A New Look at the Performance Analysis of First-Order Methods

Marc Teboulle

School of Mathematical Sciences

Tel Aviv University

Joint work with

Yoel Drori, Google's R&D Center, Tel Aviv

Optimization without Borders
In Honor of Yuri Nesterov's 60th Birthday
February 7–12, 2016 – Les Houches, France

The Problem

Problem: How to evaluate the performance (complexity) of an optimization method for a given class of problems?

The Problem

Problem: How to evaluate the performance (complexity) of an optimization method for a given class of problems?

- We focus on first order methods for smooth convex minimization

$$(M) \quad \min\{f(x) : \; x \in \mathbb{R}^d\}, \quad f \text{ convex and } C_L^{1,1}(\mathbb{R}^d).$$

Asssumption.

- (M) is solvable, i.e., the optimal set $X_*(f) := \operatorname{argmin} f$ is nonempty.
- Given any starting point $x_0, \exists R > 0$, such that $\|x_0 - x_*\| \leq R, \; x_* \in X_*(f)$.

The Problem

Problem: How to evaluate the performance (complexity) of an optimization method for a given class of problems?

- We focus on first order methods for smooth convex minimization

$$(M) \quad \min\{f(x) : x \in \mathbb{R}^d\}, \quad f \text{ convex and } C_L^{1,1}(\mathbb{R}^d).$$

Asssumption.

- (M) is solvable, i.e., the optimal set $X_*(f) := \operatorname{argmin} f$ is nonempty.
- Given any starting point $x_0, \exists R > 0$, such that $\|x_0 - x_*\| \leq R, \; x_* \in X_*(f)$.

We completely depart from conventional approaches....

Black-Box First Order Methods

- A *Black-box* optimization method [1] is an algorithm $\mathcal{A}$ which has knowledge of:
  – The underlying space $\mathbb{R}^d$
  – The family of functions $\mathcal{F}$ to be minimized

[1] Oracle Model of Optimization – Nemirovsky-Yudin (83)

Black-Box First Order Methods

- A *Black-box* optimization method [1] is an algorithm $\mathcal{A}$ which has knowledge of:
  – The underlying space $\mathbb{R}^d$
  – The family of functions $\mathcal{F}$ to be minimized

The function itself is not known.

- To gain information on the objective function f to be minimized, the algorithm $\mathcal{A}$ queries a subroutine which given an input point in $\mathbb{R}^d$, returns the value of f and its gradient f' at that point.

First Order Method: The Algorithm $\mathcal{A}$

The algorithm starts with an initial point $x_0 \in \mathbb{R}^d$ and generate a finite sequence of points $\{x_i : i = 1, \dots N\}$ where at each step, the algorithm depends only on the previous steps, their function values and gradients via some rule:

$$x_{i+1} = \mathcal{A}(x_0, \dots, x_i; f(x_0), \dots, f(x_i); f'(x_0), \dots, f'(x_i)), \ \ i = 0, 1, \dots, N-1$$

Note that the algorithm has another implicit knowledge: $\|x_0 - x_*\| \leq R$.

[1] Oracle Model of Optimization – Nemirovsky-Yudin (83)

Performance/Complexity of an Algorithm

- We measure the worst-case performance (or complexity) of an algorithm $\mathcal{A}$ by looking at the absolute inaccuracy

$$\delta(f, x_N) = f(x_N) - f(x_*),$$

where x_N is the output of the algorithm after making N calls to the oracle.
- The worst-case is taken over all possible functions $f \in \mathcal{F}$ with starting points x_0 satisfying $\|x_0 - x_*\| \leq R$, where $x^* \in X_*(f)$.

Problem

We look at finding the **maximal absolute inaccuracy over all possible inputs** to the algorithm.

This leads to the following....

Main Observation

The worst-case performance of an optimization method is by itself an optimization problem!

Main Observation

The worst-case performance of an optimization method is by itself an optimization problem!

The Performance Estimation Problem – PEP

To measure the worst-case performance of an algorithm $\mathcal{A}$ we need to solve the following *Performance Estimation Problem (PEP)*:

$$\begin{aligned} \max \quad & f(x_N) - f(x_*) \\ \text{s.t.} \quad & f \in \mathcal{F}, \\ & x_{i+1} = \mathcal{A}(x_0, \dots, x_i; f(x_0), \dots, f(x_i); f'(x_0), \dots, f'(x_i)), \; i = 0, \dots, N-1, \\ & x_* \in X_*(f), \; \|x_* - x_0\| \leq R, \\ & x_0, \dots, x_N, x_* \in \mathbb{R}^d. \end{aligned} \tag{P}$$

PEP is an abstract optimization problem in *infinite dimension* : $f \in \mathcal{F}$.

Clearly intractable!?!..

A Methodology to Tackle PEP: Basic Un-Formal Approach

A. Relax the functional constraint ($f \in \mathcal{F}$) by new variables and constraints in $\mathbb{R}^d$ to built **a finite dimensional** problem. This is done by:

1. Exploiting adequate properties of the class $\mathcal{F}$ at the points $x_0, \ldots, x_N, x_* \in \mathbb{R}^d$.
2. Using the rule(s) describing the given algorithm $\mathcal{A}$.

A Methodology to Tackle PEP: Basic Un-Formal Approach

A. Relax the functional constraint ($f \in \mathcal{F}$) by new variables and constraints in $\mathbb{R}^d$ to built **a finite dimensional** problem. This is done by:

1. Exploiting adequate properties of the class $\mathcal{F}$ at the points $x_0, \ldots, x_N, x_* \in \mathbb{R}^d$.
2. Using the rule(s) describing the given algorithm $\mathcal{A}$.

The resulting relaxed finite dimensional problem remains a **valid upper bound** on

$$f(x_N) - f(x_*).$$

Yet, this problem remains nontrivial to tackle. So what else can be done...?

A Methodology to Tackle PEP: Basic Un-Formal Approach

A. Relax the functional constraint ($f \in \mathcal{F}$) by new variables and constraints in $\mathbb{R}^d$ to built **a finite dimensional** problem. This is done by:

1. Exploiting adequate properties of the class $\mathcal{F}$ at the points $x_0, \ldots, x_N, x_* \in \mathbb{R}^d$.
2. Using the rule(s) describing the given algorithm $\mathcal{A}$.

The resulting relaxed finite dimensional problem remains a **valid upper bound** on

$$f(x_N) - f(x_*).$$

Yet, this problem remains nontrivial to tackle. So what else can be done...?

B. More Relaxations..!!...

1. For a given class of algorithms $\mathcal{A}$: Exploit structures of PEP to simplify it.
2. Develop a novel relaxation technique and duality to find an upper bound to this problem.

Despite "massive" relaxations: We derive new and better complexity bounds than currently known.

In principle... this approach is universal. It can be applied to any optimization algorithm...!

Relaxing the functional constraint $f \in \mathcal{F}$

We focus on First Order Methods (FOM) for *smooth convex* problem, that is: convex $f \in \mathcal{F} \equiv C_L^{1,1}$.

We start with the following well known fact for convex f in $\mathcal{F} \equiv C_L^{1,1}$.

Relaxing the functional constraint $f \in \mathcal{F}$

We focus on First Order Methods (FOM) for *smooth convex* problem, that is: convex $f \in \mathcal{F} \equiv C_L^{1,1}$.

We start with the following well known fact for convex f in $\mathcal{F} \equiv C_L^{1,1}$.

Proposition Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is convex and has Lipschitz continuous gradient with constant L. Then for every $x, y \in \mathbb{R}^d$:

$$\frac{1}{2L}\|f'(x) - f'(y)\|^2 \leq f(x) - f(y) - \langle f'(y), x - y\rangle. \quad (1)$$

Relaxing the functional constraint $f \in \mathcal{F}$

We focus on First Order Methods (FOM) for *smooth convex* problem, that is: convex $f \in \mathcal{F} \equiv C_L^{1,1}$.

We start with the following well known fact for convex f in $\mathcal{F} \equiv C_L^{1,1}$.

Proposition Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is convex and has Lipschitz continuous gradient with constant L. Then for every $x, y \in \mathbb{R}^d$:

$$\frac{1}{2L}\|f'(x) - f'(y)\|^2 \leq f(x) - f(y) - \langle f'(y), x - y\rangle. \qquad (1)$$

The relaxation scheme - a sort of "discretization"

- Apply (1) at the points $x_0, \ldots, x_N$ and x_*.
- Use the resulting inequalities as "constraints" instead of the functional constraint $f \in \mathcal{F}$.

Relaxing the functional constraint $f \in \mathcal{F}$

We focus on First Order Methods (FOM) for *smooth convex* problem, that is: convex $f \in \mathcal{F} \equiv C_L^{1,1}$.

We start with the following well known fact for convex f in $\mathcal{F} \equiv C_L^{1,1}$.

Proposition Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is convex and has Lipschitz continuous gradient with constant L. Then for every $x, y \in \mathbb{R}^d$:

$$\frac{1}{2L}\|f'(x) - f'(y)\|^2 \leq f(x) - f(y) - \langle f'(y), x - y\rangle. \qquad (1)$$

The relaxation scheme - a sort of "discretization"

- Apply (1) at the points $x_0, \ldots, x_N$ and x_*.
- Use the resulting inequalities as "constraints" instead of the functional constraint $f \in \mathcal{F}$.

Define

$$L\|x_* - x_0\|^2\delta_i := f(x_i) - f(x_*), \quad L\|x_* - x_0\|g_i := f'(x_i), \quad i = 0, \ldots, N, *,$$

In terms of δ_i, g_i, condition (1) becomes

$$\frac{1}{2}\|g_i - g_j\|^2 \leq \delta_i - \delta_j - \langle g_j, \frac{x_i - x_j}{\|x_* - x_0\|}\rangle, \quad i, j = 0, \ldots, N, *. \qquad (2)$$

We now treat $x_*, \{x_i, \delta_i, g_i\}_{i=0}^N$ as the optimization variables, instead of $f \in C_L^{1,1}$.

A (Relaxed) Finite Dimensional PEP

Replacing the constraint on f by the constraints (2) we reach a **relaxed finite dimensional** PEP in the variables $x_*, \{x_i, \delta_i, g_i\}_{i=0}^N$:

$$
(P) \qquad \max_{x_*, x_i, g_i \in \mathbb{R}^d, \delta_i \in \mathbb{R}} \ L\|x_* - x_0\|^2 \delta_N
$$

$$
\text{s.t.} \quad \frac{1}{2}\|g_i - g_j\|^2 \leq \delta_i - \delta_j - \langle g_j, \frac{x_i - x_j}{\|x_* - x_0\|} \rangle, \quad i, j = 0, \ldots, N, *,
$$

$$
x_{i+1} = \mathcal{A}(x_0, \ldots, x_i; \delta_0, \ldots, \delta_i; g_0, \ldots, g_i), \ i = 0, \ldots, N-1,
$$

$$
\|x_* - x_0\| \leq R.
$$

Since (P) is a relaxation of the original maximization problem, its solution still provides a valid upper bound on the complexity of the given method $\mathcal{A}$:

$$
f(x_N) - f(x^*) \leq \text{val}(P).
$$

We will now show our main results for:

1. The gradient method.
2. A broad class of first order methods.

PEP for the Gradient Method

Algorithm (GM)

0. Input: $N, h, f \in C_L^{1,1}(\mathbb{R}^d)$ convex, $x_0 \in \mathbb{R}^d$.
1. For $i = 0, \ldots, N-1$, compute $x_{i+1} = x_i - \frac{h}{L} f'(x_i), \quad (h > 0)$.

PEP for the Gradient Method

Algorithm (GM)

0. Input: $N, h, f \in C_L^{1,1}(\mathbb{R}^d)$ convex, $x_0 \in \mathbb{R}^d$.
1. For $i = 0, \dots, N-1$, compute $x_{i+1} = x_i - \frac{h}{L} f'(x_i), \quad (h > 0)$.

After some transformations, PEP for (GM) is **a Nonconvex Quadratic Problem**:

$$
(P) \qquad \begin{aligned}
\max_{g_i \in \mathbb{R}^d, \delta_i \in \mathbb{R}} \quad & LR^2 \delta_N \\
\text{s.t.} \quad & \tfrac{1}{2}\|g_i - g_j\|^2 \le \delta_i - \delta_j - \langle g_j, \textstyle\sum_{t=i+1}^{j} h g_{t-1} \rangle, \quad i < j = 0, \dots, N, \\
& \tfrac{1}{2}\|g_i - g_j\|^2 \le \delta_i - \delta_j + \langle g_j, \textstyle\sum_{t=j+1}^{i} h g_{t-1} \rangle, \quad j < i = 0, \dots, N, \\
& \tfrac{1}{2}\|g_i\|^2 \le \delta_i, \quad i = 0, \dots, N, \\
& \tfrac{1}{2}\|g_i\|^2 \le -\delta_i - \langle g_i, \nu + \textstyle\sum_{t=1}^{i} h g_{t-1} \rangle, \quad i = 0, \dots, N.
\end{aligned}
$$

Notation: $\nu \in \mathbb{R}^d$ is any unit vector;
$i < j = 0, \dots, N$ is a shorthand notation for $i = 0, \dots, N-1, j = i+1, \dots, N$.

“As is", PEP remains impossible to tackle..!?..We turn to the second phase: Reformulation and more relaxations!

Analyzing PEP for the Gradient Method

The main steps (see paper for details):

- We further drop constraints... This is still a valid upper bound!
- Reformulate it as a *Quadratic Matrix* (QM) Optimization Problem:

$$\begin{aligned} \max_{G\in\mathbb{R}^{(N+1)\times d},\delta\in\mathbb{R}^{N+1}} \quad & LR^2\delta_N \\ \text{s.t. } & \mathrm{Tr}(G^T A_{i-1,i} G) \leq \delta_{i-1} - \delta_i, \quad i = 1,\dots,N, \\ & \mathrm{Tr}(G^T D_i G + \nu e_{i+1}^T G) \leq -\delta_i, \quad i = 0,\dots,N, \end{aligned} \tag{G'}$$

The matrices $A_{i-1,i}, D_i \in \mathbb{S}^{N+1}$ are explicitly given in terms of h.
($\{e_{i+1}\}_{i=0}^N$ are the canonical unit vectors in $\mathbb{R}^{N+1}$ and $\nu \in \mathbb{R}^d$ is a unit vector.)

Analyzing PEP for the Gradient Method

The main steps (see paper for details):

- We further drop constraints... This is still a valid upper bound!
- Reformulate it as a *Quadratic Matrix* (QM) Optimization Problem:

$$\begin{aligned} \max_{G\in\mathbb{R}^{(N+1)\times d},\delta\in\mathbb{R}^{N+1}} \quad & LR^2\delta_N \\ \text{s.t. } & \mathrm{Tr}(G^T A_{i-1,i} G) \leq \delta_{i-1} - \delta_i, \quad i = 1,\dots,N, \\ & \mathrm{Tr}(G^T D_i G + \nu e_{i+1}^T G) \leq -\delta_i, \quad i = 0,\dots,N, \end{aligned} \tag{G'}$$

The matrices $A_{i-1,i}, D_i \in \mathbb{S}^{N+1}$ are explicitly given in terms of h.
($\{e_{i+1}\}_{i=0}^N$ are the canonical unit vectors in $\mathbb{R}^{N+1}$ and $\nu \in \mathbb{R}^d$ is a unit vector.)

- To find an upper bound on problem G′ we use duality.
- We further exploit the special structure of this (QM), and a dimension reduction result, to derive a tractable SDP dual.

A Dual Problem for the Quadratic Matrix Problem G′

$$\min_{\lambda\in\mathbb{R}^N, t\in\mathbb{R}} \{\frac{1}{2}LR^2 t : \ \lambda \in \Lambda, \ S(\lambda, t) \succeq 0\}, \qquad \text{(DG}'\text{)}$$

$$\Lambda := \{\lambda \in \mathbb{R}^N : \ \lambda_{i+1} - \lambda_i \geq 0, \quad i = 1, \dots, N-1, \ 1 - \lambda_N \geq 0, \ \lambda_i \geq 0, \quad i = 1, \dots, N\},$$

$$\mathbb{S}^{N+2} \ni S(\lambda, t) := \begin{pmatrix} (1-h)S_0(\lambda) + hS_1(\lambda) & q \\ q^T & t \end{pmatrix},$$

$q := (\lambda_1, \lambda_2 - \lambda_1, \dots, \lambda_N - \lambda_{N-1}, 1 - \lambda_N)^T$ and $S_0, S_1 \in \mathbb{S}^{N+1}$ are defined by:

$$S_0(\lambda) = \begin{pmatrix} 2\lambda_1 & -\lambda_1 & & & & \\ -\lambda_1 & 2\lambda_2 & -\lambda_2 & & & \\ & -\lambda_2 & 2\lambda_3 & -\lambda_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\lambda_{N-1} & 2\lambda_N & -\lambda_N \\ & & & & -\lambda_N & 1 \end{pmatrix} \qquad (3)$$

and

$$S_1(\lambda) = \begin{pmatrix} 2\lambda_1 & \lambda_2 - \lambda_1 & \dots & \lambda_N - \lambda_{N-1} & 1 - \lambda_N \\ \lambda_2 - \lambda_1 & 2\lambda_2 & & \lambda_N - \lambda_{N-1} & 1 - \lambda_N \\ \vdots & & \ddots & & \vdots \\ \lambda_N - \lambda_{N-1} & \lambda_N - \lambda_{N-1} & & 2\lambda_N & 1 - \lambda_N \\ 1 - \lambda_N & 1 - \lambda_N & \dots & 1 - \lambda_N & 1 \end{pmatrix}. \qquad (4)$$

A Feasible Analytical Solution of this SDP can be found!

A crucial and hard part of the proof!

A Feasible Analytical Solution of this SDP can be found!

A crucial and hard part of the proof!

Lemma

Let

$$t = \frac{1}{2Nh+1}, \text{ and } \lambda_i = \frac{i}{2N+1-i}, \qquad i = 1, \ldots, N.$$

Then,

- *the matrices* $S_0(\lambda), S_1(\lambda) \in \mathbb{S}^{N+1}$ *defined in (3)–(4) are positive definite for every* $N \in \mathbb{N}$.
- *The pair* (λ_i, t) *is feasible for* DG'.

Equipped with this result, invoking standard duality leads to the desired complexity result for GM.

Complexity Bound for the Gradient Method

Theorem

Let $f \in C_L^{1,1}(\mathbb{R}^d)$ *and let* $x_0, \ldots, x_N \in \mathbb{R}^d$ *be generated by (GM) with* $0 < h \leq 1$*. Then*[a]

$$f(x_N) - f(x_*) \leq \frac{LR^2}{4N+2}. \tag{5}$$

[a] The classical bound on the gradient method: $f(x_N) - f(x_*) \leq \frac{LR^2}{2N}$.

Complexity Bound for the Gradient Method

Theorem

Let $f \in C_L^{1,1}(\mathbb{R}^d)$ and let $x_0, \ldots, x_N \in \mathbb{R}^d$ be generated by (GM) with $0 < h \leq 1$. Then[a]

$$f(x_N) - f(x_*) \leq \frac{LR^2}{4N+2}. \tag{5}$$

[a]The classical bound on the gradient method: $f(x_N) - f(x_*) \leq \frac{LR^2}{2N}$.

We further prove that this bound is tight!

The Bound is Tight

Theorem

Let $L > 0$, $N \in \mathbb{N}$ *and* $d \in \mathbb{N}$. *Then for every* $h > 0$ *there exists a convex function* $\varphi \in C_L^{1,1}(\mathbb{R}^d)$ *and a point* $x_0 \in \mathbb{R}^d$ *such that after N iterations, Algorithm GM reaches an approximate solution* x_N *with the following absolute inaccuracy*

$$\varphi(x_N) - \varphi^* = \frac{LR^2}{4Nh+2}.$$

The Bound is Tight

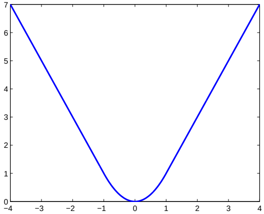
Theorem

Let $L > 0$, $N \in \mathbb{N}$ and $d \in \mathbb{N}$. Then for every $h > 0$ there exists a convex function $\varphi \in C_L^{1,1}(\mathbb{R}^d)$ and a point $x_0 \in \mathbb{R}^d$ such that after N iterations, Algorithm GM reaches an approximate solution x_N with the following absolute inaccuracy

$$\varphi(x_N) - \varphi^* = \frac{LR^2}{4Nh + 2}.$$

Interestingly...this φ is nothing else but the Moreau envelope of $\|x\|/(2Nh+1)$!

$$\varphi(x) = \begin{cases} \frac{1}{2N+1}\|x\| - \frac{1}{2(2N+1)^2}, & \|x\| \geq \frac{1}{2N+1}, \\ \frac{1}{2}\|x\|^2, & \|x\| < \frac{1}{2N+1}, \end{cases}$$



with $x_0 = e_1$.

A Conjecture for GM with $0 < h < 2$

We conclude this part by raising a conjecture on the worst-case performance of the gradient method with **a constant step size $0 < h < 2$.**

A Conjecture for GM with $0 < h < 2$

We conclude this part by raising a conjecture on the worst-case performance of the gradient method with **a constant step size** $0 < h < 2$**.**

Conjecture 1

Suppose the sequence $x_0, \ldots, x_N$ is generated by Algorithm GM with $0 < h < 2$, then

$$f(x_N) - f(x_*) \leq \frac{LR^2}{2} \max\left(\frac{1}{2Nh+1}, (1-h)^{2N} \right).$$

Note: when $0 < h \leq 1$ the bound above coincides with our previous bound.

A Wide Class of First-Order Algorithms

Consider the following class of first-order algorithms:

Algorithm (FO)

0. Input: $f \in C_L^{1,1}(\mathbb{R}^d)$, $x_0 \in \mathbb{R}^d$.
1. For $i = 0, \ldots, N-1$, compute $x_{i+1} = x_i - \frac{1}{L} \sum_{k=0}^{i} h_k^{(i+1)} f'(x_k)$.

1. We now show that the class (FO) covers some fundamental schemes beyond the gradient method.
2. For this class we establish a complexity bound that can be efficiently computed via SDP solvers.
3. Furthermore, we derive an "optimized" algorithm of this form by finding optimal step sizes $h_k^{(i)}$.

Example 1: the Heavy Ball Method

Example (The heavy ball method, HBM, Polyak (1964))

0. Input: $f \in C_L^{1,1}(\mathbb{R}^d)$, $x_0 \in \mathbb{R}^d$,
1. $x_1 \leftarrow x_0 - \frac{\alpha}{L} f'(x_0), \quad (\alpha > 0)$.
2. For $i = 1, \ldots, N-1$ compute: $x_{i+1} = x_i - \frac{\alpha}{L} f'(x_i) + \beta(x_i - x_{i-1}), \ (\beta > 0)$.

Example 1: the Heavy Ball Method

Example (The heavy ball method, HBM, Polyak (1964))

0. Input: $f \in C_L^{1,1}(\mathbb{R}^d)$, $x_0 \in \mathbb{R}^d$,
1. $x_1 \leftarrow x_0 - \frac{\alpha}{L} f'(x_0), \quad (\alpha > 0)$.
2. For $i = 1, \ldots, N-1$ compute: $x_{i+1} = x_i - \frac{\alpha}{L} f'(x_i) + \beta(x_i - x_{i-1}), \ (\beta > 0)$.

- By recursively eliminating the term $x_i - x_{i-1}$ in the last step, we can rewrite step 2 as follows:

$$x_{i+1} = x_i - \frac{1}{L} \sum_{k=0}^{i} \alpha \beta^{i-k} f'(x_k),$$

hence this methods clearly fits in the class (FO).

Example 2: Nesterov's Fast Gradient Method

Example (Nesterov's fast gradient method, FGM (1983))

0. Input: $f \in C_L^{1,1}(\mathbb{R}^d)$, $x_0 \in \mathbb{R}^d$,
1. $y_1 \leftarrow x_0$, $t_1 \leftarrow 1$,
2. For $i = 1, \ldots, N$ compute:
   1. $x_i \leftarrow y_i - \frac{1}{L} f'(y_i)$,
   2. $t_{i+1} \leftarrow \frac{1+\sqrt{1+4t_i^2}}{2}$,
   3. $y_{i+1} \leftarrow x_i + \frac{t_i - 1}{t_{i+1}}(x_i - x_{i-1})$.

- This algorithm is as simple as the gradient method, yet achieves an optimal convergence rate of $O(1/N^2)$:

$$f(x_N) - f(x_*) \leq \frac{\mathbf{2L}\|x_0 - x_*\|^2}{(N+1)^2}, \ \forall\, x_* \in X_*(f); \ (\mathbf{3L}/\mathbf{32}, \ \text{lower bound}).$$

Example 2: Nesterov's Fast Gradient Method

Example (Nesterov's fast gradient method, FGM (1983))

0. Input: $f \in C_L^{1,1}(\mathbb{R}^d)$, $x_0 \in \mathbb{R}^d$,
1. $y_1 \leftarrow x_0$, $t_1 \leftarrow 1$,
2. For $i = 1, \dots, N$ compute:
   1. $x_i \leftarrow y_i - \frac{1}{L} f'(y_i)$,
   2. $t_{i+1} \leftarrow \frac{1+\sqrt{1+4t_i^2}}{2}$,
   3. $y_{i+1} \leftarrow x_i + \frac{t_i - 1}{t_{i+1}}(x_i - x_{i-1})$.

- This algorithm is as simple as the gradient method, yet achieves an optimal convergence rate of $O(1/N^2)$:

$$f(x_N) - f(x_*) \leq \frac{\mathbf{2L}\|x_0 - x_*\|^2}{(N+1)^2}, \ \forall\, x_* \in X_*(f); (\mathbf{3L}/\mathbf{32}, \text{ lower bound}).$$

- This algorithm includes 2 sequences of points (x_i, y_i). At first glance, it does not appear to belong to the class (FO)...
- ...It can be shown that the FGM fits in the class (FO), (see the paper).

PEP for the Wide Class of First-Order Algorithms (FO)

Applying our approach to FO, (as done for GM), we derive the following PEP:

$$
\begin{aligned}
\max_{g_i \in \mathbb{R}^d, \delta_i \in \mathbb{R}} \quad & LR^2 \delta_N \\
\text{s.t.} \quad & \tfrac{1}{2}\|g_i - g_j\|^2 \le \delta_i - \delta_j - \langle g_j, \textstyle\sum_{t=i+1}^{j} \sum_{k=0}^{t-1} h_k^{(t)} g_k \rangle, \quad i < j = 0, \dots, N, \\
& \tfrac{1}{2}\|g_i - g_j\|^2 \le \delta_i - \delta_j + \langle g_j, \textstyle\sum_{t=j+1}^{i} \sum_{k=0}^{t-1} h_k^{(t)} g_k \rangle, \quad j < i = 0, \dots, N, \\
& \tfrac{1}{2}\|g_i\|^2 \le \delta_i, \quad i = 0, \dots, N, \\
& \tfrac{1}{2}\|g_i\|^2 \le -\delta_i - \langle g_i, \nu + \textstyle\sum_{t=1}^{i} \sum_{k=0}^{t-1} h_k^{(t)} g_k \rangle, \quad i = 0, \dots, N.
\end{aligned}
$$

- In this general case **an analytical solution appears unlikely...**

PEP for the Wide Class of First-Order Algorithms (FO)

Applying our approach to FO, (as done for GM), we derive the following PEP:

$$
\begin{aligned}
\max_{g_i \in \mathbb{R}^d, \delta_i \in \mathbb{R}} \quad & LR^2 \delta_N \\
\text{s.t. } \quad & \tfrac{1}{2}\|g_i - g_j\|^2 \le \delta_i - \delta_j - \langle g_j, \textstyle\sum_{t=i+1}^{j} \sum_{k=0}^{t-1} h_k^{(t)} g_k \rangle, \quad i < j = 0, \ldots, N, \\
& \tfrac{1}{2}\|g_i - g_j\|^2 \le \delta_i - \delta_j + \langle g_j, \textstyle\sum_{t=j+1}^{i} \sum_{k=0}^{t-1} h_k^{(t)} g_k \rangle, \quad j < i = 0, \ldots, N, \\
& \tfrac{1}{2}\|g_i\|^2 \le \delta_i, \quad i = 0, \ldots, N, \\
& \tfrac{1}{2}\|g_i\|^2 \le -\delta_i - \langle g_i, \nu + \textstyle\sum_{t=1}^{i} \sum_{k=0}^{t-1} h_k^{(t)} g_k \rangle, \quad i = 0, \ldots, N.
\end{aligned}
$$

- In this general case **an analytical solution appears unlikely...**
- Nevertheless, using techniques similar to the ones used for GM, we establish **a dual bound that can be efficiently computed via any SDP solver.**

More precisely, we obtain the following result.

A Bound on Algorithm FO via Convex SDP

Theorem

Fix any $N, d \in \mathbb{N}$. *Let* $f \in C_L^{1,1}(\mathbb{R}^d)$ *be convex and suppose that* $x_0, \ldots, x_N \in \mathbb{R}^d$ *are generated by Algorithm FO, and that (DQ)' is solvable. Then,*

$$f(x_N) - f(x_*) \leq LR^2 B(h)$$

Here $B(\cdot)$ is the value of the **Convex SDP** (DQ'):

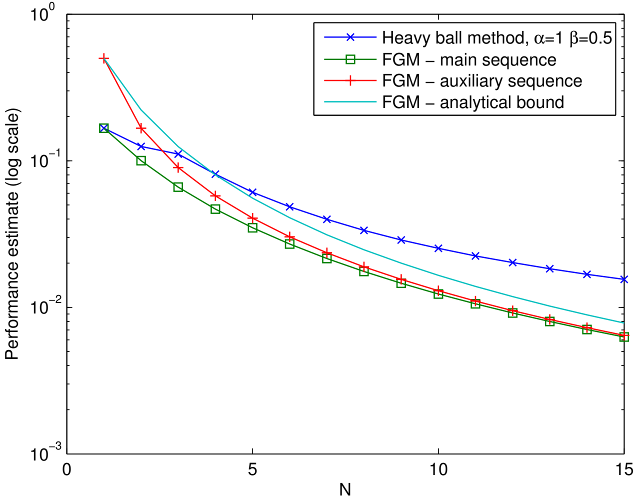
$$B(h) = \min_{\lambda,\tau,t} \frac{1}{2}LR^2 t$$
$$\text{s.t.} \begin{pmatrix} \sum_{i=1}^{N} \lambda_i \tilde{A}_{i-1,i}(h) + \sum_{i=0}^{N} \tau_i \tilde{D}_i(h) & \frac{1}{2}\tau \\ \frac{1}{2}\tau^T & \frac{1}{2}t \end{pmatrix} \succeq 0, \qquad \text{(DQ}'\text{)}$$
$$(\lambda, \tau) \in \tilde{\Lambda},$$

$\tilde{\Lambda} := \{(\lambda, \tau) \in \mathbb{R}_+^N \times \mathbb{R}_+^{N+1} : \tau_0 = \lambda_1,\ \lambda_i - \lambda_{i+1} + \tau_i = 0,\ i = 1, \ldots, N-1,\ \lambda_N + \tau_N = 1\}.$

and the matrices $\tilde{A}_i(h), \tilde{D}_i(h)$ are explicitly given in terms of $h \equiv (h_k^i)_{0\leq<k\leq i\leq N}$.

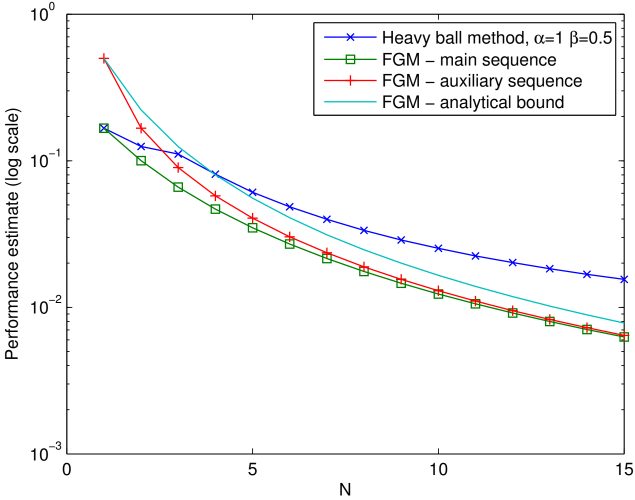
Note: The bound is independent of the dimension d.

Numerical Examples



- **FGM Analytical Bound = $\frac{2LR^2}{(N+1)^2}$. HBM is not competitive versus FGM**
- **Conjecture 2: $f(x_i), f(y_i)$ converge to optimal value with same rate of convergence.**

Numerical Examples



- **FGM Analytical Bound =** $\frac{2LR^2}{(N+1)^2}$**. HBM is not competitive versus FGM**
- **Conjecture 2:** $f(x_i), f(y_i)$ **converge to optimal value with same rate of convergence.**

...Just proven by Kim-Fessler (2015).

Finding an “Optimized" Algorithm

- Given $B(h)$, a natural question is:
how to find the “best" algorithm with respect to the bound? i.e., the best step sizes. That is find $h^* = \mathrm{argmin}_h B(h)$ which leads to the mini-max problem:

$$\min_{h_k^{(k)}} \max_{x_i, g_i \in \mathbb{R}^d, \delta_i \in \mathbb{R}} \delta_N$$
$$\text{s.t. } \frac{1}{2L}\|g_i - g_j\|^2 \le \delta_i - \delta_j - \langle g_j, x_i - x_j \rangle, \quad i, j = 0, \ldots, N, *,$$
$$x_{i+1} = x_i - \frac{1}{L}\sum_{k=0}^{i} h_k^{(i+1)} g_k, \quad i = 0, \ldots, N-1,$$
$$\|x_* - x_0\| \le R.$$

- Once again, we face a challenging problem...

Finding an “Optimized" Algorithm

- Given $B(h)$, a natural question is:
  how to find the “best" algorithm with respect to the bound? i.e., the best step sizes. That is find $h^* = \mathrm{argmin}_h B(h)$ which leads to the mini-max problem:

$$\min_{h_k^{(k)}} \max_{x_i, g_i \in \mathbb{R}^d, \delta_i \in \mathbb{R}} \delta_N$$

$$\text{s.t. } \frac{1}{2L}\|g_i - g_j\|^2 \leq \delta_i - \delta_j - \langle g_j, x_i - x_j \rangle, \quad i, j = 0, \ldots, N, *,$$
$$x_{i+1} = x_i - \frac{1}{L}\sum_{k=0}^{i} h_k^{(i+1)} g_k, \quad i = 0, \ldots, N-1,$$
$$\|x_* - x_0\| \leq R.$$

- Once again, we face a challenging problem...
- Using semidefinite relaxations, duality and linearization, a solution to this problem can be **efficiently approximated**.

An Optimized Algorithm – Solution step I

- Remove some selected constraints and eliminate x_i using the equality constraints:

$$\begin{aligned}
\min_{h_k^{(k)}} \max_{x.,g_i\in\mathbb{R}^d,\delta_i\in\mathbb{R}} \quad & \delta_N \\
\text{s.t.} \quad & \tfrac{1}{2L}\|g_{i-1}-g_i\|^2 \le \delta_{i-1}-\delta_i - \langle g_i, \textstyle\sum_{k=0}^{i-1} h_k^{(i)} g_k\rangle, \quad i=1,\dots,N, \\
& \tfrac{1}{2L}\|g_i\|^2 \le -\delta_i - \langle g_i, x_* - x_0 + \textstyle\sum_{t=1}^{i}\sum_{k=0}^{t-1} h_k^{(t)} g_k\rangle, \quad i=0,\dots,N, \\
& \|x_*-x_0\|^2 \le R^2.
\end{aligned}$$

An Optimized Algorithm – Solution step I

- Remove some selected constraints and eliminate x_i using the equality constraints:

$$\begin{aligned}\min_{h_k^{(k)}} \max_{x_\cdot, g_i \in \mathbb{R}^d, \delta_i \in \mathbb{R}} \quad & \delta_N \\ \text{s.t.} \quad & \tfrac{1}{2L}\|g_{i-1} - g_i\|^2 \le \delta_{i-1} - \delta_i - \langle g_i, \textstyle\sum_{k=0}^{i-1} h_k^{(i)} g_k \rangle, \quad i = 1, \dots, N, \\ & \tfrac{1}{2L}\|g_i\|^2 \le -\delta_i - \langle g_i, x_* - x_0 + \textstyle\sum_{t=1}^{i} \sum_{k=0}^{t-1} h_k^{(t)} g_k \rangle, \quad i = 0, \dots, N, \\ & \|x_* - x_0\|^2 \le R^2.\end{aligned}$$

- Take dual of the **inner "max" problem**, to obtain a **Nonconvex (bilinear) SDP**:

$$(BIL) \quad \min_{h,\lambda,\tau,t} \left\{ \frac{1}{2} t : \begin{pmatrix} \sum_{i=1}^N \lambda_i \tilde{A}_i(h) + \sum_{i=0}^N \tau_i \tilde{D}_i(h) & \frac{1}{2}\tau \\ \frac{1}{2}\tau^T & \frac{1}{2} t \end{pmatrix} \succeq 0, (\lambda, \tau) \in \tilde{\Lambda} \right\},$$

$$\tilde{A}_i(h) := \tfrac{1}{2}(e_i - e_{i+1})(e_i - e_{i+1})^T + \tfrac{1}{2} \textstyle\sum_{k=0}^{i-1} h_k^{(i)} (e_{i+1} e_{k+1}^T + e_{k+1} e_{i+1}^T),$$

$$\tilde{D}_i(h) := \tfrac{1}{2} e_{i+1} e_{i+1}^T + \tfrac{1}{2} \textstyle\sum_{t=1}^{i} \sum_{k=0}^{t-1} h_k^{(t)} (e_{i+1} e_{k+1}^T + e_{k+1} e_{i+1}^T)$$

$$\tilde{\Lambda} := \{(\lambda, \tau) \in \mathbb{R}_+^N \times \mathbb{R}_+^{N+1} : \tau_0 = \lambda_1,\ \lambda_i - \lambda_{i+1} + \tau_i = 0,\ i = 1, \dots, N-1,\ \lambda_N + \tau_N = 1\}.$$

Optimized Algorithm – Solution step II

- Define a new variable (Linearize the bilinear nonconvex SDP):

$$r_{i,k} = \lambda_i h_k^{(i)} + \tau_i \sum_{t=k+1}^{i} h_k^{(t)}, \quad i = 1, \ldots, N,\ k = 0, \ldots, i-1$$

to obtain a **A Convex SDP**:

$$(LIN)\ \min_{r,\lambda,\tau,t} \left\{ \frac{1}{2} t : \begin{pmatrix} S(r,\lambda,\tau) & \frac{1}{2}\tau \\ \frac{1}{2}\tau^T & \frac{1}{2} t \end{pmatrix} \succeq 0,\ (\lambda,\tau) \in \tilde{\Lambda} \right\},$$

where

$$S(r,\lambda,\tau) = \tfrac{1}{2} \sum_{i=1}^{N} \lambda_i (e_i - e_{i+1})(e_i - e_{i+1})^T + \tfrac{1}{2} \sum_{i=0}^{N} \tau_i e_{i+1} e_{i+1}^T$$
$$+ \tfrac{1}{2} \sum_{i=1}^{N} \sum_{k=0}^{i-1} r_{i,k} (e_{i+1} e_{k+1}^T + e_{k+1} e_{i+1}^T).$$

Optimized Algorithm – Solution step II

- Define a new variable (Linearize the bilinear nonconvex SDP):

$$r_{i,k} = \lambda_i h_k^{(i)} + \tau_i \sum_{t=k+1}^{i} h_k^{(t)}, \quad i = 1, \ldots, N, \; k = 0, \ldots, i-1$$

to obtain a **A Convex SDP**:

$$(LIN) \; \min_{r,\lambda,\tau,t} \left\{ \frac{1}{2} t : \begin{pmatrix} S(r,\lambda,\tau) & \frac{1}{2}\tau \\ \frac{1}{2}\tau^T & \frac{1}{2} t \end{pmatrix} \succeq 0, \; (\lambda, \tau) \in \tilde{\Lambda} \right\},$$

where

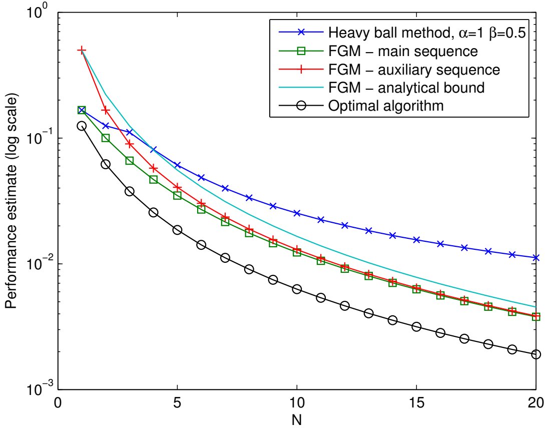
$$S(r,\lambda,\tau) = \frac{1}{2} \sum_{i=1}^{N} \lambda_i (e_i - e_{i+1})(e_i - e_{i+1})^T + \frac{1}{2} \sum_{i=0}^{N} \tau_i e_{i+1} e_{i+1}^T + \frac{1}{2} \sum_{i=1}^{N} \sum_{k=0}^{i-1} r_{i,k} (e_{i+1} e_{k+1}^T + e_{k+1} e_{i+1}^T).$$

Theorem (Use the solution of (LIN) to solve (BIL) and get optimal h.)

Suppose $(r^*, \lambda^*, \tau^*, t^*)$ is an optimal solution for (LIN), then $(h, \lambda^*, \tau^*, t^*)$ is an optimal solution for (BIL), where $h = (h_k^{(i)})_{0 \le k < i \le N}$ is defined by the following recursive rule:

$$h_k^{(i)} = \begin{cases} \frac{r_{i,k}^* - \tau_i^* \sum_{t=k+1}^{i-1} h_k^{(t)}}{\lambda_i^* + \tau_i^*} & \lambda_i^* + \tau_i^* \neq 0 \\ 0 & \text{otherwise} \end{cases}, \quad i = 1, \ldots, N, \; k = 0, \ldots, i-1.$$

An Optimized Algorithm – Numerical Results



- The bound on the new algorithm is two times better than the bound on Nesterov's FGM!

An Optimized Algorithm –Example with $N = 5$

Example

A first-order algorithm with optimal step-sizes for $N = 5$:

$$
\begin{aligned}
x_1 &\leftarrow x_0 - \tfrac{1.6180}{L} f'(x_0) \\
x_2 &\leftarrow x_1 - \tfrac{0.1741}{L} f'(x_0) - \tfrac{2.0194}{L} f'(x_1) \\
x_3 &\leftarrow x_2 - \tfrac{0.0756}{L} f'(x_0) - \tfrac{0.4425}{L} f'(x_1) - \tfrac{2.2317}{L} f'(x_2) \\
x_4 &\leftarrow x_3 - \tfrac{0.0401}{L} f'(x_0) - \tfrac{0.2350}{L} f'(x_1) - \tfrac{0.6541}{L} f'(x_2) - \tfrac{2.3656}{L} f'(x_3) \\
x_5 &\leftarrow x_4 - \tfrac{0.0178}{L} f'(x_0) - \tfrac{0.1040}{L} f'(x_1) - \tfrac{0.2894}{L} f'(x_2) - \tfrac{0.6043}{L} f'(x_3) - \tfrac{2.0778}{L} f'(x_4)
\end{aligned}
$$

We then get

$$f(x_5) - f(x_*) \leq 0.019 \times LR^2 \text{ for any } x^* \in X_*(f).$$

Concluding Remarks and Extensions

- The PEP framework offers a new approach to derive complexity bounds.
- Finding a bound for the PEP problem is challenging!
- Numerical bounds required solving SDP which dimension depends on N.

Concluding Remarks and Extensions

- The PEP framework offers a new approach to derive complexity bounds.
- Finding a bound for the PEP problem is challenging!
- Numerical bounds required solving SDP which dimension depends on N.

Two Very Recent Works Building on PEP:

- Kim-Fessler (MP-2015) confirmed our Conjecture 2. Also derived an efficient “Optimized" algorithm, with an *analytical bound* for the *auxiliary sequence* y_k.
- Taylor et al. (2015): Tightness of our relaxations and numerical bounds + smooth strongly convex case.

Concluding Remarks and Extensions

- The PEP framework offers a new approach to derive complexity bounds.
- Finding a bound for the PEP problem is challenging!
- Numerical bounds required solving SDP which dimension depends on N.

Two Very Recent Works Building on PEP:

- Kim-Fessler (MP-2015) confirmed our Conjecture 2. Also derived an efficient "Optimized" algorithm, with an *analytical bound* for the *auxiliary sequence* y_k.
- Taylor et al. (2015): Tightness of our relaxations and numerical bounds + smooth strongly convex case.

Extensions: Analyze other algorithms (e.g., constraints – done for projected gradient), and different classes $\mathcal{F}$ of input functions/optimization models...

PEP also useful as a constructive approach to design new algorithms....
In our Recent work on Nonsmooth problems we derive

an **Optimal Kelley-Like Cutting Plane Method**. [To appear in Math. Prog.]

HAPPY BIRTHDAY YURI !