OPT2019: 11th Annual Workshop on Optimization for Machine Learning

Breaking the Curse of Dimensionality (Locally) to Accelerate Conditional Gradients

Jelena Diakonikolas JELENA.D@BERKELEY.EDU
UC Berkeley, California, USA.

Alejandro Carderera ALEJANDRO.CARDERERA@GATECH.EDU
Georgia Institute of Technology, Georgia, USA.

Sebastian Pokutta POKUTTA@ZIB.DE
Zuse Institute Berlin and Technische Universität Berlin, Berlin, Germany.

Abstract

Conditional gradient methods constitute a class of first order algorithms for solving smooth convex optimization problems that are projection-free and numerically competitive. We present the *Locally Accelerated Conditional Gradients* algorithmic framework that relaxes the projection-freeness requirement to only require projection onto (typically low-dimensional) simplices and mixes accelerated steps with conditional gradient steps to achieve *dimension-independent* local acceleration for smooth strongly convex functions. We prove that the introduced class of methods attains the asymptotically optimal convergence rate. Our theoretical results are supported by numerical experiments that demonstrate a speed-up both in wall-clock time and in per-iteration progress when compared to state-of-the-art conditional gradient variants.

1. Introduction

We consider problems of the form:

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

where f is an L-smooth (gradient Lipschitz) μ-strongly convex function and $\mathcal{X} \subseteq \mathbb{R}^n$ is a polytope. We assume access to the objective function f through a first-order oracle (FO), and access to the polytope $\mathcal{X}$ through a linear optimization oracle (LO). Conditional gradient (CG) algorithms operate under this model and due to their simplicity, good practical performance, and other favorable characteristics continue to be an active area of research (see, e.g., [1–3, 7–10, 12–15, 18, 20] and references therein). While some CG variants achieve a linear convergence rate for smooth strongly convex functions [8, 15], they do not achieve the optimal accelerated convergence rate that is attained by projection-based methods for smooth strongly convex optimization.

This slower convergence is not merely an artifact of the analysis of existing CG-type methods: global dimension-independent accelerated convergence is impossible for any method whose access to the polytope is limited to an LO [12, 16]. In particular, in the worst case, any such method requires $t = \Omega(\min\{n, 1/\epsilon\})$ queries to an LO to construct a solution $\mathbf{x}_t \in \mathcal{X}$ that satisfies $f(\mathbf{x}_t) - f(\mathbf{x}^*) \leq \epsilon$, where $\mathbf{x}^* = \operatorname{argmin}_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$. Thus, if we seek a global convergence of the form $f(\mathbf{x}_t) - f(\mathbf{x}^*) \leq (1-r)^t (f(\mathbf{x}_0) - f(\mathbf{x}^*))$, then $r \leq 2\frac{\log n}{n}$,

© J. Diakonikolas, A. Carderera & S. Pokutta.

i.e., the (unaccelerated) convergence rate of methods such as the Away-step Frank-Wolfe (AFW) [11, 15] and Pairwise Frank-Wolfe (PFW) [15, 21] is optimal up to log factors.

The *Catalyst* framework [19] can provide black-box acceleration for different CG variants. However, to be compatible with the lower bound from [12, 16], this approach still incurs a large dimension-dependent factor in the resulting iteration complexity of the accelerated method. Another form of acceleration in the case of LO-based methods is achieved by *Conditional Gradient Sliding* [17], which leads to the optimal first-order oracle (FO) complexity. However, the LO complexity remains in the unaccelerated regime; this is also necessary due to the lower bound. Note that in practice a call to an LO is typically much more computationally intensive than a call to the FO.

We show that local *dimension-independent* acceleration for conditional gradient methods *is possible*. The acceleration is achieved after a burn-in phase whose length does not depend on the target accuracy ϵ (but could potentially depend on the dimension). Our contributions are summarized as follows, where we assume that f is L-smooth and μ-strongly convex.

1. (*Locally Accelerated Conditional Gradients.*) We introduce a new class of conditional gradient algorithms – *Locally Accelerated Conditional Gradients (LaCG)* – that achieve an asymptotically optimal iteration complexity of $K + O\left(\sqrt{\frac{L}{\mu}} \log \frac{1}{\epsilon}\right)$ to solve Problem (P) up to error ϵ, where K is a constant that only depends on $\mathcal{X}$ and f.

2. (*Generalized Accelerated Method.*) We generalize the algorithm μAGD+ from [4] by showing that it retains its convergence guarantees when coupled with an arbitrary alternative algorithm. Furthermore, it also tolerates inexact projections onto the feasible set and admits changes to the convex set onto which these projections are performed (as long as the convex set is contained in the convex set from the preceding iteration and it contains the minimizer $\mathbf{x}^*$).

3. (*Computational Experiments.*) We compare our methods to other conditional gradient variants and provide computational evidence that our algorithms achieve a practical speed-up, both in terms of per-iteration progress and in wall-clock time.

2. Locally Accelerated Conditional Gradients

For concreteness, we present a variant of LaCG that is used on top of the Away-Step Frank-Wolfe (AFW) algorithm. We note, however, that the same methodology can be used on top of any active-set-based CG method (such as, e.g., Pairwise Frank-Wolfe). Pseudocode of the resulting algorithm is provided in Algorithm 1.

A standing assumption in our methodology is that projections onto the convex hull of the active set can be implemented efficiently. These projections are typically cheap in applications due to the small size of the active sets; the sparsity of the final solution and the away steps keep the active set small. Further, there are various heuristics that are used in practice to reduce the size of the active set (see, e.g., [2]). Solving the projection problem amounts to minimizing a quadratic function over the probability simplex to an accuracy that is of the same order as the target accuracy, and thus normally requires a number of iterations of the order $\log(1/\epsilon)$. Furthermore, computing this projection does not require any additional calls to either the LO or the FO of the original problem. In Algorithm 1, the projection steps are carried out in Lines 16 and 12.

Our algorithm couples the steps of AFW and an adaptation of μAGD+ [4] that we introduce to handle special properties of CG-type methods. In particular, in every iteration, the method computes the AFW step and the μAGD+ step and chooses the one with the lower function value. This is crucial to ensure that the method always makes at least as much per-iteration progress as AFW. It also ensures convergence over iterations in which the convex hull $\mathcal{C}_k$ of the active set $\mathcal{S}_k$ does not contain the optimum $\mathbf{x}^*$ and in which there are no guarantees for the accelerated sequence. By a standard fact, after a constant number of iterations K (that only depends on f and $\mathcal{X}$), $\mathcal{C}_k$ is guaranteed to contain (a ball of radius r around) $\mathbf{x}^*$ in every subsequent iteration. After those K iterations, we can apply the arguments pertaining to the accelerated sequence. Our coupling ensures that we do not need to know K a priori or have the ability to detect whether $\mathbf{x}^* \in \mathcal{C}_k$.

Another ingredient of our analysis is that we allow feasible sets used in the accelerated sequence to shrink (see Lemma 1 below). This allows us to use the convex hull of the same active set as AFW, as long as AFW is not adding any vertices. When AFW adds a new vertex, LaCG freezes the active set for the analysis from Lemma 1 to apply. The discrepancy between the two active sets is resolved via scheduled restarts that are enforced not to happen too frequently, so that the convergence rate from Lemma 1 can be preserved. Due to space constraints, further details are omitted and we only state the final result in Theorem 2.

Algorithm 1: Locally Accelerated Conditional Gradients

Let $\mathbf{x}_0 \in \mathcal{X}$ be an arbitrary point, $\mathcal{S}_0^{\text{AFW}} = \{\mathbf{x}_0\}$, $\boldsymbol{\lambda}_0^{\text{AFW}} = [1]$;
Let $\mathbf{y}_0 = \hat{\mathbf{x}}_0 = \mathbf{w}_0 = \mathbf{x}_0$, $\mathbf{z}_0 = -\nabla f(\mathbf{y}_0) + L\mathbf{y}_0$, $\mathcal{C}_1 = \text{co}(\mathcal{S}_0^{\text{AFW}})$;
$a_0 = A_0 = 1$, $\theta = \sqrt{\frac{\mu}{2L}}$, $\mu_0 = L - \mu$;
$H = \frac{2}{\theta}\log(1/(2\theta^2) - 1)$; // Minimum restart period
$r_f = \text{false}$, $r_c = 0$; // Restart flag and restart counter initialization
for $i \leftarrow 2$ **to** l **do**
  $\mathbf{x}_k^{\text{AFW}}, \mathcal{S}_k^{\text{AFW}}, \boldsymbol{\lambda}_k^{\text{AFW}} = \text{AFW}(\mathbf{x}_{k-1}^{\text{AFW}}, \mathcal{S}_{k-1}^{\text{AFW}}, \boldsymbol{\lambda}_{k-1}^{\text{AFW}})$; // AFW step
  if r_f *and* $r_c \geq H$ **then** // Restart criterion is met
    $\mathbf{y}_k = \text{argmin}\{f(\mathbf{x}_k^{\text{AFW}}), f(\hat{\mathbf{x}}_k)\}$;
    $\mathcal{C}_{k+1} = \text{co}(\mathcal{S}_k^{\text{AFW}})$; // Updating feasible set for the accelerated sequence
    $a_k = A_k = 1$, $\mathbf{z}_k = -\nabla f(\mathbf{y}_k) + L\mathbf{y}_k$; // Restarting accelerated sequence
    $\hat{\mathbf{x}}_k = \mathbf{w}_k = \text{argmin}_{\mathbf{u}\in\mathcal{C}_{k+1}}\{-\langle \mathbf{z}_k, \mathbf{u}\rangle + \frac{L}{2}\|\mathbf{u}\|^2\}$;
    $r_c = 0$, $r_f = \text{false}$; // Resetting the restart indicators
  else
    $A_k = A_{k-1}/(1-\theta)$, $a_k = \theta A_k$;
    $\hat{\mathbf{x}}_k, \mathbf{z}_k, \mathbf{w}_k = \text{ACC}(\mathbf{x}_{k-1}, \mathbf{z}_{k-1}, \mathbf{w}_{k-1}, \mu, \mu_0, a_k, A_k, \mathcal{C}_k)$; // Accelerated step
    if $\mathcal{S}_k^{\text{AFW}} \setminus \mathcal{S}_{k-1}^{\text{AFW}} \neq \emptyset$ **then** // Vertex was added to the AFW active set
      $r_f = \text{true}$; // Raise restart flag
    end
    if $r_f = \text{false}$ **then** // If AFW did not add a vertex since last restart
      $\mathcal{C}_{k+1} = \text{co}(\mathcal{S}_k^{\text{AFW}})$; // Update the feasible set
    else
      $\mathcal{C}_{k+1} = \mathcal{C}_k$; // Freeze the feasible set
    end
  end
  $\mathbf{x}_k = \text{argmin}\{f(\mathbf{x}_k^{\text{AFW}}), f(\hat{\mathbf{x}}_k), f(\mathbf{x}_{k-1})\}$; // Choose the better step + monotonicity
  $r_c = r_c + 1$; // Increment the restart counter
end

Algorithm 2: Accelerated Step ACC$(\mathbf{x}_{k-1}, \mathbf{z}_{k-1}, \mathbf{w}_{k-1}, \mu, \mu_0, a_k, A_k, \mathcal{C}_k)$

1 $\theta = a_k / A_k$;
2 $\mathbf{y}_k = \frac{1}{1+\theta}\mathbf{x}_{k-1} + \frac{\theta}{1+\theta}\mathbf{w}_{k-1}$;
3 $\mathbf{z}_k = \mathbf{z}_{k-1} - a_k \nabla f(\mathbf{y}_k) + \mu a_k \mathbf{y}_k$;
4 $\mathbf{w}_k = \operatorname{argmin}_{\mathbf{u}\in\mathcal{C}_k}\{-\langle \mathbf{z}_k, \mathbf{u}\rangle + \frac{\mu A_k + \mu_0}{2}\|\mathbf{u}\|^2\}$;
5 $\hat{\mathbf{x}}_k = (1-\theta)\mathbf{x}_{k-1} + \theta \mathbf{w}_k$;
6 **return** $\hat{\mathbf{x}}_k, \mathbf{z}_k, \mathbf{w}_k$;

The workhorse of our analysis is the following lemma, which shows that it is possible to couple the sequence of steps of accelerated method μAGD+ from [4] with an arbitrary sequence of points, without paying in the convergence rate. It further shows that the convergence is unaffected by any shrinking of the feasible set, as long as it remains convex and it contains the problem solution $\mathbf{x}^*$. Finally, we also allow for inexact projection steps. The analysis relies on the use of the Approximate Duality Gap Technique [5].

Lemma 1 *(Convergence of the modified μAGD+) Given the setting in Problem (P), let $\{\mathcal{C}_i\}_{i=0}^k$ be a sequence of convex subsets of $\mathcal{X}$ such that $\mathcal{C}_i \subseteq \mathcal{C}_{i-1}$ for all i and $\mathbf{x}^* \in \bigcap_{i=0}^k \mathcal{C}_i$. Let $\{\tilde{\mathbf{x}}_i\}_{i=0}^k$ be any (fixed) sequence of points from $\mathcal{X}$. Let $a_0 = 1$, $\frac{a_k}{A_k} = \theta$ for $k \geq 1$, where $A_k = \sum_{i=0}^k a_i$ and $\theta = \sqrt{\frac{\mu}{2L}}$. Let $\mathbf{y}_0 \in \mathcal{X}$, $\mathbf{x}_0 = \mathbf{w}_0$, and $\mathbf{z}_0 = L\mathbf{y}_0 - \nabla f(\mathbf{y}_0)$. For $k \geq 1$, define iterates $\mathbf{x}_k$ by:*

$$\begin{aligned} \hat{\mathbf{x}}_k,\, \mathbf{z}_k,\, \mathbf{w}_k &= \mathrm{ACC}(\mathbf{x}_{k-1}, \mathbf{z}_{k-1}, \mathbf{w}_{k-1}, \mu, \mu_0, a_k, A_k, \mathcal{C}_k) \\ \mathbf{x}_k &= \operatorname{argmin}\{f(\hat{\mathbf{x}}_k),\, f(\tilde{\mathbf{x}}_k)\} \end{aligned} \tag{1}$$

where, for all $k \geq 0$, $\mathbf{w}_k$ is defined as an ϵ_k^m-approximate solution of the projection in Line 4 of Algorithm 2. Then, for all $k \geq 0$:

$$f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq (1-\theta)^k \frac{(L-\mu)\|\mathbf{x}^* - \mathbf{y}_0\|^2}{2} + \frac{2\sum_{i=0}^{k-1}\epsilon_i^m + \epsilon_k^m}{A_k}.$$

Theorem 2 *(Convergence of LaCG) Let $\mathbf{x}_k$ be the solution output by Algorithm 1, $D = \max_{\mathbf{x},\mathbf{y}\in\mathcal{X}}\|\mathbf{x}-\mathbf{y}\|$, and let δ be the pyramidal width of $\mathcal{X}$ (see [15]). If:*

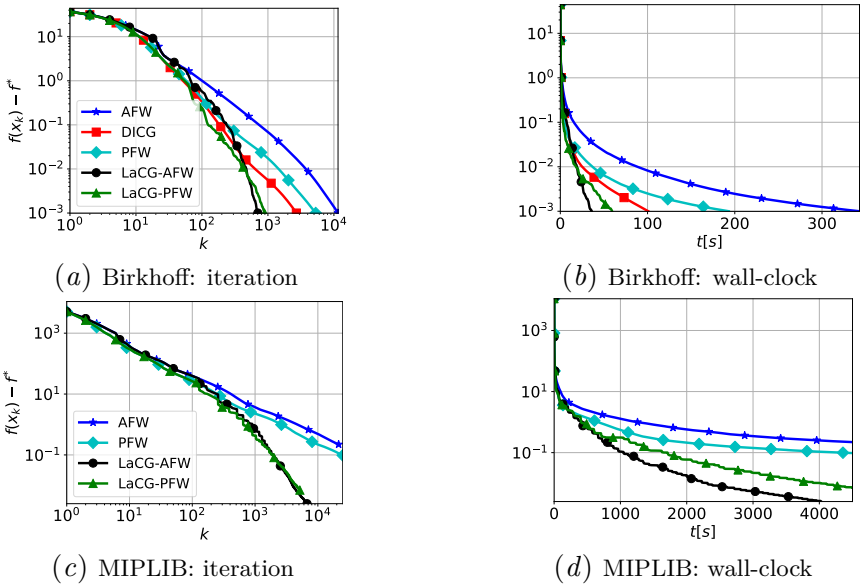
$$k \geq \min\left\{\frac{8L}{\mu}\left(\frac{D}{\delta}\right)^2 \log\left(\frac{f(\mathbf{x}_0) - f(\mathbf{x}^*)}{\epsilon}\right),\; K_0 + H + 2\sqrt{\frac{2L}{\mu}}\log\left(\frac{(L-\mu)r^2}{2\epsilon}\right)\right\},$$

where $H = 2\sqrt{2L/\mu}\log(L/\mu - 1)$ and $K_0 = \frac{8L}{\mu}\left(\frac{D}{\delta}\right)^2 \log\left(\frac{2(f(\mathbf{x}_0) - f(\mathbf{x}^*))}{\mu r^2}\right)$, then:

$$f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \epsilon.$$

3. Numerical Experiments

We implemented Algorithm 1 using Python 3 and numpy, employing the $\mathcal{O}(n \log n)$ projections onto the simplex described in [6, Algorithm 1] and Nesterov's accelerated method [22, 23] to solve the subproblems in Algorithm 1. While we give convergence guarantees for the AFW



(a) Birkhoff: iteration (b) Birkhoff: wall-clock

(c) MIPLIB: iteration (d) MIPLIB: wall-clock

Figure 1: Algorithm comparison in terms of 1(a),1(c) iteration count and 1(b),1(d) wall-clock time for the 1(a),1(b) Birkhoff and 1(c),1(d) MIPLIB examples. In both examples, LaCG coupled with either AFW or PFW exhibits faster convergence than other methods. Transition to faster accelerated convergence can be clearly observed in the plots showing convergence in terms of the iteration count.

variant, as mentioned before, other linearly converging CG variants can be used, as long as they maintain an active set; this excludes e.g., decomposition invariant CG (DICG) [9].

The two examples in Fig. 1 show the convergence both in terms of the iteration count and wall-clock time when solving Problem (P) with $L/\mu \approx 100$. The first example, shown in Fig. 1(a)-1(b), corresponds to minimization over the Birkhoff polytope of dimension $n = 1600$ with $f(\mathbf{x}) = \mathbf{x}^T \frac{M^T M + I}{2} \mathbf{x}$, where $M \in \mathbb{R}^{n \times n}$ is a sparse matrix whose 1% of the elements are drawn from a standard Gaussian distribution and I is the identity matrix (the matrix $M^T M$ has 15% non-zero elements). We compare LaCG (implemented with AFW and PFW) with AFW, PFW, and the DICG algorithm [9]. The second example, shown in Fig. 1(c)-1(d), corresponds to a structured regression problem over the convex hull of the feasible region defined by integer and linear constraints, where the LO corresponds to solving a mixed integer program (MIP) with a linear objective function. The specific instance used corresponds to `ran14x18-disj-8`, of dimension 504, from the MIPLIB library. The objective function $f(\mathbf{x}) = \mathbf{x}^T \frac{M}{2} \mathbf{x} + b$ was obtained by first generating an orthonormal basis $\mathcal{B} = \{\mathbf{u}_1, \cdots, \mathbf{u}_n\}$ in $\mathbb{R}^n$ and a set of n uniformly distributed values $\{\lambda_1, \cdots, \lambda_n\}$ between μ and L and setting $M = \sum_{i=1}^n \lambda_i \mathbf{u}_i \mathbf{u}_i^T$. The vector b was set to be outside of the feasible region, and has random entries uniformly distributed in $[0, 1]$. Note that DICG [1, 9] is not applicable here, as the representation of the MIPLIP polytope mixes continuous and integer variables, so that the away step oracle cannot be readily implemented. In fact, for polytopes over which linear optimization is NP-hard (e.g., TSP polytope), efficiently computing away steps with an away step oracle as in [1] is not possible unless NP = co-NP.

References

[1] Mohammad Ali Bashiri and Xinhua Zhang. Decomposition-invariant conditional gradient for general polytopes with line search. In *Proc. NIPS'17*, 2017.

[2] G. Braun, S. Pokutta, and D. Zink. Lazifying Conditional Gradient Algorithms. In *Proc. ICML'17*, 2017.

[3] Gábor Braun, Sebastian Pokutta, Dan Tu, and Stephen Wright. Blended conditional gradients: the unconditioning of conditional gradients. In *Proc. ICML'19*, 2019. To appear.

[4] Michael B Cohen, Jelena Diakonikolas, and Lorenzo Orecchia. On acceleration with noise-corrupted gradients. In *Proc. ICML'18*, 2018.

[5] Jelena Diakonikolas and Lorenzo Orecchia. The approximate duality gap technique: A unified theory of first-order methods. *SIOPT*, 29(1):660–689, 2019.

[6] John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proc. NIPS'08*, 2008.

[7] Robert M Freund, Paul Grigas, and Rahul Mazumder. An extended Frank-Wolfe method with "in-face" directions, and its application to low-rank matrix completion. *SIOPT*, 27(1):319–346, 2017.

[8] D. Garber and E. Hazan. A linearly convergent variant of the conditional gradient algorithm under strong convexity, with applications to online and stochastic optimization. *SIOPT*, 26(3):1493–1528, 2016.

[9] Dan Garber and Ofer Meshi. Linear-memory and decomposition-invariant linearly convergent conditional gradient algorithm for structured polytopes. In *Proc. NIPS'16*, 2016.

[10] Dan Garber, Shoham Sabach, and Atara Kaplan. Fast generalized conditional gradient method with applications to matrix recovery problems. *arXiv preprint arXiv:1802.05581*, 2018.

[11] Jacques Guélat and Patrice Marcotte. Some comments on Wolfe's 'away step'. *Math. Program.*, 35(1):110–119, 1986.

[12] Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *Proc. ICML'13*, 2013.

[13] Thomas Kerdreux, Alexandre d'Aspremont, and Sebastian Pokutta. Restarting Frank-Wolfe. In *Proc. AISTATS'18*, 2018.

[14] Thomas Kerdreux, Fabian Pedregosa, and Alexandre D'Aspremont. Frank-Wolfe with subsampling oracle. In *Proc. ICML'18*, 2018.

[15] Simon Lacoste-Julien and Martin Jaggi. On the global linear convergence of Frank-Wolfe optimization variants. In *Proc. NIPS'15*, 2015.

[16] G Lan. The complexity of large-scale convex programming under a linear optimization oracle. Technical report, 2013.

[17] Guanghui Lan and Yi Zhou. Conditional gradient sliding for convex optimization. *SIOPT*, 26(2):1379–1409, 2016.

[18] Guanghui Lan, Sebastian Pokutta, Yi Zhou, and Daniel Zink. Conditional accelerated lazy stochastic gradient descent. In *Proc. ICML'17*, 2017.

[19] Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. A universal catalyst for first-order optimization. In *Proc. NIPS'15*, 2015.

[20] F Locatello, A Raj, S Praneeth Karimireddy, G Rätsch, B Schölkopf, SU Stich, and M Jaggi. On Matching Pursuit and Coordinate Descent. In *Proc. ICML'18*, 2018.

[21] BF Mitchell, Vladimir Fedorovich Dem'yanov, and VN Malozemov. Finding the point of a polyhedron closest to the origin. *SIAM Journal on Control*, 12(1):19–26, 1974.

[22] Yu E Nesterov. An $O(1/k)$-rate of convergence method for smooth convex functions minimization. In *Dokl. Acad. Nauk SSSR*, volume 269, pages 543–547, 1983.

[23] Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.