Noname manuscript No.
(will be inserted by the editor)

Adaptive Conditional Gradient Descent

Abbas Khademi · Antonio Silveti-Falls

Received: date / Accepted: date

Abstract Selecting an effective step-size is a fundamental challenge in first-order optimization, especially for problems with non-Euclidean geometries. This paper presents a novel adaptive step-size strategy for optimization algorithms that rely on linear minimization oracles, as used in the Conditional Gradient or non-Euclidean Normalized Steepest Descent algorithms. Using a simple heuristic to estimate a local Lipschitz constant for the gradient, we can determine step-sizes that guarantee sufficient decrease at each iteration. More precisely, we establish convergence guarantees for our proposed Adaptive Conditional Gradient Descent algorithm, which covers as special cases both the classical Conditional Gradient algorithm and non-Euclidean Normalized Steepest Descent algorithms with adaptive step-sizes. Our analysis covers optimization of continuously differentiable functions in non-convex, quasar-convex, and strongly convex settings, achieving convergence rates that match state-of-the-art theoretical bounds. Comprehensive numerical experiments validate our theoretical findings and illustrate the practical effectiveness of Adaptive Conditional Gradient Descent. The results exhibit competitive performance, underscoring the potential of the adaptive step-size for applications.

Keywords Adaptive step-size · Linear minimization oracle · First-order method · Projection-free method · Non-convex optimization · Conditional Gradient · Frank-Wolfe · Steepest-descent

Mathematics Subject Classification 65K05 · 90C26 · 90C30 · 49M37

1 Introduction

1.1 Problem Context

First-order methods based on linear minimization oracles offer a powerful paradigm for structured optimization, providing either projection-free optimization for constrained problems or geometry-adaptive descent for unconstrained problems. These methods have found applications ranging from high-dimensional matrix completion [49], sparse optimization [8,36], constrained multiobjective optimization problems [2,19], submodular optimization [9,51], convex mixed-integer optimization [24], and traffic flow problems [32] to training deep neural networks with billions of parameters [45,46], where the linear minmization oracle structure naturally encodes problem geometry through its extremal points.

This framework unifies two important algorithms that originate from different problem settings. In constrained optimization, it yields the Conditional Gradient algorithm, which sidesteps potentially expensive projections onto a constraint set. In unconstrained optimization, it produces the Normalized Steepest Descent algorithm, which adapts the direction of descent to the geometry of the problem (or our

A. Khademi
School of Mathematics, Institute for Research in Fundamental Sciences (IPM), P.O.Box 19395-5746, Tehran, Iran.
School of Mathematics and Computer Science, Iran University of Science and Technology, Tehran, Iran.
E-mail: abbaskhademi92@gmail.com
https://orcid.org/0000-0002-8276-6821

A. Silveti-Falls *(Corresponding Author)*
CVN, CentraleSupélec, Université Paris-Saclay, Inria, France.
E-mail: tonys.falls@gmail.com
https://orcid.org/0000-0002-8165-6348

choosing). Both methods share a critical dependency: their convergence rates and practical performance hinge on appropriate step-size selection, which is a challenge that existing approaches address only partially.

While various step-size strategies have been proposed in the literature, ranging from conservative, descent-driven choices that require knowledge of the global Lipschitz constant of the gradient [16] to unassuming, parameter-free open-loop schedules [17] and adaptive backtracking approaches [44], none fully exploit the local geometry that the iterative nature of these algorithms naturally encounters. We develop an adaptive step-size strategy that estimates local Lipschitz constants of the gradient on-the-fly using previous gradient information followed by a backtracking correction procedure, combining the robustness of backtracking with the efficiency of curvature-aware steps.

To formalize our approach, we consider the following general problem setting

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

where $\mathcal{X} \subseteq \mathbb{R}^n$ and $f\colon \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function. We will assume that the gradient ∇f is Lipschitz-continuous with respect to a norm $\|\cdot\|$, i.e., there exists a constant $L > 0$ such that for all $x, y \in \mathcal{X}$,

$$\|\nabla f(x) - \nabla f(y)\|_* \leq L\|x - y\|,$$

which we will also sometimes refer to as f being L-smooth. While we will assume that L exists, we will not assume knowledge of the value of L in our study of the problem.

For a non-empty compact convex set $\mathcal{C}$, most crucially, we will assume that we are able to solve a so-called Linear Minimization Oracle (LMO) of the form

$$\operatorname*{lmo}_{\mathcal{C}}(x) := \operatorname*{argmin}_{v\in\mathcal{C}} \langle x, v\rangle, \tag{1}$$

where $\mathrm{lmo}_{\mathcal{C}}(x)$ can be any minimizer in the case that there are many, The LMO always returns a boundary point of $\mathcal{C}$ (typically an extremal point) and this structure is what enables both the projection-free property of the Conditional Gradient algorithm and the adaptation to the geometry of the problem in the Normalized Steepest Descent algorithm. The specific structure of the LMO output, i.e., sparse for ℓ^1-balls, dense for ℓ^∞-balls, rank-1 for nuclear norm-balls, full-rank for operator norm-balls, etc, endows these algorithms with their distinctive computational properties.

We will study a general algorithmic template, GenCG, based on the LMO for solving (P), which is applicable to both the unconstrained and constrained variants; when applied to unconstrained problems it is sometimes referred to as Normalized Steepest Descent algorithm, the Continuous Greedy algorithm, or the Generalized Matching Pursuit algorithm while it is called the Frank-Wolfe algorithm or the Conditional Gradient algorithm when applied to constrained problems. The only difference in the template between the two types of problems is a slight modification to the update direction d^k in the constrained case, to ensure feasibility. However, both algorithms share the same core structure in that they query an LMO at each iteration to obtain a direction, then take a step in that direction and it is this shared structure that allows us to develop a unified step-size strategy that works for both.

Conditional Gradient The Conditional Gradient algorithm is a classical method to solve (P) when $\mathcal{X} = \mathcal{C}$. At every iteration k, the algorithm linearizes the objective function f around the current iterate x^k and then minimizes this linearization over the constraint set $\mathcal{C}$, which amounts to querying the LMO. Then, since the LMO is always in $\mathcal{C}$, taking a convex combination between the current feasible iterate x^k and the LMO output v^k is guaranteed to remain in $\mathcal{C}$ by convexity. Since the iterates never leave $\mathcal{C}$, there is no need to project back onto $\mathcal{C}$ and the method is thus *projection-free*.

Normalized Steepest Descent The Normalized Steepest Descent algorithm is also a classic method to solve (P) when $\mathcal{X} = \mathbb{R}^n$. It is analogous to doing normalized gradient descent but in a non-Euclidean norm, which is advantageous in many problems whose geometry is not necessarily Euclidean. By selecting a particular norm, we can enforce structured updates (sparse, dense, low-rank, full-rank, etc) or improve the convergence rate for problems with particular geometries. Similar to the Conditional Gradient algorithm, it requires computing a linearization of the objective function f around the current iterate x^k that is then minimized over a unit-ball $\mathcal{C}$ for some norm to get some update direction. Rather than taking a convex combination with the current iterate, though, Normalized Steepest Descent directly steps in the direction output by the LMO.

1.2 Related Work

Step-size selection in LMO-based methods has been extensively studied since the seminal work of Frank and Wolfe [18]. Classical approaches range from the short-step step-size rule, which requires knowledge of the global Lipschitz constant L and sets $t_k = \min\{-\langle \nabla f(x^k), d^k\rangle/(L\|d^k\|^2), 1\}$ [5,17], to parameter-free open-loop schedules such as $t_k = 2/(2+k)$ that sacrifice adaptivity for simplicity [17]. While exact line search maximizes per-iteration progress, its computational cost often outweighs its benefits, especially when gradients are expensive to evaluate [15]. These classical methods either require problem-specific constants that are difficult to estimate in practice or fail to adapt to the local geometry of the objective function.

There is a closely related work [44] to ours, which introduced a unified backtracking framework for both Conditional Gradient and Normalized Steepest Descent that eliminates the need for knowing the global Lipschitz constant L. Their adaptive strategy maintains an estimate of the local Lipschitz constant that is decreased by a multiplicative factor (typically 0.9) when backtracking occurs. While this approach successfully adapts to the problem landscape without prior knowledge of L, it suffers from slow adaptation when the iterates move into regions of significantly lower curvature—requiring multiple iterations for the estimate to decrease substantially. Our work builds directly upon this foundation by incorporating gradient-based local Lipschitz constant estimation to accelerate this adaptation process.

Several other step-size strategies have been proposed for LMO-based methods. For the Conditional Gradient side, there has recently been introduced [23] a secant-based line search that adapts to local smoothness with reduced computational cost compared to exact line search, though it requires additional gradient evaluations. Other works [53] have demonstrated accelerated convergence with open-loop step-sizes under specific geometric conditions, while optimistic step-size variants that go beyond the short-step rule have also been proposed [39]. On the Normalized Steepest Descent side, the connection to continuous greedy [9] and generalized matching pursuit has been formalized in [35], which provided a unified optimization view establishing explicit convergence rates for matching pursuit methods in an optimization sense, demonstrating the deep connection between these seemingly separate algorithms.

Our analysis holds for quasar-convex functions, which are a broader class of functions than convex functions or even star-convex functions, and which have gotten more attention recently [20,21,40,52]. The convergence of the Conditional Gradient algorithm under star-convexity has been analyzed in a recent work [41] under a few different step-size schedules which include open-loop, short-step, and backtracking. Of note is that their analysis of the adaptive backtracking algorithm [44] uses the star-convexity assumption, attaining similar rates to ours for the more general quasar-convex functions.

Another relevant line of work [37,38] developed gradient-based heuristics for estimating local smoothness constants in proximal gradient methods. Their approach uses consecutive gradients and iterates to estimate the local Lipschitz constant without requiring additional function evaluations. While their work focused on proximal methods, the principle of exploiting gradient information for local curvature estimation has broader applicability, as we demonstrate in this work.

Contributions This paper presents an adaptive step-size method for GenCG that estimates the local Lipschitz constant of the gradient through a simple heuristic. Unlike methods that rely on the global Lipschitz constant (as in the short-step) or backtracking alone [44], our approach adapts efficiently to varying problem landscapes where global properties are difficult to estimate or exhibit significant variation. Our theoretical contributions are rigorous convergence analyses of both variants of GenCG for (P) under convex, quasar-convex, and non-convex settings. Notably, under an additional quasar-convexity assumption, we establish improved convergence rates in the non-convex case compared to [44]. Our use of local Lipschitz constants furthermore yields simplified proofs throughout. We validate our approach's superiority through comprehensive synthetic experiments designed to isolate and demonstrate the benefits of local adaptation across diverse problem geometries and conditioning. By warm-starting the backtracking procedure with gradient-based estimates of the local Lipschitz constant, our algorithm rapidly adapts to changes in the problem landscape, which addresses a limitation of pure backtracking approaches. These experiments confirm our hypothesis that leveraging local curvature information leads to faster practical convergence.

1.3 Outline

The structure of the remainder of the paper is as follows. Section 2 reviews the Conditional Gradient method, Normalized Steepest Descent and greedy methods, their connection through the LMO, and classic step-size choices for both methods: the short-step rule, open-loop schedules, and line-search. Section 3 presents our adaptive step-size strategy with local Lipschitz constant estimation, detailing our methods for overcoming the slow adaptation of existing pure backtracking schemes. Section 4 describes the proposed optimization algorithms in full and provides a convergence analysis that covers strongly convex, convex, quasar-convex, and non-convex settings. Section 5 presents numerous numerical experiments, evaluating the Adaptive Conditional Gradient and Normalized Steepest Descent algorithms. Finally, Section 6 concludes the paper with a summary of findings and possible avenues to continue this research direction.

Notation and Conventions

We will denote the natural numbers $\mathbb{N}$, which includes 0, and use $\mathbb{N}^*$ to denote $\mathbb{N} \setminus \{0\}$. For a Hilbert space, the inner product is denoted by $\langle \cdot, \cdot \rangle$. The dual norm of $\|\cdot\|$ is denoted by $\|\cdot\|_*$ and is defined by $\|v\|_* := \max_{\|x\| \le 1} \langle v, x \rangle$. We denote the closed ball of radius $\rho > 0$ centered at the origin by $\mathcal{B}(\rho) := \{x : \|x\| \le \rho\}$ and, when ρ is omitted, we will take by convention the unit-ball. Moreover, we define functional-value gap at $x \in \mathcal{X}$ as $h(x) := f(x) - f^\star$, where $f^\star = \inf_{x \in \mathcal{X}} f(x)$ is the optimal objective value, and the Frank-Wolfe gap as $\max_{v \in \mathcal{X}} \langle \nabla f(x), x - v \rangle$, a measure of non-stationarity. For $x \in \mathbb{R}^n$ and $q \ge 1$, let $\|x\|_q$ denote the ℓ^q-norm of x, defined as $\|x\|_q = (\sum_{i=1}^n |x_i|^q)^{1/q}$. For $X \in \mathbb{R}^{n \times m}$ and $q \in [1, +\infty]$, let $\|X\|_{\mathcal{S}^q}$ denote the Schatten-q norm of X, defined by applying the ℓ^q norm to the singular values of X. For $a \in \mathbb{R}$, define $\operatorname{sign}(a)$ as $\operatorname{sign}(a) = 1$ if $a > 0$, -1 if $a < 0$, and 0 if $a = 0$. For $v \in \mathbb{R}^n$, $\operatorname{sign}(v) \in \{-1, 0, 1\}^n$ denotes the vector of signs of the elements of v. We will denote the indicator function for a set A by $\iota_A \colon x \mapsto \begin{cases} 0, & x \in A \\ +\infty, & x \notin A \end{cases}$ and the normal cone N_A to be its subdifferential in the convex sense. Finally, we denote the linear span of a set V by $\operatorname{span}(V)$ and the convex hull of V by $\operatorname{conv}(V)$. We will denote the Minkowski multiplication of a set $A \subset \mathbb{R}^n$ by a real number $\lambda \in \mathbb{R}$ by $\lambda A := \{\lambda a \colon a \in A\}$. A function $f \colon \mathbb{R}^n \to \mathbb{R}$ will be said to be *strongly convex* with respect to the norm $\|\cdot\|$ if $\exists \mu > 0$ such that $f - \mu\|\cdot\|^2$ is convex. Moreover, a differentiable function f will be called *η-quasar-convex* if $\exists x^\star \in \operatorname*{argmin}_{x \in \mathcal{X}} f(x)$, $\eta \in]0, 1] \colon \forall y \in \mathbb{R}^n, f(x^\star) - f(y) \ge \frac{1}{\eta} \langle \nabla f(y), x^\star - y \rangle$.

2 Preliminaries

We being this section with the generic LMO template in GenCG, which was also studied in [44]. This is a very general template that encompasses both the Conditional Gradient algorithm and the Normalized Steepest Descent algorithm, depending on whether $\mathcal{X} = \mathcal{C}$ or $\mathcal{X} = \mathbb{R}^n$.

Algorithm 1 Generic Conditional Gradient (GenCG)

```
Input: Initial point $x^0 \in \mathcal{X}$, compact convex set $\mathcal{C}$ (typically a closed unit-ball $\mathcal{B}_{\mathrm{lmo}}(1)$), step-size sequence $\{t_k\}_{k \ge 0} \subseteq [0, +\infty[$.
Initialization: Set iteration counter $k \leftarrow 0$.
Repeat: For $k = 0, 1, 2, \dots$, perform the following steps:
    Compute $v^k \leftarrow \underset{\mathcal{C}}{\mathrm{lmo}}(\nabla f(x^k))$.
    Set $d^k \leftarrow \begin{cases} v^k, & \text{if } \mathcal{X} = \mathbb{R}^n \quad \text{(Normalized Steepest Descent)} \\ v^k - x^k. & \text{if } \mathcal{X} = \mathcal{C} \quad \text{(Conditional Gradient)} \end{cases}$
    Set $x^{k+1} \leftarrow x^k + t_k d^k$.
    Update $k \leftarrow k + 1$.
Stop: Terminate when convergence criteria is satisfied.
```

While the LMO is accessible for a multitude of sets $\mathcal{C}$, we will focus especially on the case where $\mathcal{C} = \mathcal{B}_{\mathrm{lmo}}(1)$ is the closed unit-ball in some norm that we will denote $\|\cdot\|_{\mathrm{lmo}}$ for clarity.[1] In this case, the LMO returns an element of the subdifferential (in the convex sense) of the dual norm $\|\cdot\|_{\mathrm{lmo}\,*}$, i.e.,

$$d \in \underset{\mathcal{C}}{\mathrm{lmo}}(x) \iff -d \in \partial \|x\|_{\mathrm{lmo}\,*}.$$

[1] All of the convergence results we will present readily extend to the more general case where $\mathcal{C}$ is a level-set of some general Minkowski gauge function.

Moreover, the output of the LMO is always on the boundary of $\mathcal{C}$, typically an extremal point/atom of the set in a convex-analytic sense, and this endows the output of the LMO with a certain structure based on what the extremal points of $\mathcal{C}$ are. For instance, when $\mathcal{C}$ is the unit-ball for the ℓ^1 norm (i.e., $\|\cdot\|_{\mathrm{lmo}} = \|\cdot\|_1$), the atoms are the 1-sparse vectors and the output of the LMO is then typically 1-sparse for non-zero inputs. In contrast, when $\mathcal{C}$ is the unit-ball for the ℓ^∞ norm (i.e., $\|\cdot\|_{\mathrm{lmo}} = \|\cdot\|_\infty$), the atoms are dense sign vectors and so the output of the LMO is then typically dense for non-zero inputs. This extends to spectral norms on matrices (also known as Schatten p-norms); for the nuclear norm or Schatten 1-norm, the atoms of $\mathcal{C}$ are rank-1 matrices and so the output of the LMO in this case is typically rank-1 while for the operator norm or Schatten ∞-norm, the atoms of $\mathcal{C}$ are orthogonal matrices and the output of the LMO is typically full-rank.

The LMO is also familiar in the constrained optimization community due to its use in the Frank-Wolfe [18] or Conditional Gradient [34] algorithms, as well as in the greedy optimization community where it has been used for greedy and matching pursuit problems. As we will soon describe, the algorithms we consider for solving (P) are centered around the LMO and are inspired by the Conditional Gradient algorithm. We will consider two distinct cases for (P), for which we will develop two corresponding algorithms and analyses: either $\mathcal{X} = \mathcal{C} \subset \mathbb{R}^n$ for some non-empty compact convex set $\mathcal{C}$ (and the problem is constrained) or $\mathcal{X} = \mathbb{R}^n$ (and the problem is unconstrained). While most of our results in the unconstrained setting hold for general non-empty sets $\mathcal{C}$ so long as they are compact and convex, we will sometimes state our results specifically for the case that $\mathcal{C}$ is the closed unit-ball $\mathcal{B}_{\mathrm{lmo}}(1)$ induced by some norm of our choosing $\|\cdot\|_{\mathrm{lmo}}$, as this is often the most relevant for problems in practice and it greatly simplifies the statement of the results in the unconstrained setting.

In both unconstrained and constrained settings, GenCG provides a certificate of criticality for (P) in the form of the quantity

$$\mathrm{Gap}(x^k) := -\langle \nabla f(x^k), d^k\rangle = \begin{cases} \langle \nabla f(x^k), x^k - v^k\rangle, & \text{(Conditional Gradient)} \\ \|\nabla f(x^k)\|_{\mathrm{lmo}\,*}. & \text{(Normalized Steepest Descent)} \end{cases} \tag{2}$$

Since $\nabla f(x^k)$ and d^k are already computed to run the algorithm, the cost to compute this certificate is cheap and can be used a stopping criterion. For the unconstrained case where $\mathcal{X} = \mathbb{R}^n$, a point $x^\star$ is a critical point if and only if $\|\nabla f(x^\star)\|_{\mathrm{lmo}\,*} = 0$, which is equivalent to $\mathrm{Gap}(x^\star) = 0$. For the constrained case where $\mathcal{X} = \mathcal{C}$, criticality is characterized by $-\nabla f(x) \in N_{\mathcal{C}}(x)$ which is equivalent to the so-called Frank-Wolfe gap $\max_{v\in\mathcal{C}}\langle \nabla f(x), x - v\rangle$ being zero, which is also equivalent to $\mathrm{Gap}(x^k)$ being zero and inspires our choice of name for this quantity. We finally remark that the choice of norm $\|\cdot\|_{\mathrm{lmo}}$ for the LMO directly relates to the convergence guarantees we will get, which are all in terms fo Gap.

2.1 Conditional Gradient Method

The Conditional Gradient method, originally proposed by Frank and Wolfe [18] in 1956 for quadratic programming and later generalized by Levitin and Polyak [34] to broader settings, has become more popular since its introduction to machine learning in [12,26]. Its appeal lies in its simplicity, relying only on gradient evaluations and the LMO instead of projections, which may be more costly for certain choices of $\mathcal{C}$ [14]. This makes it particularly applicable to many large-scale structured optimization problems that arise in machine learning and beyond [7,26]. At each iteration k, the algorithm approximates f around the current point x^k using the linear model:

$$f(x^k) + \langle \nabla f(x^k), v - x^k\rangle, \tag{3}$$

where $v \in \mathcal{C}$. Minimizing this linear model over $\mathcal{C}$ then corresponds to querying the LMO for $\mathcal{C}$ at the point $\nabla f(x^k)$, which gives a direction $v^k \in \mathcal{C}$. The next iterate x^{k+1} is finally constructed as a convex combination of the current point x^k and the oracle-selected direction v^k, parameterized by a step-size $t_k \in [0,1]$:

$$x^{k+1} = (1 - t_k)x^k + t_k v^k, \tag{4}$$

or equivalently,

$$x^{k+1} = x^k + t_k d^k, \tag{5}$$

where $d^k := v^k - x^k = \mathrm{lmo}_{\mathcal{C}}(\nabla f(x^k)) - x^k$. This update ensures that x^{k+1} remains feasible within $\mathcal{C}$, thanks to the convexity of $\mathcal{C}$, as illustrated in Figure 1. Consequently, no projection onto $\mathcal{C}$ is needed, making the Conditional Gradient method a projection-free method.

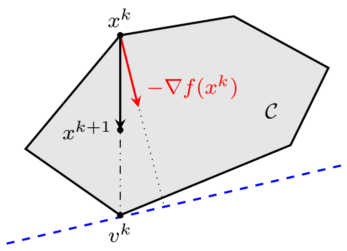


Fig. 1 Illustration of one step of the Conditional Gradient algorithm. Starting at x^k, the gradient $\nabla f(x^k)$ is computed and used to compute v^k, an output of the LMO. The next iterate x^{k+1} is obtained by moving along the line segment connecting x^k to v^k. The blue dashed line represents a supporting hyperplane to the set $\mathcal{C}$ dictated by its normal direction $-\nabla f(x^k)$.

2.2 Normalized Steepest Descent, Continuous Greedy, and Generalized Matching Pursuit

While the Conditional Gradient method discussed in the previous part is suited to solving constrained problems, by a slight modification to the definition of the update direction d^k in GenCG, one can arrive at a variant capable of solving unconstrained problems. This algorithm has appeared several times throughout the literature under different names for different cases, e.g., the Normalized Steepest Descent method [6] when $\mathcal{C}$ is the closed unit-ball for some norm $\|\cdot\|_{\mathrm{lmo}}$, the continuous greedy method [9,51], or generalized matching pursuit [35] when $\mathcal{C}$ is the convex hull of some set of atoms or a dictionary to be optimized over.

Normalized Steepest Descent Recall that the typical gradient descent algorithm with fixed step-size t can be written,

$$x^{k+1} = x^k - t\nabla f(x^k) = \operatorname*{argmin}_{x\in\mathbb{R}^n} f(x^k) + \langle \nabla f(x^k), x - x^k\rangle + \frac{1}{2t}\|x - x^k\|_2^2. \tag{6}$$

If one replaces the Euclidean term $\|x - x^k\|_2^2$ by an arbitrary norm $\|x - x^k\|_{\mathrm{lmo}}^2$ then the resulting algorithm is called *Steepest Descent with respect to the norm* $\|\cdot\|_{\mathrm{lmo}}$ [29,42,43] and is given by

$$x^{k+1} = x^k + t\|\nabla f(x^k)\|_{\mathrm{lmo}*} \operatorname*{lmo}_{\mathcal{C}}(\nabla f(x^k)), \tag{7}$$

where $\|\cdot\|_{\mathrm{lmo}*}$ is the dual norm to $\|\cdot\|_{\mathrm{lmo}}$ and $\mathcal{C}$ is the closed unit-ball induced by the norm $\|\cdot\|_{\mathrm{lmo}}$. When $\|\cdot\|_{\mathrm{lmo}}$ is the Euclidean norm, the dual norm $\|\cdot\|_{\mathrm{lmo}*}$ is Euclidean again and Steepest Descent reduces to the familiar gradient descent algorithm. When the function f has a gradient ∇f that is Lipschitz-continuous with respect to the norm $\|\cdot\|_{\mathrm{lmo}}$, taking $t = \frac{1}{L}$ recovers the analog of the short-step in the unconstrained setting. Indeed, by definition of the dual norm it follows that

$$t\|\nabla f(x^k)\|_{\mathrm{lmo}*} = t\max_{v\in\mathcal{C}}\langle \nabla f(x^k), v\rangle = t\left\langle \nabla f(x^k), -\operatorname*{lmo}_{\mathcal{C}}(\nabla f(x^k))\right\rangle. \tag{8}$$

On the other hand, if we replace $\|x - x^k\|_2^2$ by $\iota_{t\mathcal{C}}(x - x^k)$, then the resulting method can be interpreted as a non-Euclidean trust region method, as noted in [45], with an update to x^k that is always normalized in the norm $\|\cdot\|_{\mathrm{lmo}}$. This method is the so-called *Normalized Steepest Descent with respect to the norm* $\|\cdot\|_{\mathrm{lmo}}$ and is illustrated for an example in Figure 2. When t_k is chosen like in the open-loop step-size setting, the method is exactly the unconstrained analog to the Conditional Gradient algorithm.

Beyond Normalized Steepest Descent with standard norm-balls, several related methods follow similar LMO-based frameworks. The continuous greedy method [9,51] applies this approach to submodular maximization, using a specific step-size schedule $t_k = 1/N$ that ensures the iterates sum to one after N iterations. Generalized matching pursuit [35] extends the framework to arbitrary sets of atoms $\mathcal{A}$, seeking minimizers over $\mathrm{span}(\mathcal{A})$ by querying the LMO over $\mathrm{conv}(\mathcal{A}\cup -\mathcal{A})$. When $\mathcal{A}$ consists of standard basis vectors, this reduces to greedy coordinate descent [36].

While these methods demonstrate the versatility of the LMO framework, they typically employ fixed or problem-specific step-sizes. Our work focuses on adaptive step-size strategies for the core Normalized Steepest Descent method with norm-ball constraints, where the local Lipschitz constant estimation can be efficiently computed and provides clear computational benefits.

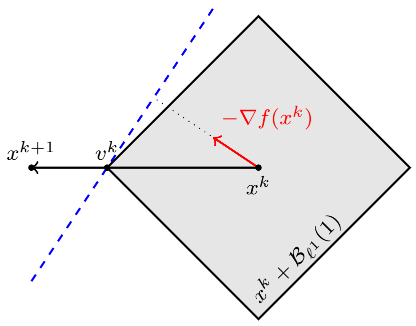


Fig. 2 One step of the Normalized Steepest Descent method using the ℓ^1 unit-ball centered at x^k. Starting at x^k, the gradient $\nabla f(x^k)$ is computed and used to compute v^k, an output of the LMO over $\mathcal{C} = \mathcal{B}_{\ell^1}(1)$. The next iterate x^{k+1} is then found by adding $t_k v^k$ to x^k. The blue dashed line represents the supporting hyperplane to the set $x^k + \mathcal{B}_{\ell^1}(1)$ dictated by its normal direction $-\nabla f(x^k)$.

2.3 Classic Step-size Strategies

The convergence and practical performance of LMO-based methods critically depend on step-size selection. We review the main strategies that have been studied for both Conditional Gradient and Normalized Steepest Descent, highlighting connections that are often overlooked in the literature.

Short-Step The short-step can be found by denoting $t_{\max} = \begin{cases} 1, & \text{(Conditional Gradient)} \\ +\infty, & \text{(Normalized Steepest Descent)} \end{cases}$ and then minimizing a quadratic upper bound on the progress from x^k to x^{k+1} from the descent lemma over $[0, t_{\max}]$:

$$\operatorname*{argmin}_{t\in[0,t_{\max}]} f(x^k) + t\langle \nabla f(x^k), d^k\rangle + \frac{L}{2}t^2\|d^k\|^2.$$

It is given in closed-form by

$$t_k = \min\left\{-\frac{\langle \nabla f(x^k), d^k\rangle}{L\|d^k\|^2}, t_{\max}\right\} = \begin{cases} \min\left\{-\frac{\langle \nabla f(x^k), d^k\rangle}{L\|d^k\|^2}, 1\right\}, & \text{(Conditional Gradient)} \\ \frac{\|\nabla f(x^k)\|_{\mathrm{lmo}*}}{L\|d^k\|^2}. & \text{(Normalized Steepest Descent)} \end{cases} \tag{9}$$

For Conditional Gradient, this appears as a quasi-adaptive step-size that depends on the Frank-Wolfe gap and the Lipschitz constant L of the gradient. For Normalized Steepest Descent with $\mathcal{C} = \mathcal{B}(\rho)$ and $\|\cdot\| = \|\cdot\|_{\mathrm{lmo}}$, this is precisely Steepest Descent with fixed step-size $1/L$ in the norm $\|\cdot\|_{\mathrm{lmo}}$. For example, in the ℓ^2 geometry, the update becomes

$$x^{k+1} = x^k + \frac{\|\nabla f(x^k)\|_2}{L}\operatorname*{lmo}_{\mathcal{B}_{\ell^2}(1)}(\nabla f(x^k)) = x^k + \frac{\|\nabla f(x^k)\|_2}{L}\left(\frac{-\nabla f(x^k)}{\|\nabla f(x^k)\|_2}\right) = x^k - \frac{1}{L}\nabla f(x^k).$$

This reveals that standard gradient descent with step-size $1/L$ can be interpreted to be Normalized Steepest Descent with the short-step. The short-step was also used in Steepest Descent under the name *sharp operator* in stochastic spectral descent [10].

Open-loop Schedule An open-loop schedule requires no knowledge of the parameter of the problem (e.g., L) and is given, for all $k \in \mathbb{N}$, by

$$t_k = \frac{2}{k+2}. \tag{10}$$

This schedule works for both Conditional Gradient and Normalized Steepest Descent without requiring knowledge of L but, on the other hand, it is not adaptive and will remain large even if the iterates are close to a critical point. While widely used in the Conditional Gradient literature [26,53], its applicability to unconstrained optimization is less appreciated. For a “parameter-free” gradient descent method, the LMO perspective suggests using open-loop schedules with Normalized Steepest Descent, with no line search or Lipschitz constant estimation needed. Not only is this method parameter-free, it is even affine-invariant [31].

Line Search A line-search schedule maximizes per-iteration progress along the current direction and is given by

$$t_k \in \underset{t\in[0,t_{\max}]}{\operatorname{argmin}}\ f(x^k + td^k). \tag{11}$$

Though optimal per iteration when the minimization is performed exactly, the computational cost of multiple function evaluations can outweigh the benefits, motivating adaptive strategies like those we develop in this work.

3 Adaptive Step-Size Selection with Local Lipschitz Estimation

In search of a step-size that is as effective as the line-search schedule without the added computational overhead, one can start by adapting the short-step strategy discussed in Section 2. Normally, this requires knowledge of the global Lipschitz constant L that is often unknown or an overly conservative upper bound of the curvature when known. We now review and develop adaptive strategies that estimate local smoothness properties to achieve better practical performance with this step-size.

Pure Backtracking In [44] the authors introduced an elegant backtracking scheme that maintains an estimate L_k of the local Lipschitz constant. At each iteration, the algorithm computes a candidate step-size using L_k in place of L in the short-step formula, then checks the sufficient decrease condition

$$f(x^k + t_k d^k) \le f(x^k) + t_k\langle \nabla f(x^k), d^k\rangle + \frac{L_k}{2} t_k^2 \|d^k\|_2^2.$$

We warn the reader that the norm used to compute the LMO (if it is over a unit-ball) and the norm used to perform the backtracking procedure do *not* have to match. The backtracking procedure as in [44] always uses the Euclidean norm $\|\cdot\|_2$ for backtracking, even when the LMO is used over a general compact convex set $\mathcal{C}$.

If the sufficient decrease condition is satisfied, the algorithm accepts the step-size and decreases the estimate of the local Lipschitz constant for the next iteration, $L_{k+1} = \eta L_k$ where $\eta \in]0,1[$ (typically 0.9). If the sufficient decrease condition is not satisfied, the algorithm increases L_k by a factor $\beta > 1$ (typically 2) and recomputes the step-size, repeating until the condition holds.

While this approach successfully eliminates dependence on the global constant L and adapts to changes in the geometry, it suffers from slow adaptation when moving to regions of lower curvature. Since L_k decreases by only a factor of 0.9 per iteration, transitioning from a high-curvature region (where $L_k \approx 100$, say) to a flat region (where the local Lipschitz constant is 1) requires over 40 iterations. Figure 3 illustrates this slow decay on an example problem on video colocalization, showing how the estimate often oscillates between two regimes rather than quickly adapting to the local geometry.

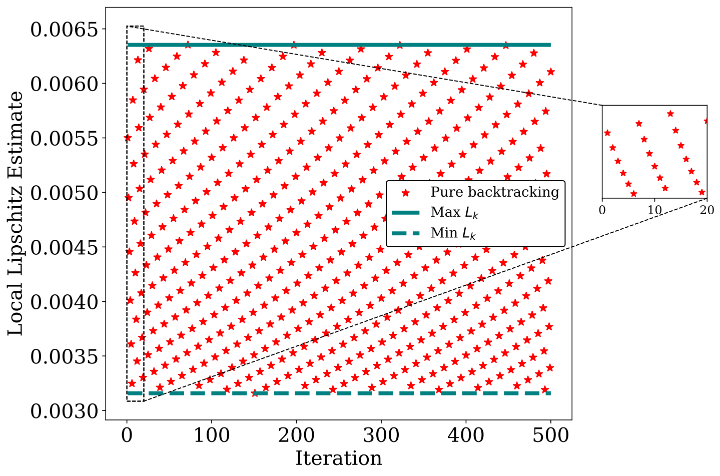


Fig. 3 The local Lipschitz constant found at each iteration with the pure backtracking procedure used in [44] is oscillating heavily between two values, roughly 0.0065 and 0.0030.

Local Lipschitz Estimation via Gradient Differences Malitsky and Mishchenko [37] proposed estimating the local Lipschitz constant directly from consecutive gradients by

$$L_k = \frac{\|\nabla f(x^k) - \nabla f(x^{k-1})\|_2}{\|x^k - x^{k-1}\|_2} + \delta,$$

where $\delta > 0$ is a small constant that we have added for numerical stability. Again, we use Euclidean norms for consistency with the backtracking framework. This estimate directly captures the local rate of gradient change, providing immediate adaptation to the local geometry rather than waiting for slow multiplicative decay. The global Lipschitz property ensures $L_k \le L + \delta$, and this estimate typically underestimates the global L in smooth regions, which is desirable for exploiting local flatness. This heuristic is also computationally efficient, requiring no additional gradient evaluations beyond those already computed, and completely eliminates the dependence on L. However, it does require one to store the previous iteration's gradient $\nabla f(x^{k-1})$ and iterate x^{k-1} in order to compute L_k.

3.1 Adaptive Step-Size Algorithm

Integrating the heuristic of [37] into GenCG is not so simple when $\mathcal{C}$ is not Euclidean. One could take the maximum between the current L_k at each iteration compared with the previous, so that the estimate becomes a tighter and tighter approximation of the constant L as the algorithm progresses, but this does not take advantage of the local curvature, which might be very low and amenable to large steps. Instead, we will use the heuristic as a warm-start for an optimistic candidate step-size, followed by a modification of the backtracking procedure of [44], if sufficient decrease is not met. We propose two adaptive step-size scaling strategies, called constant scaling and adjustable scaling, that leverage local smoothness combined with improved backtracking. The algorithm is outlined as AdaStep.

Algorithm 2 Adaptive Step-Size (AdaStep)

Input: $x^{k-1}, x^k \in \mathcal{X}$, direction d^k, $\delta > 0$, and some $\beta > 1$.
General Steps:

(Step 1) Estimate the local Lipschitz constant:

$$L_k \leftarrow \frac{\|\nabla f(x^k) - \nabla f(x^{k-1})\|_*}{\|x^k - x^{k-1}\|} + \delta.$$

(Step 2) Compute the initial step-size:

$$t_k \leftarrow \begin{cases} \min\left\{-\frac{\langle \nabla f(x^k), d^k\rangle}{L_k\|d^k\|^2}, 1\right\}, & \text{(Conditional Gradient)} \\ \frac{\|\nabla f(x^k)\|_{\mathrm{lmo}*}}{L_k\|d^k\|^2}. & \text{(Normalized Steepest Descent)} \end{cases}$$

(Step 3) While the sufficient decrease condition is not satisfied, i.e.,

$$f(x^k + t_k d^k) > f(x^k) + t_k\langle \nabla f(x^k), d^k\rangle + \frac{L_k}{2}t_k^2\|d^k\|^2,$$

update the Lipschitz estimate and step-size:

$$L_k \leftarrow \beta L_k \quad \text{and} \quad t_k \leftarrow \begin{cases} \min\left\{-\frac{\langle \nabla f(x^k), d^k\rangle}{L_k\|d^k\|^2}, 1\right\}, & \text{(Conditional Gradient)} \\ \frac{\|\nabla f(x^k)\|_{\mathrm{lmo}*}}{L_k\|d^k\|^2}. & \text{(Normalized Steepest Descent)} \end{cases}$$

Repeat until the condition is met.
Output: Step-size t_k.

Upon termination, the updated iterate $x^{k+1} = x^k + t_k d^k$ satisfies the local descent property:

$$f(x^{k+1}) \le f(x^k) + t_k\langle \nabla f(x^k), d^k\rangle + \frac{L_k}{2}t_k^2\|d^k\|^2. \tag{12}$$

This ensures decrease in either the norm of the gradient ∇f or the Frank-Wolfe gap without knowing the Lipschitz constant L of the gradient ∇f. It also allows for the step-size to increase if a larger step will provide more progress.

AdaStep will continue until the termination criterion is met, which differs depending on whether or not the problem is non-convex and whether or not the minimal functional-value $\inf_{x\in\mathcal{X}} f(x)$ is known. When f is convex (or quasar-convex) and $\inf_{x\in\mathcal{X}} f(x)$ is (approximately) known, we will use as a termination criterion the functional-value gap

$$f(x^k) - \inf_{x\in\mathcal{X}} f(x).$$

Meanwhile when f is non-convex or if the $\inf_{x\in\mathcal{X}} f(x)$ is not known, we will use the general LMO gap discussed in the beginning of Section 2,

$$\mathrm{Gap}(x^k) := -\langle \nabla f(x^k), d^k\rangle = \begin{cases} \langle \nabla f(x^k), x^k - v^k\rangle, & \text{(Conditional Gradient)} \\ \|\nabla f(x^k)\|_{\mathrm{lmo}\,*}, & \text{(Normalized Steepest Descent)} \end{cases}$$

as the termination criterion in GenCG. We recall that this quantity is relatively cheap to compute, since d^k and $\nabla f(x^k)$ are both already computed to run the algorithm.

3.1.1 Adaptive Step-Size with Constant Scaling

The constant scaling method enhances AdaStep by applying an initial scaling to L_k to promote larger step-sizes. After estimating L_k in step 1, we adjust by scaling:

$$L_k \leftarrow \gamma L_k,$$

where $\gamma \in]0, 1]$ is a fixed scaling factor. This reduction in L_k potentially increases t_k. The algorithm then proceeds to step 3, checking the sufficient decrease condition. If the condition is not met, L_k is iteratively increased by the backtracking factor $\beta \in]1, +\infty[$,

$$L_k \leftarrow \beta L_k,$$

and t_k is recomputed. This method, termed *adaptive constant scaling*, uses fixed parameters γ and β to balance aggressive step-size selection with robustness.

3.1.2 Adaptive Step-Size with Adjustable Scaling

The adjustable scaling method builds on the constant scaling approach by dynamically updating the scaling factor γ to adapt to the function's local curvature. After estimating L_k and applying the initial scaling $L_k \leftarrow \gamma L_k$, the algorithm proceeds as in the constant scaling method. However, γ is periodically updated, e.g., every r iterations, based on the total number of backtracking steps in the previous period:

– **Conservative**: If no backtracking occurs, indicating a conservative L_k, reduce γ:

$$\gamma \leftarrow 0.9\gamma.$$

– **Aggressive**: If backtracking exceeds r iterations, indicating an aggressive L_k, increase γ:

$$\gamma \leftarrow 1.1\gamma.$$

– **Balanced**: If backtracking occurs but totals r or fewer steps, γ remains unchanged, as the algorithm is well-calibrated.

This *adaptive adjustable scaling* method enhances flexibility by adjusting γ to match the function's smoothness, potentially accelerating convergence in problems with varying curvature. Both scaling methods are computationally efficient, requiring only scalar updates. In our experiments, we used consistent initial parameters for both methods: $r = 10$, $\gamma = 1/4$, $\delta = 10^{-10}$, and $\beta = 2$.

4 Algorithm and Convergence Analysis

We present here the *Adaptive Conditional Gradient Descent* algorithm (ACGD) which is simply the template from GenCG with our adaptive step-size strategy AdaStep. Afterwards, we give some remarks followed by its convergence analysis under a variety of assumptions on the regularity of f, ranging from non-convex to strongly convex.

Algorithm 3 Adaptive Conditional Gradient Descent (ACGD)

```
Input: Compact convex set $\mathcal{C}$ (typically a closed unit-ball $\mathcal{B}_{\text{lmo}}(1)$), feasible set $\mathcal{X}$ (either $\mathbb{R}^n$ or $\mathcal{C}$), parameters $\gamma \in ]0,1]$, $\delta > 0$, $t_{\max} > 0$ and $\beta > 1$.
Initialization: Set $k = 0$ and choose $x^{k-1}, x^k \in \mathcal{X}$.
General Steps:
    (Step 1) Compute
        $v^k \leftarrow \underset{\mathcal{C}}{\text{lmo}}(\nabla f(x^k)).$
    (Step 2) Set descent direction
        $d^k \leftarrow \begin{cases} v^k, & \text{if } \mathcal{X} = \mathbb{R}^n \\ v^k - x^k. & \text{if } \mathcal{X} = \mathcal{C} \end{cases}$
    (Step 3) Estimate the local Lipschitz constant:
        $L_k \leftarrow \frac{\|\nabla f(x^k) - \nabla f(x^{k-1})\|_*}{\|x^k - x^{k-1}\|} + \delta.$
    (Step 4) Scale local Lipschitz
        $L_k \leftarrow \gamma L_k.$
    (Step 5) Compute the initial step-size:
        $t_k \leftarrow \min\left\{-\frac{\langle \nabla f(x^k), d^k\rangle}{L_k\|d^k\|^2}, t_{\max}\right\}.$
    (Step 6) While $f(x^k + t_k d^k) > f(x^k) + t_k\langle \nabla f(x^k), d^k\rangle + \frac{L_k}{2}t_k^2\|d^k\|^2$, do:
        $L_k \leftarrow \beta L_k \quad \text{and} \quad t_k \leftarrow \min\left\{-\frac{\langle \nabla f(x^k), d^k\rangle}{L_k\|d^k\|^2}, t_{\max}\right\}.$
    (Step 7) Update iterate
        $x^{k+1} \leftarrow x^k + t_k d^k \quad \text{and} \quad k \leftarrow k+1.$
Stop: Terminate when a convergence criterion is met.
```

4.1 Algorithm

The backtracking procedure used terminates in a finite number of iterations because the estimated local Lipschitz constant L_k is bounded above by $L+\delta$, where L is the global Lipschitz constant of ∇f. Moreover, the exponential growth of L_k (via $\beta > 1$) ensures that L_k will eventually exceed $L + \delta$, guaranteeing termination.

Remark 1 In the unconstrained case ($\mathcal{X} = \mathbb{R}^n$), the norm $\|\cdot\|_{\text{lmo}}$ used in Steps 3 to 6 does *not* have to be the same norm used to define the closed unit-ball $\mathcal{B}_{\text{lmo}}(1)$ taken for $\mathcal{C}$. This means it is possible to compute the normalized steepest descent direction with respect to an arbitrary norm $\|\cdot\|_{\text{lmo}}$ (Step 1) in combination with backtracking in a *different* norm $\|\cdot\|$. As we will explain the numerical section, it turns out that doing the backtracking procedure in the ℓ^2 norm works well even when $\mathcal{C}$ is not the ℓ^2 closed unit-ball, although in the theoretical section we will also show that the convergence guarantee is for $\|\nabla f(x^k)\|_{\text{lmo}\,*}$. □

After termination of Step 6 in ACGD, the updated iterate satisfies the following local descent property

$$f(x^k + t_k d^k) \le f(x^k) + t_k\langle \nabla f(x^k), d^k\rangle + \frac{L_k}{2}t_k^2\|d^k\|^2, \tag{13}$$

which is the key inequality that we will use to prove the convergence rates.

Remark 2 In our adaptive step-size method, we define the update direction as $v^k = \text{lmo}_{\mathcal{B}_{\text{lmo}}(1)}(\nabla f(x^k))$, where the LMO is taken over the closed unit-ball $\mathcal{B}_{\text{lmo}}(1)$ induced by some norm that we denote $\|\cdot\|_{\text{lmo}}$. Formally, $\text{lmo}_{\mathcal{C}}(v) = \text{argmin}_{\|s\|_{\text{lmo}}\le 1}\langle s, v\rangle$, which guarantees $\|v^k\|_{\text{lmo}} = 1$. To estimate the local Lipschitz constant L_k efficiently, we could use the same norm in both the lmo computation and the backtracking, i.e., $\|\cdot\|_{\text{lmo}} = \|\cdot\|$. In that case, the calculation for L_k simplifies as follows

$$L_k = \frac{\|\nabla f(x^k) - \nabla f(x^{k-1})\|_*}{\|x^k - x^{k-1}\|} + \delta = \frac{\|\nabla f(x^k) - \nabla f(x^{k-1})\|_*}{t_{k-1}} + \delta,$$

because $\|x^k - x^{k-1}\| = \|t_{k-1}v^{k-1}\| = t_{k-1}$ since, under this assumption, $\|v^{k-1}\| = \|v^{k-1}\|_{\text{lmo}} = 1$. This formulation reduces computational cost by leveraging the fact that the output of the LMO has a fixed norm, so the quantity $\|x^k - x^{k-1}\|$ can be known in advance, eliminating the need to compute distances between iterates explicitly and thus avoiding having to store x^{k-1}. However, we will see that it is eventually useful to allow the norms used to calculate L_k and the norm used to calculate the LMO to be *decoupled*, since estimating the local Lipschitz constant using ℓ^2 norms performs well in practice with directions coming from non-Euclidean LMOs. □

Corollary 1 *Let $k \in \mathbb{N}$ and consider the step-size t_k, iterate x_k, and direction d^k generated by ACGD. Then,*

$$t_k \langle \nabla f(x^k), d^k \rangle + \frac{L_k}{2} t_k^2 \|d^k\|^2 \leq 0.$$

Proof For $t_k = 0$ the claim obviously holds so consider $t_k > 0$.

Starting from the inequality: For all $k \in \mathbb{N}$, by the construction of the step-size t_k, it holds

$$0 < t_k \leq -\frac{\langle \nabla f(x^k), d^k \rangle}{L_k \|d^k\|^2} \leq -2\frac{\langle \nabla f(x^k), d^k \rangle}{L_k \|d^k\|^2}.$$

Multiplying both sides by $\frac{L_k\|d^k\|^2}{2}$ and rearranging, we find

$$\langle \nabla f(x^k), d^k \rangle + \frac{L_k}{2} t_k \|d^k\|^2 \leq 0,$$

which, after multiplying by $t_k \geq 0$, gives the desired claim. □

Corollary 2 *Let $k \in \mathbb{N}$ and consider consecutive iterates x^k and x^{k+1} generated by ACGD. Then, $f(x^{k+1}) \leq f(x^k)$, i.e., the functional-values are non-increasing.*

Proof From Corollary 1, we know that:

$$t_k \langle \nabla f(x^k), d^k \rangle + \frac{L_k}{2} t_k^2 \|d^k\|^2 \leq 0.$$

Thus, it follows that $f(x^k + t_k d^k) \leq f(x^k)$. This ensures that the functional-values are non-increasing, guaranteeing descent at each iteration. □

Remark 3 The main difference between our adaptive step-size method and the step-size method proposed in [44] lies in the handling of the Lipschitz constant estimate. Our method computes L_k directly from gradient differences at each iteration and then modifies it if needed, allowing it to adapt more quickly to the local curvature of a region. In contrast, the step-size method proposed in [44] initializes L_{-1} heuristically as

$$\frac{\|\nabla f(x^0) - \nabla f(x^0 + \epsilon d^0)\|_2}{\epsilon \|d^0\|_2},$$

where ϵ is a small constant, and updates it iteratively by increasing it during backtracking or decreasing it when the sufficient decrease condition is satisfied. This approach avoids recomputing the Lipschitz estimate from scratch but may result in conservative step-sizes, particularly in iterations where the sufficient decrease condition does not require adjustment through the backtracking loop. Note also that, technically speaking, there is no guarantee that this heuristic initialization for L_{-1} is non-zero. □

4.2 Unconstrained Problems ($\mathcal{X} = \mathbb{R}^n$)

In the subsection, we will always assume that $\mathcal{X} = \mathbb{R}^n$, so that (P) is unconstrained. We start with a lemma that characterizes how much the functional-values must decrease across iterations.

Lemma 1 *Let $k \in \mathbb{N}$ and consider the step-size t_k, iterate x^k, and direction d^k generated by ACGD. Then,*

$$f(x^{k+1}) \leq f(x^k) - \frac{1}{2L_k\|d^k\|^2} \|\nabla f(x^k)\|^2_{\text{lmo}*}.$$

Proof By the local descent property of ACGD given in (13) and noting that $d^k = \text{lmo}_{\mathcal{C}}(\nabla f(x^k))$ in the unconstrained case, we get

$$f(x^{k+1}) \leq f(x^k) - t_k \|\nabla f(x^k)\|_{\text{lmo}\,*} + \frac{L_k}{2} t_k^2 \|d^k\|^2.$$

By definition of t_k in ACGD, $t_k = \frac{\|\nabla f(x^k)\|_{\text{lmo}\,*}}{L_k \|d^k\|^2}$, which leads to

$$\begin{aligned} f(x^{k+1}) &\leq f(x^k) - \left(\frac{\|\nabla f(x^k)\|_{\text{lmo}\,*}}{L_k \|d^k\|^2} \right) \|\nabla f(x^k)\|_{\text{lmo}\,*} + \frac{L_k}{2} \left(\frac{\|\nabla f(x^k)\|_{\text{lmo}\,*}}{L_k \|d^k\|^2} \right)^2 \|d^k\|^2 \\ &= f(x^k) - \frac{1}{2L_k \|d^k\|^2} \|\nabla f(x^k)\|_{\text{lmo}\,*}^2. \end{aligned}$$

□

The following theorem establishes the convergence rate (to a critical point) for ACGD in the unconstrained setting under the assumption that f is bounded from below by $\inf_{x \in \mathbb{R}^n} f(x) > -\infty$.

Theorem 1 (Unconstrained Non-convex Convergence Rate) *Let $\mathcal{C} = \mathcal{B}_{\text{lmo}}(1)$ be the closed unit-ball for some norm $\|\cdot\|_{\text{lmo}}$ and let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function, bounded from below by $f^\star > -\infty$, with a gradient that is Lipschitz-continuous in the norm $\|\cdot\|_*$ with constant $L > 0$. Let $\{x^k\}_{k \in \mathbb{N}}$ be the sequence of iterates generated by ACGD. Then, for any $N \in \mathbb{N}^*$,*

$$\min_{0 \leq k \leq N} \|\nabla f(x^k)\|_{\text{lmo}\,*}^2 \leq \frac{2\beta(L+\delta)\zeta^2(f(x^0) - f^\star)}{N+1},$$

where $\zeta \in]0, +\infty[$ is the norm-equivalence constant such that, for all $x \in \mathbb{R}^n$, $\|x\| \leq \zeta \|x\|_{\text{lmo}}$.

Proof Using the local descent property given by Lemma 1 for f, it holds for all $k \in \mathbb{N}$,

$$f(x^{k+1}) \leq f(x^k) - \frac{1}{2L_k \|d^k\|^2} \|\nabla f(x^k)\|_{\text{lmo}\,*}^2. \tag{14}$$

Letting $N \in \mathbb{N}^*$ and summing from $k = 0$ to N yields

$$\begin{aligned} \sum_{k=0}^{N} \|\nabla f(x^k)\|_{\text{lmo}\,*}^2 &\leq \sum_{k=0}^{N} 2L_k \|d^k\|^2 \left(f(x^k) - f(x^{k+1}) \right) \\ &\leq 2\beta(L+\delta)\zeta^2 \sum_{k=0}^{N} \left(f(x^k) - f(x^{k+1}) \right) \\ &\leq 2\beta(L+\delta)\zeta^2 \left(f(x^0) - f^\star \right), \end{aligned}$$

where $\zeta \in]0, +\infty[$ is the norm-equivalence constant such that, for all $x \in \mathbb{R}^n$, $\|x\| \leq \zeta \|x\|_{\text{lmo}}$. Lower bounding by taking the minimum over $k \in \{0, \ldots, N\}$ finally gives

$$\min_{0 \leq k \leq N} \|\nabla f(x^k)\|_{\text{lmo}\,*}^2 \leq \frac{2\beta(L+\delta)\zeta^2(f(x^0) - f^\star)}{N+1}.$$

□

Remark 4 The previous theorem gives a convergence guarantee in terms of $\|\nabla f(x^k)\|_{\text{lmo}\,*}$ where the norm $\|\cdot\|_{\text{lmo}\,*}$ is the dual norm to $\|\cdot\|_{\text{lmo}}$, the one used to compute v^k via the LMO. This norm does *not* have to be the same norm that is used in the backtracking procedure. In practice, we find that using the ℓ^2 for the backtracking procedure works well even when the LMO is computed with respect to a different norm (e.g., ℓ^1 or ℓ^∞). □

The next theorem guarantees a stronger convergence rate (to the optimal functional-value) for ACGD when minimizing a quasar-convex function. It requires the additional hypothesis that the sub-level sets of f are bounded. Note that a sub-level set for a continuous function is a closed set, so that the boundedness assumption ensures compactness of sub-level sets of f.

Theorem 2 (Unconstrained Quasar-convex Convergence Rate) *Let $\mathcal{C} = \mathcal{B}_{\mathrm{lmo}}(1)$ be a closed unit-ball for some norm $\|\cdot\|_{\mathrm{lmo}}$ and let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function with a gradient that is Lipschitz-continuous in the norm $\|\cdot\|_*$ with constant $L > 0$. Assume also that f is η-quasar-convex and that the level set $\mathcal{S} := \{x \in \mathbb{R}^n : f(x) \le f(x^0)\}$ associated with the initialization x^0 is bounded and let $\{x^k\}_{k\in\mathbb{N}}$ the sequence of iterates generated by ACGD. Then, for any $N \in \mathbb{N}^*$,*

$$f(x^N) - f^\star \le \frac{2\beta(L+\delta)R^2}{\left(\frac{2\beta(L+\delta)R^2}{f(x^0)-f^\star}\right) + N\eta^2}. \tag{15}$$

Proof First we note that, for any $N \in \mathbb{N}^*$, if $f(x^N) = f^\star$ then inequality (15) clearly holds. So, we assume that $f(x^N) \neq f^\star$, which means that, for $k \in \{0,\dots,N\}$, we have $h_k > 0$.

From the descent property in Lemma 1 and the definition of the step-size t_k, it holds for all $k \in \{0,\dots,N-1\}$,

$$f(x^{k+1}) \le f(x^k) - \frac{\|\nabla f(x^k)\|^2_{\mathrm{lmo}*}}{2L_k\|d^k\|^2}.$$

Define $h_k := f(x^k) - f^\star \ge 0$; subtracting $f^\star$ from both sides of the above inequality gives

$$h_{k+1} \le h_k - \frac{\|\nabla f(x^k)\|^2_{\mathrm{lmo}*}}{2L_k\|d^k\|^2}.$$

Since each $x^k \in \mathcal{S}$ and $\mathcal{S}$ is bounded by assumption, define

$$R := \max\left\{\max_{x\in\mathcal{S}} \|x - x^\star\|_{\mathrm{lmo}}, 1\right\} < \infty.$$

By η-quasar-convexity of f, it follows that

$$h_k \le \frac{1}{\eta}\langle \nabla f(x^k), x^k - x^\star\rangle \le \frac{1}{\eta}\|\nabla f(x^k)\|_{\mathrm{lmo}*}\|x^k - x^\star\|_{\mathrm{lmo}} \Rightarrow \|\nabla f(x^k)\|_{\mathrm{lmo}*} \ge \frac{h_k}{\|x^k - x^\star\|_{\mathrm{lmo}}} \ge h_k\frac{\eta}{R},$$

since $x^k \in \mathcal{S}$, i.e., $\left\|x^k - x^\star\right\|_{\mathrm{lmo}} \le R$ for all $k \in \mathbb{N}$. It therefore follows that

$$h_{k+1} \le h_k - \frac{\|\nabla f(x^k)\|^2_{\mathrm{lmo}*}}{2L_k\|d^k\|^2} \le h_k - \frac{1}{2L_k\|d^k\|^2}\left(h_k\frac{\eta}{R}\right)^2 \le h_k - \frac{h_k^2\eta^2}{2\beta(L+\delta)\zeta^2R^2},$$

since $L_k \le \beta(L+\delta)$ and $\zeta \in]0,+\infty[$ is the norm-equivalence constant such that, for all $x \in \mathbb{R}^n$, $\|x\| \le \zeta\|x\|_{\mathrm{lmo}}$. Manipulating the above we find

$$\frac{h_{k+1}}{h_{k+1}h_k} \le \frac{h_k - \frac{h_k^2\eta^2}{2\beta(L+\delta)\zeta^2R^2}}{h_{k+1}h_k} \implies \frac{1}{h_k} \le \frac{1}{h_{k+1}} - \frac{h_k\eta^2}{2\beta(L+\delta)\zeta^2R^2h_{k+1}}.$$

Since $h_{k+1} \le h_k$, we know that $\frac{h_k}{h_{k+1}} \ge 1$ and thus

$$-\frac{h_k\eta^2}{2\beta(L+\delta)\zeta^2R^2h_{k+1}} \le -\frac{\eta^2}{2\beta(L+\delta)\zeta^2R^2}.$$

Substituting this into our earlier equation, we find that

$$\frac{1}{h_k} \le \frac{1}{h_{k+1}} - \frac{\eta^2}{2\beta(L+\delta)\zeta^2R^2},$$

which we rewrite as

$$\frac{1}{h_{k+1}} \ge \frac{1}{h_k} + \frac{\eta^2}{2\beta(L+\delta)\zeta^2R^2}.$$

Summing this telescoping inequality from $k = 0$ to $N-1$,

$$\frac{1}{h_N} \ge \frac{1}{h_0} + \frac{N\eta^2}{2\beta(L+\delta)\zeta^2R^2}.$$

Inverting and simplifying gives

$$h_N \le \frac{2\beta(L+\delta)\zeta^2R^2}{\left(\frac{2\beta(L+\delta)\zeta^2R^2}{h_0}\right) + N\eta^2},$$

which completes the proof. □

Theorem 3 (Unconstrained Strongly Convex Convergence Rate) *Let $\mathcal{C} = \mathcal{B}_{\mathrm{lmo}}(1)$ be a closed ball of radius $\rho > 0$ for some norm $\|\cdot\|_{\mathrm{lmo}}$ and let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable with a gradient that is Lipschitz-continuous in the norm $\|\cdot\|_*$ with constant $L > 0$. Assume also that f is μ-strongly convex with respect to $\|\cdot\|_{\mathrm{lmo}}$ and let $\{x^k\}_{k\in\mathbb{N}}$ be the sequence of iterates generated by ACGD. Then, for any $N \in \mathbb{N}^*$,*

$$f(x^N) - f^\star \leq \left(1 - \frac{\mu}{\beta(L+\delta)\zeta^2}\right)^N (f(x^0) - f^\star), \tag{16}$$

where $\zeta \in]0, +\infty[$ is the norm-equivalence constant such that, for all $x \in \mathbb{R}^n$, $\|x\| \leq \zeta \|x\|_{\mathrm{lmo}}$.

Proof From the local descent property established in Lemma 1 for f with the definition of the step-size t_k, we have, for all $k \in \mathbb{N}$,

$$f(x^{k+1}) \leq f(x^k) - \frac{\|\nabla f(x^k)\|^2_{\mathrm{lmo}*}}{2L_k\|d^k\|^2}. \tag{17}$$

By the assumed strong convexity of f with constant $\mu > 0$, we have the Polyak-Łojasiewicz inequality (or more precisely the Kurdyka-Łojasiewicz inequality with exponent $1/2$)

$$\|\nabla f(x^k)\|^2_{\mathrm{lmo}*} \geq 2\mu(f(x^k) - f^\star).$$

Applying the above inequality to (17), we find

$$f(x^{k+1}) \leq f(x^k) - \frac{\mu(f(x^k) - f^\star)}{L_k\|d^k\|^2}.$$

Subtracting $f^\star$ from both sides and letting $h_k := f(x^k) - f^\star \geq 0$ we can rewrite to get

$$h_{k+1} \leq h_k - \frac{\mu h_k}{L_k\|d^k\|^2} = \left(1 - \frac{\mu}{L_k\|d^k\|^2}\right) h_k.$$

From the backtracking procedure (c.f., AdaStep), we know that $L_k \leq \beta(L + \delta)$ upon termination, i.e.,

$$1 - \frac{\mu}{L_k\|d^k\|^2} \geq 1 - \frac{\mu}{\beta(L+\delta)\zeta^2},$$

where $\zeta \in]0, +\infty[$ is the norm-equivalence constant such that, for all $x \in \mathbb{R}^n$, $\|x\| \leq \zeta\|x\|_{\mathrm{lmo}}$. Thus,

$$h_{k+1} \leq \left(1 - \frac{\mu}{\beta(L+\delta)\zeta^2}\right) h_k.$$

Applying this recursively from $k = 0$ to $N - 1$ yields

$$h_N \leq \left(1 - \frac{\mu}{\beta(L+\delta)\zeta^2}\right)^N h_0,$$

which completes the proof. □

4.3 Constrained Problems ($\mathcal{X} = \mathcal{C}$)

In this subsection, we will always assume that $\mathcal{X} = \mathcal{C}$ is some non-empty compact convex set. This makes (P) a constrained problem that can be solved using the Conditional Gradient algorithm with our adaptive step-size, i.e., ACGD.

The next theorem establishes a convergence rate for the Frank-Wolfe gaps of ACGD when minimizing a smooth function over a non-empty compact convex set to a stationary point in the general non-convex setting. The rate is given on $\mathrm{Gap}(x^k)$, which we recall is the Frank-Wolfe gap when $\mathcal{X} = \mathcal{C}$ and which, when bounded above by ϵ, certifies ϵ-criticality of x^k for (P). We make use of the fact that $\mathcal{C}$ is compact and thus admits a finite diameter $D = \max_{x,y\in\mathcal{C}} \|x - y\| < +\infty$.

Theorem 4 (Constrained Non-convex Convergence Rate) *Let $\mathcal{C} \subset \mathbb{R}^n$ be a non-empty compact convex set and let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable with $\{x^k\}_{k\in\mathbb{N}}$ the sequence of iterates generated by ACGD. Then, for any $N \in \mathbb{N}^*$,*

$$\min_{0\le k\le N} \operatorname{Gap}(x^k) \le \sqrt{\frac{2C(f(x^0)-f^\star)}{N+1}},$$

where $C := \left(\max\{\beta(L+\delta), 1, \max_{x\in\mathcal{C}} \|\nabla f(x)\|\} \cdot (\max\{D,1\})^2\right)$.

Proof Using the local descent property for f given by (13) we have, for all $k \in \mathbb{N}$,

$$f(x^{k+1}) \le f(x^k) + t_k\langle \nabla f(x^k), d^k\rangle + \frac{L_k}{2}t_k^2\|d^k\|^2 = f(x^k) - t_k\operatorname{Gap}(x^k) + \frac{L_k}{2}t_k^2\|d^k\|^2$$

which can be rearranged to get a bound on the gap,

$$t_k\operatorname{Gap}(x^k) \le f(x^k) - f(x^{k+1}) + \frac{L_k}{2}t_k^2\|d^k\|^2 \tag{18}$$

By design, we have $t_k = \min\left\{-\frac{\langle\nabla f(x^k), d^k\rangle}{L_k\|d^k\|^2}, 1\right\} = \min\left\{\frac{\operatorname{Gap}(x^k)}{L_k\|d^k\|^2}, 1\right\}$, which gives two cases that we examine separately.

Case one: $t_k = \frac{\operatorname{Gap}(x^k)}{L_k\|d^k\|^2} \le 1$. In this case we find Starting with (18) and using the definition of t_k we find

$$\frac{(\operatorname{Gap}(x^k))^2}{L_k\|d^k\|^2} \le f(x^k) - f(x^{k+1}) + \frac{(\operatorname{Gap}(x^k))^2}{2L_k\|d^k\|^2}$$

which is equivalent to

$$\frac{(\operatorname{Gap}(x^k))^2}{2L_k\|d^k\|^2} \le f(x^k) - f(x^{k+1}).$$

Case two: $t_k = 1$, $\frac{\operatorname{Gap}(x^k)}{L_k\|d^k\|^2} \ge 1$. In this case, starting from (18) and using $t_k = 1$ we find

$$\operatorname{Gap}(x^k) \le f(x^k) - f(x^{k+1}) + \frac{L_k}{2}\|d^k\|^2 \le f(x^k) - f(x^{k+1}) + \frac{\operatorname{Gap}(x^k)}{2}.$$

Rearranging this yields

$$\frac{\operatorname{Gap}(x^k)}{2} \le f(x^k) - f(x^{k+1}).$$

Combining both cases we have, for all $k \in \mathbb{N}$,

$$\min\left\{\frac{\operatorname{Gap}(x^k)}{2}, \frac{(\operatorname{Gap}(x^k))^2}{2L_k\|d^k\|^2}\right\} \le f(x^k) - f(x^{k+1}).$$

Set $C := \left(\max\{\beta(L+\delta), 1, \max_{x\in\mathcal{C}} \|\nabla f(x)\|\} \cdot (\max\{D,1\})^2\right) > 0$, then

$$\min\left\{\frac{\operatorname{Gap}(x^k)}{2}, \frac{(\operatorname{Gap}(x^k))^2}{2C}\right\} \le f(x^k) - f(x^{k+1}).$$

Now, we claim that $\min\left\{\frac{\operatorname{Gap}(x^k)}{2}, \frac{(\operatorname{Gap}(x^k))^2}{2C}\right\} = \frac{(\operatorname{Gap}(x^k))^2}{2C}$, i.e., that $\frac{\operatorname{Gap}(x^k)}{2} \ge \frac{(\operatorname{Gap}(x^k))^2}{2C}$ since

$$\frac{\operatorname{Gap}(x^k)}{2} - \frac{(\operatorname{Gap}(x^k))^2}{2C} = \underbrace{\frac{\operatorname{Gap}(x^k)}{2C}}_{\ge 0}\underbrace{(C - \operatorname{Gap}(x^k))}_{\ge 0} \ge 0,$$

and

$$0 \le \operatorname{Gap}(x^k) = -\langle\nabla f(x^k), d^k\rangle \le \|\nabla f(x^k)\|\|d^k\| < C.$$

Thus, it holds for all $k \in \mathbb{N}$ that

$$\frac{(\operatorname{Gap}(x^k))^2}{2C} \le f(x^k) - f(x^{k+1}).$$

Letting $N \in \mathbb{N}^*$ and summing from $k = 0$ to N we find

$$\sum_{k=0}^{N} \frac{(\mathrm{Gap}(x^k))^2}{2C} \leq f(x^0) - f(x^{N+1}) \leq f(x^0) - f^\star.$$

Finally,

$$\begin{aligned}
&(N+1)\frac{\left(\min\limits_{0\leq k\leq N} \mathrm{Gap}(x^k)\right)^2}{2C} \leq \sum_{k=0}^{N} \frac{(\mathrm{Gap}(x^k))^2}{2C} \leq f(x^0) - f^\star \\
&\Rightarrow \left(\min_{0\leq k\leq N} \mathrm{Gap}(x^k)\right)^2 \leq \frac{2Ch_0}{N+1} \\
&\Rightarrow \min_{0\leq k\leq N} \mathrm{Gap}(x^k) \leq \sqrt{\frac{2C(f(x^0) - f^\star)}{N+1}},
\end{aligned}$$

which is the desired claim. □

We now examine the convergence rate of the functional-value gap when ACGD is employed for constrained problems with η-quasar-convex objective functions.

Theorem 5 (Constrained Quasar-convex Convergence Rate) *Let $\mathcal{C} \subset \mathbb{R}^n$ be a non-empty compact convex set and let $f\colon \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable and η-quasar-convex with $\{x^k\}_{k\in\mathbb{N}}$ the sequence of iterates generated by ACGD. Then, for any $N \in \mathbb{N}^*$,*

$$f(x^N) - f^\star \leq \frac{2C}{N\eta^2 + (\frac{2C}{f(x^0)-f^\star})},$$

where $C := \max\left\{\beta(L+\delta), \max\limits_{x\in\mathcal{C}} \|\nabla f(x)\|, 1\right\} \cdot (\max\{D, 1\})^2.$

Proof Using the local descent property of f from (13),

$$f(x^{k+1}) \leq f(x^k) + t_k\langle\nabla f(x^k), d^k\rangle + \frac{L_k}{2}t_k^2\|d^k\|^2. \tag{19}$$

Recall that the step-size is defined in this case to be $t_k = \min\left\{-\frac{\langle\nabla f(x^k),d^k\rangle}{L_k\|d^k\|^2}, 1\right\}$, which gives two cases.

Case one: $t_k = -\frac{\langle\nabla f(x^k),d^k\rangle}{L_k\|d^k\|^2}$. Substituting t_k into (19), we have:

$$\begin{aligned}
f(x^{k+1}) &\leq f(x^k) + \left(-\frac{\langle\nabla f(x^k), d^k\rangle}{L_k\|d^k\|^2}\right)\langle\nabla f(x^k), d^k\rangle + \frac{L_k}{2}\left(-\frac{\langle\nabla f(x^k), d^k\rangle}{L_k\|d^k\|^2}\right)^2\|d^k\|^2 \\
&= f(x^k) - \frac{\langle\nabla f(x^k), d^k\rangle^2}{L_k\|d^k\|^2} + \frac{\langle\nabla f(x^k), d^k\rangle^2}{2L_k\|d^k\|^2} \\
&= f(x^k) - \frac{\langle\nabla f(x^k), d^k\rangle^2}{2L_k\|d^k\|^2}.
\end{aligned}$$

By η-quasar-convexity of f, we have

$$f(x^\star) \geq f(x^k) + \frac{1}{\eta}\langle\nabla f(x^k), x^\star - x^k\rangle \geq f(x^k) + \frac{1}{\eta}\langle\nabla f(x^k), d^k\rangle,$$

where $x^\star \in \operatorname*{argmin}_{x\in\mathcal{C}} f(x)$ and $\eta > 0$. Thus, denoting $h_k := f(x^k) - f^\star$, it follows $\langle\nabla f(x^k), d^k\rangle \leq -\eta h_k \leq 0$. Squaring this and multiplying by -1 gives

$$-\langle\nabla f(x^k), d^k\rangle^2 \leq -\eta^2 h_k^2.$$

Therefore, using (19) and since $\|d^k\| \leq D := \max_{x,y\in\mathcal{C}}\|x - y\|$, it follows that

$$h_{k+1} \leq h_k - \frac{\eta^2 h_k^2}{2L_k D^2} \leq h_k - \frac{\eta^2 h_k^2}{2C},$$

where $C := \big(\max\{\beta(L+\delta), \max_{x\in\mathcal{C}}\|\nabla f(x)\|, 1\} \cdot (\max\{D,1\})^2\big) > 0$.

Case two: $t_k = 1$, so $-\frac{\langle \nabla f(x^k), d^k\rangle}{L_k\|d^k\|^2} \geq 1$. Substituting $t_k = 1$ into (19), we have

$$\begin{aligned} f(x^{k+1}) &\leq f(x^k) + \langle \nabla f(x^k), d^k\rangle + \frac{L_k}{2}\|d^k\|^2 \\ &\leq f(x^k) + \langle \nabla f(x^k), d^k\rangle - \frac{1}{2}\langle \nabla f(x^k), d^k\rangle \\ &= f(x^k) + \frac{1}{2}\langle \nabla f(x^k), d^k\rangle \\ &\leq f(x^k) - \frac{\eta h_k}{2}, \end{aligned}$$

so that, finally,

$$h_{k+1} \leq h_k - \frac{\eta h_k}{2} = \frac{(1-\eta)h_k}{2}.$$

In both cases, we obtain

$$h_{k+1} \leq \max\left\{h_k - \frac{\eta^2 h_k^2}{2C}, \frac{(1-\eta)h_k}{2}\right\}.$$

Also,

$$\eta h_k \leq -\langle \nabla f(x^k), d^k\rangle \leq \|\nabla f(x^k)\|\|d^k\| \leq C < 2C.$$

Now we show that $\max\left\{h_k - \frac{\eta^2h_k^2}{2C}, \frac{(1-\eta)h_k}{2}\right\} = h_k - \frac{\eta^2h_k^2}{2C}$, i.e., $h_k - \frac{\eta^2h_k^2}{2C} \geq \frac{(1-\eta)h_k}{2}$:

$$h_k - \frac{\eta^2h_k^2}{2C} - \frac{(1-\eta)h_k}{2} = \frac{(1+\eta)h_k}{2} - \frac{\eta^2h_k^2}{2C} = \underbrace{\frac{h_k}{2C}}_{\geq 0}\underbrace{(C(1+\eta) - \eta^2 h_k)}_{\geq 0} \geq 0.$$

Therefore

$$h_{k+1} \leq h_k - \frac{\eta^2h_k^2}{2C}.$$

Since $h_k \geq 0$ for all $k \in \mathbb{N}$, the sequence $\{h_k\}_{k\in\mathbb{N}}$ is non-increasing and bounded below by 0. Thus, $\{h_k\}_{k\in\mathbb{N}}$ converges to some limit $h^\star \geq 0$. Taking the limit as $k \to \infty$ of the inequality above gives

$$h^\star \leq h^\star - \frac{\eta^2(h^\star)^2}{2C},$$

which implies that $\frac{(h^\star)^2}{2C} \leq 0$. Since $C > 0$, this can only happen if $h^\star = 0$. Therefore, $\lim_{k\to\infty} h_k = 0$. Furthermore, for all $k \in \mathbb{N}$,

$$\frac{h_{k+1}}{h_{k+1}h_k} \leq \frac{h_k - \frac{\eta^2h_k^2}{2C}}{h_{k+1}h_k} \implies \frac{1}{h_k} \leq \frac{1}{h_{k+1}} - \frac{\eta^2 h_k}{2Ch_{k+1}}.$$

Since $h_{k+1} \leq h_k$, we know that $\frac{h_k}{h_{k+1}} \geq 1$. Therefore,

$$-\frac{\eta^2h_k}{2Ch_{k+1}} \leq -\frac{\eta^2}{2C}.$$

Substituting this into our earlier equation, we get

$$\frac{1}{h_k} \leq \frac{1}{h_{k+1}} - \frac{\eta^2}{2C}.$$

Adding $\frac{\eta^2}{2C}$ to both sides, we get:

$$\frac{1}{h_{k+1}} \geq \frac{1}{h_k} + \frac{\eta^2}{2C}.$$

We can apply this repeatedly and unroll the recurrence to find

$$\frac{1}{h_{k+1}} \geq \frac{1}{h_0} + \frac{(k+1)\eta^2}{2C}.$$

To get h_{k+1} back, we take the reciprocal of both sides of the inequality and arrive at

$$h_{k+1} \leq \frac{1}{\frac{1}{h_0} + \frac{(k+1)\eta^2}{2C}} = \frac{1}{\frac{2C+(k+1)\eta^2h_0}{2Ch_0}} = \frac{2Ch_0}{2C+(k+1)\eta^2h_0} = \frac{2C}{\frac{2C}{h_0} + (k+1)\eta^2}.$$

□

The above theorem applies also to convex objective functions f and establishes that the sequence of function values converges to the optimal value at a sub-linear rate. As with all of our convergence guarantees, while the global Lipschitz gradient condition is employed in the analysis, from a computational perspective, it is not necessary to know the value of the global Lipschitz constant for the gradient.

5 Numerical Experiments

We performed extensive numerical experiments to assess the efficacy of ACGD, benchmarking its performance against some other step-size strategies for both constrained and unconstrained optimization problems. All experiments were conducted using Julia [3] on the Google Colaboratory cloud-based computational platform (Google Colab) [4].

In what follows, we present these numerical results for ACGD. To ensure a fair comparison, all methods were initialized from the same starting point and were run for a maximum of 3,000 iterations. The complete dataset and source code are publicly accessible using this link.

5.1 Constrained Problems

To evaluate the impact of step-size selection, we compare several strategies across a range of problems: the adaptive step-size schemes proposed in this work (i.e., ACGD), the pure backtracking step-size from [44], the classical short-step rule, and the open-loop step-size schedule. This comparison aims to assess the practical efficiency and convergence behavior of our adaptive methods relative to established step-sizes as baselines.

Convergence was determined using a termination criterion of 10^{-5}. We remind the reader that, for constrained problems, this criterion is chosen to be the functional-value gap when the optimal value is known; otherwise, the Frank-Wolfe gap is used.

5.1.1 Lasso Problem

We first consider a rewritten variant of the classical Lasso problem, which employs an ℓ^1-norm constraint and is widely used for sparse statistical recovery [50]. The problem is formulated as

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \ & \|b - Ax\|_2^2 \\ \text{s.t.} \ & \|x\|_1 \leq \tau \end{aligned} \tag{20}$$

where $A \in \mathbb{R}^{m\times n}$ is the measurement matrix, $b \in \mathbb{R}^m$ is the observed signal, and $\tau > 0$ controls the sparsity of the solution $x \in \mathbb{R}^n$.

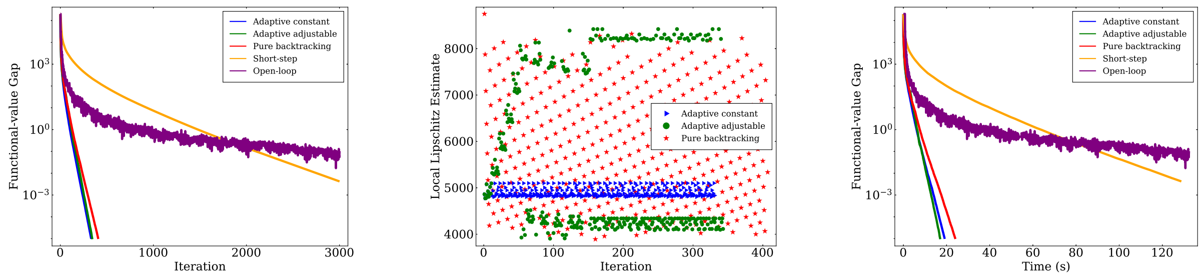
We generated a random problem instance to comprehensively assess step-size methods with $m = 2,000$, $n = 10,000$, and $\tau = 10$. For each instance, we generate a random ground truth solution x^{true} as a sparse vector with τ non-zero entries sampled from a standard Gaussian distribution. The matrix A is also random with Gaussian entries, and the signal b was computed as $b := Ax^{\text{true}}$, resulting in an optimal objective value of zero. The Lipschitz constant of the gradient was computed as $L = 2\lambda_{\max}(A^\top A)$.

Table 1 presents a comparative analysis of the performance of various step-size strategies on this Lasso instance. The adaptive adjustable, adaptive constant, and pure backtracking methods all successfully converged to the optimal solution, satisfying the functional-value gap stopping criterion. Bold values in the table indicate the best performance in each column (e.g., lowest runtime, smallest objective value, or smallest gap).

Table 1 Performance comparison of step-size strategies on Lasso problem

Step-size strategy	Iterations	Time (s)	Objective value	Functional-value gap
Adaptive constant	331	19.23	**0.0000**	**0.0000**
Adaptive adjustable	343	**17.15**	**0.0000**	**0.0000**
Pure backtracking	407	24.22	**0.0000**	**0.0000**
Short-step	3000	128.84	0.0042	0.0042
Open-loop	3000	132.39	0.1025	0.1025

Among these, the adaptive adjustable method demonstrated superior efficiency, achieving convergence in 17.15 seconds over 343 iterations. The adaptive constant method followed closely, requiring 19.23 seconds and 331 iterations. The pure backtracking method, while convergent, was significantly slower, taking 24.22 seconds over 407 iterations. In contrast, both the short-step and open-loop strategies failed to converge within the limit of 3,000 iterations, exhibiting significant sub-optimality. Their final objective values were 0.0042 and 0.1025, respectively, indicating a clear performance gap compared to adaptive approaches.



(a) Gaps in Log Scale (b) Lipschitz Constant Evolution (c) Gaps over Time

Fig. 4 Convergence behavior and computational efficiency for Lasso problem

Figure 4 illustrates the convergence behavior of the step-size strategies. The open-loop and short-step methods exhibit the slowest convergence, as they fail to adapt to the local geometry of the problem. Among the adaptive methods, pure backtracking shows high variability in Lipschitz estimates, leading to inefficient progress. In contrast, the adaptive adjustable method achieves the fastest convergence by warm-starting the backtracking procedure, effectively balancing aggressiveness with stability.

As shown in Figure 4(b), the adaptive adjustable method (green dots) maintains Lipschitz estimates tightly concentrated in a low, effective range. In contrast, pure backtracking (red stars) exhibits wide fluctuations — ranging from below 4,000 to over 8,000 — resulting in either unstable or overly conservative steps. By consistently operating near the lower end of this spectrum, the adaptive adjustable method achieves faster and more stable convergence.

5.1.2 Matrix Balancing Problem

Matrix balancing is a fundamental problem in optimization with applications in network analysis and numerical linear algebra [13,28]. We consider the following constrained optimization problem

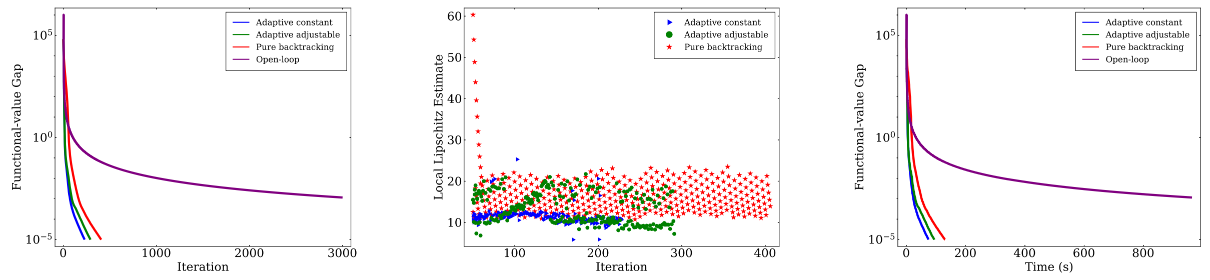
$$\begin{aligned} \min_{x\in\mathbb{R}^n} \ & \textstyle\sum_{i,j=1}^{n} a_{ij}\exp(x_i - x_j) \\ \text{s.t.} \ & x \in [a,b]^n \end{aligned} \tag{21}$$

where $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ is a square matrix with non-negative entries, satisfying the balanced condition that the sum of each row equals the sum of its corresponding column [13]. While the gradient's Lipschitz constant L lacks a closed-form expression, a conservative upper bound can be derived over the feasible set $\mathcal{C}$.

We generate synthetic data with $n = 100$, $a = 1$, and $b = 10$. The matrix A is constructed as a sparse, symmetric, and non-negative matrix. Specifically, we set a density factor $d = 5$ and initialize a random sparse matrix $A \in \mathbb{R}^{n\times n}$ using a sparsity level of d/n. To ensure symmetry and non-negativity, we compute $A = |A + A^\top| + 0.05 I_n$, where I_n is the identity matrix, introducing a small diagonal shift to enhance numerical stability.

Table 2 compares step-size strategies. All adaptive methods (adaptive constant, adaptive adjustable, and pure backtracking) converge to the optimal value of 480.2835. The adaptive constant method is fastest (228 iterations, 75.44 s), followed by adaptive adjustable (291 iterations, 94.70 s). Pure backtracking requires 406 iterations (131.08 s). The open-loop method fails to converge within 3,000 iterations, yielding a primal gap of 0.0011. The short-step rule, reliant on a conservative Lipschitz bound, was prohibitively slow and is omitted.

Figure 5 illustrates convergence behavior. Adaptive methods maintain low, stable Lipschitz estimates L_k, enabling aggressive yet safe steps. In contrast, pure backtracking exhibits wider L_k fluctuations, leading to slower progress. The open-loop method, lacking adaptivity, stagnates early.



(a) Gaps in Log Scale (b) Lipschitz Constant Evolution (c) Gaps over Time

Fig. 5 Convergence behavior and computational efficiency for Matrix Balancing problem

5.1.3 Sparse Logistic Regression Problem

Logistic regression is a cornerstone of binary classification in supervised learning [33]. Let us consider the following logistic regression problem

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \ & \frac{1}{m}\sum_{i=1}^{m} \log\left(1+\exp(-y_i x^\top a_i)\right) \\ \text{s.t.} \ & \|x\|_1 \le \tau \end{aligned} \tag{22}$$

where $A \in \mathbb{R}^{m\times n}$ is the feature matrix whose i-th row is $a_i \in \mathbb{R}^n$, $y \in \{-1,1\}^m$ is a vector whose entries are labels, and $\tau > 0$ controls solution sparsity. Here, The logistic loss function is L-smooth with $L = \|A\|_{\text{op}}^2/(4m)$, where $\|A\|_{\text{op}}$ denotes the operator norm (or spectral norm) of the matrix A, defined as the largest singular value of A.

We evaluate step-size strategies on LIBSVM `a1a` dataset [11] ($m = 1605$ and $n = 123$). All methods hit the 3,000-iteration limit and results are summarized in Table 3. Adaptive constant yields the best objective (0.3114) and smallest Frank-Wolfe gap (0.0086). Adaptive adjustable is fastest (1.35 s) with near-optimal accuracy (gap: 0.0112). Pure backtracking lags in both speed and accuracy. Non-adaptive methods underperform: short-step yields a poor objective (0.4522) and large gap (1.1077); open-loop is fastest (1.03 s) but plateaus at a suboptimal gap (0.0258).

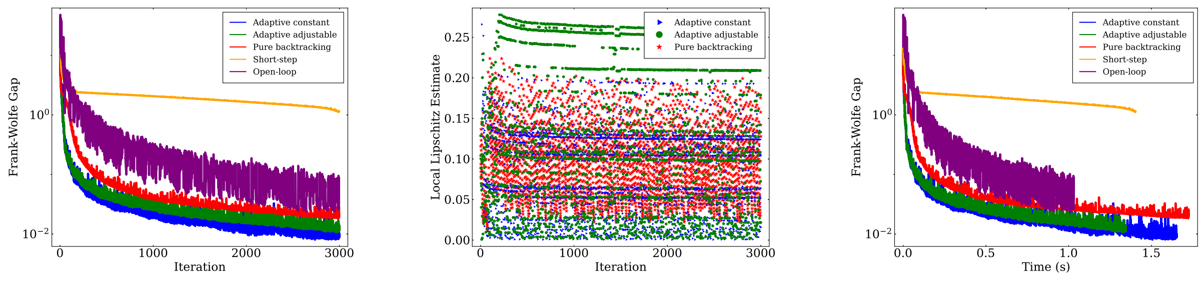
Figure 6 illustrates the convergence behavior across all methods. The left plot shows that adaptive strategies achieve significantly lower Frank-Wolfe gaps by the iteration limit, while non-adaptive methods stagnate at higher values. The center plot reveals that the adaptive adjustable method maintains stable and efficient local Lipschitz estimates, contributing to consistent progress. The right plot highlights the trade-off between speed and accuracy.

Table 2 Performance comparison of step-size strategies on Matrix Balancing problem

Step-size strategy	Iterations	Time (s)	Objective value	Functional-value gap
Adaptive constant	**228**	**75.44**	**480.2835**	**0.0000**
Adaptive adjustable	291	94.70	**480.2835**	**0.0000**
Pure backtracking	406	131.08	**480.2835**	**0.0000**
Open-loop	3000	964.16	480.2846	0.0011

Table 3 Performance comparison of step-size strategies on Logistic Regression problem

Step-size strategy	Iterations	Time	Objective value	Frank-Wolfe gap
Adaptive constant	3000	1.65	**0.3114**	**0.0086**
Adaptive adjustable	3000	1.35	0.3135	0.0112
Pure backtracking	3000	1.73	0.3171	0.0190
Short-step	3000	1.41	0.4522	1.1077
Open-loop	3000	**1.03**	0.3176	0.0258



(a) Gaps in Log Scale (b) Lipschitz Constant Evolution (c) Gaps over Time

Fig. 6 Convergence behavior and computational efficiency for Logistic Regression problem

5.1.4 Sparse Least Squares Sigmoid Regression Problem (non-convex)

We now evaluate the step-size strategies on a non-convex optimization problem as follows

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \ & \frac{1}{m}\sum_{i=1}^{m}\left(y_i - \tfrac{1}{1+\exp(-x^\top a_i)}\right)^2 \\ \text{s.t.} \ & \|x\|_1 \le \tau, \end{aligned} \tag{23}$$

where the loss function is non-convex. The experiment is conducted on the LIBSVM `a1a` dataset [11], which has $m = 1,605$ samples and $n = 123$ features. The regularization parameter is set to $\tau = 50$.

The performance of the five step-size strategies is summarized in Table 4 and visualized in Figure 7. Since the global optimum is unknown for this non-convex problem, we assess performance based on both the final objective value, the Frank-Wolfe gap as a measure of stationarity, and the total runtime.

Table 4 Performance comparison of step-size strategies on non-convex Sigmoid Regression problem

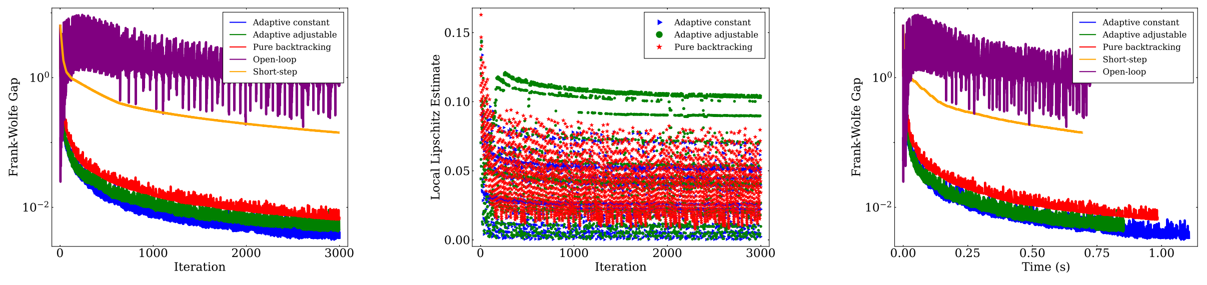
Step-size strategy	Iterations	Time (s)	Objective value	Frank-Wolfe gap
Adaptive constant	3000	1.11	**0.0989**	**0.0032**
Adaptive adjustable	3000	**0.85**	0.0995	0.0043
Pure backtracking	3000	0.99	0.1007	0.0071
Short-step	3000	0.69	0.1184	0.1418
Open-loop	3000	0.72	0.1279	1.6103

The adaptive methods and pure backtracking demonstrate superior performance. The Adaptive constant strategy achieves the lowest final objective value and the smallest Frank-Wolfe gap, indicating the best approximately-stationary point. The Adaptive adjustable strategy is the most computationally efficient, exhibiting the lowest runtime while achieving a comparable solution quality. Pure backtracking also yields a competitive result, though it converges to a slightly higher Frank-Wolfe gap. As seen in Figure 7, these three methods effectively and rapidly reduce the Frank-Wolfe gap, both per iteration and over wall-clock time. In contrast, the non-adaptive strategies struggle. The Short-step method converges very slowly, resulting in a suboptimal solution with a high Frank-Wolfe gap. The open-loop strategy is particularly ill-suited for this problem; its Frank-Wolfe gap oscillates wildly and fails to decrease consistently, indicating a failure to converge.

Figure 7(b) offers insight into the dynamics of the adaptive methods, showing the evolution of their local Lipschitz constant estimates. The estimates fluctuate significantly throughout the optimization process, which is expected in a non-convex setting, yet they remain sufficiently bounded to guide the algorithms toward a solution. The relative stability of the adaptive methods highlights their robustness in complex optimization landscapes where a fixed step-size is inadequate.

5.1.5 Video Co-Localization

We investigate the video co-localization task using the aeroplane class from the YouTube-Objects dataset [47], adopting the approach of [27]. The aim is to detect and follow the aeroplane across video frames by



(a) Gaps in Log Scale (b) Lipschitz Constant Evolution (c) Gaps over Time

Fig. 7 Convergence behavior and computational efficiency for non-convex Sigmoid Regression problem

computing bounding boxes. Mathematically, this is expressed as the optimization problem

$$\begin{aligned} \min_{x \in \mathbb{R}^{660}} \quad & \frac{1}{2} x^\top A x + b^\top x \\ \text{s.t.} \quad & x \in \mathcal{P} \end{aligned} \tag{VCL}$$

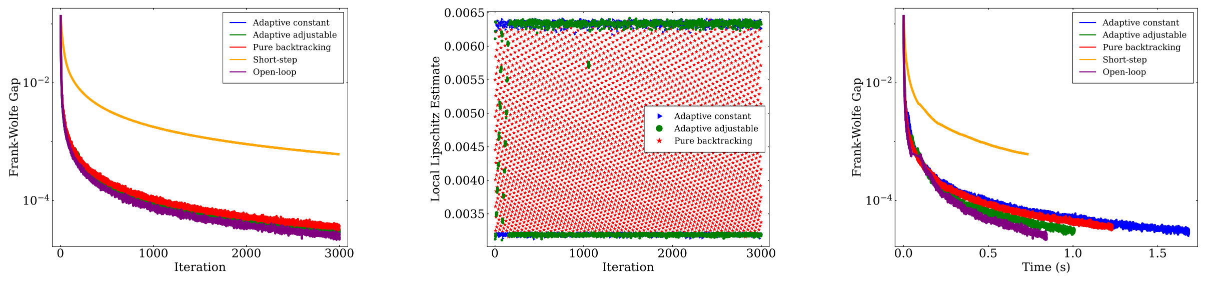
where $A \in \mathbb{R}^{660\times 660}$ and $b \in \mathbb{R}^{660}$ are data-derived matrices, and $\mathcal{P}$ denotes a flow polytope encoding temporal consistency across frames. Note that, the LMO over $\mathcal{P}$ reduces to computing a shortest path in a corresponding directed acyclic graph.

Results are summarized in Table 5. The adaptive constant, adaptive adjustable, pure backtracking, and open-loop methods all converge to the optimal value with zero Frank-Wolfe gap. Only the short-step method exhibits slight sub-optimality. In terms of speed, the short-step method is fastest but sacrifices accuracy. Among fully convergent methods, adaptive adjustable is most efficient, followed by open-loop, pure backtracking, and adaptive constant.

Table 5 Performance comparison of step-size strategies on Video Co-Localization problem

Step-size strategy	Iterations	Time	Objective value	Frank-Wolfe gap
Adaptive constant	3000	1.66	**0.0984**	**0.0000**
Adaptive adjustable	3000	0.91	**0.0984**	**0.0000**
Pure backtracking	3000	1.16	**0.0984**	**0.0000**
Short-step	3000	**0.80**	0.0990	0.0006
Open-loop	3000	1.02	**0.0984**	**0.0000**

Figure 8 illustrates convergence behavior. The left plot confirms that all methods except short-step achieve near-zero duality gap, with adaptive methods converging fastest. The center plot shows that adaptive constant and adaptive adjustable maintain low, stable Lipschitz estimates (L_k), while pure backtracking exhibits higher variability. The right plot highlights efficiency: adaptive adjustable and open-loop achieve the fastest initial progress, while short-step — though fastest overall — plateaus at a suboptimal level.



(a) Gaps in Log Scale (b) Lipschitz Constant Evolution (c) Gaps over Time

Fig. 8 Convergence behavior and computational efficiency for Video Co-Localization problem

5.1.6 Collaborative Filtering

Collaborative filtering lies at the heart of modern recommendation systems, leveraging patterns in user-item interactions to predict unobserved preferences. This task is formulated as the following constrained optimization problem

$$\begin{aligned} \min_{X \in \mathbb{R}^{m \times n}} \quad & \frac{1}{|\mathcal{I}|} \sum_{(i,j) \in \mathcal{I}} h_\rho(Y_{ij} - X_{ij}) \\ \text{s.t.} \quad & \|X\|_{\text{nuc}} \le \tau \end{aligned} \tag{24}$$

where $Y \in \mathbb{R}^{m \times n}$ is the partially observed rating matrix, $\mathcal{I} \subset \{1, \dots, m\} \times \{1, \dots, n\}$ is the index set of observed entries, and h_ρ is the Huber loss [25]:

$$h_\rho(\alpha) := \begin{cases} \frac{\alpha^2}{2}, & \text{if } |\alpha| \le \rho \\ \rho(|\alpha| - \frac{\rho}{2}), & \text{if } |\alpha| > \rho \end{cases} \tag{25}$$

and $\|\cdot\|_{\text{nuc}}$ denotes the nuclear norm, given by

$$\|X\|_{\text{nuc}} = \operatorname{tr}\left(\sqrt{X^\top X}\right) = \sum_{i=1}^{\min\{m,n\}} \sigma_i(X)$$

where $\sigma_i(X)$ are the singular values of X.

We apply this formulation to the MovieLens 100K dataset [22]. For this problem, the number of users is $m = 943$, the number of items is $n = 1,682$, and the number of observed entries is $|\mathcal{I}| = 10^5$. We set the parameters to $\rho = 1$ and $\tau = 5,000$.

Table 6 summarizes performance across step-size strategies. Open-loop remains the most efficient in practice, achieving the best objective value (0.1557) in the shortest time (1904.24s), despite not having the smallest Frank-Wolfe gap. Among adaptive methods, adaptive adjustable is the most accurate, attaining the smallest Frank-Wolfe gap (0.0134), and performs comparably to adaptive constant in objective value (0.1625 vs. 0.1622), while requiring less time. Pure backtracking is faster than both adaptive variants but yields a slightly worse objective and larger Frank-Wolfe gap. The short-step method lags significantly, producing a substantially worse objective and the largest Frank-Wolfe gap, indicating poor convergence. Figure 9 provides a visual illustration of the performance results and convergence behavior.

Table 6 Performance comparison of step-size strategies on Collaborative Filtering problem

Step-size Strategy	Iterations	Time (s)	Objective Value	Frank-Wolfe gap
Adaptive constant	3000	2485.48	0.1622	0.0137
Adaptive adjustable	3000	2329.82	0.1625	**0.0134**
Pure backtracking	3000	2191.40	0.1638	0.0146
Short-step	3000	2250.36	0.1897	0.0422
Open-loop	3000	**1904.24**	**0.1557**	0.0155

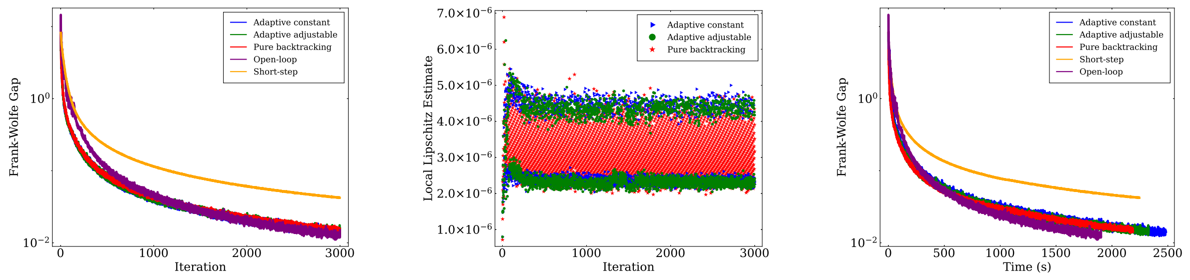
It is important to note that, in problem (24) with the data we used, the objective function is smooth with gradient whose Lipschitz constant is $L = 1/|\mathcal{I}| = 10^{-5}$, which is an extremely small value. Step-size strategies that rely on the Lipschitz constant of the gradient become too large in this regime, whereas the open-loop rule $t_k = 2/(k+2)$ achieves faster and more effective convergence on this problem. This behavior is clearly reflected in Table 6 and Figure 9.

5.1.7 Simplex Quadratic Optimization Problem (non-convex)

In this part, we consider the standard quadratic optimization problem, which entails minimizing a homogeneous quadratic objective function over the unit simplex set, formulated as

$$\begin{aligned} \min_{x \ge 0} \quad & x^\top Q x \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i = 1 \end{aligned} \tag{26}$$

where $Q \in \mathbb{R}^{n \times n}$ is symmetric. When Q is indefinite (with both positive and negative eigenvalues), the problem is non-convex and NP-hard [1,30]. We use a symmetric indefinite matrix Q of size $n = 900$ from [30] to evaluate the Conditional Gradient algorithm using different step-size strategies.



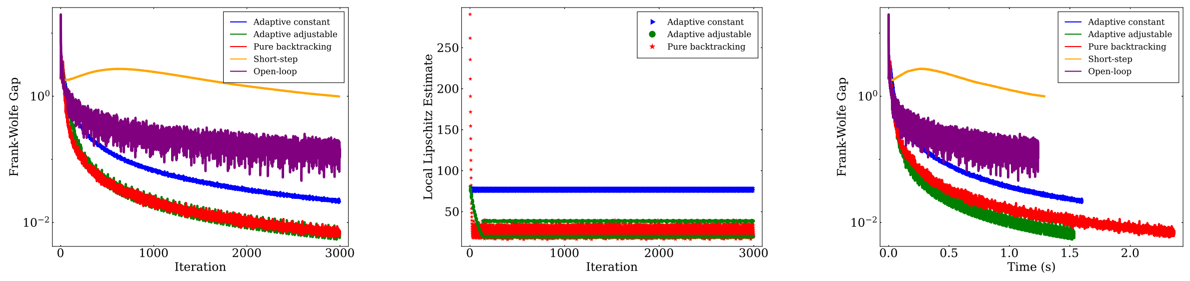
(a) Gaps in Log Scale (b) Lipschitz Constant Evolution (c) Gaps over Time

Fig. 9 Convergence behavior and computational efficiency for Collaborative Filtering problem

The results are summarized in Table 7. The Adaptive adjustable strategy achieves the best objective value and the smallest Frank-Wolfe gap, demonstrating superior solution quality and convergence. The Pure backtracking method delivers nearly identical accuracy in terms of objective value and Frank-Wolfe gap, at a moderately higher computational cost. The Adaptive constant strategy produces a competitive objective but is slower than Adaptive adjustable. In contrast, the Short-step approach is the fastest but fails to converge to a meaningful solution, yielding a poor objective value and a very large Frank-Wolfe gap. The open-loop method is nearly as fast as Short-step but produces the worst Frank-Wolfe gap of all the methods. These performance trends are further illustrated in Figure 10, which depicts the convergence behavior and comparative efficiency of the different step-size strategies.

Table 7 Performance comparison of step-size strategies on non-convex Standard Quadratic Optimization problem

Step-size Strategy	Iterations	Time (s)	Objective Value	Frank-Wolfe gap
Adaptive constant	3000	1.90	-6.9860	0.0221
Adaptive adjustable	3000	1.68	**-7.0014**	**0.0065**
Pure backtracking	3000	2.06	-7.0009	0.0068
Short-step	3000	**1.29**	-5.9066	0.9923
Open-loop	3000	1.30	-6.9158	0.1495



(a) Gaps in Log Scale (b) Lipschitz Constant Evolution (c) Gaps over Time

Fig. 10 Convergence behavior and computational efficiency for Standard Quadratic Optimization problem

Until now, we have observed that for the Conditional Gradient method, the adaptive adjustable step-size strategy outperforms other adaptive step-size methods (pure backtracking and constant) in most cases, achieving faster convergence and competitive gaps. While the open-loop step-size method occasionally performs well, its inconsistency makes it less reliable. The short-step-size method consistently underperforms and fails to converge due to overly conservative steps. This makes the adaptive adjustable method the most reliable and efficient choice for both convex and non-convex problems that we considered in this part.

5.2 Unconstrained Problems

We analyze the impact of step-size on the performance of ACGD applied to unconstrained optimization problems. We evaluate three step-size strategies—adaptive constant, adaptive adjustable, and pure backtracking. The process terminates when the gradient norm falls below a tolerance of 10^{-5} or after a maximum of 3,000 iterations, whichever occurs first.

5.2.1 Least Squares Problem Under Different Geometries

The least squares problem is a fundamental concept in linear algebra. It is typically used to find the best (approximate) solution to a system of linear equations. The problem is formally stated as

$$\min_{x\in\mathbb{R}^n} \|b - Ax\|_2^2 \tag{27}$$

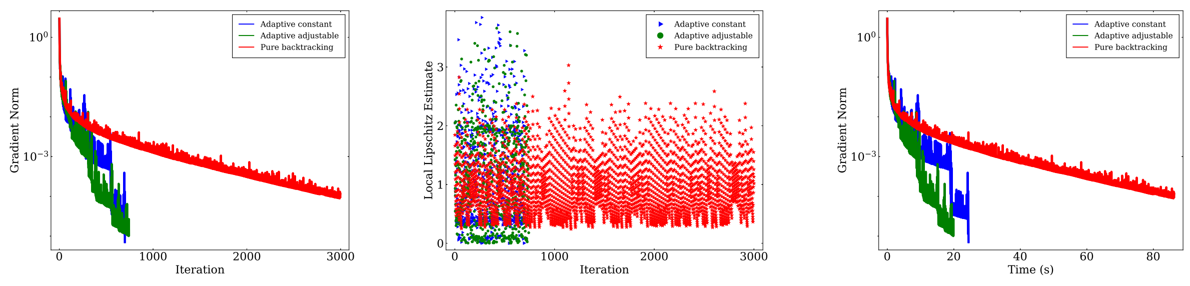
where $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. For our numerical experiments, we synthetically generate A and b with dimensions $m = 20,000$ and $n = 1,000$.

Table 8 compares the performance of various step-size strategies, while Figure 11 illustrates that adaptive methods efficiently reduce the gradient norm. These results demonstrate that adaptive step-size strategies achieve optimal solutions with superior computational efficiency compared to pure backtracking in the Normalized Steepest Descent algorithm.

For estimating a local Lipschitz constant and performing backtracking, we used ℓ^2 norm. We observe that, in terms of runtime, objective value, and gradient (LMO dual) norm, the ℓ^2 geometry for the LMO yields better performance.

Table 8 Performance comparison of step-size strategies on the unconstrained Least Squares problem with different Geometry.

LMO Norm	Step-size Strategy	Iterations	Time (s)	Objective Value	Gradient (LMO Dual) Norm
ℓ^2	Adaptive constant	698	24.47	**1.8857**	**0.0000**
	Adaptive adjustable	742	**19.95**	**1.8857**	**0.0000**
	Pure backtracking	3000	86.39	**1.8857**	0.0001
ℓ^1	Adaptive constant	3000	95.40	1.9217	**0.0015**
	Adaptive adjustable	3000	**68.65**	**1.9193**	0.0016
	Pure backtracking	3000	85.08	1.9216	0.0016
ℓ^∞	Adaptive constant	3000	103.40	**1.9039**	0.0953
	Adaptive adjustable	3000	**69.59**	1.9046	0.0950
	Pure backtracking	3000	86.01	1.9094	**0.0868**



(a) Gradient Norms in Log Scale (b) Lipschitz Constant Evolution (c) Gradient Norms over Time

Fig. 11 Convergence behavior and computational efficiency for the Least Squares problem (Backtracking and LMO ℓ^2).

5.2.2 Least Squares Sigmoid Problem (non-convex)

The least squares sigmoid problem involves minimizing a non-convex loss function, specifically the squared error between observed outputs and a sigmoid function. It is formulated as

$$\min_{x\in\mathbb{R}^n} \frac{1}{m}\sum_{i=1}^{m}\left(y_i - \frac{1}{1+\exp\left(-x^\top a_i\right)}\right)^2. \tag{28}$$

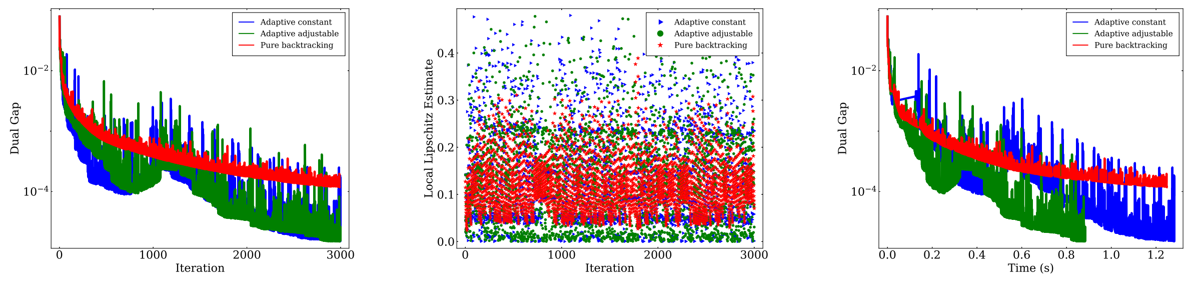
For our numerical experiments, we use the LIBSVM `a1a` dataset [11] with $m = 1{,}618$ samples and $n = 119$ features.

Table 9 compares the performance of different step-size strategies. The adaptive adjustable method is the most time-efficient, achieving a better objective value in 0.88 seconds. The adaptive constant method matches the same objective value of 0.0941 but requires more computational time (1.28 seconds). In contrast, pure backtracking is less efficient, taking 1.25 seconds and yielding a slightly higher objective value of 0.0956.

Figure 12 visually complements the numerical findings in Table 9, illustrating the convergence behavior and computational efficiency of our adaptive step-size strategies.

Table 9 Performance comparison of step-size strategies on non-convex least squares sigmoid problem

Step-size Strategy	Iterations	Time (s)	Objective Value	Gradient Norm
Adaptive constant	3000	1.28	**0.0941**	0.0001
Adaptive adjustable	3000	**0.88**	**0.0941**	**0.0000**
Pure backtracking	3000	1.25	0.0956	0.0001



(a) Gradient Norms in Log Scale (b) Lipschitz Constant Evolution (c) Gradient Norms over Time

Fig. 12 Convergence behavior and computational efficiency for non-convex least squares sigmoid problem

5.2.3 Sum of Squares Problem (non-convex)

Next, we consider a non-convex sum of squares problem as follows

$$\min_{x\in\mathbb{R}^n} (x_1-3)^2 + \sum_{i=2}^{n}\left(x_1 - 3 - 2\left(x_1+\ldots+x_i\right)^2\right)^2. \tag{29}$$

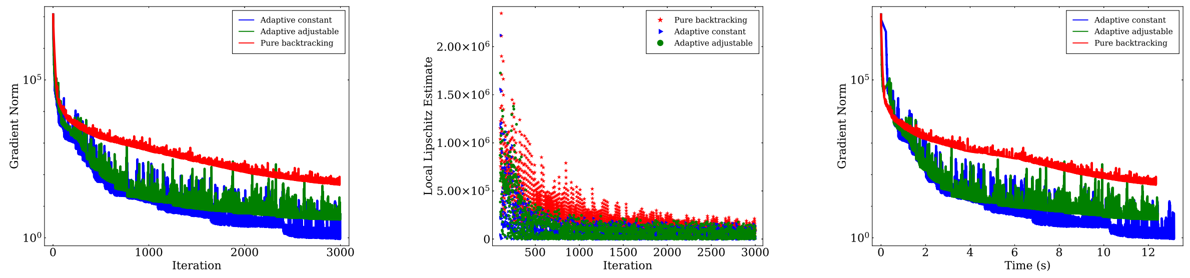
For our numerical experiments, we set the dimension to $n = 200$.

Table 10 presents the performance comparison results. The adaptive constant method achieves the best objective value and the lowest gradient norm, indicating superior solution quality. However, it is the slowest in terms of runtime. The adaptive adjustable method offers a faster runtime but results in a higher objective value and gradient norm. The pure backtracking method is the fastest but yields the highest objective value and gradient norm, suggesting convergence to a significantly poorer local minimum. The consistently large maximum and mean Lipschitz constant values across all methods reflect the challenging nature of this non-convex problem.

Figure 13 illustrates the convergence behavior. Subfigure 13(b) shows that, for $k \geq 100$, the adaptive constant method yields more stable and lower Lipschitz estimates compared to the pure backtracking method, which exhibits greater variability and higher values.

Table 10 Performance comparison of step-size strategies on the non-convex Sum of Squares problem.

Step-size Strategy	Iterations	Time (s)	Objective Value	Gradient Norm
Adaptive constant	3000	13.17	**0.6239**	**0.9193**
Adaptive adjustable	3000	12.45	4.3445	6.0382
Pure backtracking	3000	**12.38**	66.9148	47.8900



(a) Gradient Norms in Log Scale (b) Lipschitz Constant Evolution ($k \geq 100$) (c) Gradient Norms over Time

Fig. 13 Convergence behavior and computational efficiency for the non-convex Sum of Squares problem.

5.2.4 Rosenbrock Function (non-convex)

We also evaluate our method on the Rosenbrock function, a classic benchmark problem renowned for its challenging non-convex structure [48]. Consider the following problem

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{n-1} \left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right). \tag{30}$$

Its narrow, parabolic valley poses significant convergence difficulties. For numerical experiments, we set $n = 1,000$.

Table 11 shows that the adaptive adjustable method achieves the best performance with the lowest objective value and gradient norm in the fastest time. The adaptive constant method is slower with a higher objective value and gradient norm. Pure backtracking is slightly faster than adaptive constant but yields the worst objective value and gradient norm.

These observations are visually supported by Figure 14, which illustrates the convergence behavior and computational efficiency.

Table 11 Performance comparison of step-size strategies on the non-convex Rosenbrock function.

Step-size Strategy	Iterations	Time (s)	Objective Value	Gradient Norm
Adaptive constant	3000	82.37	928.6291	3.2603
Adaptive adjustable	3000	**81.19**	**923.9158**	**2.6786**
Pure backtracking	3000	81.80	979.7630	3.5251

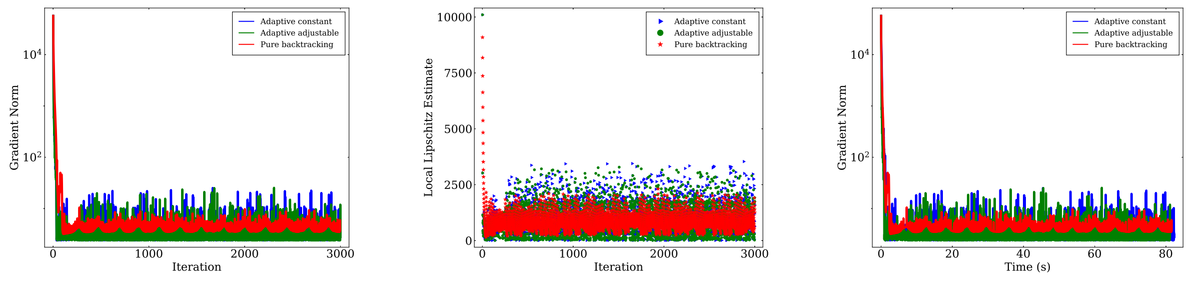
5.2.5 Lévy Function

We evaluate the non-convex multimodal Lévy function (a common benchmark for global optimization algorithms) defined as

$$\min_{x \in \mathbb{R}^n} \sin^2(\pi w_1) + \sum_{i=1}^{n-1} (w_i - 1)^2 \left(1 + 10\sin^2(\pi w_{i+1})\right) + (w_n - 1)^2 \left(1 + \sin^2(2\pi w_n)\right) \tag{31}$$

where $w_i = 1 + \frac{x_i - 1}{4}$ for all $i = 1, \ldots, n$. We set $n = 1,000$

Table 12 compares step-size strategies for the Normalized Gradient Descent algorithm. The adaptive constant method achieves the lowest objective value and gradient norm with fewer iterations. The adaptive adjustable method is slightly faster but yields higher objective value and gradient norm. Pure



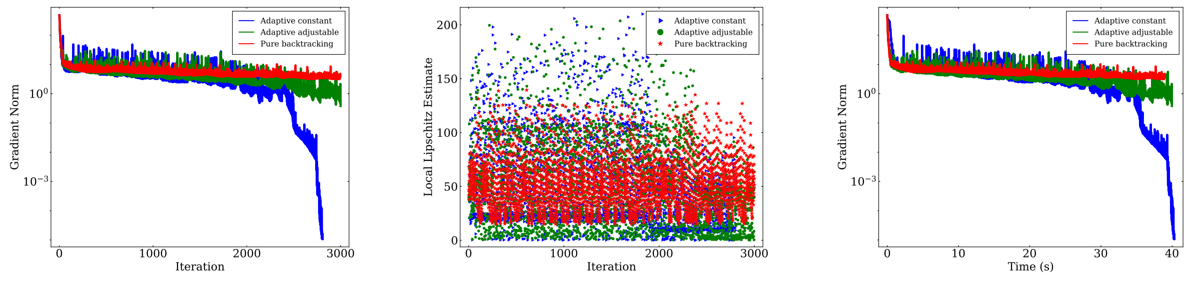
(a) Gradient Norms in Log Scale (b) Lipschitz Constant Evolution (c) Gradient Norms over Time

Fig. 14 Convergence behavior and computational efficiency for the non-convex Rosenbrock function.

backtracking is the fastest but has the highest objective value and gradient norm. Figure 15 illustrates the convergence behavior and computational efficiency.

Table 12 Performance comparison of step-size strategies on the non-convex Lévy function.

Step-size Strategy	Iterations	Time (s)	Objective Value	Gradient Norm
Adaptive constant	2808	40.34	**24.7208**	**0.0000**
Adaptive adjustable	3000	39.98	25.2061	0.3505
Pure backtracking	3000	**39.02**	522.6736	3.2346



(a) Gradient Norms in Log Scale (b) Lipschitz Constant Evolution (c) Gradient Norms over Time

Fig. 15 Convergence behavior and computational efficiency for the non-convex Lévy function.

5.2.6 Zakharov Function

We evaluate the Zakharov function, a convex benchmark for optimization algorithms, defined as

$$\min_{x\in\mathbb{R}^n} \sum_{i=1}^{n} x_i^2 + \left(\frac{1}{2}\sum_{i=1}^{n} ix_i\right)^2 + \left(\frac{1}{2}\sum_{i=1}^{n} ix_i\right)^4 . \tag{32}$$

We set $n = 1,000$.

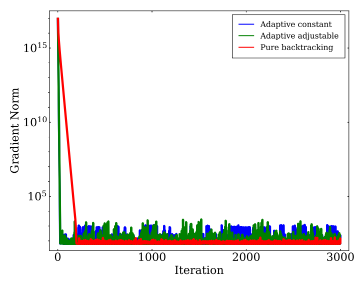
Table 13 compares step-size strategies for the Normalized Gradient Descent algorithm. The adaptive adjustable method achieves the lowest objective value and gradient norm. The adaptive constant method is slightly slower with a higher objective value and gradient norm. Pure backtracking is the fastest but yields the highest objective value and a comparable gradient norm. Figure 16 illustrates the convergence behavior and local Lipschitz constant estimation for $k \geq 250$.

6 Conclusion

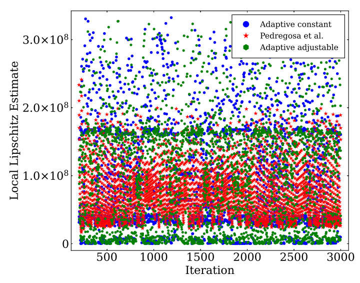
In this paper, we have presented adaptive step-size strategies for two important classes of first-order optimization algorithms that rely on LMOs: the Frank-Wolfe or Conditional Gradient algorithm for

Table 13 Performance comparison of step-size strategies on the convex Zakharov function

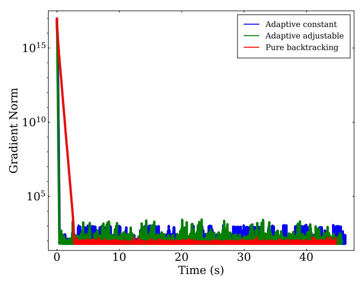
Step-size Strategy	Iterations	Time (s)	Objective Value	Gradient Norm
Adaptive constant	3000	46.27	1041.89	112.34
Adaptive adjustable	3000	45.73	**1039.44**	**64.58**
Pure backtracking	3000	**44.71**	1044.64	64.64



(a) Gradient Norms in Log Scale



(b) Lipschitz Constant Evolution ($k \geq 250$)



(c) Gradient Norms over Time

Fig. 16 Convergence behavior and computational efficiency for the convex Zakharov function. Note that we truncate the first 250 iterations, which are quite noisy.

constrained optimization and the Normalized Steepest Descent method for unconstrained optimization. This culminated in the Adaptive Conditional Gradient Descent algorithm, ACGD, for which we provided theoretically sound, practical methods for automatically selecting step-sizes without requiring prior knowledge of the Lipschitz constant.

For constrained settings, we demonstrated that ACGD directly estimates the local Lipschitz constant from gradient differences at each iteration, adapting more rapidly to the local curvature compared to previous approaches [44]. We established convergence rates of $\mathcal{O}(1/k)$ for quasar-convex problems and $\mathcal{O}(1/\sqrt{k})$ for general non-convex problems, matching the worst-case theoretical guarantees of the classical Conditional Gradient method while eliminating the need for manual parameter tuning.

Similarly, for unconstrained settings, we proved a linear convergence rate for strongly convex problems and convergence rates of $\mathcal{O}(1/k)$ in the quasar-convex case and $\mathcal{O}(1/\sqrt{k})$ in the non-convex case. Our unified analysis framework gives a harmonized presentation of both methods, highlighting their theoretical connections despite addressing different optimization scenarios.

The key advantage of our approach lies in its practicality and simplified theoretical analysis. By dynamically adjusting step-sizes based on local function behavior, our methods maintain robust performance across diverse problem instances without requiring problem-specific parameter tuning. Allowing for fine-grained choices of the norms involved in both the backtracking procedure and the LMO computation gives more versatility, as shown in the numerical experiments. This adaptivity is particularly valuable in machine learning applications where problem characteristics may be initially unknown or change during the optimization process.

Acknowledgements The research of the first author was in part supported by a grant from IPM (No.1404900034).

Appendices

References

1. Ahmadi, A.A., Zhang, J.: On the complexity of finding a local minimizer of a quadratic function over a polytope. Mathematical Programming **195**(1), 783–792 (2022)
2. Assunção, P., Ferreira, O.P., Prudente, L.: Conditional gradient method for multiobjective optimization. Computational Optimization and Applications **78**(3), 741–768 (2021)
3. Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B.: Julia: A fresh approach to numerical computing. SIAM review **59**(1), 65–98 (2017)
4. Bisong, E.: Google colaboratory. In: Building machine learning and deep learning models on Google cloud platform: a comprehensive guide for beginners, pp. 59–64. Springer (2019)
5. Bomze, I.M., Rinaldi, F., Zeffiro, D.: Frank–Wolfe and friends: a journey into projection-free first-order optimization methods. Annals of Operations Research **343**(2), 607–638 (2024)
6. Boyd, S.P., Vandenberghe, L.: Convex optimization. Cambridge University Press (2004)

Table 14 Closed-Form Solutions for selections of the LMO over Specified Feasible Regions

Name	Feasible Region $\mathcal{C}$	Closed Form $\underset{\mathcal{C}}{\mathrm{lmo}}(x)$
Scaled ℓ^2-Ball	$\{v \mid \|v\|_2 \le \tau\}$	$-\frac{\tau x}{\|x\|_2}$ if $x \neq 0$, else 0
Scaled Simplex	$\{v \mid \sum v_i = \tau, v_i \ge 0\}$	τe_j, where $j = \mathrm{argmin}_i\, x_i$
Scaled ℓ^1-Ball	$\{v \mid \|v\|_1 \le \tau\}$	$-\tau e_j \cdot \mathrm{sign}(x_j)$, where $j = \arg\max_i \|x_i\|$
General Box	$\{v \mid -l_i \le v_i \le u_i\}$	$v_i = -l_i$ if $x_i > 0$, u_i if $x_i < 0$, 0 if $x_i = 0$
Scaled ℓ^∞-Ball	$\{v \mid \|v\|_\infty \le \tau\}$	$v_i = -\tau \mathrm{sign}(x_i)$
Nuclear Norm-Ball	$\{V \mid \|V\|_{\mathrm{nuc}} \le \tau\}$	$-\tau uv^\top$, where u, v are top singular vectors of X
Spectral Norm Ball	$\{V \mid \|V\|_{\mathrm{op}} \le \tau\}$	$-\tau UV^\top$, where $X = U\Sigma V^\top$ is the reduced-SVD of X

7. Braun, G., Carderera, A., Combettes, C.W., Hassani, H., Karbasi, A., Mokhtari, A., Pokutta, S.: Conditional Gradient Methods: From Core Principles to AI Applications. MOS-SIAM Series on Optimization (2025)
8. Cai, T.T., Wang, L.: Orthogonal matching pursuit for sparse signal recovery with noise. IEEE Transactions on Information Theory **57**(7), 4680–4688 (2011)
9. Calinescu, G., Chekuri, C., Pal, M., Vondrák, J.: Maximizing a monotone submodular function subject to a matroid constraint. SIAM Journal on Computing **40**(6), 1740–1766 (2011)
10. Carlson, D., Cevher, V., Carin, L.: Stochastic spectral descent for restricted boltzmann machines. In: Artificial Intelligence and Statistics, pp. 111–119. PMLR (2015)
11. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST) **2**(3), 1–27 (2011)
12. Clarkson, K.L.: Coresets, sparse greedy approximation, and the Frank–Wolfe algorithm. ACM Transactions on Algorithms (TALG) **6**(4), 1–30 (2010)
13. Cohen, M.B., Madry, A., Tsipras, D., Vladu, A.: Matrix scaling and balancing via box constrained newton's method and interior point methods. In: 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS), pp. 902–913. IEEE (2017)
14. Combettes, C.W., Pokutta, S.: Complexity of linear minimization and projection on some sets. Operations Research Letters **49**(4), 565–571 (2021)
15. Dem'yanov, V.F., Rubinov, A.M.: Approximate methods in optimization problems. Elsevier Publishing Company (1970)
16. Dem'yanov, V.F., Rubinov, A.M.: The minimization of a smooth convex functional on a convex set. SIAM Journal on Control **5**(2), 280–294 (1967)
17. Dunn, J.C., Harshbarger, S.: Conditional gradient algorithms with open loop step size rules. Journal of Mathematical Analysis and Applications **62**(2), 432–444 (1978)
18. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval Research Logistics Quarterly **3**(1-2), 95–110 (1956)
19. Gonçalves, D.S., Gonçalves, M.L., Melo, J.G.: An away-step Frank–Wolfe algorithm for constrained multiobjective optimization. Computational Optimization and Applications **88**(3), 759–781 (2024)
20. Guminov, S., Gasnikov, A., Kuruzov, I.: Accelerated methods for weakly-quasi-convex optimization problems. Computational Management Science **20**(1), 36 (2023)
21. Guminov, S., Nesterov, Y.E., Dvurechensky, P., Gasnikov, A.: Primal-dual accelerated gradient descent with line search for convex and nonconvex optimization problems. Doklady Akademii Nauk **485**(1), 15–18 (2019)
22. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. ACM Transactions on Interactive Iintelligent Systems (TIIS) **5**(4), 1–19 (2015)
23. Hendrych, D., Besançon, M., Martínez-Rubio, D., Pokutta, S.: Secant line search for Frank–Wolfe algorithms. In: International Conference on Machine Learning. PMLR (2025)
24. Hendrych, D., Troppens, H., Besançon, M., Pokutta, S.: Convex mixed-integer optimization with Frank–Wolfe methods. Mathematical Programming Computation pp. 1–27 (2025)
25. Huber, P.J.: Robust estimation of a location parameter. In: Breakthroughs in statistics: Methodology and distribution, pp. 492–518. Springer (1992)
26. Jaggi, M.: Revisiting Frank–Wolfe: Projection-free sparse convex optimization. In: International Conference on Machine Learning, pp. 427–435. PMLR (2013)
27. Joulin, A., Tang, K., Fei-Fei, L.: Efficient image and video co-localization with Frank–Wolfe algorithm. In: European Conference on Computer Vision, pp. 253–268. Springer (2014)
28. Kalantari, B., Khachiyan, L., Shokoufandeh, A.: On the complexity of matrix balancing. SIAM Journal on Matrix Analysis and Applications **18**(2), 450–463 (1997)
29. Kelner, J.A., Lee, Y.T., Orecchia, L., Sidford, A.: An almost-linear-time algorithm for approximate max flow in undirected graphs, and its multicommodity generalizations. In: Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 217–226. SIAM (2014)
30. Khademi, A., Marandi, A.: Quadratic optimization through the lens of adjustable robust optimization. INFORMS Journal on Computing pp. 1–25 (2025)
31. Lacoste-Julien, S., Jaggi, M.: An affine invariant linear convergence analysis for Frank–Wolfe algorithms. arXiv:1312.7864 (2013)
32. LeBlanc, L.J., Morlok, E.K., Pierskalla, W.P.: An efficient approach to solving the road network equilibrium traffic assignment problem. Transportation Research **9**(5), 309–318 (1975)
33. Lee, S.I., Lee, H., Abbeel, P., Ng, A.Y.: Efficient L1 regularized logistic regression. In: Aaai, vol. 6, pp. 401–408 (2006)
34. Levitin, E.S., Polyak, B.T.: Constrained minimization methods. USSR Computational Mathematics and Mathematical Physics **6**(5), 1–50 (1966)
35. Locatello, F., Khanna, R., Tschannen, M., Jaggi, M.: A unified optimization view on generalized matching pursuit and Frank–Wolfe. In: Artificial Intelligence and Statistics, pp. 860–868. PMLR (2017)

36. Locatello, F., Raj, A., Karimireddy, S.P., Rätsch, G., Schölkopf, B., Stich, S., Jaggi, M.: On matching pursuit and coordinate descent. In: International Conference on Machine Learning, pp. 3198–3207. PMLR (2018)
37. Malitsky, Y., Mishchenko, K.: Adaptive gradient descent without descent. arXiv:1910.09529 (2019)
38. Malitsky, Y., Mishchenko, K.: Adaptive proximal gradient method for convex optimization. Advances in Neural Information Processing Systems **37**, 100,670–100,697 (2024)
39. Martínez-Rubio, D., Pokutta, S.: Beyond short steps in Frank–Wolfe algorithms. arXiv:2501.18773 (2025)
40. Martínez-Rubio, D.: Smooth quasar-convex optimization with constraints. arXiv:2510.01943 (2025)
41. Millan, R.D., Ferreira, O.P., Ugon, J.: Frank–Wolfe algorithm for star-convex functions. arXiv:2507.17272 (2025)
42. Nesterov, Y.: Efficiency of coordinate descent methods on huge-scale optimization problems. SIAM Journal on Optimization **22**(2), 341–362 (2012)
43. Nutini, J., Schmidt, M., Laradji, I., Friedlander, M., Koepke, H.: Coordinate descent converges faster with the gauss-southwell rule than random selection. In: International Conference on Machine Learning, pp. 1632–1641. PMLR (2015)
44. 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)
45. Pethick, T., Xie, W., Antonakopoulos, K., Zhu, Z., Silveti-Falls, A., Cevher, V.: Training deep learning models with norm-constrained LMOs. arXiv:2502.07529 (2025)
46. Pethick, T., Xie, W., Erdogan, M., Antonakopoulos, K., Silveti-Falls, T., Cevher, V.: Generalized gradient norm clipping & non-Euclidean (l_0, l_1)-smoothness. arXiv:2506.01913 (2025)
47. Prest, A., Leistner, C., Civera, J., Schmid, C., Ferrari, V.: Learning object class detectors from weakly annotated video. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3282–3289. IEEE (2012)
48. Rosenbrock, H.: An automatic method for finding the greatest or least value of a function. The Computer Journal **3**(3), 175–184 (1960)
49. Shalev-Shwartz, S., Gonen, A., Shamir, O.: Large-scale convex minimization with a low-rank constraint. arXiv:1106.1622 (2011)
50. Tibshirani, R.: Regression shrinkage and selection via the Lasso. Journal of the Royal Statistical Society Series B: Statistical Methodology **58**(1), 267–288 (1996)
51. Vondrák, J.: Optimal approximation for the submodular welfare problem in the value oracle model. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, pp. 67–74 (2008)
52. Wang, J.K., Wibisono, A.: Continuized acceleration for quasar convex functions in non-convex optimization. arXiv:2302.07851 (2023)
53. Wirth, E., Pena, J., Pokutta, S.: Accelerated affine-invariant convergence rates of the Frank–Wolfe algorithm with open-loop step-sizes. Mathematical Programming pp. 1–45 (2025)