqrt{p}\rceil$. The decision variables are vectors $x \in \mathbb{R}^n$ for $n = p^2$, so that $\mathrm{mat}(x)$ represents the $p \times p$ matrix constructed from the p^2-dimensional vector x. It can be seen that f is standard self-concordant with domain $\mathbb{S}^n_{++}$. Hence, $M_f = 2$ and $\nu = 3$. One can see that the gradient $\nabla f(x) = \hat{\Sigma} - \mathrm{mat}(x)^{-1}$ and Hessian $\nabla^2 f(x) = \mathrm{mat}(x)^{-1} \otimes \mathrm{mat}(x)^{-1}$. Since $\mathrm{mat}(x)$ is positive definite, we can compute the inverse via a Cholesky decomposition, which in the worst case needs $O(p^3)$ arithmetic steps. To compute the search direction, we have to solve the LP

$$s(x) \in \mathrm{argmin}_{s\in\mathcal{X}} \langle \hat{\Sigma} - \mathrm{mat}(x)^{-1}, \mathrm{mat}(s)\rangle,$$

where $\langle A, B\rangle = \mathrm{tr}(AB)$ for $A, B \in \mathbb{S}^n$. This Linear Minimization Oracle requires to identify the minimal elements of the matrix $\hat{\Sigma} - \mathrm{mat}(x)^{-1}$. Moreover, for the backtracking procedures as well as line search, we also need to construct a domain oracle. This requires to find the maximal step size $t > 0$ for which $x + t(s(x) - x) \succeq 0$, which is equivalent to finding the maximal $t \in (0, 1]$ such that $\frac{1}{t}\,\mathrm{mat}(x) \succ \mathrm{mat}(x) - \mathrm{mat}(s(x))$ or $\frac{1}{t} > \lambda_{\max}(I - \mathrm{mat}(x)^{-1/2}\,\mathrm{mat}(s(x))\,\mathrm{mat}(x)^{-1/2})$. Note that this step oracle is not needed when using the theoretical step size in FWGSC and ASFWGSC.



(a) Average iteration ratio $\tilde{\rho}(\varepsilon)$. **(b)** Average time ratio $\hat{\rho}(\varepsilon)$. **(c)** Average success ratio $\rho(\varepsilon)$.

Fig. 5 Performance Profile for Covariance estimation problem (59) averaged on 10 data sets

We test our method on synthetically generated data sets. We generated the data by first creating the matrix $\hat{\Sigma}$ randomly, by generating a random orthonormal basis or $\mathbb{R}^p$, $B = \{v_1, \ldots, v_p\}$, and then set

$$\hat{\Sigma} = \sum_{i=1}^{p} \sigma_i v_i v_i^\top,$$

where σ_i are independently and uniformly distributed between 0.5 and 1. We generated 10 such data sets, for p ranging between 50 and 300. For each data set, the methods were ran for 10 randomly generated starting points. Each starting point has been chosen as a diagonal matrix where the diagonal was randomly chosen from the R-simplex. Figure 5 collects results on the average performance of our methods and Table 5 shows the results obtained for each individual data set. We observe that `ASFWGSC` has the lowest time of obtaining any relative error below 10^{-3}. Moreover, though `PG` has a lower iteration complexity in some instances, the higher computational cost of projection vs. linear oracle computations, makes it significantly inferior to `ASFWGSC`. These results highlight the need for a linearly convergent projection free method such as `ASFWGSC` in the GSC setting in high dimension. Specifically, `ASFWGSC` outperforms FW based methods both theoretically and practically, while obtaining similar iteration complexity to `PG` with significantly lower per-iteration computational cost.

7 Conclusion

Motivated by the recent interest in computational statistics and machine learning in functions displaying generalized self-concordant properties, this paper develops a set of projection-free algorithms for minimizing generalized self-concordant functions as defined in [50]. This function class covers several well-known examples, including logistic, power, reciprocal and, of course, standard self-concordant functions. In particular, members of this function class are potentially ill-conditioned: they may neither have a Lipschitz continuous gradient nor be strongly convex on their domain. Hence, no provably convergent Frank–Wolfe method has been available so far for minimizing generalized self-concordant functions. This paper fills this important gap by developing a set of new provably convergent FW algorithms with

Table 5 Results for covariance estimation example

Problem			FW-Standard			FWGSC			LBTFWGSC		
	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−04											
cov_50	2500	50	1370.7 (20.6)	0.66 (0.12)	9.93e−05 (6.86e−07)	429.7 (84.3)	0.29 (0.07)	9.96e−05 (2.99e−07)	457.4 (109.5)	0.85 (0.22)	9.98e−05 (1.97e−07)
cov_80	6400	80	2100.6 (11.1)	2.46 (0.39)	9.96e−05 (2.03e−07)	665.6 (129.0)	0.97 (0.22)	9.98e−05 (1.25e−07)	719.1 (162.6)	3.27 (0.82)	9.98e−05 (9.62e−08)
cov_120	14400	120	3056.3 (21.7)	5.75 (0.30)	9.99e−05 (8.43e−08)	1035.5 (120.5)	2.42 (0.30)	9.98e−05 (1.03e−07)	1712.1 (265.0)	12.04 (1.91)	9.99e−05 (3.51e−08)
cov_150	22500	150	3787.5 (13.6)	10.89 (2.64)	9.99e−05 (7.50e−08)	1188.0 (103.6)	4.21 (0.41)	9.98e−05 (1.02e−07)	1720.2 (199.1)	10.60 (1.27)	9.99e−05 (4.92e−08)
cov_170	28900	170	4268.2 (7.2)	18.48 (1.99)	9.99e−05 (5.84e−08)	1414.9 (197.1)	8.04 (1.62)	9.99e−05 (4.47e−08)	1834.6 (333.0)	29.92 (5.11)	9.99e−05 (3.57e−08)
cov_200	40000	200	4963.6 (12.8)	45.46 (3.27)	9.99e−05 (3.43e−08)	1728.0 (180.7)	17.93 (2.45)	9.99e−05 (4.24e−08)	1824.3 (215.1)	53.20 (8.39)	9.99e−05 (4.35e−08)
cov_220	48400	220	5348.4 (13.6)	58.24 (5.08)	1.00e−04 (2.18e−08)	1782.2 (196.5)	23.17 (2.83)	9.99e−05 (4.50e−08)	2449.2 (258.4)	68.85 (7.65)	9.99e−05 (4.17e−08)
cov_250	62500	250	6022.7 (6.3)	92.52 (14.39)	1.00e−04 (2.38e−08)	2009.6 (175.2)	36.01 (5.14)	9.99e−05 (4.97e−08)	3363.5 (406.1)	204.46 (22.85)	1.00e−04 (2.34e−08)
cov_270	72900	270	6495.2 (4.6)	112.35 (16.95)	1.00e−04 (2.04e−08)	2211.2 (193.3)	48.04 (9.08)	9.99e−05 (4.54e−08)	3636.2 (408.0)	241.96 (36.67)	1.00e−04 (1.13e−08)
cov_300	90000	300	7183.0 (11.1)	145.69 (1.04)	1.00e−04 (2.18e−08)	2338.4 (237.4)	57.24 (6.23)	9.99e−05 (4.38e−08)	3330.5 (416.7)	259.57 (44.82)	1.00e−04 (1.87e−08)

Problem			MBTFWGSC			FW-Line Search			ASFWGSC		
	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−04											
cov_50	2500	50	370.4 (79.2)	0.79 (0.18)	9.96e−05 (2.72e−07)	276.4 (72.3)	2.05 (0.71)	9.96e−05 (2.40e−07)	137.5 (11.2)	**0.20 (0.03)**	9.25e−05 (2.90e−06)
cov_80	6400	80	575.2 (118.7)	2.75 (0.60)	9.96e−05 (1.68e−07)	429.9 (103.8)	6.60 (1.63)	9.98e−05 (2.06e−07)	210.0 (11.5)	**0.74 (0.14)**	9.40e−05 (1.98e−06)
cov_120	14400	120	897.6 (114.1)	5.55 (1.43)	9.99e−05 (8.34e−08)	677.6 (105.9)	17.14 (3.03)	9.98e−05 (1.08e−07)	310.0 (19.0)	**1.26 (0.20)**	9.71e−05 (1.13e−06)
cov_150	22500	150	1025.0 (97.0)	6.82 (0.71)	9.99e−05 (7.49e−08)	763.1 (88.7)	41.25 (5.69)	9.99e−05 (8.65e−08)	382.7 (20.3)	**2.17 (0.24)**	9.82e−05 (1.20e−06)
cov_170	28900	170	1223.4 (184.3)	21.83 (4.67)	9.99e−05 (8.93e−08)	916.1 (167.1)	69.58 (11.15)	9.99e−05 (7.73e−08)	447.3 (33.0)	**6.12 (0.97)**	9.78e−05 (8.26e−07)
cov_200	40000	200	1493.0 (163.0)	51.21 (8.58)	9.99e−05 (5.37e−08)	1116.0 (137.7)	130.65 (18.04)	9.99e−05 (6.64e−08)	530.4 (28.6)	**9.21 (0.61)**	9.87e−05 (7.32e−07)
cov_220	48400	220	1541.7 (180.9)	53.43 (9.05)	9.99e−05 (6.03e−08)	1155.5 (159.0)	178.02 (26.72)	9.99e−05 (5.25e−08)	557.4 (27.2)	**10.40 (1.23)**	9.85e−05 (1.02e−06)
cov_250	62500	250	1725.2 (165.4)	114.56 (21.08)	9.99e−05 (5.92e−08)	1267.0 (154.8)	324.70 (63.34)	9.99e−05 (5.10e−08)	662.7 (39.0)	**27.99 (3.76)**	9.88e−05 (7.57e−07)

Table 5 continued

Problem			MBTFWGSC			FW-Line Search			ASFWGSC		
	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
cov_270	72900	270	1907.5 (180.4)	121.16 (31.03)	9.99e−05 (3.46e−08)	1419.6 (163.4)	426.59 (65.65)	9.99e−05 (4.84e−08)	690.2 (28.2)	**23.27 (3.75)**	9.89e−05 (7.85e−07)
cov_300	90000	300	2006.8 (216.5)	145.92 (19.94)	9.99e−05 (2.49e−08)	1472.0 (186.6)	560.21 (66.08)	9.99e−05 (4.10e−08)	775.4 (36.5)	**25.96 (2.28)**	9.89e−05 (5.45e−07)

Problem			PN			PG		
	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−04								
cov_50	2500	50	15.2 (2.0)	80.19 (13.33)	3.37e−05 (2.95e−05)	120.8 (87.4)	0.96 (0.59)	2.21e−05 (1.87e−05)
cov_80	6400	80	18.1 (3.0)	309.32 (74.70)	3.98e−05 (3.02e−05)	325.0 (234.9)	7.19 (5.05)	1.54e−05 (1.24e−05)
cov_120	14400	120	19.4 (2.0)	569.46 (33.40)	4.11e−05 (3.13e−05)	314.9 (172.8)	10.71 (5.35)	2.83e−05 (2.41e−05)
cov_150	22500	150	20.0 (2.4)	665.23 (96.11)	5.13e−05 (2.74e−05)	362.5 (201.7)	18.49 (10.58)	5.20e−05 (3.28e−05)
cov_170	28900	170	22.6 (2.9)	1930.93 (316.94)	4.05e−05 (2.75e−05)	611.8 (395.7)	64.61 (38.46)	5.63e−05 (2.64e−05)
cov_200	40000	200	23.7 (2.5)	1794.82 (207.54)	3.51e−05 (2.32e−05)	582.4 (266.1)	57.60 (24.23)	5.45e−05 (2.88e−05)
cov_220	48400	220	23.6 (1.6)	1855.80 (135.15)	5.43e−05 (2.21e−05)	655.6 (216.1)	73.64 (23.54)	6.98e−05 (2.29e−05)
cov_250	62500	250	27.6 (3.7)	5039.51 (753.80)	5.28e−05 (2.30e−05)	2454.3 (4178.9)	438.30 (637.84)	7.57e−05 (1.93e−05)
cov_270	72900	270	28.2 (4.6)	4963.79 (736.25)	5.06e−05 (1.47e−05)	3337.9 (5712.1)	589.54 (1004.08)	6.08e−05 (1.53e−05)
cov_300	90000	300	27.1 (2.0)	4651.52 (468.64)	4.82e−05 (2.66e−05)	971.1 (291.6)	214.63 (62.11)	6.46e−05 (1.72e−05)

Problem			FW-Standard			FWGSC			LBTFWGSC		
	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−06											
cov_50	2500	50	13178.1 (416.7)	6.56 (0.75)	9.95e−07 (2.45e−09)	1650.5 (412.4)	1.08 (0.35)	9.99e−07 (8.02e−10)	2288.4 (614.9)	4.28 (1.38)	9.99e−07 (4.89e−10)
cov_80	6400	80	19819.2 (494.7)	23.01 (3.69)	9.97e−07 (1.79e−09)	3380.9 (678.2)	4.86 (1.10)	9.99e−07 (2.97e−10)	4930.8 (1058.8)	21.94 (4.75)	1.00e−06 (2.23e−10)
cov_120	14400	120	29325.2 (567.2)	54.18 (2.16)	9.99e−07 (1.09e−09)	4514.2 (851.3)	10.77 (2.21)	1.00e−06 (2.38e−10)	10587.9 (2156.0)	74.15 (14.24)	1.00e−06 (8.35e−11)
cov_150	22500	150	36789.8 (544.6)	116.11 (18.38)	9.99e−07 (6.05e−10)	5061.4 (1030.4)	17.72 (3.83)	1.00e−06 (2.51e−10)	10517.3 (2303.2)	64.70 (13.95)	1.00e−06 (5.64e−11)

Table 5 continued

Problem			FW-Standard			FWGSC			LBTFWGSC		
	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
cov_170	28900	170	41651.4 (237.0)	179.79 (21.21)	9.99e−07 (9.11e−10)	6865.5 (1430.3)	39.43 (12.59)	1.00e−06 (1.54e−10)	12660.4 (2797.2)	204.40 (47.65)	1.00e−06 (8.71e−11)
cov_200	40000	200	48118.1 (683.4)	439.85 (26.28)	9.99e−07 (4.86e−10)	7773.0 (1347.1)	82.98 (16.96)	1.00e−06 (1.47e−10)	10872.1 (1983.0)	300.09 (49.33)	1.00e−06 (6.17e−11)
cov_220	48400	220	*50001.0 (0.0)	552.84 (44.79)	1.16e−06 (3.46e−08)	7832.2 (1012.3)	100.95 (11.50)	1.00e−06 (1.56e−10)	8826.2 (1085.6)	246.51 (29.06)	1.00e−06 (7.30e−11)
cov_250	62500	250	*50001.0 (0.0)	767.94 (119.51)	1.46e−06 (3.57e−08)	8705.8 (1498.3)	159.21 (44.27)	1.00e−06 (1.08e−10)	21310.9 (3991.2)	1289.64 (221.73)	1.00e−06 (2.94e−11)
cov_270	72900	270	*50001.0 (0.0)	869.79 (131.12)	1.71e−06 (3.12e−08)	9900.4 (1396.7)	217.19 (57.43)	1.00e−06 (9.54e−11)	23551.3 (3621.0)	1511.20 (248.37)	1.00e−06 (3.08e−11)
cov_300	90000	300	*50001.0 (0.0)	1017.68 (9.11)	2.06e−06 (3.52e−08)	10531.1 (1653.3)	258.92 (40.44)	1.00e−06 (6.88e−11)	22015.4 (3712.3)	1578.08 (312.05)	1.00e−06 (5.01e−11)

Problem			MBTFWGSC			FW-Line Search			ASFWGSC		
	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−06											
cov_50	2500	50	1584.7 (410.5)	3.29 (1.07)	9.99e−07 (5.24e−10)	1481.5 (407.5)	11.12 (3.76)	9.99e−07 (6.58e−10)	172.8 (10.9)	**0.25 (0.03)**	9.11e−07 (4.26e−08)
cov_80	6400	80	3277.8 (677.6)	16.10 (4.96)	9.99e−07 (3.25e−10)	3116.0 (677.3)	47.82 (10.03)	9.99e−07 (2.50e−10)	267.9 (11.7)	**0.91 (0.17)**	9.51e−07 (2.96e−08)
cov_120	14400	120	4358.0 (846.9)	25.02 (5.85)	1.00e−06 (2.61e−10)	4112.2 (840.5)	99.25 (23.82)	1.00e−06 (2.03e−10)	399.6 (19.6)	**1.56 (0.25)**	9.73e−07 (1.96e−08)
cov_150	22500	150	4876.0 (1022.6)	38.89 (10.48)	1.00e−06 (2.19e−10)	4583.5 (1011.6)	246.57 (56.16)	1.00e−06 (2.12e−10)	495.0 (20.6)	**2.73 (0.31)**	9.69e−07 (1.57e−08)
cov_170	28900	170	6646.7 (1419.3)	119.90 (38.07)	1.00e−06 (1.79e−10)	6301.6 (1403.9)	477.78 (106.69)	1.00e−06 (1.70e−10)	575.3 (33.0)	**7.51 (1.11)**	9.79e−07 (1.52e−08)
cov_200	40000	200	7505.3 (1333.1)	289.89 (58.84)	1.00e−06 (1.25e−10)	7084.0 (1312.4)	838.13 (176.41)	1.00e−06 (8.71e−11)	681.4 (29.7)	**11.31 (0.69)**	9.78e−07 (1.64e−08)
cov_220	48400	220	7556.7 (1005.9)	335.30 (51.38)	1.00e−06 (1.58e−10)	7121.9 (996.9)	1091.47 (142.32)	1.00e−06 (8.41e−11)	724.1 (29.2)	**12.66 (1.38)**	9.82e−07 (1.32e−08)
cov_250	62500	250	8381.2 (1492.7)	566.67 (171.05)	1.00e−06 (1.44e−10)	7868.1 (1486.4)	2067.88 (583.44)	1.00e−06 (1.04e−10)	851.6 (35.8)	**34.29 (4.60)**	9.87e−07 (5.59e−09)
cov_270	72900	270	9553.1 (1394.7)	702.23 (198.56)	1.00e−06 (1.02e−10)	9004.9 (1393.2)	2750.95 (605.23)	1.00e−06 (9.68e−11)	895.6 (27.2)	**28.86 (4.55)**	9.80e−07 (5.52e−09)
cov_300	90000	300	10151.0 (1640.5)	847.26 (146.52)	1.00e−06 (1.22e−10)	9549.6 (1622.9)	3655.96 (604.43)	1.00e−06 (1.16e−10)	1002.3 (38.5)	**33.18 (2.92)**	9.92e−07 (5.68e−09)

Problem			PN			PG		
	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error
Relative error = 1e−06								
cov_50	2500	50	16.6 (2.3)	84.52 (13.55)	4.45e−08 (4.76e−08)	122.8 (86.9)	0.98 (0.58)	7.53e−08 (1.59e−07)

Table 5 continued

Problem			PN			PG		
	n	p	Iter	Time (s)	Error	Iter	Time (s)	Error
cov_80	6400	80	19.5 (3.1)	325.16 (73.64)	1.96e−07 (1.66e−07)	326.0 (235.0)	7.22 (5.06)	9.70e−08 (2.07e−07)
cov_120	14400	120	21.0 (2.2)	603.12 (32.05)	1.02e−07 (8.71e−08)	316.4 (172.8)	10.76 (5.35)	4.52e−08 (8.35e−08)
cov_150	22500	150	21.8 (2.6)	717.20 (105.85)	1.60e−07 (2.48e−07)	364.3 (201.4)	18.58 (10.56)	6.98e−09 (4.58e−09)
cov_170	28900	170	24.2 (3.2)	2038.72 (344.85)	3.16e−07 (2.88e−07)	614.3 (395.6)	64.89 (38.41)	1.16e−08 (2.66e−08)
cov_200	40000	200	25.2 (2.9)	1881.89 (217.82)	3.69e−07 (3.18e−07)	584.9 (265.9)	57.86 (24.19)	6.69e−08 (1.94e−07)
cov_220	48400	220	25.5 (1.6)	1977.83 (140.60)	2.54e−07 (2.34e−07)	658.3 (215.5)	73.91 (23.49)	7.18e−08 (1.94e−07)
cov_250	62500	250	29.6 (3.6)	5324.53 (762.52)	2.35e−07 (2.41e−07)	2457.6 (4179.1)	438.91 (637.89)	7.91e−08 (2.35e−07)
cov_270	72900	270	30.2 (4.6)	5256.39 (756.05)	2.50e−07 (2.22e−07)	3340.9 (5711.7)	589.97 (1004.03)	1.02e−07 (2.36e−07)
cov_300	90000	300	29.2 (2.0)	4986.63 (453.33)	2.12e−07 (3.05e−07)	973.7 (291.6)	215.23 (62.14)	2.08e−07 (2.69e−07)

Number of iterations and CPU time in seconds to achieve a certain relative error or best relative error achieved by methods, as well as the relative error achieved at that iteration

sublinear convergence rates. The key innovation of this paper is the design of new adaptive step-size policies and backtracking formulations, exploiting the specific problem structure of GSC-minimization problems. This paper also derives new linearly convergent projection-free methods for the minimization of GSC functions. Specifically, we show how to adapt the local linear minimization ideas of [24] to the current, potentially ill-conditioned, setup. Together with the concurrent paper [10], which appeared on arXive after this work has been submitted for publication, we also derive a new linearly convergent variant of the FW method featuring linear global convergence rates for GSC functions. With the help of extensive numerical experiments, we demonstrate the practical efficiency of our approach.

We conclude by mentioning some interesting potential extensions. First, our theory could be used to derive distributed versions of the algorithms presented in this paper in order to develop a generalized and projection-free variants of the DISCO algorithm [56] or distributed cubic-regularized Newton's methods [1,14]. These are Newton methods capable to minimize a self-concordant function using distributed computations. Projection-free methods which are able to handle the same problem, but now including generalized self-concordant functions, have the potential to be serious competitors in practice. Second, it will be interesting to incorporate gradient sliding techniques [31], and stochastic versions of our algorithms. Recently, a Newton Frank–Wolfe method has been introduced in [34]. It seems natural to us that their algorithm can be extended to GSC functions. All these are important extensions, which we are planning to pursue in the near future.

Acknowledgements The authors sincerely thank Professor Shoham Sabach for his contribution in the early stages of this project, including his part in developing the basic ideas developed in this paper. We would also like to thank Professor Quoc Tranh-Dinh for fruitful discussions on this topic and in sharing MATLAB codes of SCOPT with us. Feedback from Professors Robert M. Freund and Sebastian Pokutta are also gratefully acknowledged. Finally, we would like to thank the Associate Editor and the Reviewers for their valuable remarks and suggestions. M. Staudigl acknowledges support by the COST Action CA16228 "European Network for Game Theory". S. Shtern was partially supported by the Israel Science Foundation, Grant 1460/19. The research in Sections 3.1, 3.2, 4.1 was supported by Russian Science Foundation (project No. 21-71-30005).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

A Additional facts about GSC functions

In order to make this paper self-contained we are collecting in this appendix finer estimates provided by generalized self-concordance. For a complete treatise the reader should consult the seminal paper [50]. An important feature of GSC functions is their invariance under affine transformations. This is made precise in the following Lemma.

Lemma 9 ([50], Prop. 2) *Let $f \in \mathcal{F}_{M_f,\nu}$ and $\mathcal{A}(x) = Ax + b : \mathbb{R}^n \to \mathbb{R}^p$ a linear operator. Then*

(a) *If $\nu \in [2, 3]$, then $\tilde{f}(x) \triangleq f(\mathcal{A}(x))$ is $(M_{\tilde{f}}, \nu)$-GSC with $M_{\tilde{f}} = M_f \|A\|^{3-\nu}$.*

(b) *If $\nu > 3$ and $\lambda_{\min}(A^\top A) > 0$, then $\tilde{f}(x) = f(\mathcal{A}(x))$ is $(M_{\tilde{f}}, \nu)$-GSC with $M_{\tilde{f}} = M_f \lambda_{\min}(A^\top A)^{\frac{3-\nu}{2}}$, where $\lambda_{\min}(A^\top A)$ is the smallest eigenvalue of $A^\top A$.*

When we apply FW to the minimization of a function $f \in \mathcal{F}_M$, the search direction at position x is determined by the target state $s(x) = s$ defined in (1). If $A : \tilde{\mathcal{X}} \to \mathcal{X}$ is a surjective linear re-parametrization of the domain $\mathcal{X}$, then the new optimization problem $\min_{\tilde{\mathcal{X}}} \tilde{f}(\tilde{x}) = f(A\tilde{x})$ is still within the frame of problem (P). Furthermore, the updates produced by FW are not affected by this re-parametrization since $\langle \nabla \tilde{f}(\tilde{x}), \hat{s} \rangle = \langle \nabla f(A\tilde{x}), A\hat{s} \rangle = \langle \nabla f(x), s \rangle$ for $x = A\tilde{x} \in \mathcal{X}$, $s = A\hat{s} \in \mathcal{X}$.

Beside affine invariance, we will use some stability properties of GSC functions.

Proposition 5 ([50], Prop. 1) *Let $f_i \in \mathcal{F}_{M_{f_i},\nu}$ where $M_{f_i} \geq 0$ and $\nu \geq 2$ for $i = 1, \ldots, N$. Then, given scalars $w_i > 0$, $1 \leq i \leq N$, the function $f \triangleq \sum_{i=1}^N w_i f_i$ is well defined on $\operatorname{dom} f \triangleq \bigcap_{i=1}^N \operatorname{dom} f_i$ and belongs to $\mathcal{F}_{M_f,\nu}$, where $M_f \triangleq \max_{1 \leq i \leq N} w_i^{1-\frac{\nu}{2}} M_{f_i}$.*

As corollary of this Proposition and invariance under linear transformations, we obtain the next characterization theorem, which is of particular importance in machine learning applications.

Given N functions $\varphi_i \in \mathcal{F}_{M_{\varphi_i},\nu}$. For $(a_i, b_i) \in \mathbb{R}^n \times \mathbb{R}$, $q \in \mathbb{R}^n$ and $Q \in \mathbb{R}^{n \times n}$ a positive definite and symmetric matrix, consider the finite-sum model

$$f(x) \triangleq \sum_{i=1}^N \varphi_i(\langle a_i, x \rangle + b_i) + \langle q, x \rangle + \frac{1}{2}\langle Qx, x \rangle \tag{60}$$

Proposition 6 ([50], Prop. 5) *If $\varphi_i \in \mathcal{F}_{M_{\varphi_i},\nu}$ for $\nu \in (0, 3]$, then $f : \mathbb{R}^n \to (-\infty, \infty]$ defined in (60) belongs to $\mathcal{F}_{M_f,3}$, where $M_f \triangleq \lambda_{\min}(Q)^{(\nu-3)/2} \max_{1 \leq i \leq N} M_{\varphi_i} \|a_i\|_2^{3-\nu}$.*

B Proof of Theorem 1

B.1 Preparations

The proof of Proposition 1 is an application of the technical Lemma below.

Lemma 10 *Consider the function*

$$\psi_\nu(t) \triangleq t - \xi\omega_\nu(t\delta)t^2, \tag{61}$$

where $\xi, \delta \geq 0$ *are parameters and* $\nu \in [2, 3]$. *For all* $\nu \in [2, 3]$, *the function* $t \mapsto \psi_\nu(t)$ *is concave and differentiable. The unique maximum of this function is achieved at*

$$t_\nu^* \triangleq \begin{cases} \frac{1}{\delta}\ln\left(1+\frac{\delta}{\xi}\right) & \text{if } \nu = 2, \\ \frac{1}{\delta}\left[1-\left(1+\frac{\delta}{\xi}\frac{4-\nu}{\nu-2}\right)^{-\frac{\nu-2}{4-\nu}}\right] & \text{if } \nu \in (2,3), \\ \frac{1}{\delta+\xi} & \text{if } \nu = 3, \end{cases} \tag{62}$$

Proof We will organize the proof of Lemma 10 according to the generalized self-concordance parameter $\nu \in [2, 3]$.

The case $\nu = 2$: For this parameter we have $\omega_2(t) = \frac{1}{t^2}[e^t - t - 1]$, and thus

$$\psi_2(t) = t - \frac{\xi}{\delta^2}[e^{t\delta} - t\delta - 1].$$

This is a strictly concave function with unique maximum at

$$t_2^* = \frac{1}{\delta}\ln\left(1+\frac{\delta}{\xi}\right). \tag{63}$$

The case $\nu \in (2, 3)$: Since $\omega_\nu(t) = \left(\frac{\nu-2}{4-\nu}\right)\frac{1}{t}\left[\frac{\nu-2}{2(3-\nu)t}((1-t)^{\frac{2(3-\nu)}{2-\nu}} - 1) - 1\right]$, some simple algebra shows that

$$\psi_\nu(t) = t\left(1+\frac{\xi}{\delta}\frac{\nu-2}{4-\nu}\right) - \frac{\xi}{\delta^2}\frac{(\nu-2)^2}{2(3-\nu)(4-\nu)}\left[(1-t\delta)^{\frac{2(3-\nu)}{2-\nu}} - 1\right].$$

Setting $\psi_\nu'(t) = 0$, yields the value

$$t_\nu^* = \frac{1}{\delta}\left[1-\left(1+\frac{\delta}{\xi}\frac{4-\nu}{\nu-2}\right)^{-\frac{\nu-2}{4-\nu}}\right].$$

It is easy to check that $\psi_\nu''(t) = -\xi(1-t\delta)^{\frac{2}{2-\nu}} < 0$, so that t^* is the global maximum of $\psi_\nu(t)$.

The case $\nu = 3$: For this case, we have $\omega_3(t) = \frac{-t-\ln(1-t)}{t^2}$. It is easy to see that

$$\psi_3(t) = t + \frac{\xi}{\delta^2}[t\delta + \ln(1-t\delta)] \qquad t \in (0, 1/\delta).$$

Therefore, for $t \in (0, 1/\delta)$, we see that

$$\psi_3'(t) = 1 + \frac{\xi}{\delta^2}\left(\delta - \frac{\delta}{1 - t\delta}\right), \text{ and } \psi_3''(t) = -\frac{\xi}{\delta}(1 - t\delta)^{-2} < 0.$$

The unique maximum is attained at

$$t_3^* = \frac{1}{\delta + \xi}.$$

B.2 Proof of Theorem 1

Identifying the parameters involved in (61) as $\delta = M_f \delta_\nu(x)$, and $\xi = \frac{\mathsf{e}(x)^2}{\mathsf{Gap}(x)}$ gives us

$$\eta_{x, M_f, \nu}(t) = \mathsf{Gap}(x)\psi_\nu(t).$$

Hence, the following explicit expressions for the step-size parameters are immediate consequences of Lemma 10.

$\nu = 2$: Since $M_f \delta_2(x) = M_f \beta(x)$ we get the relation

$$\mathtt{t}_{M_f, 2}(x) = \frac{1}{M_f \beta(x)} \ln\left(1 + \frac{M_f \beta(x)}{\mathsf{e}(x)^2}\mathsf{Gap}(x)\right).$$

$\nu \in (2, 3)$: Set $\delta = M_f \delta_\nu(x) = \frac{\nu - 2}{2} M_f \beta(x)^{3-\nu} \mathsf{e}(x)^{\nu-2}$ and $\xi = \frac{\mathsf{e}(x)^2}{\mathsf{Gap}(x)}$, we get

$$\mathtt{t}_{M_f, \nu}(x) = \frac{2}{\nu - 2}\frac{1}{M_f}\beta(x)^{\nu-3}\mathsf{e}(x)^{2-\nu} \times \left[1 - \left(1 + \frac{4 - \nu}{2} M_f \beta(x)^{3-\nu}\mathsf{e}(x)^{\nu-4}\mathsf{Gap}(x)\right)^{\frac{2-\nu}{4-\nu}}\right].$$

$\nu = 3$: Since $M_f \delta_3(x) = \frac{M_f}{2}\mathsf{e}(x)$, we get

$$\mathtt{t}_{M_f, 3}(x) = \frac{\mathsf{Gap}(x)}{\frac{M}{2}\mathsf{e}(x)(\frac{2}{M}\mathsf{e}(x) + \mathsf{Gap}(x))}$$

This completes the proof of Theorem 1. □

C Auxiliary results needed in the proof of Theorem 1

C.1 Proof of Lemma 4

Set $x \equiv x^k$. Since $\mathtt{t}_{M_f, \nu}(x) > 1$, the decrease of the objective function is

$$\eta_{x, M_f, \nu}(1) = \mathsf{Gap}(x)\left(1 - \frac{\mathsf{e}(x)^2}{\mathsf{Gap}(x)}\omega_\nu(M_f \delta_\nu(x))\right).$$

If $\nu > 2$ we know that $M_f\delta_\nu(x) \le \mathtt{t}_\nu(x) M_f\delta_\nu(x) < 1$, and the expression above is well-defined. If $\nu = 2$, the domain of the function ω_2 is full, and again the expression above is well-defined. Set $\zeta_\nu(t) \triangleq \omega_\nu(t M_f\delta_\nu(x))t^2$ and $\xi(x) \triangleq \frac{\mathsf{e}(x)^2}{\mathsf{Gap}(x)}$, so that

$$\frac{\eta_{x,M_f,\nu}(t)}{\mathsf{Gap}(x)} = t - \zeta_\nu(t)\xi(x),$$

where $t \in (0,\infty)$ if $\nu = 2$ and $t \in (0, \frac{1}{M_f\delta_\nu(x)})$ for $\nu \in (2,3]$. By definition, $\mathtt{t}_{M_f,\nu}(x)$ is the unconstrained maximizer of the right-hand-side above. Therefore, $1 - \xi(x)\zeta'_\nu(\mathtt{t}_{M_f,\nu}(x)) = 0$. Since $t \mapsto \zeta_\nu(t)$ is convex, its derivative is a non-decreasing function. Thus, since we assume that $1 < \mathtt{t}_{M_f,\nu}(x)$, it follows $\xi(x) = \frac{1}{\zeta'_\nu(\mathtt{t}_{M_f,\nu}(x))} \le \frac{1}{\zeta'_\nu(1)}$. Moreover, $\zeta_\nu(1) \ge 0$, so that

$$\begin{aligned}\frac{\eta_{x,M_f,\nu}(1)}{\mathsf{Gap}(x)} &= 1 - \xi(x)\zeta_\nu(1) = 1 - \frac{\zeta_\nu(1)}{\zeta'_\nu(\mathtt{t}_{M_f,\nu}(x))} \ge 1 - \frac{\zeta_\nu(1)}{\zeta'_\nu(1)} \\ &= 1 - \frac{\omega_\nu(M_f\delta_\nu(x))}{2\omega_\nu(M_f\delta_\nu(x)) + M_f\delta_\nu(x)\omega'_\nu(M_f\delta_\nu(x))} \\ &\ge \frac{1}{2}.\end{aligned}$$

where we used that $\omega'_\nu(t) \ge 0$ for $t > 0$. □

C.2 Proof of Lemma 5

We first prove a general lower estimate on the per-iteration progress.

Lemma 11 *Suppose that $\mathtt{t}_\nu(x^k) \le 1$. Then, the per-iteration progress in the objective function value is lower bounded by*

$$\Delta_k \ge \begin{cases} \frac{2\ln(2)-1}{\mathsf{e}(x^k)} \min\left\{\frac{\mathsf{e}(x)\,\mathsf{Gap}(x^k)}{M_f\beta(x^k)}, \frac{\mathsf{Gap}(x^k)^2}{\mathsf{e}(x^k)}\right\} & \text{if } \nu = 2, \\ \tilde{\gamma}_\nu \min\left\{\frac{\mathsf{Gap}(x^k)}{\frac{\nu-2}{2}M_f\beta(x^k)^{3-\nu}\mathsf{e}(x^k)^{\nu-2}}, \frac{-1}{\mathtt{b}}\frac{\mathsf{Gap}(x^k)^2}{\mathsf{e}(x^k)^2}\right\} & \text{if } \nu \in (2,3), \\ \frac{2(1-\ln(2))}{M_f\mathsf{e}(x^k)} \min\left\{\mathsf{Gap}(x^k), \frac{M_f\,\mathsf{Gap}(x^k)^2}{\mathsf{e}(x^k)}\right\} & \text{if } \nu = 3. \end{cases} \tag{64}$$

where $\tilde{\gamma}_\nu \triangleq 1 + \frac{4-\nu}{2(3-\nu)}\left(1 - 2^{2(3-\nu)/(4-\nu)}\right)$ and $\mathtt{b} \triangleq \frac{2-\nu}{4-\nu}$.

We demonstrate this result as a corollary of the technical lemma below.

Lemma 12 *Consider function $t \mapsto \psi_\nu(t)$ defined in eq. (61) with unique maximum t_ν^* as described in eq. (62). It holds that*

$$\psi_\nu(t_\nu^*) = \begin{cases} \frac{1}{\delta}\left((1+\frac{\xi}{\delta})\ln\left(1+\frac{\delta}{\xi}\right)-1\right) & \text{if } \nu = 2, \\ \frac{1}{\delta}\left(1-\frac{\mathtt{a}\mathtt{b}\xi}{\delta}+\frac{\mathtt{a}\mathtt{b}\xi}{\delta}\left(1-\frac{1}{\mathtt{b}}\frac{\delta}{\xi}\right)^{\mathtt{b}+1}\right) & \text{if } \nu \in (2,3), \\ \frac{1}{\delta}\left(1-\frac{\xi}{\delta}\ln\left(1+\frac{\delta}{\xi}\right)\right) & \text{if } \nu = 3. \end{cases} \tag{65}$$

where $\mathtt{a} \triangleq \frac{4-\nu}{2(3-\nu)}$ *and* $\mathtt{b} \triangleq \frac{2-\nu}{4-\nu} < 0$. *Moreover, the following lower bound holds*

$$\psi_\nu(t_\nu^*) \geq \begin{cases} \frac{2\ln 2-1}{\delta}\min\{1, \frac{\delta}{\xi}\} & \text{if } \nu = 2, \\ \frac{\tilde{\gamma}_\nu}{\delta}\min\left\{1, -\frac{\delta}{\xi\mathtt{b}}\right\} & \text{if } \nu \in (2,3), \\ \frac{1-\ln 2}{\delta}\min\{1, \frac{\delta}{\xi}\} & \text{if } \nu = 3. \end{cases} \tag{66}$$

where

$$\tilde{\gamma}_\nu \triangleq 1 + \frac{4-\nu}{2(3-\nu)}\left(1-2^{2(3-\nu)/(4-\nu)}\right). \tag{67}$$

Proof We organize the proof according to the value of $\nu \in [2,3]$.

The case $\nu = 2$: Since $\psi_2(t) = t - \frac{\xi}{\delta^2}[e^{t\delta} - t\delta - 1]$, once we plug in t_2^* from eq. (63) we arrive, after some computations, at

$$\psi_2(t_2^*) = \frac{1}{\delta}\left((1+\frac{\xi}{\delta})\ln(1+\frac{\delta}{\xi}) - 1\right)$$

We next establish the lower bound formulated in (66). Denote $\phi(t) \triangleq (1+t)\ln\left(1+\frac{1}{t}\right) - 1$. Then $\psi(t_2^*) = \phi(\frac{\xi}{\delta})/\delta$. At the same time,

$$\frac{d\phi(t)}{dt} = \ln\left(1+\frac{1}{t}\right) + (1+t)\cdot\frac{t}{1+t}\cdot\left(-\frac{1}{t^2}\right) = \ln\left(1+\frac{1}{t}\right) - \frac{1}{t} < 0.$$

Thus, $\phi(t)$ is decreasing and $\phi(t) \geq \phi(1) = 2\ln 2 - 1$ when $t \in (0,1]$.

Let us now consider the function $t \mapsto \frac{\phi(t)}{1/t}$.

$$\frac{d}{dt}\left(\frac{\phi(t)}{1/t}\right) = \phi(t) + t\phi'(t) = (2t+1)\ln\left(1+\frac{1}{t}\right) - 2 \geq 0.$$

Hence, $\frac{\phi(t)}{1/t} \geq \phi(1) = 2\ln 2 - 1$ when $t \in (1, +\infty)$. Combining these two cases, we see that

$$\psi_2(t_2^*) = \frac{1}{\delta}\phi(\xi/\delta) \geq (2\ln(2)-1)\min\{1/\delta, 1/\xi\}. \tag{68}$$

The case $\nu \in (2,3)$: A computation shows that

$$\psi_\nu(t_\nu^*) = \frac{1}{\delta}\left[1 - \frac{4-\nu}{2(3-\nu)}\left(1 + \frac{\delta}{\xi}\frac{4-\nu}{\nu-2}\right)^{\frac{2-\nu}{4-\nu}}\right] + \frac{\xi}{\delta^2}\frac{(\nu-2)}{2(3-\nu)}\left[1 - \left(1 + \frac{\delta}{\xi}\frac{4-\nu}{\nu-2}\right)^{\frac{2-\nu}{4-\nu}}\right].$$

Set $\mathtt{a} \triangleq \frac{4-\nu}{2(3-\nu)} > 0$ and $\mathtt{b} \triangleq \frac{2-\nu}{4-\nu} < 0$. Then, setting $u = 1 - \frac{1}{\mathtt{b}}\frac{\delta}{\xi}$, we see that

$$\begin{aligned}\psi_\nu(t_\nu^*) &= \frac{1}{\delta}\left(1 - \frac{\xi\mathtt{ab}}{\delta} - \mathtt{a}u^{\mathtt{b}} + \mathtt{ab}\frac{\xi}{\delta}u^{\mathtt{b}}\right)\\ &= \frac{1}{\delta}\left[1 - \frac{\mathtt{ab}\xi}{\delta} + \frac{\mathtt{ab}\xi}{\delta}\left(1 - \frac{1}{\mathtt{b}}\frac{\delta}{\xi}\right)^{\mathtt{b}+1}\right]\end{aligned}$$

To verify the lower bound, we rewrite $\psi_\nu(t_\nu^*)$ as follows:

$$\begin{aligned}\psi_\nu(t_\nu^*) &= \frac{1}{\delta}\left(1 - \mathtt{a}u^{\mathtt{b}} + \frac{\mathtt{a}}{u-1}(1-u^b)\right)\\ &= \frac{1}{\delta}\left(1 + \frac{\mathtt{a}}{u-1} - \frac{\mathtt{a}u^{\mathtt{b}+1}}{u-1}\right)\\ &= \frac{1}{\delta}\gamma(u),\end{aligned}$$

where $\gamma(u) \triangleq 1 + \frac{\mathtt{a}}{u-1} - \frac{\mathtt{a}u^{\mathtt{b}+1}}{u-1}$. Our next goal is to show that, for $u \in [2,+\infty)$, $\gamma(u)$ is below bounded by some positive constant and, for $u \in (1,2]$, $\gamma(u)$ is below bounded by some positive constant multiplied by $u-1$.

1. $u \in [2,+\infty)$. We will show that $\gamma'(u) \geq 0$, whence $\gamma(u) \geq \gamma(2)$. Thus, we need to show that

$$0 \leq \gamma'(u) = -\frac{\mathtt{a}}{(u-1)^2}\underbrace{\left(1 - (\mathtt{b}+1)u^{\mathtt{b}} + \mathtt{b}u^{\mathtt{b}+1}\right)}_{=h(u)}.$$

Since $\mathtt{a} > 1$, to show that $\gamma'(u) \geq 0$ it is enough to show that $h(u) \leq 0$. Since $\mathtt{b} \in (-1,0)$ and $t \geq 2$,

$$h'(u) = \mathtt{b}(\mathtt{b}+1)u^{\mathtt{b}} - \mathtt{b}(\mathtt{b}+1)u^{\mathtt{b}-1} = \mathtt{b}(\mathtt{b}+1)u^{\mathtt{b}-1}(u-1) \leq 0.$$

Whence, $h(u) \leq h(2)$ for all $u \in [2,+\infty)$. It remains to show that $h(2) \leq 0$. Let us consider $h(2) = \varphi(\mathtt{b}) := 1 - (\mathtt{b}+1)2^{\mathtt{b}} + \mathtt{b}2^{\mathtt{b}+1} = 1 + \mathtt{b}2^{\mathtt{b}} - 2^{\mathtt{b}}$ as a function of $\mathtt{b} \in (-1,0)$. Clearly, $\varphi(-1) = \varphi(0) = 0$, and it is easy to check via the intermediate value theorem that $\varphi(b) < 0$ for all $b \in (-1,0)$. We conclude that for $u \geq 2$ we get $\psi_\nu(t_2^*) \geq \frac{1}{\delta}\gamma(2)$.

2. $t \in (1, 2]$. We will show that $\frac{d}{du}\left(\gamma(u)/(u-1)\right) \leq 0$, whence $\gamma(u) \geq (u-1)\gamma(2)$. Thus, we need to show that

$$\begin{aligned} 0 &\geq \frac{d}{dt}\left(\frac{1}{u-1} + \frac{\mathtt{a}}{(u-1)^2} - \frac{\mathtt{a}u^{\mathtt{b}+1}}{(u-1)^2}\right) \\ &= \frac{1}{(u-1)^3}\left(-u + 1 - 2\mathtt{a} + \mathtt{a}(\mathtt{b}+1)u^{\mathtt{b}} - \mathtt{a}(\mathtt{b}-1)u^{\mathtt{b}+1}\right) \equiv \frac{1}{(u-1)^3}h(u). \end{aligned}$$

Therefore, our next step is to show that $h(u) \leq 0$. We have

$$\begin{aligned} h'(u) &= -1 + \mathtt{a}(\mathtt{b}+1)\mathtt{b}u^{\mathtt{b}-1} - \mathtt{a}(\mathtt{b}-1)(\mathtt{b}+1)u^{\mathtt{b}}, \\ h''(u) &= \mathtt{ab}(\mathtt{b}+1)(\mathtt{b}-1)u^{\mathtt{b}-2} - \mathtt{a}(\mathtt{b}-1)\mathtt{b}(\mathtt{b}+1)u^{\mathtt{b}-1} \\ &= \mathtt{ab}(\mathtt{b}+1)(\mathtt{b}-1)u^{\mathtt{b}-2}(1-u). \end{aligned}$$

By definition, $\mathtt{a}(\mathtt{b}+1) = 1$. Hence, since $u > 1$ and $\mathtt{b} \in (-1, 0)$, we observe that $h''(u) \leq 0$. Thus, $h'(u) \leq h'(1) = 0$, and consequently, $h(u) \leq h(1) = 0$, for all $u \in (1, 2]$. This proves the claim $\gamma(u)/(u-1) \geq \gamma(2)$ for $u \in (1, 2]$.

Combining both cases, we obtain that $\gamma(u) \geq \min\{\gamma(2), (u-1)\gamma(2)\}$, where $\gamma(2) = 1 - \mathtt{a} + \mathtt{a}2^{1/\mathtt{a}}$, using the fact that $\mathtt{b}+1 = 1/\mathtt{a}$. Unraveling this expression by using the definition of the constant $\mathtt{a}$, we see that $\gamma(2)$ depends only on the self-concordance parameter $\nu \in (2, 3)$. In light of this, let us introduce the constant

$$\tilde{\gamma}_\nu \triangleq 1 + \frac{4-\nu}{2(3-\nu)}\left(1 - 2^{2(3-\nu)/(4-\nu)}\right). \tag{69}$$

Observe that $\tilde{\gamma}_2 = 0$ and, by a simple application of l'Hôpital's rule, $\lim_{\nu\uparrow 3}\hat{\gamma}_\nu = 1 - \log(2) \in (0, 1)$. Hence $\gamma(2) \equiv \tilde{\gamma}_\nu \in (0, 1)$ for all $\nu \in (2, 3)$. We conclude,

$$\psi_\nu(t_\nu^*) \geq \frac{\tilde{\gamma}_\nu}{\delta}\min\left\{1, \frac{-1}{\mathtt{b}}\frac{\delta}{\xi}\right\} \tag{70}$$

The case $\nu = 3$: A direct substitution for $\psi_3(t)$ gives us

$$\psi_3(t_3^*) = \frac{1}{\delta} + \frac{\xi}{\delta^2}\ln\left(\frac{\xi}{\delta+\xi}\right). \tag{71}$$

Denote $u = \xi/\delta$. Then $t_3^* = \frac{1}{\delta+\xi}$, so that

$$\psi_3(t_3^*) = \frac{1}{\delta}\left[1 + u\ln\left(\frac{u}{u+1}\right)\right].$$

Consider the function $\phi : (0,\infty) \to (0,\infty)$, given by $\phi(t) := 1 + t\ln\left(\frac{t}{1+t}\right)$. Then, $\psi_3(t_3^*) = \frac{1}{\delta}\phi(\xi/\delta)$. For $t \in (0,1)$, one sees

$$\phi'(t) = \ln\left(\frac{t}{1+t}\right) + t\frac{1+t}{t}\left(\frac{1}{1+t} - \frac{t}{(1+t)^2}\right) = \ln\left(1 - \frac{1}{1+t}\right) + \frac{1}{1+t} < 0.$$

Consequently, $\phi(t)$ is decreasing for $t \in (0,1)$. Hence, $\phi(t) \geq \phi(1) = 1 - \ln 2$, for all $t \in (0,1)$. On the other hand, if $t \geq 1$,

$$\frac{d}{dt}\left(\frac{\phi(t)}{1/t}\right) = \frac{d}{dt}(t\phi(t)) = 1 + 2t\ln\left(\frac{t}{1+t}\right) + \frac{t}{1+t} \geq 0.$$

Hence, $t \mapsto \frac{\phi(t)}{1/t}$ is an increasing function for $t \geq 1$, and thus $\phi(t) \geq \frac{1-\ln 2}{t}$, for all $t \geq 1$. Summarizing these two cases we see

$$\psi_3(t_3^*) \geq \frac{1}{\delta}\min\{1, \delta/\xi\}(1 - \ln(2)) = (1 - \ln(2))\min\{1/\delta, 1/\xi\}. \tag{72}$$

Proof of Lemma 11 Recall that $\eta_{x,M_f,\nu}(t) = \mathsf{Gap}(x)\psi_\nu(t)$. By identifying the parameters appropriately, we can give the proof of Lemma 11 as a straightforward exercise derived from Lemma 12. We provide the explicit derivation for each GSC parameter ν below.

$\nu = 2$: Substitute in (63) the parameter values $\xi = \frac{\mathsf{e}(x)^2}{\mathsf{Gap}(x)}$ and $\delta = M_f\delta_2(x) = M_f\beta(x)$, the lower bound turns into

$$\psi_2(\mathtt{t}_{M_f,2}(x)) \geq \frac{2\ln(2) - 1}{\mathsf{e}(x)}\min\left\{\frac{\mathsf{e}(x)}{M_f\beta(x)}, \frac{\mathsf{Gap}(x)}{\mathsf{e}(x)}\right\}. \tag{73}$$

Hence,

$$\begin{aligned}\Delta_k &\geq \mathsf{Gap}(x^k)\frac{2\ln(2) - 1}{\mathsf{e}(x)}\min\left\{\frac{\mathsf{e}(x)}{M_f\beta(x)}, \frac{\mathsf{Gap}(x)}{\mathsf{e}(x)}\right\} \\ &= \frac{2\ln(2) - 1}{\mathsf{e}(x)}\min\left\{\frac{\mathsf{e}(x^k)\,\mathsf{Gap}(x^k)}{M_f\beta(x^k)}, \frac{\mathsf{Gap}(x^k)^2}{\mathsf{e}(x^k)}\right\}.\end{aligned}$$

$\nu \in (2,3)$: Substitute in (70) the parameter values $\delta \equiv M_f\delta_\nu(x) = \frac{\nu-2}{2}M_f\beta(x)^{3-\nu}\mathsf{e}(x)^{\nu-2}$, $\xi \equiv \frac{\mathsf{e}(x)^2}{\mathsf{Gap}(x)}$, so that

$$\psi_\nu(\mathtt{t}_{M_f,\nu}(x)) \geq \tilde{\gamma}_\nu\min\left\{\frac{1}{\frac{\nu-2}{2}M_f\beta(x)^{3-\nu}\mathsf{e}(x)^{\nu-2}}, \frac{-1}{\mathtt{b}}\frac{\mathsf{Gap}(x)}{\mathsf{e}(x)^2}\right\}. \tag{74}$$

$$\text{Hence, } \Delta_k \geq \tilde{\gamma}_\nu\min\left\{\frac{\mathsf{Gap}(x^k)}{\frac{\nu-2}{2}M_f\beta(x^k)^{3-\nu}\mathsf{e}(x^k)^{\nu-2}}, \frac{-1}{\mathtt{b}}\frac{\mathsf{Gap}(x^k)^2}{\mathsf{e}(x^k)^2}\right\}.$$

$\nu = 3$: Substitute in (72) the parameter values $\delta \equiv \delta_3(x) = \frac{M_f}{2}\mathsf{e}(x)$, $\xi \equiv \frac{\mathsf{e}(x)^2}{\mathsf{Gap}(x)}$, to get

$$\psi_3(\mathsf{t}_3(x)) \geq \frac{2(1-\ln(2))}{M_f \mathsf{e}(x)} \min\left\{1, \frac{M_f\, \mathsf{Gap}(x)}{\mathsf{e}(x)}\right\}. \tag{75}$$

Hence, $\Delta_k \geq \frac{2(1-\ln(2))}{M_f \mathsf{e}(x^k)} \min\left\{\mathsf{Gap}(x^k), \frac{M_f\, \mathsf{Gap}(x^k)^2}{\mathsf{e}(x^k)}\right\}$.

Proof of Lemma 5 Use the estimates $\beta(x) \leq \operatorname{diam}(\mathcal{X})$ and $\mathsf{e}(x) \leq \sqrt{L_{\nabla f}}\beta(x) \leq \sqrt{L_{\nabla f}}\operatorname{diam}(\mathcal{X})$ in the expressions provided in Lemma 64.

References

1. Agafonov, A., Dvurechensky, P., Scutari, G., Gasnikov, A., Kamzolov, D., Lukashevich, A., Daneshmand, A.: An accelerated second-order method for distributed stochastic optimization. In: 2021 60th IEEE Conference on Decision and Control (CDC) (2021). arXiv:2103.14392
2. Bach, F.: Self-concordant analysis for logistic regression. Electron. J. Stat. **4**, 384–414 (2010). https://doi.org/10.1214/09-EJS521
3. Baes, M.: Estimate Sequence Methods: Extensions and Approximations. Institute for Operations Research, ETH, Zürich (2009)
4. Beck, A., Shtern, S.: Linearly convergent away-step conditional gradient for non-strongly convex functions. Math. Program. **164**(1), 1–27 (2017). https://doi.org/10.1007/s10107-016-1069-4
5. Beck, A., Teboulle, M.: A conditional gradient method with linear rate of convergence for solving convex linear systems. Math. Methods Oper. Res. **59**(2), 235–247 (2004)
6. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**(1), 183–202 (2009). https://doi.org/10.1137/080716542
7. Ben-Tal, A., Nemirovski, A.: Lectures on Modern Convex Optimization (Lecture Notes). Personal webpage of A. Nemirovski (2020). https://www2.isye.gatech.edu/~nemirovs/LMCOLN2020WithSol.pdf
8. Bomze, I.M., Mertikopoulos, P., Schachinger, W., Staudigl, M.: Hessian barrier algorithms for linearly constrained optimization problems. SIAM J. Optim. **29**(3), 2100–2127 (2019)
9. Candes, E.J., Strohmer, T., Voroninski, V.: PhaseLift: exact and stable signal recovery from magnitude measurements via convex programming. Commun. Pure Appl. Math. **66**(8), 1241–1274 (2013)
10. Carderera, A., Besancon, M., Pokutta, S.: Simple steps are all you need: Frank-Wolfe and generalized self-concordant functions. arXiv:2105.13913 (2021)
11. Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, Cambridge (2006)
12. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. (2011). https://doi.org/10.1145/1961189.1961199
13. Cover, T.M.: Universal portfolios. Math. Finance **1**(1), 1–29 (1991). https://doi.org/10.1111/j.1467-9965.1991.tb00002.x
14. Daneshmand, A., Scutari, G., Dvurechensky, P., Gasnikov, A.: Newton method over networks is fast up to the statistical precision. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 139, pp. 2398–2409. PMLR (2021). http://proceedings.mlr.press/v139/daneshmand21a.html
15. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002). https://doi.org/10.1007/s101070100263
16. Dvurechensky, P., Nesterov, Y.: Global performance guarantees of second-order methods for unconstrained convex minimization (2018). CORE Discussion Paper 2018/32
17. Dvurechensky, P., Ostroukhov, P., Safin, K., Shtern, S., Staudigl, M.: Self-concordant analysis of Frank-Wolfe algorithms. In: Singh, H.D.A. (eds.) Proceedings of the 37th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 119, pp. 2814–2824. PMLR, Virtual (2020). http://proceedings.mlr.press/v119/dvurechensky20a.html. arXiv:2002.04320

18. Dvurechensky, P., Shtern, S., Staudigl, M.: First-order methods for convex optimization. EURO J. Comput. Optim. (2021). https://doi.org/10.1016/j.ejco.2021.100015. arXiv:2101.00935
19. Dvurechensky, P., Staudigl, M.: Hessian barrier algorithms for non-convex conic optimization. arXiv:2111.00100 (2021)
20. Dvurechensky, P., Staudigl, M., Uribe, C.A.: Generalized self-concordant hessian-barrier algorithms. Preprint arXiv:1911.01522 (2019)
21. Epelman, M., Freund, R.M.: Condition number complexity of an elementary algorithm for computing a reliable solution of a conic linear system. Math. Program. **88**(3), 451–485 (2000). https://doi.org/10.1007/s101070000136
22. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval Res. Logist. Q. **3**(1–2), 95–110 (1956). https://doi.org/10.1002/nav.3800030109
23. Freund, R.M., Grigas, P., Mazumder, R.: An extended Frank-Wolfe method with "in-face"directions, and its application to low-rank matrix completion. SIAM J. Optim. **27**(1), 319–346 (2017). https://doi.org/10.1137/15M104726X
24. Garber, D., Hazan, E.: A linearly convergent variant of the Conditional Gradient algorithm under strong convexity, with applications to online and stochastic optimization. SIAM J. Optim. **26**(3), 1493–1528 (2016). https://doi.org/10.1137/140985366
25. GuéLat, J., Marcotte, P.: Some comments on Wolfe's 'away step'. Math. Program. **35**(1), 110–119 (1986). https://doi.org/10.1007/BF01589445
26. Gutman, D.H., Peña, J.F.: The condition number of a function relative to a set. Math. Program. (2020). https://doi.org/10.1007/s10107-020-01510-4
27. Harchaoui, Z., Juditsky, A., Nemirovski, A.: Conditional gradient algorithms for norm-regularized smooth convex optimization. Math. Program. **152**(1), 75–112 (2015). https://doi.org/10.1007/s10107-014-0778-9
28. Jaggi, M.: Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In: International Conference on Machine Learning, pp. 427–435 (2013)
29. Lacoste-Julien, S., Jaggi, M.: On the global linear convergence of Frank-Wolfe optimization variants. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 28, pp. 496–504. Curran Associates, Inc. (2015). https://proceedings.neurips.cc/paper/2015/file/c058f544c737782deacefa532d9add4c-Paper.pdf
30. Lan, G.: The complexity of large-scale convex programming under a linear optimization oracle. Preprint arXiv:1309.5550 (2013)
31. Lan, G., Zhou, Y.: Conditional gradient sliding for convex optimization. SIAM J. Optim. **26**(2), 1379–1409 (2016). https://doi.org/10.1137/140992382
32. Levitin, E.S., Polyak, B.T.: Constrained minimization methods. USSR Comput. Math. Math. Phys. **6**(5), 1–50 (1966). https://doi.org/10.1016/0041-5553(66)90114-5
33. Li, Y.H., Cevher, V.: Convergence of the exponentiated gradient method with Armijo line search. J. Optim. Theory Appl. **181**(2), 588–607 (2019). https://doi.org/10.1007/s10957-018-1428-9
34. Liu, D., Cevher, V., Tran-Dinh, Q.: A Newton Frank-Wolfe method for constrained self-concordant minimization. Preprint arXiv:2002.07003 (2020)
35. Marron, J.S., Todd, M.J., Ahn, J.: Distance-weighted discrimination. J. Am. Stat. Assoc. **102**(480), 1267–1271 (2007)
36. Marteau-Ferey, U., Bach, F., Rudi, A.: Globally convergent newton methods for ill-conditioned generalized self-concordant losses. Preprint arXiv:1907.01771 (2019)
37. Marteau-Ferey, U., Ostrovskii, D., Bach, F., Rudi, A.: Beyond least-squares: Fast rates for regularized empirical risk minimization through self-concordance. In: Beygelzimer, A., Hsu, D. (eds.) Proceedings of the Thirty-Second Conference on Learning Theory, Proceedings of Machine Learning Research, vol. 99, pp. 2294–2340. PMLR, Phoenix, USA (2019). http://proceedings.mlr.press/v99/marteau-ferey19a.html
38. Merhav, N., Feder, M.: Universal prediction. IEEE Trans. Inf. Theory **44**(6), 2124–2147 (1998)
39. Nesterov, Y.: A method of solving a convex programming problem with convergence rate $O(1/k^2)$. Soviet Math. Doklady **27**(2), 372–376 (1983)
40. Nesterov, Y.: Complexity bounds for primal-dual methods minimizing the model of objective function. Math. Program. **171**(1), 311–330 (2018). https://doi.org/10.1007/s10107-017-1188-6
41. Nesterov, Y.: Lectures on Convex Optimization, Springer Optimization and Its Applications, vol. 137. Springer (2018)

42. Nesterov, Y., Nemirovski, A.: Interior Point Polynomial methods in Convex programming. SIAM Publications (1994)
43. Nocedal, J., Wright, S.J.: Numerical Optimization, 2nd edn. Springer (2000)
44. Odor, G., Li, Y.H., Yurtsever, A., Hsieh, Y.P., Tran-Dinh, Q., El Halabi, M., Cevher, V.: Frank-Wolfe works for non-Lipschitz continuous gradient objectives: Scalable poisson phase retrieval. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6230–6234 (2016)
45. Ostrovskii, D.M., Bach, F.: Finite-sample analysis of m-estimators using self-concordance. Electron. J. Stat. **15**(1), 326–391 (2021). https://doi.org/10.1214/20-EJS1780
46. Owen, A.B.: Self-concordance for empirical likelihood. Can. J. Stat. **41**(3), 387–397 (2013). https://doi.org/10.1002/cjs.11183
47. Pedregosa, F., Negiar, G., Askari, A., Jaggi, M.: Linearly convergent Frank-Wolfe with backtracking line-search. In: International Conference on Artificial Intelligence and Statistics, pp. 1–10. PMLR (2020)
48. Peña, J., Rodríguez, D.: Polytope conditioning and linear convergence of the Frank-Wolfe algorithm. Math. Oper. Res. **44**(1), 1–18 (2018). https://doi.org/10.1287/moor.2017.0910
49. Stonyakin, F., Tyurin, A., Gasnikov, A., Dvurechensky, P., Agafonov, A., Dvinskikh, D., Alkousa, M., Pasechnyuk, D., Artamonov, S., Piskunova, V.: Inexact model: A framework for optimization and variational inequalities. Optimization Methods and Software (2021). 10.1080/10556788.2021.1924714. WIAS Preprint No. 2709, arXiv:2001.09013, arXiv:1902.00990
50. Sun, T., Tran-Dinh, Q.: Generalized self-concordant functions: a recipe for Newton-type methods. Math. Program. (2018). https://doi.org/10.1007/s10107-018-1282-4
51. Tran-Dinh, Q., Kyrillidis, A., Cevher, V.: An inexact proximal path-following algorithm for constrained convex minimization. SIAM J. Optim. **24**(4), 1718–1745 (2014). https://doi.org/10.1137/130944539
52. Tran-Dinh, Q., Kyrillidis, A., Cevher, V.: Composite self-concordant minimization. J. Mach. Learn. Res. **16**(1), 371–416 (2015)
53. Tran-Dinh, Q., Li, Y.H., Cevher, V.: Composite convex minimization involving self-concordant-like cost functions. In: LeThi, H.A., Pham Dinh, T., Nguyen, N.T. (eds.) Modelling, Computation and Optimization in Information Systems and Management Sciences, pp. 155–168. Springer, Cham (2015)
54. Tunçel, L., Nemirovski, A.: Self-concordant barriers for convex approximations of structured convex sets. Found. Comput. Math. **10**(5), 485–525 (2010). https://doi.org/10.1007/s10208-010-9069-x
55. Wolfe, P.: Integer and Nonlinear Programming, chap. Convergence Theory in Nonlinear Programming. North-Holland Publishing Company (1970)
56. Zhang, Y., Lin, X.: DiSCO: Distributed optimization for self-concordant empirical loss. In: Proceedings of the 32nd International Conference on Machine Learning, pp. 362–370. PMLR (2015). http://proceedings.mlr.press/v37/zhangb15.html
57. Zhao, R., Freund, R.M.: Analysis of the Frank-Wolfe method for convex composite optimization involving a logarithmically-homogeneous barrier. Preprint arXiv:2010.08999 (2020)
58. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **67**(2), 301–320 (2005)