Frank-Wolfe algorithm: introduction

Andersen Ang

Department of Combinatorics and Optimization,
University of Waterloo, Waterloo, Canada

msxang@uwaterloo.ca, where $\mathbf{x} = \lfloor \pi \rfloor$
Homepage: angms.science

First draft: November 15, 2021 Last update: November 15, 2021

Introduction

- What is Frank-Wolfe (FW) algorithm?
  A first-order algorithm for solving constrained convex optimization.

- What are the other names of FW algorithm?
  Conditional gradient, convex commination algorithm

- For problem we can solve by FW algorithm, what is the alternative method?
  Projected gradient descent (PGD). Or in other words, for problem that can be solved by PGD, we can also solve it by FW.

- As FW is closely related to PGD, so to understand better about FW, we first review about PGD and constrained convex optimization.

Problem setup: constrained convex optimization

- Consider the following problem

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

where

- $f : \mathbb{R}^n \to \mathbb{R}$ is the cost function
- f is assumed to be L-smooth and convex
- $\mathcal{C} \subseteq \mathbb{R}^n$ is the constrained set
- $\mathcal{C}$ is assumed to be convex, closed and compact
- $\mathbf{x} \in \mathbb{R}^n$ is the optimization variable
- When $\mathbf{x} \in \mathcal{C}$, the variable is feasible. Otherwise, the variable is infeasible.
- We are interested in designing an iterative algorithm that solve problem (P) by producing an optimal and feasible solution $\mathbf{x}^*$.

Projected gradient descent

- For solving the problem

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

one way is to use the Projected Gradient Descent (PGD):

Algorithm 1: Projected gradient descent

Result: $\mathbf{x}^*$ that solves (P)

Initialize $\mathbf{x}_0 \in \mathcal{C}$;

for $k = 1, 2, \ldots$ **do**

  $\mathbf{y}_{k+1} = \mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k)$; `// Gradient step`

  $\mathbf{x}_{k+1} = \mathrm{proj}_{\mathcal{C}}\big(\mathbf{y}_{k+1}\big)$; `// Projection step`

end

where the projection step consists of solving an optimization subproblem

$$\mathrm{proj}_{\mathcal{C}}(\mathbf{y}) \coloneqq \operatorname*{argmin}_{\mathbf{u}\in\mathcal{C}} \frac{1}{2}\|\mathbf{u}-\mathbf{y}\|_2^2$$

- PGD in one line: $\mathbf{x}_{k+1} = \mathrm{proj}_{\mathcal{C}}\big(\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k)\big)$

The problem of PGD

- One problem of PGD iteration

$$\mathbf{x}_{k+1} = \mathrm{proj}_{\mathcal{C}}\big(\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k)\big)$$

  is on the projection step. It is possible that the projection subproblem cannot be solved "quickly" or "cheaply".

- This motivates the use of Frank-Wolfe algorithm.

Frank-Wolfe algorithm

Algorithm 2: Frank-Wolfe algorithm

Result: $\mathbf{x}^*$ that solves (P)

Initialize $\mathbf{x}_0 \in \mathcal{C}$;

for $k = 1, 2, \ldots$ **do**

  $\mathbf{y}_{k+1} \in \underset{\mathbf{y} \in \mathcal{C}}{\operatorname{argmin}} \langle \nabla f(\mathbf{x}_k), \mathbf{y} \rangle$; `// FW step`

  $\alpha_k = \frac{2}{k+1}$; `// Convex combination weight`

  $\mathbf{x}_{k+1} = (1 - \alpha_k)\mathbf{x}_k + \alpha_k \mathbf{y}_{k+1}$; `// Convex combination`

end

- Why iterate $\mathbf{x}_{k+1}$ is feasible:
  - Notice that $\mathbf{x}_0$ is feasible and $\mathbf{y}_1$ is feasible in the FW step.
  - The iterate $\mathbf{x}_1$ is a convex combination of $\mathbf{x}_0$ and $\mathbf{y}_1$
  - The set $\mathcal{C}$ is a convex set
  - So by induction, all $\mathbf{x}_k$ are feasible

A closer look at the FW iteration

- The FW step is a constrained linear optimization in the form of

$$\operatorname*{argmin}_{\mathbf{u}\in\mathcal{C}} \langle \mathbf{c}, \mathbf{u} \rangle, \tag{LO}$$

  which can be expensive to solve in general.

- If $\mathcal{C}$ represents a linear constraint, then FW step is a linear programming subproblem, i.e., it can be expressed as

$$\operatorname*{argmin}_{\mathbf{u}} \langle \mathbf{c}, \mathbf{u} \rangle \text{ s.t. } \mathbf{A}\mathbf{u} \leq \mathbf{b}, \mathbf{u} \geq \mathbf{0}, \tag{LP}$$

  for some $\mathbf{A}, \mathbf{b}$.

- Note that in general LO can be expensive to solve, while for some LP with some specific $\mathbf{A}, \mathbf{b}$ can be solved very cheaply and fast.

The no-free lunch of Frank-Wolfe algorithm

- The most expensive step in PGD is the projection

$$\mathrm{proj}_{\mathcal{C}}\big(\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k)\big) = \operatorname*{argmin}_{\mathbf{u}\in\mathcal{C}} \frac{1}{2}\|\mathbf{u} - \big(\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k)\big)\|_2^2,$$

  which is a constrained problem with quadratic cost function.

- The most expensive step in FW is the constrained linear optimization step

$$\operatorname*{argmin}_{\mathbf{u}\in\mathcal{C}} \langle \mathbf{c}, \mathbf{u}\rangle. \tag{LO}$$

- We now see the "no-free lunch" aspect of FW: it replaces the projection subproblem by a LO subproblem, in which both subproblems can be equally expensive to solve.

- When $\mathcal{C}$ is a linear set that the LP subproblem is easy to solve, a potential advantage of FW over PGD is that FW may converge faster than PGD because the most costly step in FW is only a cheap LP, while the PGD subproblem is a quadratic programming with linear constraint.

Does FW run faster than PGD?

- In general, FW and PGD share the same asymptotic convergence rate.
- **Theorem** The sequence $\{f(\mathbf{x}_k)\}$ produced by FW converges to the optimal value f^* at a rate of $\mathcal{O}(\frac{1}{k})$, assuming that f is convex and L-smooth.
- Similarly, the convergence rate of PGD is $\mathcal{O}(\frac{1}{k})$.
- We will look at the proof of the convergence of FW next time.
- Ultimately, it depends on the structure of $\mathcal{C}$ to determine which algorithm to use.

Last page - summary

Brief introduction to Frank-Wolfe algorithm

- Review of constrained convex optimization and projected gradient descent
- Frank-Wolfe algorithm

What's next

- Examples of Frank-Wolfe algorithm
- Convergence analysis

End of document