Projection-Free First Order Optimization

2020 INFORMS Annual Meeting

Alejandro Carderera

Georgia Institute of Technology

alejandro.carderera@gatech.edu

November 11th, 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:

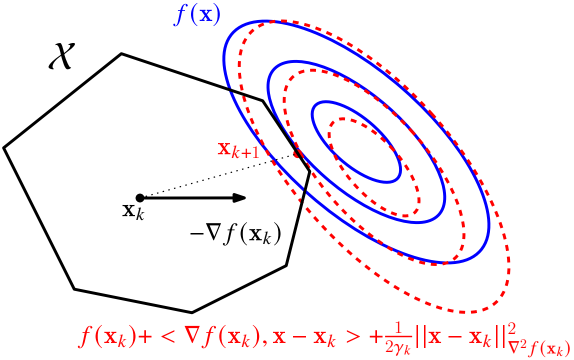
For $t \geq 0$ and $0 < \gamma_t \leq 1$ do:

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

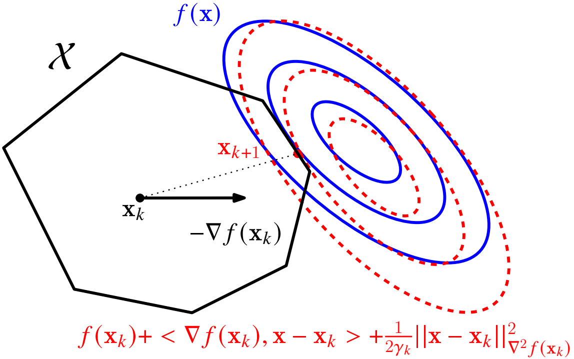
This is equivalent to:

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

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:

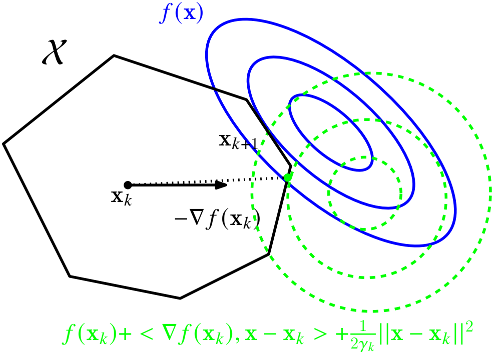
For $t \geq 0$ and $0 < \gamma_t \leq 1$ do:

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

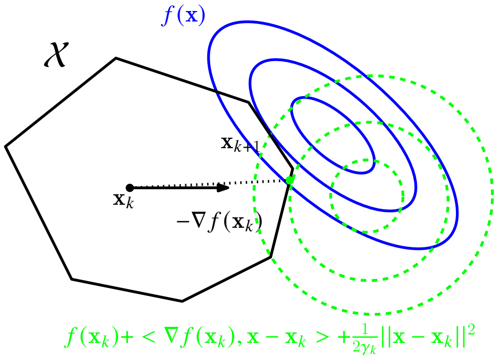
This is equivalent to:

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

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]:

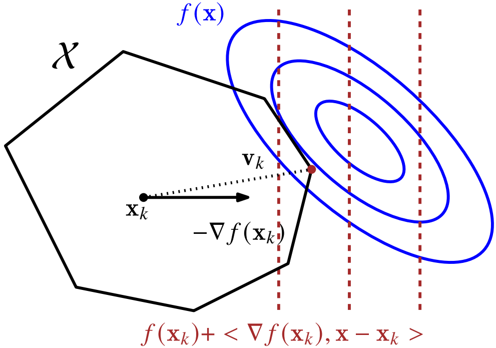
Also known as the Frank-Wolfe (FW) algorithm ([FW56]). For $t \geq 0$ do:

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

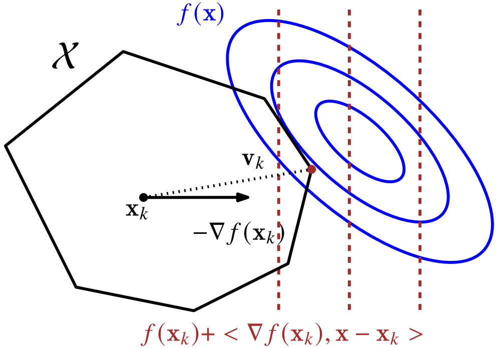
And for some $0 < \gamma_t \leq 1$ take:

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

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 immediately compute the *Frank-Wolfe-gap* $g(\mathbf{x}_t)$:

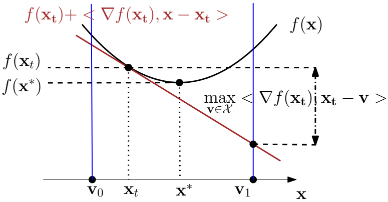
$$g(\mathbf{x}_t) \stackrel{\text{def}}{=} \langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{v}_t \rangle = \max_{\mathbf{v} \in \mathcal{X}} \langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{v} \rangle$$

Frank-Wolfe gap.

The Frank-Wolfe gap is an upper bound on the primal gap, and can therefore 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 $\min_{0 \le t \le T} g(\mathbf{x}_t) = \mathcal{O}(1/t)$ for $T \ge 1$ [Jag13].

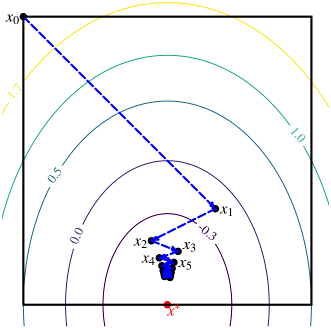
The aforementioned primal gap rate is optimal for the class of algorithms that only add a single vertex at each iteration [Jag13; Lan13].

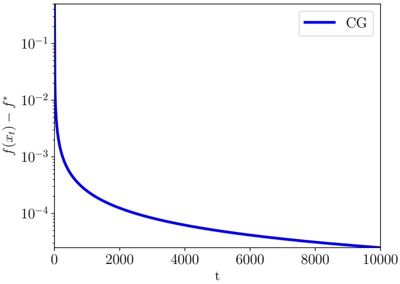
What about L-smooth and μ-strongly convex f?

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)

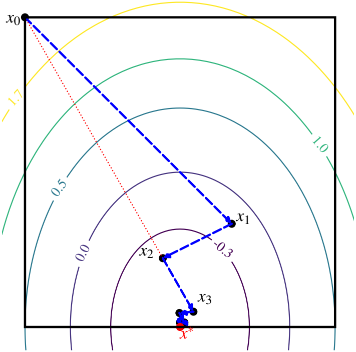


Figure: Away-step CG (ACG)

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)

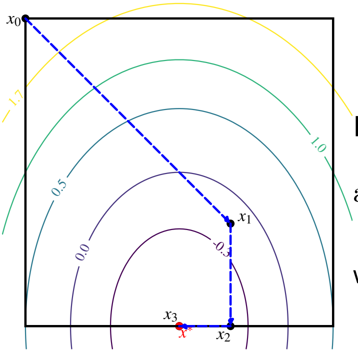


Figure: Pairwise-step CG

Move along:

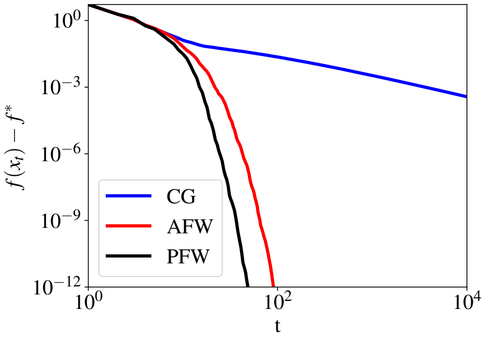
$$\underset{\mathbf{v}\in\mathcal{X}}{\operatorname{argmin}} \langle \nabla f(\mathbf{x}_t), \mathbf{v}\rangle - \underset{\mathbf{u}\in\mathcal{S}}{\operatorname{argmax}} \langle \nabla f(\mathbf{x}_t), \mathbf{u}\rangle ,$$

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 $\mathcal{X}$



Video Co-localization.

Objective

Given a set of videos, locate with bounding boxes an object that is present in the frames. It can be used to generate data from weakly-labelled videos.

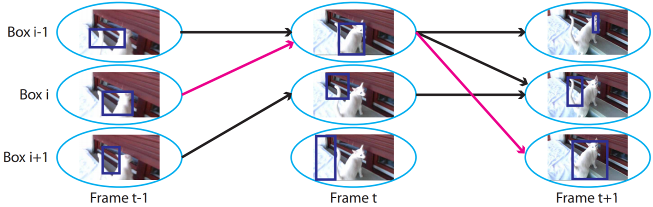
(a) Frame t_{i-1} (b) Frame t_i (c) Frame t_{i+1}

Formulation sketch

1. Generate a series of bounding boxes for each frame
2. Compute a temporal similarity metric between all the bounding boxes in frames t_i and t_{i+1}, for all i
3. Build a directed graph using the bounding boxes as nodes. For every bounding box at time t_i, connect it with a weighted edge to all the bounding boxes at time t_{i+1} where the weight is given by the similarity metric.
4. Get rid of edges with similarity weight below a given threshold.
5. Construct a convex quadratic function that encodes the *temporal* and *spatial* similarity of the bounding boxes.

Video Co-localization.

Find the path between the first and last frame that maximizes this quadratic:



A relaxation of the previous problem can be formulated as:

$$\min_{\mathbf{x} \in \mathcal{X}} \langle \mathbf{x}, Q\mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{x} \rangle ,$$

where $\mathcal{X}$ is the convex hull of the vertices of the flow polytope, Q is a symmetric positive semi-definite matrix, and $\mathbf{b}$ is a vector.

Matrix Completion.

Why use a CG/FW algorithm?
Solving an LP over the flow polytope is equivalent to solving a shortest path problem.

Matrix Completion.

Objective

Given a matrix $Y \in \mathbb{R}^{n\times m}$, assume we only observe a subset of all its entries, denoted by $\mathcal{I} \subseteq \{(i,j) \| 1 \leq i \leq m, 1 \leq j \leq n\}$. Find a low rank matrix $X \in \mathbb{R}^{n\times m}$ that approximates Y (useful in recommendations systems). A convex surrogate of the previous problem can be phrased

$$\min_{\|X\|_{\mathrm{nuc}} \leq \tau} \frac{1}{\|\mathcal{I}\|} \sum_{(i,j)\in\mathcal{I}} (Y_{i,j} - X_{i,j})^2,$$

where $\|X\|_{\mathrm{nuc}}$ denotes the nuclear norm of X, which is equal to the sum of the singular values of X.

Matrix Completion.

Why use a CG/FW algorithm?
Computing a projection onto the nuclear norm ball requires computing a full SVD decomposition of the matrix X, whereas solving a linear minimization problem over the nuclear norm ball requires computing only the top left and right singular vectors!

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.

[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.

References II

[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.