Local Acceleration of Conditional Gradients

IOL-COGA Research Seminar

Alejandro Carderera

Georgia Institute of Technology

alejandro.carderera@gatech.edu

December 3rd, 2020

Georgia Tech H. Milton Stewart School of Industrial and Engineering Systems

Goal is to solve:

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

Where $f(\mathbf{x})$ is a convex function and $\mathcal{X}$ is a compact convex set.
How can we tackle the problem?

1. Projected Newton Method:

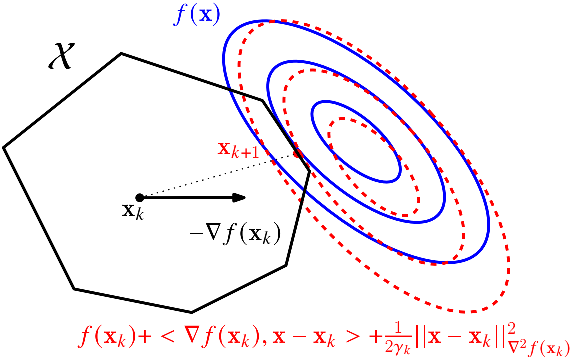
For $t \geq 0$ and $0 < \gamma_t \leq 1$ do:

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}_t) + \langle \nabla f(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle + \frac{1}{2\gamma_t} \|\mathbf{x} - \mathbf{x}_t\|_{\nabla^2 f(\mathbf{x}_t)} .$$

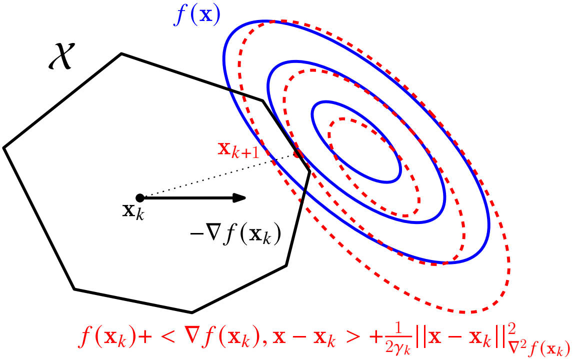
This is equivalent to:

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} \left\| \mathbf{x} - \left(\mathbf{x}_t - \gamma_t [\nabla^2 f(\mathbf{x}_t)]^{-1} \nabla f(\mathbf{x}_t) \right) \right\|^2_{\nabla^2 f(\mathbf{x}_t)} .$$

1. Projected Newton Method:



1. Projected Newton Method:



Downside:

- Computing $\nabla^2 f(\mathbf{x}_t)$ can be very expensive
- Need to solve a quadratic problem over $\mathcal{X}$

2. Projected Gradient Descent:

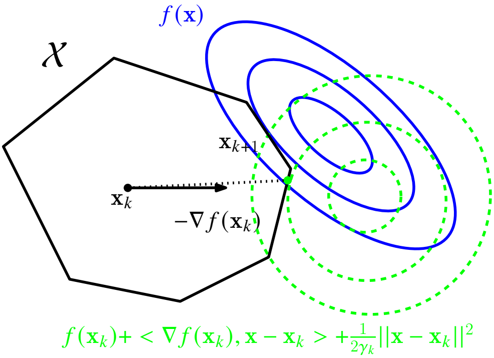
For $t \geq 0$ and $0 < \gamma_t \leq 1$ do:

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}_t) + \langle \nabla f(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle + \frac{1}{2\gamma_t} \|\mathbf{x} - \mathbf{x}_t\|^2$$

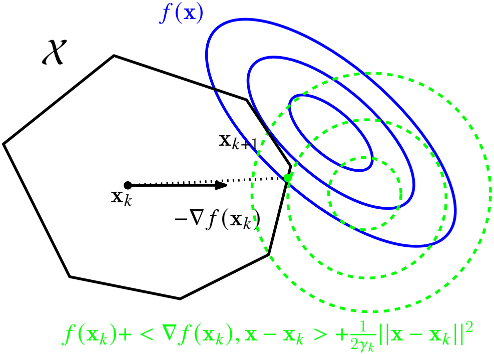
This is equivalent to:

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x} - (\mathbf{x}_t - \gamma_t \nabla f(\mathbf{x}_t))\|^2 .$$

2. Projected Gradient Descent:



2. Projected Gradient Descent:



Downside:

- ~~Computing $\nabla^2 f(\mathbf{x}_t)$ can be very expensive~~
- Need to solve a quadratic problem over $\mathcal{X}$

3. Conditional Gradients (CG) [LP66]:

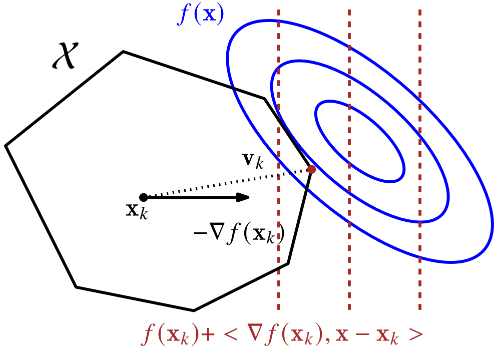
Also known as the Frank-Wolfe (FW) algorithm ([FW56]). For $t \geq 0$ do:

$$\mathbf{v}_{t+1} = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}_t) + \langle \nabla f(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle .$$

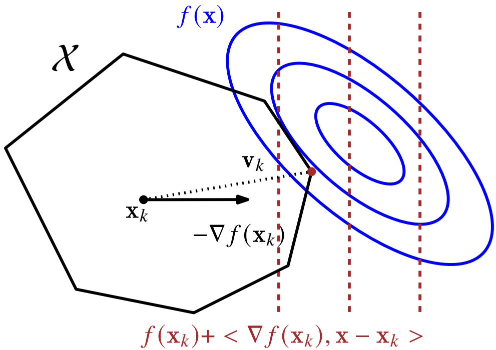
And for some $0 < \gamma_t \leq 1$ take:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \gamma_t (\mathbf{v}_{t+1} - \mathbf{x}_t)$$

3. Conditional Gradients (CG) [LP66]:



3. Conditional Gradients (CG) [LP66]:



Downside:

- ~~Computing $\nabla^2 f(\mathbf{x}_t)$ can be very expensive~~
- ~~Need to solve a quadratic problem over $\mathcal{X}$~~

This leads to the "The Poor Man's Approach to Convex Optimization and Duality" [Jag11]:

Algorithm 1 CG algorithm.

Input: $x_0 \in \mathcal{X}$, stepsizes $\gamma_t \in (0, 1]$.

1: **for** $t = 0$ to T **do**
2: $\mathbf{v}_t = \mathrm{argmin}_{\mathbf{x} \in \mathcal{X}} \langle \nabla f(\mathbf{x}_t), \mathbf{x} \rangle$
3: $\mathbf{x}_{t+1} = \mathbf{x}_t + \gamma_t (\mathbf{v}_t - \mathbf{x}_t)$
4: **end for**

At each iterate we can use:

- First-order (FO) oracle to access $\nabla f(\mathbf{x})$
- Linear optimization (LO) oracle to solve $\mathrm{argmin}_{\mathbf{x} \in \mathcal{X}} \left\langle \vec{}, \mathbf{x} \right\rangle$ for some $\vec{\in} \mathbb{R}^n$

Frank-Wolfe gap.

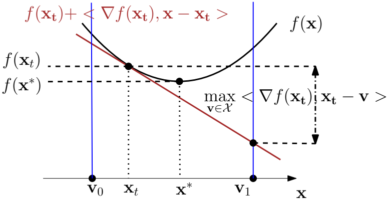
At each iterate we can immediately compute the *Frank-Wolfe-gap* $g(\mathbf{x}_t)$:

$$g(\mathbf{x}_t) \stackrel{\text{def}}{=} \langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{v}_t \rangle = \max_{\mathbf{v} \in \mathcal{X}} \langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{v} \rangle ,$$

an upper bound on the primal gap, which can be used as a stopping criterion when running these algorithms:

$$\begin{aligned} g(\mathbf{x}_t) &= \max_{\mathbf{v} \in \mathcal{X}} \langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{v} \rangle \\ &\geq \langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^* \rangle \\ &\geq f(\mathbf{x}_t) - f(\mathbf{x}^*). \end{aligned}$$

Where the last inequality follows from the convexity of f.



Advantages of CG.

First-order. Dimensionality of modern problems makes computing second-order information infeasible.

Advantages of CG.

First-order. Dimensionality of modern problems makes computing second-order information infeasible.

Projection-free. Projection into certain feasible regions is computationally expensive: Birkhoff polytope and flow polytope are a few examples.

Advantages of CG.

First-order. Dimensionality of modern problems makes computing second-order information infeasible.

Projection-free. Projection into certain feasible regions is computationally expensive: Birkhoff polytope and flow polytope are a few examples.

Sparse solutions. Solution is a convex combination of (a typically sparse set of) extreme points.

Advantages of CG.

First-order. Dimensionality of modern problems makes computing second-order information infeasible.

Projection-free. Projection into certain feasible regions is computationally expensive: Birkhoff polytope and flow polytope are a few examples.

Sparse solutions. Solution is a convex combination of (a typically sparse set of) extreme points.

Stopping criterion. At each iteration the Frank-Wolfe gap gives us an upper bound on the primal gap.

Convergence rate for L-smooth and convex f

Theorem (Primal gap convergence rate of CG/FW)

The CG/FW algorithm using $\gamma_t = 2/(2+t)$ converges at a rate of $f(\mathbf{x}_t) - f(\mathbf{x}^*) = \mathcal{O}(1/t)$ [FW56; DH78]. Moreover, the Frank-Wolfe gap satisfies $\min_{0 \leq t \leq T} g(\mathbf{x}_t) = \mathcal{O}(1/t)$ for $T \geq 1$ [Jag13].

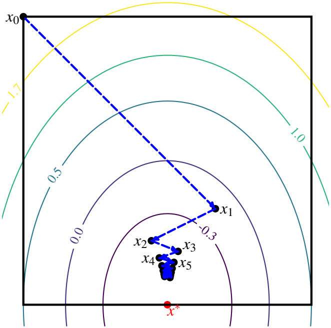
The aforementioned primal gap convergence rate is optimal for the class of algorithms that only add a single vertex at each iteration [Jag13; Lan13].

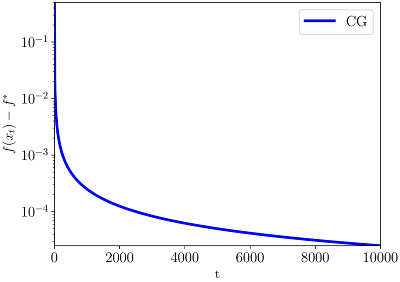
What about L-smooth and μ-strongly convex f?

In general: **Sublinear convergence**.

Example (CG Convergence.)

L-smooth and μ-strongly convex f with $x \in \mathbb{R}^2$, and x^* in boundary of $\mathcal{X}$ using line search.





Linear convergence when $\mathcal{X}$ is a polytope is achieved by allowing steps that decrease the weight of *bad* vertices [GH15]. This has led to various CG variants:

Linear convergence when $\mathcal{X}$ is a polytope is achieved by allowing steps that decrease the weight of *bad* vertices [GH15]. This has led to various CG variants:

Away-step Conditional Gradients (ACG)

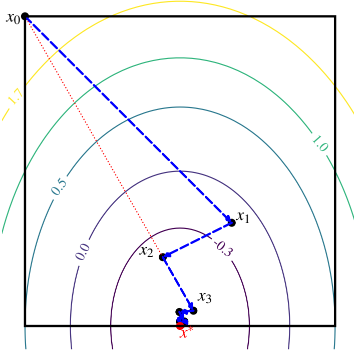


Figure: Away-step CG (ACG)

Allow steps in the direction of:

$$\mathbf{x}_t - \operatorname*{argmax}_{\mathbf{u} \in \mathcal{S}} \langle \nabla f(\mathbf{x}_t), \mathbf{u} \rangle,$$

where $\mathcal{S}$ is the active set of $\mathbf{x}_t$.

Pairwise-step Conditional Gradients (PCG)

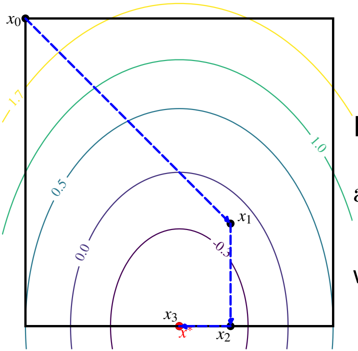


Figure: Pairwise-step CG

Move along:

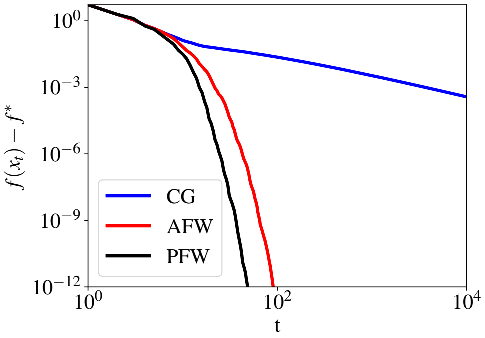
$$\operatorname*{argmin}_{\mathbf{v}\in\mathcal{X}} \langle \nabla f(\mathbf{x}_t), \mathbf{v}\rangle - \operatorname*{argmax}_{\mathbf{u}\in\mathcal{S}} \langle \nabla f(\mathbf{x}_t), \mathbf{u}\rangle ,$$

where $\mathcal{S}$ is the active set of $\mathbf{x}_t$.

Convergence rate for L-smooth μ-strongly convex f.

Theorem (Convergence rate of ACG and PCG.)

If $\mathcal{X}$ is a polytope, then the ACG and PCG algorithms with line search satisfy that $f(\mathbf{x}_t) - f(\mathbf{x}^*) = \mathcal{O}\left(1 - \frac{\mu}{L}\left(\frac{\delta}{D}\right)^2\right)^{k(t)}$ [LJ15] where D and δ are the diameter and pyramidal width of the polytope $\mathcal{X}$



CG Global Acceleration

However, we know that optimal methods for this class of functions achieve an ϵ solution in $T = \mathcal{O}\left(\sqrt{\frac{L}{\mu}} \log \frac{1}{\epsilon}\right)$ first-order calls [NY83; Nes83].

Can CG achieve these convergence rates **globally**?

CG Global Acceleration

However, we know that optimal methods for this class of functions achieve an ϵ solution in $T = \mathcal{O}\left(\sqrt{\frac{L}{\mu}} \log \frac{1}{\epsilon}\right)$ first-order calls [NY83; Nes83].

Can CG achieve these convergence rates **globally**?

Dimension independent global acceleration is not possible [Jag13; Lan13].

Conditional Gradient Sliding

Idea: Run Nesterov's Accelerated Gradient Descent, use CG to solve the projection subproblems approximately [LZ16].

Conditional Gradient Sliding

Idea: Run Nesterov's Accelerated Gradient Descent, use CG to solve the projection subproblems approximately [LZ16].

Results:

- Separate LO and FO oracle calls.
- Globally optimal $\mathcal{O}\left(\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right)$ calls to FO and $\mathcal{O}\left(\frac{LD^2}{\epsilon}+\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right)$ calls to LO oracles.
- Convergence rates independent of the dimension n.

Catalyst Augmented ACG.

Idea: Run Accelerated Proximal Method and solve proximal problems with a linearly convergent CG [LMH15].

Catalyst Augmented ACG.

Idea: Run Accelerated Proximal Method and solve proximal problems with a linearly convergent CG [LMH15].

Results:

- $\mathcal{O}\left(\sqrt{\frac{L-\mu}{\mu}}\left(\frac{D}{\delta}\right)^2 \log\frac{1}{\epsilon}\right)$ Calls to FO and LO oracles.
- Convergence rates dependent of the dimension n.

Summary

Complexity for L-smooth μ-strongly convex f.

Algorithm	LO Calls	FO Calls
CG Variants	$\mathcal{O}\left(\frac{L}{\mu}\left(\frac{D}{\delta}\right)^2 \log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{L}{\mu}\left(\frac{D}{\delta}\right)^2 \log\frac{1}{\epsilon}\right)$
CGS	$\mathcal{O}\left(\frac{LD^2}{\epsilon} + \sqrt{\frac{L}{\mu}} \log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\sqrt{\frac{L}{\mu}} \log\frac{1}{\epsilon}\right)$
Catalyst	$\mathcal{O}\left(\sqrt{\frac{L-\mu}{\mu}}\left(\frac{D}{\delta}\right)^2 \log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\sqrt{\frac{L-\mu}{\mu}}\left(\frac{D}{\delta}\right)^2 \log\frac{1}{\epsilon}\right)$

Objectives:

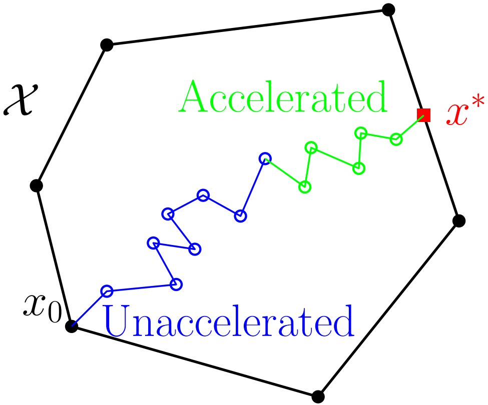
- Dimension independent global acceleration.

Objectives:

- ~~Dimension independent global acceleration.~~
- Dimension independent local acceleration.

Locally Accelerated Conditional Gradients (LaCG).

What do we mean by **local acceleration**?



After a constant number of iterations, accelerate the convergence.

Locally Accelerated Conditional Gradients (LaCG).

The key ingredients is the *Approximate Duality Gap* technique [DO19] and a *Modified μAGD* algorithm [CDO18; DCP20].

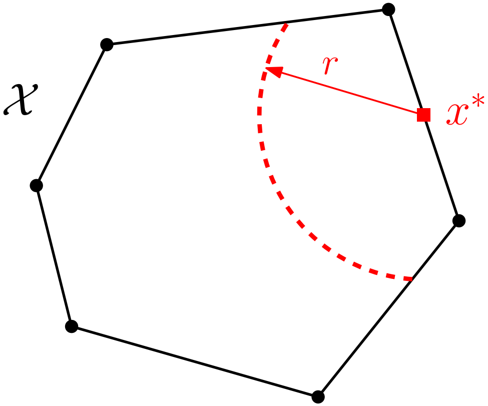
Theorem (Convergence rate of μAGD.)

Let f be L-smooth and μ-strongly convex and let $\{C_i\}_{i=0}^t$ be a sequence of convex subsets of $\mathcal{X}$ such that $C_i \subseteq C_{i-1}$ for all i and $x^* \in \cap_{i=0}^t C_i$, then the μAGD achieves an ϵ-optimal solution in:

$$T = \mathcal{O}\left(\sqrt{\frac{L}{\mu}} \log \frac{1}{\epsilon}\right)$$

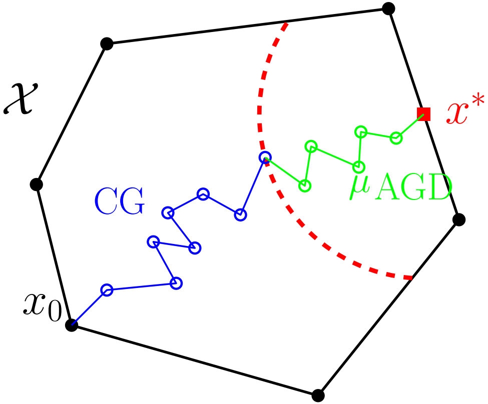
How do we build $\{C_i\}_{i=0}^t$ in an efficient way?

There exists an $r > 0$ (that depends only on f and $\mathcal{X}$) s.t. if $\|x^* - x_K\| \leq r \Rightarrow x^* \in conv\left(\mathcal{S}_t\right)$ for all $t \geq K$, where $\mathcal{S}_t$ is the active set at iteration t.

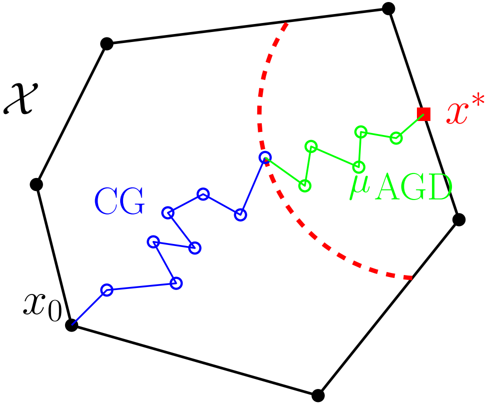


So when we are inside the red semicircle and we use $C_t = \mathcal{S}_t$, acceleration is possible.

Naively, what we would like:

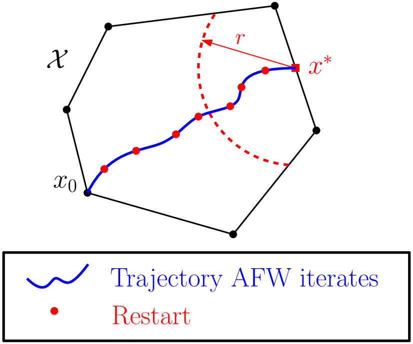


Naively, what we would like:

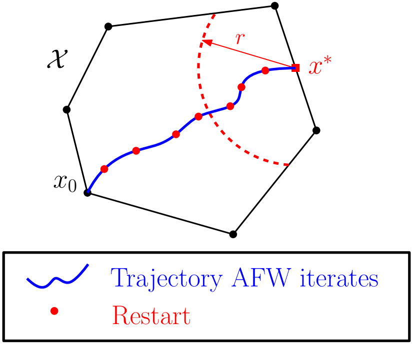


But since the value of r is not known, we don't know when to switch from CG to μAGD.

Run ACG and restart AGD by running it over a new conv $(\mathcal{S}_t)$ every H iterations.

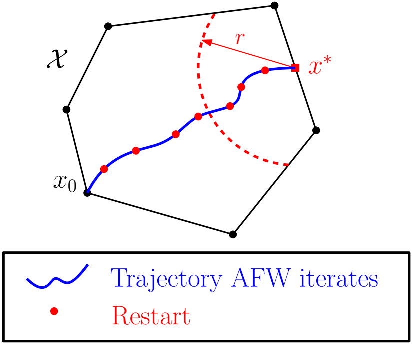


Run ACG and restart AGD by running it over a new conv $(\mathcal{S}_t)$ every H iterations.



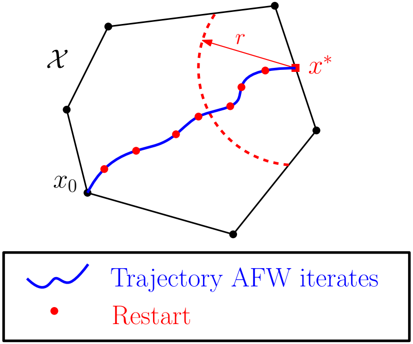
- Every H iterations restart AGD and run it over conv $(\mathcal{S}_t)$.

Run ACG and restart AGD by running it over a new conv $(\mathcal{S}_t)$ every H iterations.



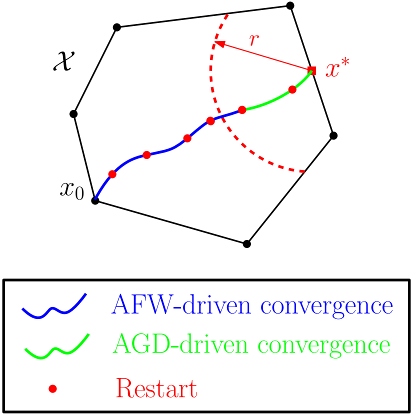
- Every H iterations restart AGD and run it over conv $(\mathcal{S}_t)$.
- Have AGD and ACG compete for progress at each iteration between restarts.

Run ACG and restart AGD by running it over a new conv $(\mathcal{S}_t)$ every H iterations.



- Every H iterations restart AGD and run it over conv $(\mathcal{S}_t)$.
- Have AGD and ACG compete for progress at each iteration between restarts.
- Space out restarts so that you only loose a factor of 2 in the AGD convergence rate.

What we will obtain:



Locally Accelerated Conditional Gradients (LaCG)

Algorithm 2 Locally Accelerated Conditional Gradients

```
Initialize C_0 = S_0, x_0 = x_0^{ACG} = x_0^{AGD}, H = O( sqrt(L/μ) log(L/μ) )
for t = 1 to T do
  x_{t+1}^{ACG}, S_{t+1} ← ACG(x_t^{ACG}, S_t)                              ▷ ACG step
  if Vertex has been added to S since restart then
    if t = Hn for some n ∈ ℕ then
      x_{t+1}^{AGD} ← argmin_{x ∈ {x_t^{ACG}, x_t^{AGD}}} f(x)              ▷ Restart AGD
      C_{t+1} ← Update based on previous line.
    else
      x_{t+1}^{AGD} ← AGD(x_t^{AGD}, C_t)                   ▷ Run AGD decoupled from ACG
      C_{t+1} ← C_t
    end if
  else
    x_{t+1}^{AGD} ← AGD(x_t, C_t)                             ▷ Run AGD coupled with ACG
    C_{t+1} ← conv(S_{t+1})
  end if
  x_{t+1} ← argmin_{x ∈ {x_{t+1}^{ACG}, x_{t+1}^{AGD}, x_t}} f(x)           ▷ Monotonicity
end for
```

Convergence rate of LaCG

Theorem (Convergence rate of LaCG)

Let f be L-smooth and μ-strongly convex and let r be the critical radius. The number of steps T required to reach an ϵ-optimal solution to the minimization problem satisfies:

$$t = min \left\{ \mathcal{O}\left(\frac{L}{\mu}\left(\frac{D}{\delta}\right)^2 \log\frac{1}{\epsilon}\right), K + \mathcal{O}\left(\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right)\right\},$$

where $K = \frac{8L}{\mu}\left(\frac{D}{\delta}\right)^2 \log\left(\frac{2(f(x_0)-f^*)}{\mu r^2}\right)$.

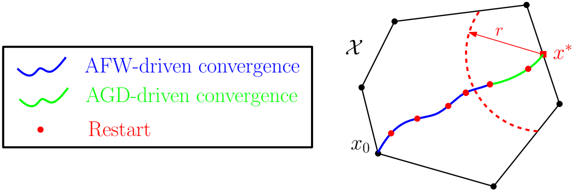
Convergence rate of LaCG

Theorem (Convergence rate of LaCG)

Let f be L-smooth and μ-strongly convex and let r be the critical radius. The number of steps T required to reach an ϵ-optimal solution to the minimization problem satisfies:

$$t = min \left\{ \mathcal{O}\left(\frac{L}{\mu} \left(\frac{D}{\delta} \right)^2 \log \frac{1}{\epsilon} \right), K + \mathcal{O}\left(\sqrt{\frac{L}{\mu}} \log \frac{1}{\epsilon} \right) \right\},$$

where $K = \frac{8L}{\mu} \left(\frac{D}{\delta} \right)^2 \log \left(\frac{2(f(x_0) - f^*)}{\mu r^2} \right)$.



Computational Results.

Despite the faster convergence rate after the burn-in phase, how does LaCG perform with respect to other projection-free algorithms?

Simplex in $\mathbb{R}^{1500}$ with $L/\mu = 1000$.

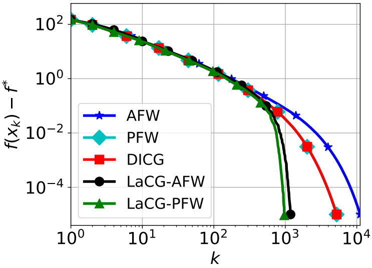


Figure: Primal gap vs. iteration

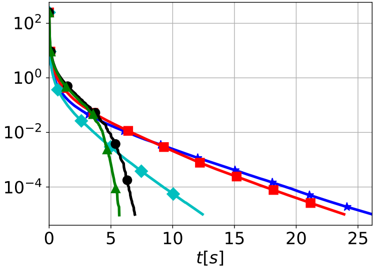


Figure: Primal gap vs. time

When close enough to $x*$ (after burn-in phase), there is a significant speedup in the convergence rate.

Birkhoff polytope in $\mathbb{R}^{400x400}$ with $L/\mu = 100$.

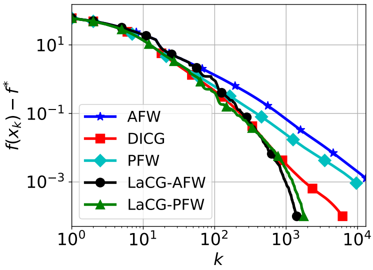


Figure: Primal gap vs. iteration

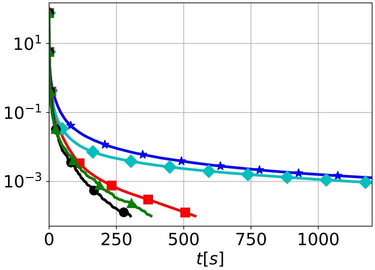


Figure: Primal gap vs. time

Structured Regression over MIPLIB Polytope (`ran14x18-disj-8`).

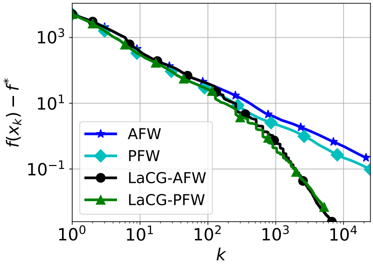


Figure: Primal gap vs. iteration

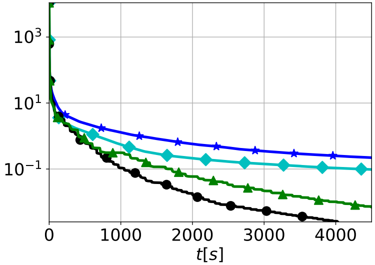


Figure: Primal gap vs. time

Congestion Balancing in Traffic Networks.

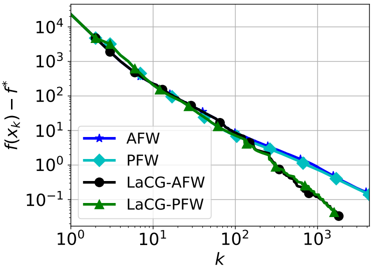


Figure: Primal gap vs. iteration

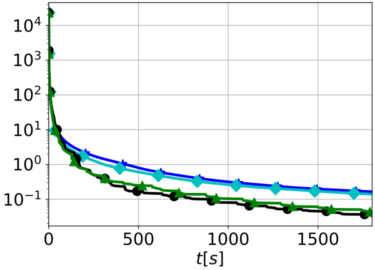


Figure: Primal gap vs. time

Joint work with Jelena Diakonikolas and Sebastian Pokutta. See Locally Accelerated Conditional Gradients in International Conference on Artificial Intelligence and Statistics (2020) for more details.

Problem setting

Consider the case where:

- Computing $\nabla f(\mathbf{x})$ and $\nabla^2 f(\mathbf{x})$, although possible, is expensive.
- There is no access to a stochastic oracle for $\nabla f(\mathbf{x})$.
- The feasible region is a polytope $\mathcal{X}$

Unfortunately, zeroth-order algorithms (those that only use function value oracles) are not efficient in high dimensions, and so we must try to make as much primal progress as possible per first and second order oracle call.

Background

Conditional Gradient Sliding: Run Nesterov's Accelerated Gradient Descent, use CG to solve the projection subproblems approximately [LZ16].

Background

Conditional Gradient Sliding: Run Nesterov's Accelerated Gradient Descent, use CG to solve the projection subproblems approximately [LZ16].

Idea: Why not use ACG to **approximately solve the scaled-projection subproblems in Newton's method with unit step size**? That is, compute:

$$\begin{aligned}\mathbf{x}_{t+1} &= \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}_t) + \langle \nabla f(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle + \frac{1}{2} \|\mathbf{x} - \mathbf{x}_t\|_{\nabla^2 f(\mathbf{x}_t)} \\ &= \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}} \left\| \mathbf{x} - \left(\mathbf{x}_t - [\nabla^2 f(\mathbf{x}_t)]^{-1} \nabla f(\mathbf{x}_t) \right) \right\|^2_{\nabla^2 f(\mathbf{x}_t)} .\end{aligned}$$

Why?: If these scaled projections are computed exactly, the steps contract $\|\mathbf{x}_t - \mathbf{x}^*\|$ quadratically once close enough to the optimum.

What we want:

- Global linear convergence in primal gap
- Local quadratic convergence in primal gap
- Use of inexact second-order oracles, use H_k, an approximation to $\nabla^2 f(\mathbf{x}_k)$

What we want:

- Global linear convergence in primal gap
- Local quadratic convergence in primal gap
- Use of inexact second-order oracles, use H_k, an approximation to $\nabla^2 f(\mathbf{x}_k)$

Template:

- Compute an ε_k-optimal scaled projection (Newton step with unit step size) using ACG
- Compute an independent ACG step with line search
- Take the iterate with lowest function value

Assumptions

Accuracy of the Hessian oracle:
The oracle Ω queried with a point $\mathbf{x}_k$ returns a matrix H_k with a parameter $\eta = \max\{\lambda_{\max}(H_k^{-1}\nabla^2 f(\mathbf{x}_k)), \lambda_{\max}([\nabla^2 f(\mathbf{x}_k)]^{-1} H_k)\}$ such that:

$$\frac{\eta - 1}{\|\mathbf{x}_k - \mathbf{x}^*\|^2} \leq \omega,$$

where $\omega \geq 0$ denotes a known constant.

Assumptions

Accuracy of the Hessian oracle:
The oracle Ω queried with a point $\mathbf{x}_k$ returns a matrix H_k with a parameter $\eta = \max\{\lambda_{\max}(H_k^{-1}\nabla^2 f(\mathbf{x}_k)), \lambda_{\max}([\nabla^2 f(\mathbf{x}_k)]^{-1}H_k)\}$ such that:

$$\frac{\eta - 1}{\|\mathbf{x}_k - \mathbf{x}^*\|^2} \leq \omega,$$

where $\omega \geq 0$ denotes a known constant.

Lower bound on the primal gap:
We compute ε_k using a lower bound on the primal gap that satisfies $lb\,(\mathbf{x}_k) \leq f\,(\mathbf{x}_k) - f\,(\mathbf{x}^*)$.

Assumptions

Accuracy of the Hessian oracle:
The oracle Ω queried with a point $\mathbf{x}_k$ returns a matrix H_k with a parameter $\eta = \max\{\lambda_{\max}(H_k^{-1}\nabla^2 f(\mathbf{x}_k)), \lambda_{\max}([\nabla^2 f(\mathbf{x}_k)]^{-1} H_k)\}$ such that:

$$\frac{\eta - 1}{\|\mathbf{x}_k - \mathbf{x}^*\|^2} \leq \omega,$$

where $\omega \geq 0$ denotes a known constant.

Lower bound on the primal gap:
We compute ε_k using a lower bound on the primal gap that satisfies $lb\left(\mathbf{x}_k\right) \leq f\left(\mathbf{x}_k\right) - f\left(\mathbf{x}^*\right)$.

Strict Complementarity:
We have that $\langle \nabla f\left(\mathbf{x}^*\right), \mathbf{x} - \mathbf{x}^* \rangle = 0$ if and only if $\mathbf{x} \in \mathcal{F}\left(\mathbf{x}^*\right)$, where $\mathcal{F}\left(\mathbf{x}^*\right)$ is the minimal face that contains $\mathbf{x}^*$.

Algorithm 3 Second-order Conditional Gradient Sliding Algorithm

$\mathbf{x}_0, \mathbf{x}_0^{\text{ACG}} \leftarrow \operatorname{argmin}_{\mathbf{v}\in\mathcal{X}} \langle \nabla f(\mathbf{x}), \mathbf{v}\rangle$

$\mathcal{S}_{k+1}^{\text{ACG}} \leftarrow \{\mathbf{x}_0\}$

for $t = 1$ to T **do**

$\mathbf{x}_{k+1}^{\text{ACG}}, \mathcal{S}_{k+1}^{\text{ACG}} \leftarrow \text{ACG}\left(\mathbf{x}_k^{\text{ACG}}, \mathcal{S}_k^{\text{ACG}}\right)$ ▷ ACG step

$\hat{f}_k(\mathbf{x}) \leftarrow \langle \nabla f(\mathbf{x}_k), \mathbf{x} - \mathbf{x}_k\rangle + \frac{1}{2}\|\mathbf{x} - \mathbf{x}_k\|_{H_k}^2$ ▷ Quadratic Approximation

$\varepsilon_k \leftarrow \left(\frac{lb(\mathbf{x}_k)}{\|\nabla f(\mathbf{x}_k)\|}\right)^4$

Find $\tilde{\mathbf{x}}_{k+1}$ such that $\max_{\mathbf{v}\in\mathcal{X}} \langle \nabla \hat{f}_k(\tilde{\mathbf{x}}_{k+1}), \tilde{\mathbf{x}}_{k+1} - \mathbf{v}\rangle < \varepsilon_k$ using ACG ▷ Minimize $\hat{f}_k$

if $f(\tilde{\mathbf{x}}_{k+1}) \leq f(\mathbf{x}_{k+1}^{\text{ACG}})$ **then**

$\mathbf{x}_{k+1} \leftarrow \tilde{\mathbf{x}}_{k+1}$ ▷ Choose PVM step

else

$\mathbf{x}_{k+1} \leftarrow \mathbf{x}_{k+1}^{\text{ACG}}$ ▷ Choose ACG step

end if

end for

Convergence rate of SOCGS

Theorem (Convergence rate of SOCGS)

Let f be L-smooth and μ-strongly convex and $\mathcal{X}$ be a polytope. Under the assumptions given before, the SOCGS algorithm achieves a ε-optimal solution after $\mathcal{O}\left(\log\log 1/\varepsilon\right)$ first and second order oracle calls and $\mathcal{O}\left(\log\left(1/\varepsilon\right)\log\log 1/\varepsilon\right)$ linear oracle calls, after a burn-in phase independent of ε.

Informal proof sketch:

- The inexact Newton steps converge quadratically in distance to the optimum.
- After a finite number of iterations, both the ACG and Newton iterations are contained in $\mathcal{F}^*$
- Using smoothness and strong convexity one can show that then the quadratic rate in distance to the optimum is a quadratic rate in primal gap.

Computational Results.

Sparse coding over the Birkhoff polytope in $\mathbb{R}^{80\times 80}$ with 100000 samples.

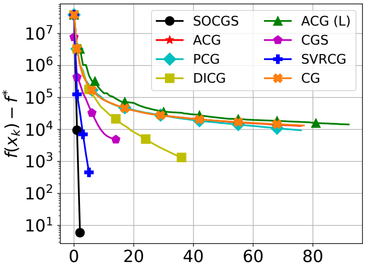


Figure: Primal gap vs. iteration

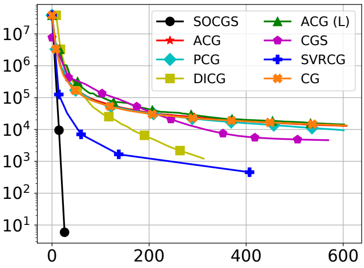


Figure: Primal gap vs. time

Logistic regression over the ℓ_1 ball in $\mathbb{R}^{5000}$.

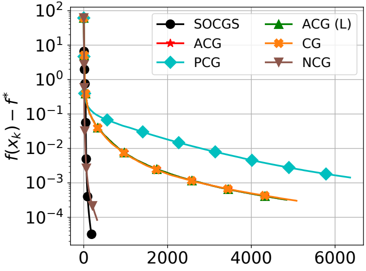


Figure: Primal gap vs. iteration

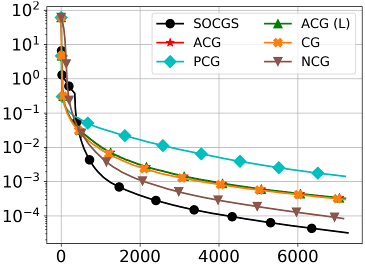


Figure: Primal gap vs. time

Inverse covariance estimation over the spectrahedron in $\mathbb{R}^{50\times50}$.

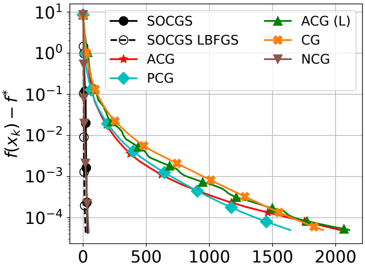


Figure: Primal gap vs. iteration

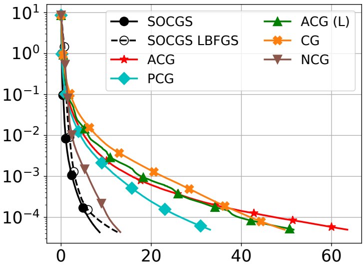


Figure: Primal gap vs. time

Joint work with Sebastian Pokutta. See Second-order Conditional Gradient Sliding on arXiv for the full details.

Thank you
for your attention.

References I

[FW56] Marguerite Frank and Philip Wolfe. "An algorithm for quadratic programming". In: *Naval research logistics quarterly* 3.1-2 (1956), pp. 95–110.

[LP66] E. S. Levitin and B. T. Polyak. "Constrained minimization methods". In: *USSR Computational Mathematics and Mathematical Physics* 6.5 (1966), pp. 1–50.

[Jag11] Martin Jaggi. "Sparse convex optimization methods for machine learning". PhD thesis. ETH Zurich, 2011.

[DH78] Joseph C Dunn and S Harshbarger. "Conditional gradient algorithms with open loop step size rules". In: *Journal of Mathematical Analysis and Applications* 62.2 (1978), pp. 432–444.

References II

[Jag13] Martin Jaggi. "Revisiting Frank-Wolfe: Projection-free sparse convex optimization". In: *Proceedings of the 30th international conference on machine learning*. CONF. 2013, pp. 427–435.

[Lan13] Guanghui Lan. "The complexity of large-scale convex programming under a linear optimization oracle". In: *arXiv preprint arXiv:1309.5550* (2013).

[GH15] Dan Garber and Elad Hazan. "Faster rates for the frank-wolfe method over strongly-convex sets". In: *32nd International Conference on Machine Learning, ICML 2015*. 2015.

[LJ15] Simon Lacoste-Julien and Martin Jaggi. "On the Global Linear Convergence of Frank-Wolfe Optimization Variants". In: *Advances in Neural Information Processing Systems 28*. 2015, pp. 496–504.

References III

[NY83] Arkadii Semenovich Nemirovsky and David Borisovich Yudin. "Problem complexity and method efficiency in optimization". In: *Wiley-Interscience Series in Discrete Mathematics* 15 (1983).

[Nes83] Y Nesterov. "A method of solving a convex programming problem with convergence rate $O(\frac{1}{k^2})$". In: *Soviet Math. Dokl*. Vol. 27. 1983.

[LZ16] Guanghui Lan and Yi Zhou. "Conditional gradient sliding for convex optimization". In: *SIAM Journal on Optimization* 26.2 (2016), pp. 1379–1409.

[LMH15] Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. "A universal catalyst for first-order optimization". In: *Advances in neural information processing systems*. 2015, pp. 3384–3392.

References IV

[DO19] Jelena Diakonikolas and Lorenzo Orecchia. "The approximate duality gap technique: A unified theory of first-order methods". In: *SIAM Journal on Optimization* 29.1 (2019), pp. 660–689.

[CDO18] Michael B Cohen, Jelena Diakonikolas, and Lorenzo Orecchia. "On acceleration with noise-corrupted gradients". In: *arXiv preprint arXiv:1805.12591* (2018).

[DCP20] Jelena Diakonikolas, Alejandro Carderera, and Sebastian Pokutta. "Locally accelerated conditional gradients". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 1737–1747.