Optimization for Data Science
ETH Zürich, FS 2023 261-5110-00L

Lecture 7: The Frank-Wolfe Algorithm

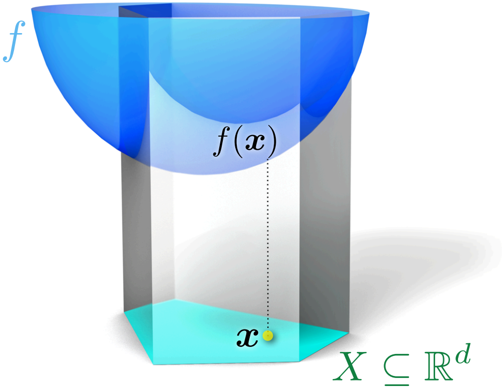
Bernd Gärtner
Niao He

https://www.ti.inf.ethz.ch/ew/courses/ODS23/index.html

March 27, 2023

Constrained optimization reloaded

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in X \end{array}$$



Algorithm	Nontrivial primitive
Projected gradient descent	projection onto X
Frank-Wolfe algorithm	linear optimization over X

In many cases, linear optimization is easier / faster than projection.

The Frank-Wolfe Algorithm

History:

- Discovered by Marguerite Frank and Philip Wolfe in 1956 [FW56].
- After the second world war, linear programming (minimize a linear function over set of linear constraints) had significant impact for many industrial applications.
- Frank and Wolfe studied if similar methods could be generalized to non-linear objectives and constraints, in particular to quadratic programming.

The Primitive and the Algorithm

Linear minimization oracle: Given $\mathbf{g} \in \mathbb{R}^d$,

$$\mathrm{LMO}_X(\mathbf{g}) := \operatorname*{argmin}_{\mathbf{z} \in X} \; \mathbf{g}^\top \mathbf{z}$$

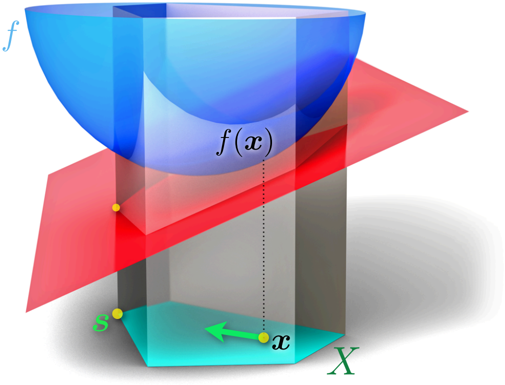
is any minimizer of the linear function $\mathbf{g}^\top \mathbf{z}$ over X.

We assume that a minimizer exists whenever we apply the oracle. If X is closed and bounded, this is guaranteed.

Algorithm: Given an initial feasible point $\mathbf{x}_0 \in X$, and (time-dependent) stepsizes $\gamma_t \in [0,1]$, repeat the following for $t = 0, 1, \ldots$:

$$\begin{aligned} \mathbf{s} &:= \mathrm{LMO}_X(\nabla f(\mathbf{x}_t)), \\ \mathbf{x}_{t+1} &:= (1-\gamma_t)\mathbf{x}_t + \gamma_t \mathbf{s}. \end{aligned}$$

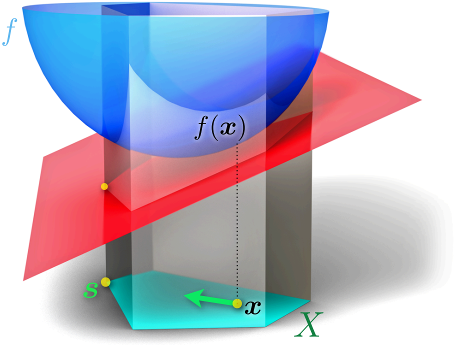
The Frank-Wolfe algorithm, visually



Minimize the linear approximation of f over X.

Make a step towards the minimizer.

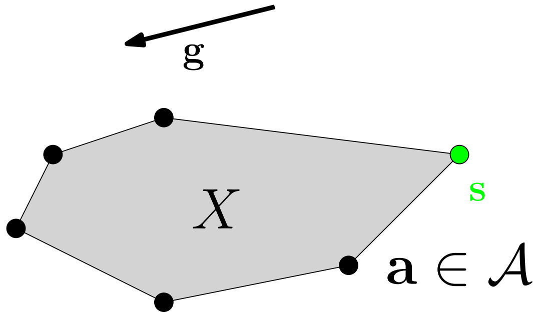
Attractive features



- Iterates are always feasible, if the constraint set X is convex. In other words, $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in X$.
- The algorithm is projection-free. Primitive LMO_X is often easier to implement than projection onto X (two examples will follow).
- Iterates have a simple sparse representation: $\mathbf{x}_t$ is a convex combination of the initial iterate and the minimizers $\mathbf{s}$ used so far.

Linear minimization oracles: Atoms

The Frank-Wolfe algorithm is particularly useful when X is the convex hull of a finite or otherwise "nice" set of points $\mathcal{A}$ (the atoms), $X = \mathrm{conv}(\mathcal{A})$.



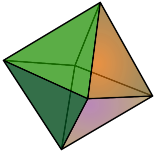
- $\mathrm{LMO}_X(\mathbf{g}) = \mathrm{argmin}_{\mathbf{z} \in X} \ \mathbf{g}^\top \mathbf{z}$ is always attained by some atom.
- This may significantly simplify the search for $\mathbf{s} = \mathrm{LMO}_X(\mathbf{g})$.

Linear minimization oracles   Example: ℓ_1-ball

LASSO (in standard primal form):

$$\min_{\mathbf{x}\in\mathbb{R}^d} \|A\mathbf{x}-\mathbf{b}\|^2 \quad \text{subject to} \quad \|\mathbf{x}\|_1 \le 1$$

$X = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\|_1 \le 1\}$, the cross polytope (atoms $= \{\pm\mathbf{e}_i\}$)



Projection (Section 4.5):

- $O(d \log d)$ time (not obvious)
- Improvement to $O(d)$ (nontrivial)

LMO (easy in $O(d)$ time):

$$\begin{aligned}
\mathrm{LMO}_X(\mathbf{g}) &= \operatorname*{argmin}_{\mathbf{z}\in X} \mathbf{z}^\top\mathbf{g} \\
&= \operatorname*{argmin}_{\mathbf{z}\in\{\pm\mathbf{e}_1,\ldots,\pm\mathbf{e}_n\}} \mathbf{z}^\top\mathbf{g} \\
&= -\mathrm{sgn}(g_i)\mathbf{e}_i \quad \text{with} \quad i := \operatorname*{argmax}_{i\in[d]} |g_i|
\end{aligned}$$

Linear minimization oracles   Example: Spectahedron

Hazan's algorithm [Haz08]: an application of the Frank-Wolfe algorithm to semidefinite programming.

$\mathrm{LMO}_X(G)$:

$$\begin{array}{ll} \text{argmin} & G \bullet Z \\ \text{subject to} & \mathrm{Tr}(Z) = 1 \\ & Z \succeq 0. \end{array}$$

- X is the spectahedron, the set of all (symmetric) positive semidefinite matrices $Z \in \mathbb{R}^{d \times d}$ of trace 1.
- G is a symmetric matrix.
- $A \bullet B$ stands for the "scalar product" of two square matrices A and B, $A \bullet B = \sum_{i,j} a_{ij} b_{ij}$.

The LMO is a semidefinite program itself, but of a simple form that allows an explicit solution.

Linear minimization oracles Example: Spectahedron (Atoms)

Spectahedron: $X = \{Z \in \mathbb{R}^{d\times d} : \mathrm{Tr}(Z) = 1, Z \succeq 0\}$

Atoms:

- The matrices of the form $\mathbf{z}\mathbf{z}^\top$ with $\mathbf{z} \in \mathbb{R}^d$, $\|\mathbf{z}\| = 1$ (these are positive semidefinite of trace 1 and hence in X).

Need to show: every $Z \in X$ is a convex combination of atoms.

- diagonalize: $Z = TDT^\top$ where T is orthogonal and D is diagonal, of trace 1.
- D's diagonal elements $\lambda_1, \ldots, \lambda_d$ are the (nonnegative) eigenvalues of Z.
- Let $\mathbf{a}_i$ be the i-th column of T. As T is orthogonal, we have $\|\mathbf{a}_i\| = 1$.
- $Z = \sum_{i=1}^d \lambda_i \mathbf{a}_i \mathbf{a}_i^\top$ is the desired convex combination of atoms.

Linear minimization oracles Example: Spectahedron (LMO)

Spectahedron: $X = \{Z \in \mathbb{R}^{d\times d} : \mathrm{Tr}(Z) = 1, Z \succeq 0\}$

$$\mathrm{LMO}_X(G) = \begin{array}{ll} \text{argmin} & G \bullet Z \\ \text{subject to} & Z \in X \end{array}$$

Lemma 7.1

Let λ_1 be the smallest eigenvalue of G, and let $\mathbf{s}_1$ be a corresponding eigenvector of unit length. Then we can choose $\mathrm{LMO}_X(G) = \mathbf{s}_1\mathbf{s}_1^\top$.

Proof.

$$\min_{\mathrm{Tr}(Z)=1, Z\succeq 0} G \bullet Z \overset{\text{(atoms)}}{=} \min_{\|\mathbf{z}\|=1} G \bullet \mathbf{z}\mathbf{z}^\top \overset{\text{(rewrite)}}{=} \min_{\|\mathbf{z}\|=1} \mathbf{z}^\top G\mathbf{z} \overset{\text{(linear algebra)}}{=} \lambda_1.$$

The eigenvector $\mathbf{s}_1$ is easily seen to attain the last minimum, hence the atom $\mathbf{s}_1\mathbf{s}_1^\top$ attains the first minimum. $\mathrm{LMO}_X(G) = \mathbf{s}_1\mathbf{s}_1^\top$ follows. □

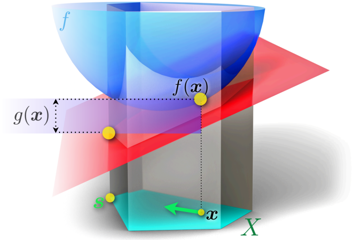
The duality gap

Even if $f(\mathbf{x}^\star)$ is unknown, $\mathrm{LMO}_X(\mathbf{g})$ gives us an upper bound for the optimality gap $f(\mathbf{x}_t) - f(\mathbf{x}^\star)$ (see next slide).

Duality gap:

$$g(\mathbf{x}) := \nabla f(\mathbf{x})^\top (\mathbf{x} - \mathbf{s}) \quad \text{for} \quad \mathbf{s} := \mathrm{LMO}_X(\nabla f(\mathbf{x})).$$

- $g(\mathbf{x}) \geq 0$ is the optimality gap $\nabla f(\mathbf{x})^\top \mathbf{x} - \nabla f(\mathbf{x})^\top \mathbf{s}$ of the linear subproblem.



Function value of the linear approximation

- at $\mathbf{x}$: $f(\mathbf{x})$
- at $\mathbf{s}$: $f(\mathbf{x}) + \nabla f(\mathbf{x})^\top (\mathbf{s} - \mathbf{x}) = f(\mathbf{x}) - g(\mathbf{x})$

The duality gap bound

Lemma 7.2

Suppose that the constrained minimization problem $\min\{f(\mathbf{x}) : \mathbf{x} \in X\}$ has a minimizer $\mathbf{x}^\star$. Let $\mathbf{x} \in X$. Then

$$g(\mathbf{x}) \geq f(\mathbf{x}) - f(\mathbf{x}^\star),$$

meaning that the duality gap is an upper bound for the optimality gap.

Proof.

Using that $\mathbf{s}$ minimizes $\nabla f(\mathbf{x})^\top \mathbf{z}$ over X, we argue that

$$\begin{aligned} g(\mathbf{x}) &= \nabla f(\mathbf{x})^\top (\mathbf{x} - \mathbf{s}) \\ &\geq \nabla f(\mathbf{x})^\top (\mathbf{x} - \mathbf{x}^\star) \\ &\geq f(\mathbf{x}) - f(\mathbf{x}^\star). \end{aligned}$$

In the last inequality we have used the first-order characterization of convexity of f (Lemma 2.16). $\square$

Convergence in $\mathcal{O}(1/\varepsilon)$ steps

Standard stepzise in the Frank-Wolfe algorithm: $\gamma_t = 2/(t+2)$.

We need to assume that f is smooth, but the smoothness parameter L does not enter the stepsize.

it is infinitely differentiable, or that it has derivatives of all orders at every point in its domain.

Theorem 7.3

Consider the constrained minimization problem $\min\{f(\mathbf{x}) : \mathbf{x} \in X\}$ where $f : \mathbb{R}^d \to \mathbb{R}$ is convex and smooth with parameter L, and set X is convex, closed and bounded (in particular, a minimizer $\mathbf{x}^\star$ of f over X exists, and all linear minimization oracles have minimizers). With any $\mathbf{x}_0 \in X$, and with stepsizes $\gamma_t = 2/(t+2)$, the Frank-Wolfe algorithm yields

$$f(\mathbf{x}_T) - f(\mathbf{x}^\star) \leq \frac{2L\,\mathrm{diam}(X)^2}{T+1}, \quad T \geq 1,$$

where $\mathrm{diam}(X) := \max_{\mathbf{x},\mathbf{y}\in X} \|\mathbf{x}-\mathbf{y}\|$ is the diameter of X (which exists since X is closed and bounded).

Main proof ingredient: The descent lemma

Lemma 7.4

For a step $\mathbf{x}_{t+1} := \mathbf{x}_t + \gamma_t(\mathbf{s} - \mathbf{x}_t)$ with stepsize $\gamma_t \in [0, 1]$, it holds that

$$f(\mathbf{x}_{t+1}) \leq f(\mathbf{x}_t) - \gamma_t g(\mathbf{x}_t) + \gamma_t^2 \frac{L}{2} \|\mathbf{s} - \mathbf{x}_t\|^2,$$

where $\mathbf{s} = \mathrm{LMO}_X(\nabla f(\mathbf{x}_t))$.

Proof.

From the definition of smoothness of f, we have

$$\begin{aligned} f(\mathbf{x}_{t+1}) &= f(\mathbf{x}_t + \gamma_t(\mathbf{s} - \mathbf{x}_t)) \\ &\leq f(\mathbf{x}_t) + \nabla f(\mathbf{x}_t)^\top \gamma_t(\mathbf{s} - \mathbf{x}_t) + \gamma_t^2 \frac{L}{2} \|\mathbf{s} - \mathbf{x}_t\|^2 \\ &= f(\mathbf{x}_t) - \gamma_t g(\mathbf{x}_t) + \gamma_t^2 \frac{L}{2} \|\mathbf{s} - \mathbf{x}_t\|^2, \end{aligned}$$

using the definition of the duality gap. $\square$

Convergence bound proof

Descent lemma: $f(\mathbf{x}_{t+1}) \leq f(\mathbf{x}_t) - \gamma_t g(\mathbf{x}_t) + \gamma_t^2 \frac{L}{2} \|\mathbf{s} - \mathbf{x}_t\|^2$.

Writing $h(\mathbf{x}) := f(\mathbf{x}) - f(\mathbf{x}^\star)$ for the (unknown) optimization gap at point $\mathbf{x}$, und using $h(\mathbf{x}) \leq g(\mathbf{x})$ (Lemma 7.4), the descent lemma implies that

$$
\begin{aligned}
h(\mathbf{x}_{t+1}) &\leq h(\mathbf{x}_t) - \gamma_t g(\mathbf{x}_t) + \gamma_t^2 \frac{L}{2} \|\mathbf{s} - \mathbf{x}_t\|^2 \\
&\leq h(\mathbf{x}_t) - \gamma_t h(\mathbf{x}_t) + \gamma_t^2 \frac{L}{2} \|\mathbf{s} - \mathbf{x}_t\|^2 \\
&= (1 - \gamma_t) h(\mathbf{x}_t) + \gamma_t^2 \frac{L}{2} \|\mathbf{s} - \mathbf{x}_t\|^2 \\
&\leq (1 - \gamma_t) h(\mathbf{x}_t) + \gamma_t^2 C,
\end{aligned}
$$

where $C := \frac{L}{2} \operatorname{diam}(X)^2$.

Result follows by induction (Exercise 47):

$$
f(\mathbf{x}_t) - f(\mathbf{x}^\star) = h(\mathbf{x}_t) \leq \frac{4C}{t+1}, \quad t \geq 1.
$$

Stepsize variants

Runtime analysis also holds for two alternative stepsizes.

In practice, convergence might even be faster with these alternatives, since they are trying to optimize progress, in two different ways.

Line search stepsize:

$$\gamma_t := \operatorname*{argmin}_{\gamma \in [0,1]} f\big((1-\gamma)\mathbf{x}_t + \gamma \mathbf{s}\big).$$

Let $\mathbf{y}_{t+1}$ be the iterate obtained from $\mathbf{x}_t$ with the standard stepsize $\mu_t = 2/(t+2)$. We return to the previous analysis:

$$h(\mathbf{x}_{t+1}) \leq h(\mathbf{y}_{t+1}) \leq (1-\mu_t)h(\mathbf{x}_t) + \mu_t^2 C.$$

Proof finishes as before by induction.

Stepsize variants

Gap-based stepsize: choose γ_t such that the term $-\gamma_t g(\mathbf{x}_t) + \gamma_t^2 \frac{L}{2} \left\| \mathbf{s} - \mathbf{x}_t \right\|^2$ on the right-hand side of the inequality for $h(\mathbf{x}_{t+1})$ a is minimized.

$$\gamma_t := \min \left(\frac{g(\mathbf{x}_t)}{L \left\| \mathbf{s} - \mathbf{x}_t \right\|^2}, 1 \right).$$

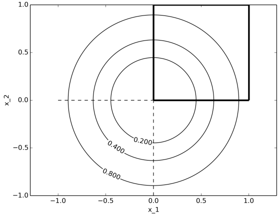
We now return to the previous analysis as follows:

$$\begin{aligned}
h(\mathbf{x}_{t+1}) &\leq h(\mathbf{x}_t) - \gamma_t g(\mathbf{x}_t) + \gamma_t^2 \frac{L}{2} \left\| \mathbf{s} - \mathbf{x}_t \right\|^2 \\
&\leq h(\mathbf{x}_t) - \mu_t g(\mathbf{x}_t) + \mu_t^2 \frac{L}{2} \left\| \mathbf{s} - \mathbf{x}_t \right\|^2 \\
&\leq h(\mathbf{x}_t) - \mu_t h(\mathbf{x}_t) + \mu_t^2 \frac{L}{2} \left\| \mathbf{s} - \mathbf{x}_t \right\|^2 \\
&\leq (1 - \mu_t) h(\mathbf{x}_t) + \mu_t^2 C.
\end{aligned}$$

Proof finishes as before by induction.

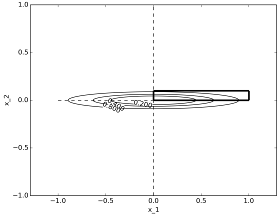
Affine invariance?

Convergence bound:
$$f(\mathbf{x}_T) - f(\mathbf{x}^\star) \leq \frac{2L\,\mathrm{diam}(X)^2}{T+1}.$$



Scenario 1:

- minimize $f(x_1, x_2) = x_1^2 + x_2^2$ over the unit square $X = \{(x_1, x_2) : 0 \leq x_1 \leq 1, 0 \leq x_2 \leq 1\}$.
- $L = 2$ (supermodel), and $\mathrm{diam}(X)^2 = 2$.



Scenario 2:

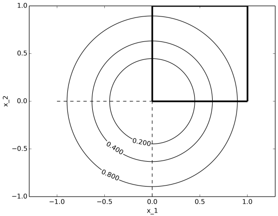
- minimize $f'(x_1, x_2) = x_1^2 + (10x_2)^2$ over the rectangle $X' = \{(x_1, x_2) : 0 \leq x_1 \leq 1, 0 \leq x_2 \leq 1/10\}$.
- $L' = 200$ and $\mathrm{diam}(X')^2 = 1 + 1/100$.

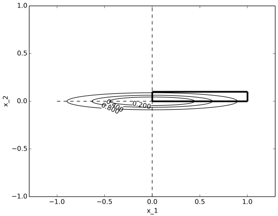
Is the algorithm indeed around 100 times slower on (f', X') than on (f, X)?

Affine invariance!

No difference in runtime! The Frank-Wolfe algorithm is invariant under all affine transformations of space.

(f, X) and (f', X') are called affinely equivalent if $f'(\mathbf{x}) = f(A\mathbf{x} + \mathbf{b})$ for some invertible matrix A and some vector $\mathbf{b}$, and $X' = \{A^{-1}(\mathbf{x} - \mathbf{b}) : \mathbf{x} \in X\}$.





$$(f, X) \quad A = \begin{pmatrix} 1 & 0 \\ 0 & 10 \end{pmatrix}, \mathbf{b} = \mathbf{0} \quad (f', X')$$

We have $\mathbf{x} \in X$ with function value $f(\mathbf{x})$ if and only if $\mathbf{x}' = A^{-1}(\mathbf{x} - \mathbf{b}) \in X'$ with the same function value $f'(\mathbf{x}') = f(AA^{-1}(\mathbf{x} - \mathbf{b}) + \mathbf{b}) = f(\mathbf{x})$.

Affine invariance of the Frank-Wolfe algorithm

Let (f, X) and (f', X') be affinely equivalent as before.

The points $\mathbf{x}$ and $\mathbf{x}' = A^{-1}(\mathbf{x} - \mathbf{b}) \in X'$ are said to correspond to each other.

Chain rule: $\nabla f'(\mathbf{x}') = A^\top \nabla f(A\mathbf{x}' + \mathbf{b}) = A^\top \nabla f(\mathbf{x})$.

Now consider performing an iteration of the Frank-Wolfe algorithm

(a) on (f, X), starting from some iterate $\mathbf{x}$, and
(b) on (f', X'), starting from the corresponding iterate $\mathbf{x}'$,

in both cases with the same stepsize.

Corresponding linear function values:

$$\nabla f'(\mathbf{x}')^\top \mathbf{z}' = \nabla f(\mathbf{x})^\top A A^{-1}(\mathbf{z} - \mathbf{b}) = \nabla f(\mathbf{x})^\top \mathbf{z} - c,$$

c some constant.

Corresponding steps: $\mathbf{s} = \mathrm{LMO}_X(\nabla f(\mathbf{x}))$ if and only if $\mathbf{s}' = \mathrm{LMO}_{X'}(\nabla f'(\mathbf{x}'))$.

The curvature constant

A good analysis of the Frank-Wolfe algorithm should provide a bound that is invariant under affine transformations, unlike the bound of Theorem 7.3.

Curvature constant (notion of complexity of (f, X)):

$$C_{(f,X)} := \sup_{\substack{\mathbf{x},\mathbf{s}\in X,\gamma\in(0,1] \\ \mathbf{y}=(1-\gamma)\mathbf{x}+\gamma\mathbf{s}}} \frac{1}{\gamma^2}\big(f(\mathbf{y}) - f(\mathbf{x}) - \nabla f(\mathbf{x})^\top(\mathbf{y}-\mathbf{x})\big).$$

Observation (arguments as for the algorithm): the curvature constant is affine invariant, i.e. if (f, X) and (f', X') are affinely equivalent, then $C_{(f,X)} = C_{(f',X')}$.

Theorem 7.5

Consider the constrained minimization problem $\min\{f(\mathbf{x}) : \mathbf{x} \in X\}$ where $f : \mathbb{R}^d \to \mathbb{R}$ is convex, X is convex, closed and bounded. Let $C_{(f,X)}$ be the curvature constant of f over X. With $\mathbf{x}_0 \in X$, and with stepsizes $\gamma_t = 2/(t+2)$, the Frank-Wolfe algorithm yields

$$f(\mathbf{x}_T) - f(\mathbf{x}^\star) \le \frac{4C_{(f,X)}}{T+1}, \quad T \ge 1.$$

Proof of convergence in terms of the curvature constant

Crucial step: prove the following version of the decrease lemma (featuring $C_{(f,X)}$ instead of $\frac{L}{2}\|\mathbf{s}-\mathbf{x}_t\|^2$):

$$f(\mathbf{x}_{t+1}) \leq f(\mathbf{x}_t) - \nabla f(\mathbf{x}_t)^\top \gamma_t(\mathbf{s}-\mathbf{x}) + \gamma_t^2 C_{(f,X)}. \tag{1}$$

After this, we can follow the remainder of the proof of Theorem 7.3, with $C_{(f,X)}$ instead of the upper bound $C = \frac{L}{2}\mathrm{diam}(X)^2$ on $\frac{L}{2}\|\mathbf{s}-\mathbf{x}_t\|^2$.

Towards (1), define

$$\mathbf{x} := \mathbf{x}_t, \quad \mathbf{y} := \mathbf{x}_{t+1} = (1-\gamma_t)\mathbf{x}_t + \gamma_t\mathbf{s}, \quad \mathbf{y}-\mathbf{x} = -\gamma_t(\mathbf{x}-\mathbf{s}).$$

Rewrite the definition of the curvature constant to get

$$f(\mathbf{y}) \leq f(\mathbf{x}) + \nabla f(\mathbf{x})^\top(\mathbf{y}-\mathbf{x}) + \gamma_t^2 C_{(f,X)}.$$

Plugging in the previous definitions of $\mathbf{x}$ and $\mathbf{y}$, (1) follows.

How good is the bound from the curvature constant?

Is there a price to pay for an affinely independent analysis? No, the new bound is always at least as good as the previous bound!

Lemma 7.6 (Exercise 48)

Let f be a convex function which is smooth with parameter L over X. Then

$$C_{(f,X)} \leq \frac{L}{2}\operatorname{diam}(X)^2.$$

Convergence in duality gap

[Jag13, Theorem 2]

Let $f : \mathbb{R}^d \to \mathbb{R}$ be convex and smooth with parameter L, and $\mathbf{x}_0 \in X$, $T \geq 2$. Then choosing any of the three stepsizes that we have discussed, the Frank-Wolfe algorithm guarantees some $t, 1 \leq t \leq T$ such that

$$g(\mathbf{x}_t) \leq \frac{27/2 \cdot C_{(f,X)}}{T+1}, \quad T \geq 2.$$

The smallest value $g(\mathbf{x}_t), t = 1, \ldots, T$ bounds the optimality gap at iteration t:

$$f(\mathbf{x}_t) - f(\mathbf{x}^\star) \leq g(\mathbf{x}_t) \leq \frac{27/2 \cdot C_{(f,X)}}{T+1}.$$

This is a computable bound that certifies small optimality gap!

Sparsity

Convergence bound of Theorem 7.5:

$$f(\mathbf{x}_T) - f(\mathbf{x}^\star) \leq \frac{4C_{(f,X)}}{T+1}, \quad T \geq 1.$$

$O(1/\varepsilon)$ many iterations are sufficent to obtain optimality gap at most ε.

At this time, the current solution is a convex combination of $\mathbf{x}_0$ and $O(1/\varepsilon)$ many atoms of the constraint set X.

Thinking of ε as a constant (such as 0.01): constantly many atoms are sufficient in order to get an almost optimal solution.

This connects to the notion of coresets in computational geometry.

Coreset: a small subsets of a given set of objects that is representative (with respect to some measure) for the set of all objects.

Some algorithms for finding small coresets are variants of or inspired by the Frank-Wolfe algorithm [Cla10].

Extensions

Approximate LMO:

- use a linear minimization oracle which is not exact but is of a certain additive or multiplicative approximation quality. Essentially, everything still works [Jag13].

Randomized LMO:

- solve the linear minimization oracle only over a random subset of X; Convergence in $O(1/\varepsilon)$ steps still holds [KPd18].

Stochastic LMO:

- LMO_X is fed with a stochastic gradient (unbiased estimator of the true gradient). Still $O(1/\varepsilon)$ steps [HL20].

Unconstrained problems:

- This is achieved by considering growing versions of a constraint set X [LKTJ17].

Use cases

Lasso and other L1-constrained problems, as discussed in Section 7.3.1.

Matrix Completion. For several low-rank approximation problems, including matrix completion as in recommender systems, the Frank-Wolfe algorithm is a very scalable algorithm, and has much lower iteration cost compared to projected gradient descent. For a more formal treatment, see Exercise 50.

Bibliography

Kenneth L. Clarkson.
Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm.
ACM Trans. Algorithms, 6(4), sep 2010.

Marguerite Frank and Philip Wolfe.
An algorithm for quadratic programming.
Naval Research Logistics Quarterly, 3(1-2):95–110, 1956.

Elad Hazan.
Sparse approximate solutions to semidefinite programs.
In Eduardo Sany Laber, Claudson Bornstein, Loana Tito Nogueira, and Luerbio Faria, editors, *LATIN 2008: Theoretical Informatics*, pages 306–316, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

Robert M. Freund Haihao Lu.
Generalized stochastic Frank-Wolfe algorithm with stochastic substitute gradient for structured convex optimization.
Mathematical Programming, pages 317–349, 2020.

Martin Jaggi.
Revisiting Frank-Wolfe: Projection-free sparse convex optimization.
In *ICML - International Conference on Machine Learning*, pages 427–435, 2013.

Thomas Kerdreux, Fabian Pedregosa, and Alexandre d'Aspremont.
Frank-wolfe with subsampling oracle, 2018.

Francesco Locatello, Rajiv Khanna, Michael Tschannen, and Martin Jaggi.
A Unified Optimization View on Generalized Matching Pursuit and Frank-Wolfe.
In *AISTATS - Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *PMLR*, pages 860–868, 2017.