An introduction to Performance Estimation of First-Order Methods

François Glineur
Université catholique de Louvain (UCLouvain)

Center for Operations Research and Econometrics and *Information and Communication Technologies, Electronics and Applied Mathematics Institute*

joint work with
Adrien B.M. Taylor (INRIA, Paris)
Julien M. Hendrickx (UCLouvain)
Funding: FNRS/FRIA

LNMB conference on the Mathematics of Operations Research
Jan. 16, 2023

What are first-order methods (FOM)?

- Continuous optimization methods

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{or} \quad \min_{x \in C \subset \mathbb{R}^n} f(x)$$

- Only use first-order information = gradient at given point

- Gradient method (Cauchy, 1847):

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k)$$

- Information about function given only by first-order oracle (this is a black-box method)

- Able to deal with unstructured problems, convergence rates require some regularity assumptions on objective f

Some comments you may have heard

FOM are slow

FOM require differentiability

FOM are really slow to converge

FOM step size is difficult to choose

FOM are unable to find global optima

Computing gradients for a FOM is expensive

FOM are an old concept, much better methods exists now

Are any of these valid ?

The seven wonders of first-order methods

Why you should probably
give first-order methods
a try for your problem

Apologies for /simplified/simplistic taglines

First wonder

FOM have performance guarantees

Performance guarantees for FOM

First-order methods require (in principle) a differentiable objective

To obtain guarantees, we need to measure **how differentiable** (how smooth) it is

Define constant L to measure how fast the gradient can change

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \qquad \text{(Lipschitz constant)}$$

Equivalently L is an upper bound on the (absolute value of) **curvature** (i.e. largest eigenvalue of hessian/second derivative):

$$-L \leq \lambda_i\{\nabla^2 f(x)\} \leq L \text{ for all } i \text{ and } x$$

If $L \to \infty$, function becomes nondifferentiable $\to$ no guarantee

Convergence guarantee for gradient method

Stepsize should be chosen as a fixed constant equal to $\frac{1}{L}$!

$$x_{k+1} = x_k - \frac{1}{L}\nabla f(x_k)$$

(in practice: dynamical estimation of L is possible)

In general (nonconvex) case, we get

$$\min_{0\leq i\leq k} \|\nabla f(x_i)\| \leq \sqrt{\frac{2L(f(x_0) - f(x^*))}{k+1}} = \mathcal{O}(\frac{1}{\sqrt{k}})$$

Only converging to stationary points (including local optima),

Actually no back-blox method can be guaranteed to find a global optimum efficiently
(theorem: exponential lower bound in n for any black-box method!)

Convergence guarantee in convex case

Things are even better when function is **convex**
(which will be assumed for most of the talk)

All local optima are automatically global

Assuming we know an upper bound R on the initial distance to the solution $\|x_0 - x^*\| \leq R$, gradient method with step $\frac{1}{L}$ now satisfies

$$f(x_k) - f^* \leq \frac{LR^2}{2k} = \mathcal{O}(\frac{1}{k})$$

Nice explicit rate, **sublinear**

Relating accuracy to number of oracle calls/gradient evaluations

Only a **worst-case** rate, still often very useful

Second wonder

FOM can deal with constraints

FOM can deal with constraints

To deal with constrained problems

$$\min_{x \in C \subset \mathbb{R}^n} f(x)$$

simply **project** on the (convex) feasible region at each iteration

$$x_{k+1} = P_C\left[x_k - \frac{1}{L}\nabla f(x_k)\right]$$

(projection computes closest point in feasible region)

Projection is computationally cheap for many useful sets:
subspaces, simplex, unit balls, separable sets, etc.

Previous convergence guarantees hold **unmodified**!

Third wonder

Using FO steps is always cheap

Computing the gradient is never too expensive

Theorem: For any computable function, the gradient can be evaluated at a cost that does not exceed a small constant (less than 5) times the cost of evaluating the function itself

This is obtained for example using **automatic differentiation**

This is *not* finite differences (and usually a much better option)

Fourth wonder

It is easy to accelerate FOM

A faster method at nearly no extra cost

Adding a **momentum** term to the method

$$y_k = x_k + \beta_k(x_k - x_{k-1})$$

$$x_{k+1} = P_C\left[y_k - \frac{1}{L}\nabla f(y_k)\right]$$

with a well-chosen parameter ($\beta_k = \frac{k-1}{k+2}$)
does not significantly increase the computational cost
but **accelerates** i.e. improves the performance guarantee to

$$f(x_k) - f^* \leq \frac{2LR^2}{(k+1)^2} = \mathcal{O}(\frac{1}{k^2})$$

(Nesterov, 1984)

Can make a huge difference in convergence speed
Rate is asymptotically optimal (no FOM can have better order)

Fifth wonder

FOM like curves

Flat is bad

FOM become faster when objective function is **never flat**

If curvature of objective is bounded below by μ

$$0 < \mu \leq \lambda_i\{\nabla^2 f(x)\} \leq L \text{ for all } i \text{ and } x$$

(i.e. f is a μ-strongly convex function) then

$$f(x_k) - f^* \leq \left(1 - \frac{\mu}{L}\right)^{2k}(f(x_0) - f^*)$$

Linear convergence, much better than all previous sublinear rates

Accelerated methods also benefit (with better rate: $\mu/L \to \sqrt{\mu/L}$)

Sixth wonder

FOM can deal with nondifferentiable problems

Not differentiable? not a problem!

For convex nondifferentiable functions, gradient method can be easily adapted by replacing the gradient by the concept of **subgradient** $\partial f(x)$ (slope of any linear lower bound)

$$x_{k+1} = P_C\Big[x_k - h_k \partial f(x_k)\Big]$$

Need to adapt stepisze (decreasing sequence)

Drawback: slower than differentiable case

$$f(x_k) - f^* \leq \frac{MR}{\sqrt{k}} = \mathcal{O}(\frac{1}{\sqrt{k}})$$

(M is maximum size of subgradient)

Smoothing to the rescue

A nondifferentable can often be approximated by a smooth function

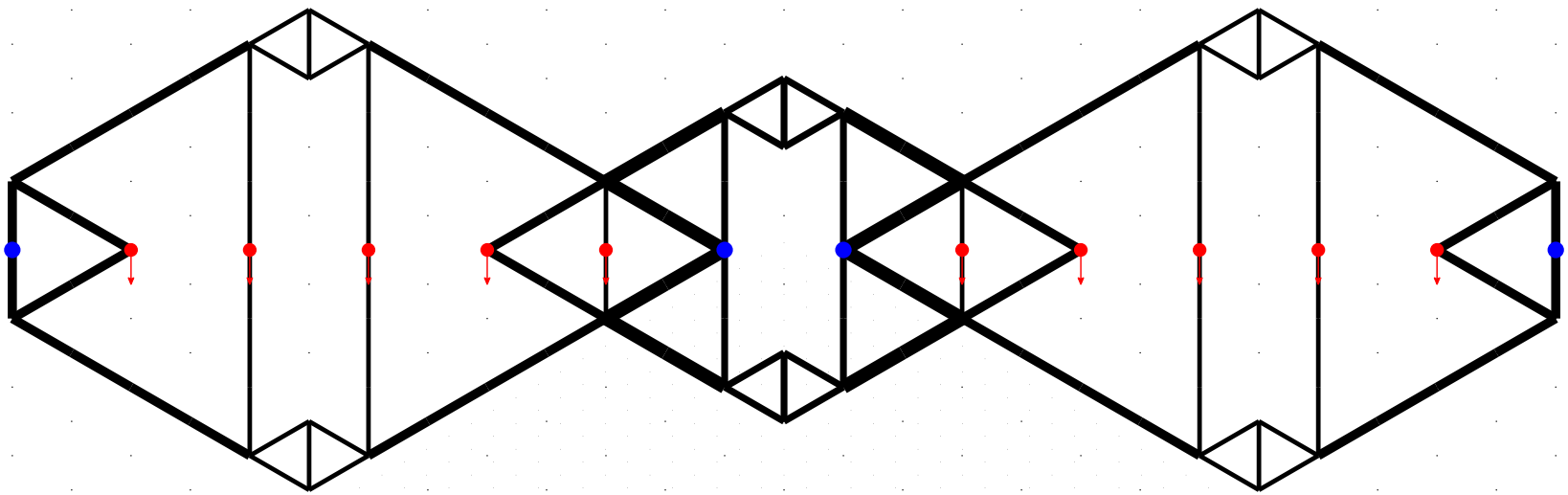
Example

$$|x| \approx \frac{1}{\rho} \log\left(e^{\rho x} + e^{-\rho x}\right)$$

Tradeoff between speed and accuracy of smooth approximation (error is $\mathcal{O}(\frac{1}{\rho})$) and its Lipschitz constant $L = \mathcal{O}(\rho)$
i.e. either fast convergence to a bad approximation, or slower convergence to a better approximate solution

Using best compromise for ρ and accelerated method will achieve a $\mathcal{O}(\frac{1}{k})$ rate (instead of $\mathcal{O}(\frac{1}{\sqrt{k}})$ for subgradient method)

Truss topology design



(can be solved with 1M potential bars in less than one hour)

Seventh wonder

FOM can deal with randomness

Randomness is no big deal for FOM

$$\min \frac{1}{|S|} \sum_{i \in S} f_i(x)$$

Instead of computing full gradient $\frac{1}{|S|} \sum_{i \in S} \nabla f_i(x)$
pick a **random** index $i \in S$ (uniformly) and
perform a **partial** gradient step with

$$x_{k+1} = x_k - h_k \nabla f_i(x)$$

This term $\nabla f_i(x)$ is an unbiased estimate of the full gradient
This **stochastic gradient** method will still converge!

Many applications in machine learning
(each f_i is the error for a different training sample)

Of course, not everything can be wonderful with FOM!

- High-accuracy solutions can be expensive to obtain (sublinear rates, or ill-conditioned linear rates $\mu/L \to 0$)

- Some functions are not smooth/regular enough
- Smoothness/regularity constants can be hard to estimate (especially lower curvature)

- Projection on some feasible sets can be too costly

- Nonconvexity (esp. combined with nonsmoothness) may still be a challenge
- Even a gradient computation can be too expensive (for huge scale problems)

Still, FOM are now inescapable in a modern optimizer's toolbox

An introduction to Performance Estimation of First-Order Methods

François Glineur
Université catholique de Louvain (UCLouvain)

Center for Operations Research and Econometrics and *Information and Communication Technologies, Electronics and Applied Mathematics Institute*

joint work with
Adrien B.M. Taylor (INRIA, Paris)
Julien M. Hendrickx (UCLouvain)
Funding: FNRS/FRIA

LNMB conference on the Mathematics of Operations Research
Jan. 16, 2023

A motivating example

Consider the standard gradient method for constrained smoooth minimization:

$$\min_{x \in \mathbb{R}^n} f(x)$$

with f convex, smooth with L-Lipschitz gradient

For $i = 0 : N - 1$

$$x_{i+1} = x_i - \frac{1}{L} \nabla f(x_i)$$

Exact convergence rate of projected gradient

$$\min_x f(x) \quad \text{from } x_0 \text{ with } x_{i+1} = x_i - \frac{1}{L}\nabla f(x_i)$$

It is well known (and proved in countless references) that

$$f(x_N) - f(x_*) \leq \frac{L\|x_0 - x_*\|^2}{2N}$$

where x_* is an optimal solution

However this worst-case convergence rate is not tight!
For example, for $N = 1$: iterate x_1 actually satisfies

$$f(x_1) - f(x_*) \leq \frac{L\|x_0 - x_*\|^2}{4}$$

(twice better!)

Bouding the error after one gradient step

Convex functions are known to satisfy

$$f(x) \geq f(y) + \nabla f(y)(x - y) \text{ for all } x, y$$

Moreover smooth convex functions actually verify condition $SC(x, y)$

$$f(x) \geq f(y) + \nabla f(y)(x - y) + \frac{\|\nabla f(x) - \nabla f(y)\|^2}{2L} \text{ for all } x, y$$

Proof simply relies on writing this condition for three pairs (x, y)

$$(x, y) \in \{(x_0, x_1), (x_*, x_0), (x_*, x_1)\}$$

Proof of the exact convergence rate of projected gradient

$$f(x) \geq f(y) + \nabla f(y)(x-y) + \frac{\|\nabla f(x) - \nabla f(y)\|^2}{2L} \text{ for all } x, y$$

We write

- $(x, y) = (x_0, x_1)$

$$f(x_0) \geq f(x_1) + \nabla f(x_1)(x_0 - x_1) + \frac{\|\nabla f(x_0) - \nabla f(x_1)\|^2}{2L}$$

- $(x, y) = (x_*, x_0)$ (since x_* optimal implies $\nabla f(x_*) = 0$)

$$f(x_*) \geq f(x_0) + \nabla f(x_0)(x_* - x_0) + \frac{\|\nabla f(x_0)\|^2}{2L}$$

- $(x, y) = (x_*, x_1)$

$$f(x_*) \geq f(x_1) + \nabla f(x_1)(x_* - x_1) + \frac{\|\nabla f(x_1)\|^2}{2L}$$

and sum these three inequalities

Proof of the exact convergence rate of gradient

We obtain

$$2f(x_*) \geq 2f(x_1) + \Big(\nabla f(x_1) - \nabla f(x_0)\Big)^T (x_0 - x_*) - 2\nabla f(x_1)^T (x_1 - x_*) + \frac{1}{L}\Big(\|\nabla f(x_0)\|^2 + \|\nabla f(x_1)\|^2 - \nabla f(x_1)^T \nabla f(x_0)\Big)$$

which can be rewritten (after dividing by 2) as

$$f(x_1) - f(x_*) \leq -\frac{1}{2}\Big(\nabla f(x_1) - \nabla f(x_0)\Big)^T (x_0 - x_*) + \nabla f(x_1)^T (x_1 - x_*) - \frac{1}{2L}\Big(\|\nabla f(x_0)\|^2 + \|\nabla f(x_1)\|^2 - \nabla f(x_1)^T \nabla f(x_0)\Big)$$

Proof of the exact convergence rate of gradient

Now use definition of the method $x_1 = x_0 - \frac{g_0}{L}$ and let (for simplicity of notation) $g_0 = \nabla f(x_0)$, $g_1 = \nabla f(x_1)$ and $d = x_0 - x_*$

Our upper bound on $f(x_1) - f(x_*)$

$$
\begin{aligned}
&- \frac{1}{2}\Big(\nabla f(x_1) - \nabla f(x_0)\Big)^T (x_0 - x_*) + \nabla f(x_1)^T (x_1 - x_*) \\
&- \frac{1}{2L}\Big(\|\nabla f(x_0)\|^2 + \|\nabla f(x_1)\|^2 - \nabla f(x_1)^T \nabla f(x_0)\Big)
\end{aligned}
$$

becomes

$$
\begin{aligned}
&- \frac{1}{2}(g_1 - g_0)^T d + g_1^T (d - \frac{g_0}{L}) - \frac{1}{2L}\Big(\|g_0\|^2 + \|g_1\|^2 - g_1^T g_0\Big) \\
&= \frac{1}{2}(g0 + g1)^T d - \frac{1}{2L}\Big(\|g_0\|^2 + \|g_1\|^2 + g_1^T g_0\Big) \\
&= \frac{1}{4}L\|d\|^2 - \frac{1}{4}L\Big\|d - \frac{g0}{L} - \frac{g1}{L}\Big\|^2 - \frac{1}{2L}\Big\|\frac{g0}{L}\Big\|^2 - \frac{1}{2L}\Big\|\frac{g1}{L}\Big\|^2
\end{aligned}
$$

Proof of the exact convergence rate of gradient

We have obtained

$$f(x_1)-f(x_*) \leq \frac{1}{4}L\|d\|^2-\frac{1}{4}L\left\|d-\frac{g0}{L}-\frac{g1}{L}\right\|^2-\frac{1}{2L}\left\|\frac{g0}{L}\right\|^2-\frac{1}{2L}\left\|\frac{g1}{L}\right\|^2$$

which establishes the claim

$$f(x_1) - f(x_*) \leq \frac{1}{4}L\|d\|^2 = \frac{L\|x_0 - x_*\|^2}{4}$$

□

Goal of this talk is to introduce and describe a systematic technique to find this type of proof, called

performance estimation

Outline

Goal

Study methods for smooth unconstrained convex minimization

$$\min_{x \in \mathbb{R}^d} f(x),$$

- function $f \in \mathcal{F}_{0,L}$: convex, L-Lipschitz gradient,
- method: first-order, oracle based, fixed-step coefficients (ability to compute f and ∇f given a x)

- Methods include fixed-step gradient, fast gradient (Nesterov acceleration), heavy ball method, etc.
- Compute exact worst-case guarantees after N iterations
- Performance criteria include for example
  - maximum possible value of $f(x_N) - f^*$
  - maximum possible distance of $\|x_N - x_*\|$
  - maximum possible value of $\max_{0 \leq k \leq N} \|\nabla f(x_k)\|$

over all functions $f \in \mathcal{F}_{0,L}$ and starting points x_0

Performance estimation problem

Formally, for a given

- problem class $\mathcal{F}$ whose members are equipped with oracle $\mathcal{O}_f$
- method $\mathcal{M}$ defined for a number of iterations N
- bound on the initial distance to the solution R
- performance criterion $\mathcal{P}$

we want to evalutate worst-case performance $w(\mathcal{F}, R, \mathcal{M}, N, \mathcal{P})$ defined as

$$\sup_{\mathbf{f},\mathbf{x_0},\ldots,\mathbf{x_N},\mathbf{x}_*} \quad \mathcal{P}(\mathcal{O}_f, x_0, \ldots, x_N, x_*) \qquad \text{(PEP)}$$

$$\text{s.t. } f \in \mathcal{F}, x_* \text{ is optimal for } f, \|x_0 - x_*\| \leq R$$

$$x_1, \ldots, x_N \text{ is generated by method } \mathcal{M} \text{ starting from } x_0,$$

Variable f is a function, and infinite-dimensional

No explicit constraint on dimension of domain of function f

How to solve this difficult optimization problem?

Let us do it step by step ...

A black-box method

First N iterates generated by a first-order black-box method $\mathcal{M}$ (N calls of the oracle), starting from initial x_0 are

$$\begin{aligned} x_1 &= \mathcal{M}_1\left(x_0, \mathcal{O}_f(x_0)\right), \\ x_2 &= \mathcal{M}_2\left(x_0, \mathcal{O}_f(x_0), \mathcal{O}_f(x_1)\right), \\ &\vdots \\ x_N &= \mathcal{M}_N\left(x_0, \mathcal{O}_f(x_0), \ldots, \mathcal{O}_f(x_{N-1})\right). \end{aligned}$$

Only depends on x_0 and the finite list of outputs from the oracle

$\rightarrow$ (PEP) can be reformulated as a finite-dimensional problem

i.e. replace (infnite-dimensional) functional variable $f \in \mathcal{F}$
by a list of N (finite-dimensional) oracle outputs $\mathcal{O}_f(x_k)$

A finite-dimensional reformulation

Since method is oracle based, (PEP) can be reformulated in a finite way using only only iterates $\{x_i\}_{i\in I}$, their function values $\{f_i\}_{i\in I}$ and their gradients $\{g_i\}_{i\in I}$ as

$$\sup_{\{\mathbf{x_i},\mathbf{g_i},\mathbf{f_i}\}_{\mathbf{i}\in\mathbf{I}}} \quad \mathcal{P}\left(\{x_i, g_i, f_i\}_{i\in I}\right), \qquad \text{(f-PEP)}$$

$$\begin{aligned} \text{s.t. } & \text{there exists } f \in \mathcal{F} \text{ such that } \mathcal{O}_f(x_i) = \{f_i, g_i\}\ \forall i \in I, \\ & x_1, \ldots, x_N \text{ is generated by method } \mathcal{M} \text{ from } x_0, \\ & \|x_0 - x_*\|_2 \leq R, \\ & g_* = 0 \end{aligned}$$

Crucial part is the first constraint, which says that $\{x_i, g_i, f_i\}_{i\in I}$ can be interpolated on $\mathcal{F} \rightarrow$ find an equivalent tractable condition for smooth convex functions

Brief history of performance estimation

- Concept of performance estimation introduced by Drori and Teboulle (2012, published 2014)
- Nonconvex formulation proposed by Drori and Teboulle $\rightarrow$ they solve a convex relaxation $\rightarrow$ only provides bounds, but shown to be tight for some algorithms (using matching worst-case functions)
- Other approaches similar in spirit:
  integral quadratic constraints by Lessard, Recht, Packard (2014); Lyapunov/potential functions by Taylor, Bach (2019)

- This talk's: an exact formulation for performance estimation of unconstrained first-order optimization algorithms

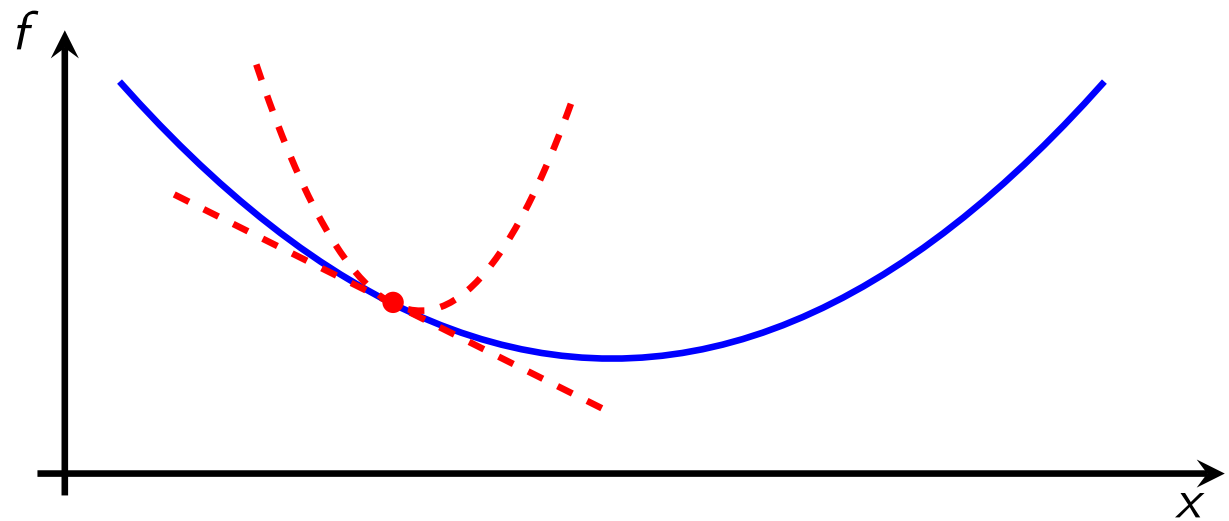
Outline

Preview of our main result

- A reformulation as a semidefinite optimization problem
  (dimension proportional to N)
- formulation is exact
  $\rightarrow$ optimal value provides the exact worst-case performance
- any dual feasible solution
  $\rightarrow$ upper bound on the worst-case performance
  can be converted into a standard proof
  (a weighted sum of valid inequalities)
- any primal feasible solution
  $\rightarrow$ lower bound on the worst case performance
  can be converted into a concrete function

The function class $\mathcal{F}_{0,L}$

A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, f is called convex and L-smooth if $\forall x, y \in \mathbb{R}^n$ we have:



(1) (Convexity) $f(x) \geq f(y) + \langle \nabla f(y), x - y \rangle$,

(2) (L-smoothness) $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$,

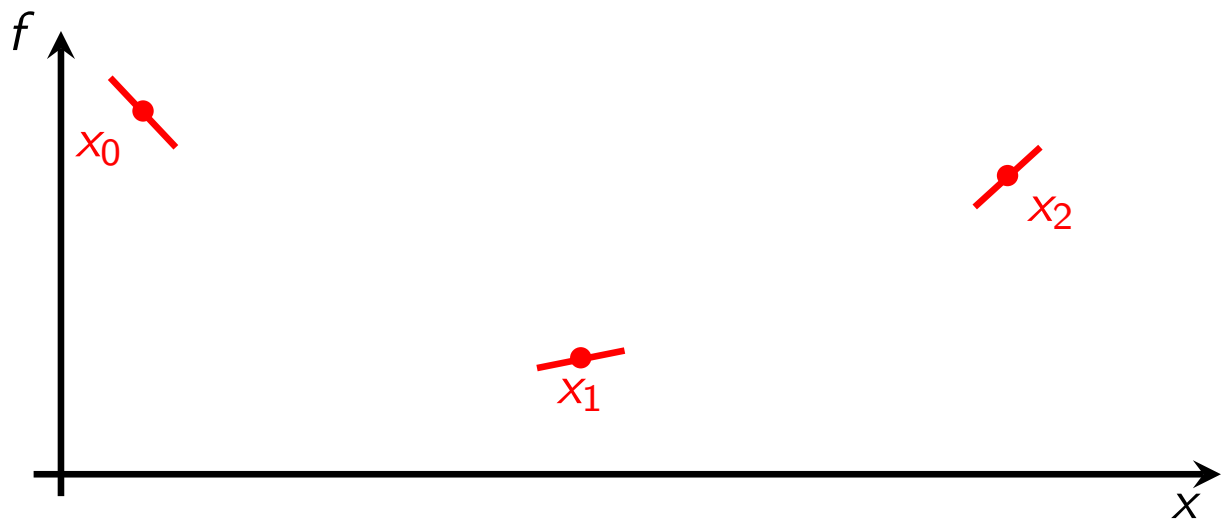
(2b) (L-smoothness) $f(x) \leq f(y) + \langle \nabla f(y), x - y \rangle + \frac{L}{2}\|x - y\|^2$.

Smooth convex interpolation problem

Consider a set S, and its associated values $\{(x_i, g_i, f_i)\}_{i \in S}$ with coordinates x_i, subgradients g_i and function values f_i.

- Is there $f \in \mathcal{F}_{0,L}$ (L-Lipschitz gradient, convex) s.t.

$$f(x_i) = f_i, \quad \text{and} \quad g_i \in \partial f(x_i), \qquad \forall i \in S.$$



- We want necessary and sufficient conditions for existence of f
- These conditions will appear as a constraints in our PEP formulation

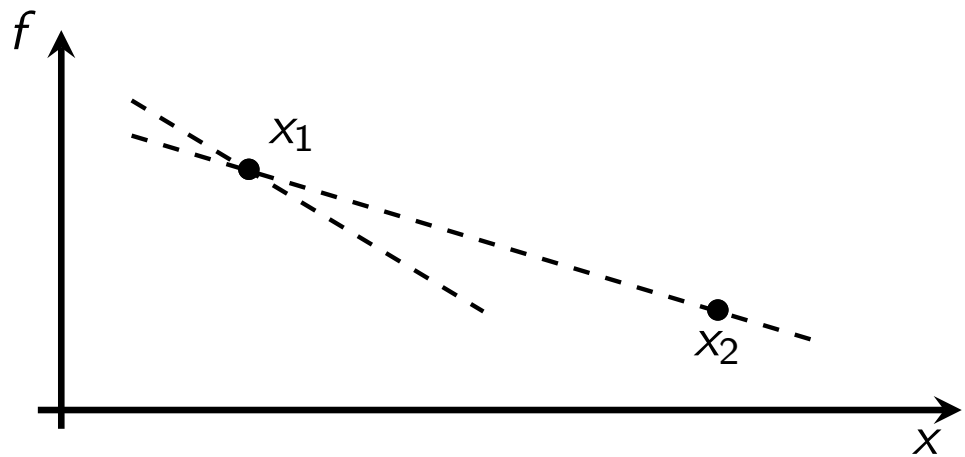
Smooth convex interpolation ($L < \infty$)

First attempt: following set conditions is necessary

$$f_i \geq f_j + g_j^T(x_i - x_j), \qquad i, j \in S, \qquad \text{(C1)}$$
$$||g_i - g_j||_2 \leq L||x_i - x_j||_2.$$

but not sufficient!

$(x_1, g_1, f_1) = (-1, -2, 1)$
$(x_2, g_2, f_2) = (0, -1, 0)$



satisfies (C1) with $L = 1$ but cannot be differentiable...

(of course set of conditions does work if set S is whole domain)

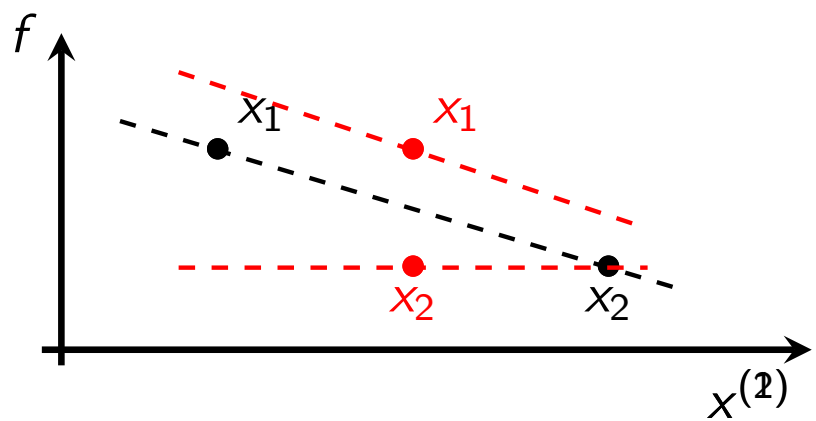
Smooth convex interpolation ($L < \infty$)

Second attempt: following set conditions is necessary

$$f_i \geq f_j + g_j^T(x_i - x_j), \qquad i, j \in S, \qquad \text{(C2)}$$
$$f_i \leq f_j + g_j^T(x_i - x_j) + \frac{L}{2}||x_i - x_j||_2^2.$$

but not sufficient!

$$(x_1, g_1, f_1) = \left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}, 0\right)$$
$$(x_2, g_2, f_2) = \left(\begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}1\\\alpha\end{pmatrix}, 1\right)$$



satisfies (C2) but do not even satisfy the basic conditions (C1)...

conditions (C2) may also satisfy previous example for some L...

A different approach

Idea: reduce smooth convex interpolation to convex interpolation.

Using two basic operations to transform the problem:

- Conjugation: f is closed, proper and convex, then:
  f L-Lipschitz gradient $\Leftrightarrow$ f^* $\frac{1}{L}$-strongly convex

  $(x_i, g_i, f_i)_{i\in S}$ is $\mathcal{F}_{0,L}$-interp. $\Leftrightarrow$ $(g_i, x_i, x_i^T g_i - f_i)_{i\in S}$ is $\mathcal{F}_{1/L,\infty}$-interp.

- Minimal curvature subtraction:
  $f(x)$ μ-strongly convex $\Leftrightarrow$ $f(x) - \frac{\mu}{2}||x||_2^2$ convex

  Since $\nabla\left(f(x) - \frac{\mu}{2}||x||^2\right) = \nabla f(x) - \mu x$ we have

$$
\begin{aligned}
&(x_i, g_i, f_i)_{i\in S} \text{ is } \mathcal{F}_{\mu,L}\text{-interpolable} \\
\Leftrightarrow &(x_i, g_i - \mu x_i, f_i - \frac{\mu}{2}||x_i||^2)_{i\in S} \text{ is } \mathcal{F}_{0,L-\mu}\text{-interpolable}
\end{aligned}
$$

Reminder: Conjugation

Given a proper function $f : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$, the (Legendre-Fenchel) conjugate of f is defined as:

$$f^*(y) = \sup_{x \in \mathbb{R}^d} y^T x - f(x),$$

with $f^* \in \mathcal{F}_{0,\infty}$ (proper, closed and convex).

For $f \in \mathcal{F}_{0,\infty}$, we have a one-to-one correspondence between f and f^*, and the following propositions are equivalent:

(a) $f(x) + f^*(g) = g^T x$,
(b) $g \in \partial f(x)$,
(c) $x \in \partial f^*(g)$.

For $f \in \mathcal{F}_{0,\infty}$, we have: $f \in \mathcal{F}_{0,L} \Leftrightarrow f^* \in \mathcal{F}_{1/L,\infty}$

$$(x_i, g_i, f_i)_{i \in S} \text{ is } \mathcal{F}_{0,L}\text{-interpolable}$$
$$\Leftrightarrow (g_i, x_i, x_i^T g_i - f_i)_{i \in S} \text{ is } \mathcal{F}_{1/L,\infty}\text{-interpolable}$$

Smooth convex interpolation

- Correct necessary and sufficient conditions are given by following Theorem [Taylor, Hendrickx, G. 2016]

- Set $\{(x_i, g_i, f_i)\}_{i \in S}$ is $\mathcal{F}_{0,L}$-interpolable if and only

$$f_j \geq f_i + g_i^T (x_j - x_i) + \frac{1}{2L} ||g_i - g_j||_2^2 \qquad \forall i, j \in S$$

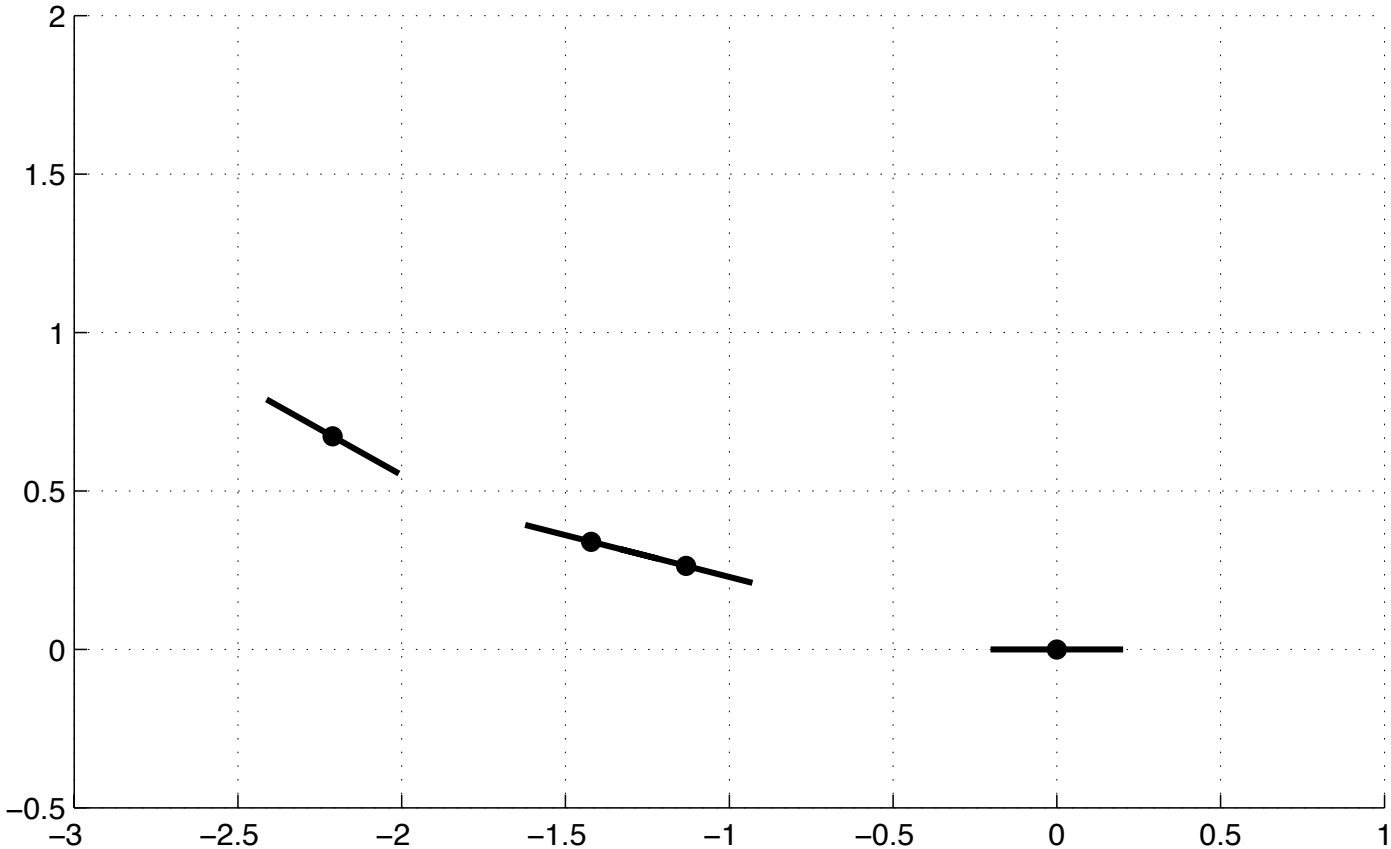
holds for every pair of indices $i \in I$ and $j \in S$

- These correspond to (slighlty less) well-known necessary conditions for L-smoothness (but only those are also sufficient)

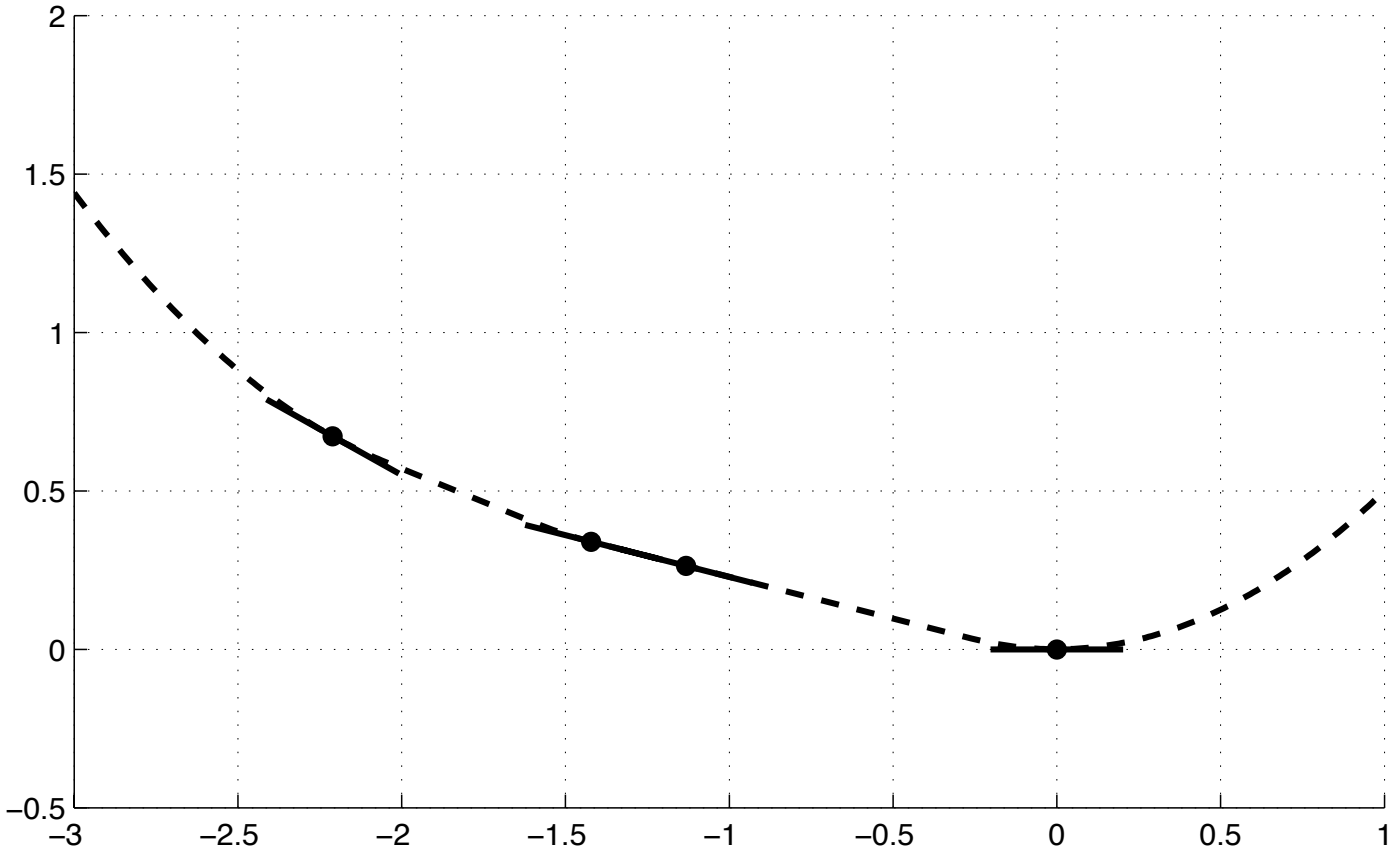
$$f(x) \geq f(y) + \nabla f(y)^T (x - y) + \frac{1}{2L} \|\nabla f(x) - \nabla f(y)\|^2$$

- Proof provides an explicit interpolating function (piecewise linear-quadratic, not unique)

Example



Example



First-order methods : a brief introduction

Introduction to performance estimation

A convex formulation for performance estimation

- Smooth convex interpolation
- Semidefinite optimization formulation

Performance estimation of standard algorithms

A semidefinite optimization formulation

Let $I = \{0, 1, \ldots, N, *\}$, assume w.l.o.g. $f^* = 0$, $x^* = 0$, $g^* = 0$

Our performance estimation problem is now

$$\sup_{\{x_i, g_i, f_i\}_{i \in I \setminus \{*\}}} \quad \mathcal{P}\left(\{x_i, g_i, f_i\}_{i \in I}\right), \qquad \text{(f-PEP2)}$$

$$\text{such that } \{x_i, g_i, f_i\}_{i \in I} \text{ is } \mathcal{F}_{\mu,L}\text{-interpolable},$$

$$x_1, \ldots, x_N \text{ is generated by method } \mathcal{M},$$

$$\|x_0 - x_*\|_2 \leq R.$$

- We want an exact and convex reformulation

Fixed-step first-order algorirhms

- In order to deal with the method constraint, we only consider fixed-step first-order methods

$$x_i = x_0 - \frac{1}{L} \sum_{k=0}^{i-1} h_{i,k} g_k$$

  where $h_{i,k}$ and fixed constants
- Iterates are defined by linear expressions involving past oracle outputs
- Many classic, standard black-box methods can be reformulated in this way (including methods based on multiple sequences)
- Recursively express all iterates in terms of gradients (and initial iterate x_0)
- This leads to tractable linear equalities in our formulation, involving variables x_i and g_i only

Simple example: PEP for gradient method

$$\max_{f,x_0,\ldots,x_N,x^*} f(x_N) - f^*,$$

subject to the constraints

$f \in \mathcal{F}_{0,L}$ i.e. convex and has L-Lipschitz gradient

$\nabla f(x^*) = 0,$

$x_{i+1} = x_i - \frac{h}{L}\nabla f(x_i),$

$||x_0 - x^*||_2^2 \leq R^2.$

can be reformulated as finite-dimensional problem using our $\mathcal{F}_{0,L}$-interpolation conditions

Formulation for performance optimization

Worst-case estimation problem translates into an exact finite problem

$$\max\ f_N - f^*$$

$$\text{s.t. } f_j \geq f_i + g_i^T(x_j - x_i) + \frac{1}{2L}||g_i - g_j||_2^2 \qquad \forall i \neq j \in \{0, ..., N, *\}$$

$$x_{i+1} = x_i - \frac{h}{L}g_i \qquad \forall i \in \{0, ..., N-1\}$$

$$(x_0 - x^*)^T(x_0 - x^*) \leq R^2$$

- Only remaining difficulty: scalar products, i.e. nonconvex quadratic constraints
- Solution: Gram matrix $G \succeq 0$, containing all scalar products which turns the problem turns into an equivalent semidefinite optimization problem

Formulation for performance optimization

Example: $N = 1$ iteration, assume wlog $x^* = g^* = 0$, and choose

$$P = (x_0 \mid x_1 \mid g_0 \mid g_1)$$

$$G = P^T P = \begin{pmatrix} x_0^T x_0 & x_0^T x_1 & x_0^T g_0 & x_0^T g_1 \\ x_1^T x_0 & x_1^T x_1 & x_1^T g_0 & x_1^T g_1 \\ g_0^T x_0 & g_0^T x_1 & g_0^T g_0 & g_0^T g_1 \\ g_1^T x_0 & g_1^T x_1 & g_1^T g_0 & g_1^T g_1 \end{pmatrix} \succeq 0.$$

Our interpolating constraints become linear in the elements of G
From $G \succeq 0$, we can recover x_0, x_1, g_0 and g_1.

Function f has d variables $\Leftrightarrow \operatorname{rank}(G) \leq d$
Formulation is exact and dimension-free in large-scale case, i.e.

$$N \ll d \quad \text{(actually when } 2N + 2 \leq d\text{)}$$

Final semidefinite formulation for performance optimization

Assuming the performance criterion idepends linearly on function values f_i and elements of G ($g_i^T g_j$ and $x_0^T g_j$) we now have

$$\begin{aligned} \max_{G \in \mathbb{S}^{N+2}, f \in \mathbb{R}^{N+1}} \quad & b^T f + \mathrm{Tr}(CG) && \text{(sdp-PEP)} \\ \text{s.t.} \quad & f_j - f_i + \mathrm{Tr}\left(GA_{ij}\right) \leq 0, && \text{for all } i,j \in I, \\ & \mathrm{Tr}\left(GA_R\right) - R^2 \leq 0, \\ & G \succeq 0. \end{aligned}$$

- Constant data matrices A_{ij}, and A_R that depend on method $\mathcal{M}$ (i.e. coefficients $h_{i,k}$) and function class parameters L, μ
- Exact formulation, matrix variable G is size $N+2$, has $\mathcal{O}(N^2)$ linear constraints
- Solution of (primal) provides an explicit function attaining the worst-case performance (using interpolation of the solution) $\rightarrow$ optimal value $=$ lower bound on worst-case performance

Obtaining a proof for the exact worst-case performance

To obtain a proof that computed value is an upper bound on the worst-case performance, use solution of the dual SDP problem

$$\inf_{S,\lambda_{ij},\tau} \ \tau R^2 \ \text{ such that } \ \tau A_R - C + \sum_{i,j\in I} \lambda_{ij} A_{ij} = S, \quad \text{(d-sdp-PEP)}$$

$$\sum_{i,j\in I} \lambda_{ij}(u_j - u_i) = b,$$

$$S \succeq 0,$$

$$\lambda_{ij} \geq 0, \qquad i,j \in I,$$

$$\tau \geq 0,$$

to derive a bound in the spirit of the introductory example

Obtaining a proof for the exact worst-case performance

- Dual solution will provide a coefficient λ_{ij} for each interpolating condition
  $\rightarrow$ weights for the list of inequalities valid for each pairs of iterates

  In the example: $\lambda_{01} = \lambda_{*0} = \lambda_{*1} = \frac{1}{2}$ and $\lambda_{10} = 0$, leading to a sum of three valid inequalities $SC(i,j) \geq 0$

$$\frac{1}{2}SC(0,1) + \frac{1}{2}SC(*,0) + \frac{1}{2}SC(*,1) \geq 0$$

Obtaining a proof for the exact worst-case performance

- Dual solution will provide a positive semidefinite slack matrix S

  $\rightarrow$ quadratic form (in the x_i and g_i) expressing the slack for the weighted sum of valid inequalities will be nonnegative

  In the example $0 \leq \frac{1}{2}SC(0,1) + \frac{1}{2}SC(*,0) + \frac{1}{2}SC(*,1) = \frac{1}{4}L\|d\|^2 - \Big(f(x_1) - f(x_*)\Big) - QUAD(x,g)$ with

$$QUAD(x,g) = \frac{L}{4}\left\|d - \frac{g0}{L} - \frac{g1}{L}\right\|^2 + \frac{1}{2L}\left\|\frac{g0}{L}\right\|^2 + \frac{1}{2L}\left\|\frac{g1}{L}\right\|^2 \geq 0$$

First-order methods : a brief introduction

Introduction to performance estimation

A convex formulation for performance estimation

Performance estimation of standard algorithms

- Gradient method
- Other methods and criteria

Performance estimation of standard algorithms

- We compute numerical results from our formulation (for values of N up to a few dozens/hundreds)
- In many cases we can identify the analytical form of a primal solution valid for all N (hence an explicit function f)
- This gives us rigorous lower bounds on the worst-case performance for all N

- Dependence on some constants can be derived a priori using homogeneity considerations
- In several cases we can also identify the analytical form of a dual solution with matching objective function for all $N \rightarrow$ rigorous proof of the worst-case
- If no analytical dual solution: (one-sided) conjectures strongly supported by numerical evidence

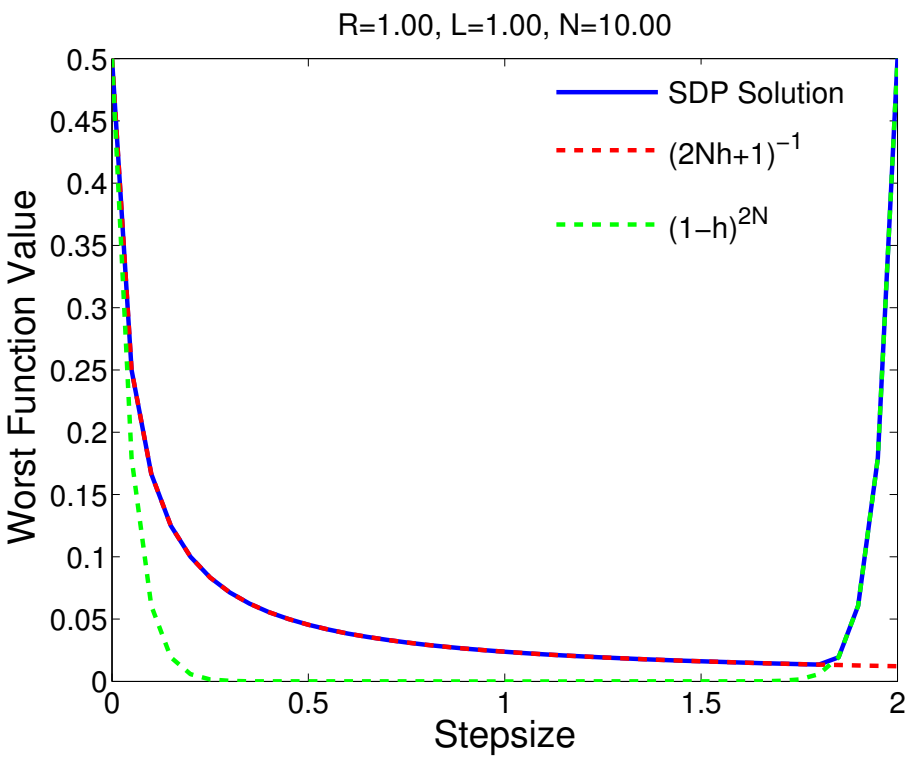
Gradient method: results

Criterion $f_N - f^*$ and gradient method with constant step-size h

$$x_{i+1} = x_i - \frac{h}{L} g_i.$$

Our results match, for all tested $0 < h < 2$ and $1 \leq N \leq 100$

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



Conjectured by Drori and Teboulle, 2014, who proved it analytically for $h \leq 1$

Suggests two worst-case regimes

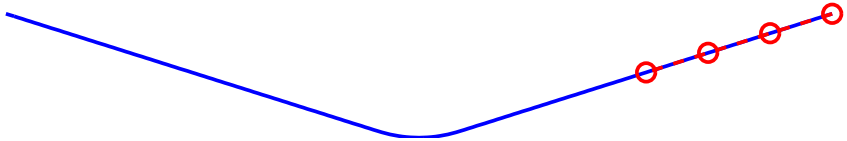
Class theoretical bound for $h = 1$ from literature is

$$\frac{LR^2}{2}\frac{1}{N} \text{ (2× worse)}$$

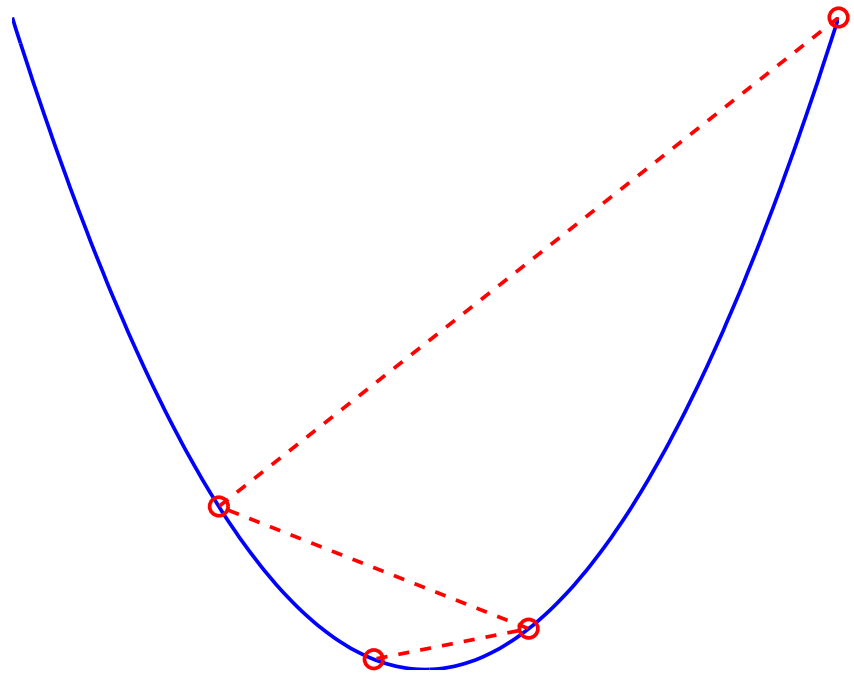
Intuition for each term in the bound

Each term corresponds to an explicit worst-case function
(which one is active depends on h and N)

These are very simple: 1D and piecewise linear-quadratic



Stays on linear part until last iteration



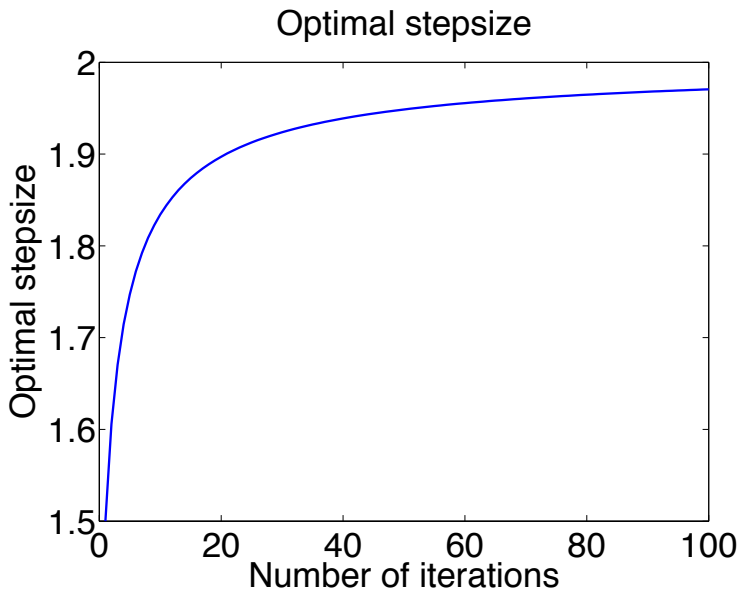
Purely quadratic, controlled overshooting at each iteration

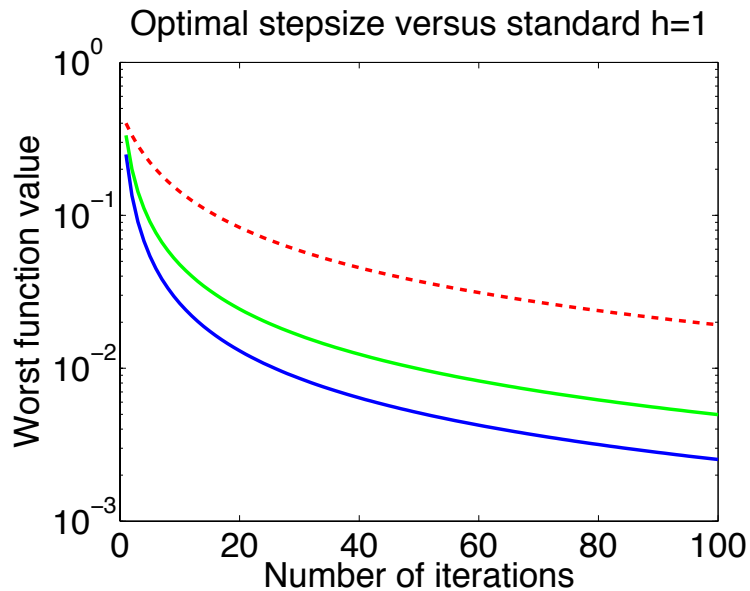
Optimal gradient step-size

Using the conjectured worst-case, compute optimal $h_{\text{opt}}(N)$

$$2-\frac{\log 4N}{2N} \sim 1+(1+4N)^{-1/(2N)} \leq h_{\text{opt}}(N) \leq 1+(1+2N)^{-1/(2N)} \sim 2-\frac{\log 2N}{2N}.$$

(equalize both terms in bounds, no closed-form solution)





(red: $h = 1$ classic bound, green: $h = 1$ exact bound, blue: optimal h exact bound)

Optimal step-size tends quickly to 2

Exact bound for h_{opt} approximately twice better than for $h = 1$

A few words about numerics

Problems solved with YALMIP+MOSEK, verified with interval-arithmetic VSDP toolbox

N	h_{opt}	Conjecture	DT relaxation	Rel. error	SDP-PEP	Rel. error
1	1.5000	$LR^2/8.00$	$LR^2/8.00$	0.00	$LR^2/8.00$	7e-09
2	1.6058	$LR^2/14.85$	$LR^2/14.54$	2e-02	$LR^2/14.85$	5e-09
5	1.7471	$LR^2/36.94$	$LR^2/32.57$	1e-01	$LR^2/36.94$	1e-08
10	1.8341	$LR^2/75.36$	$LR^2/59.80$	3e-01	$LR^2/75.36$	3e-08
20	1.8971	$LR^2/153.77$	$LR^2/109.58$	4e-01	$LR^2/153.77$	6e-08
30	1.9238	$LR^2/232.85$	$LR^2/156.23$	5e-01	$LR^2/232.85$	7e-08
40	1.9388	$LR^2/312.21$	$LR^2/201.10$	6e-01	$LR^2/312.21$	3e-08
50	1.9486	$LR^2/391.72$	$LR^2/244.70$	6e-01	$LR^2/391.72$	1e-07
100	1.9705	$LR^2/790.22$	$LR^2/451.72$	7e-01	$LR^2/790.22$	1e-07

Table: Gradient Method for convex L-smooth function, worst-case computed with relaxation from Drori and Teboulle and worst-case obtained by exact formulation (SDP-PEP) for the criterion $f(x_N) - f^*$.

Results on gradient norm

- Residual gradient norm $\|\nabla f(x_N)\|$, smooth case

$$\max \ \|\nabla f(x_N)\|_2 = LR \max\left(\frac{1}{Nh+1}, |1-h|^N\right)$$

- Simple $1D$ piecewise linear-quadratic solutions in all cases (different for each case)
- All results lead to optimal step-sizes

Nesterov's accelerated gradient method

Belongs to the class of fixed-step methods
Algorithm:

Initialization: $x_1 = x_0 - \frac{g_0}{L}$, $t_1 = 1$:

For $i = 1 : N - 2$

$$t_{i+1} = \frac{1 + \sqrt{1 + 4t_i^2}}{2}$$

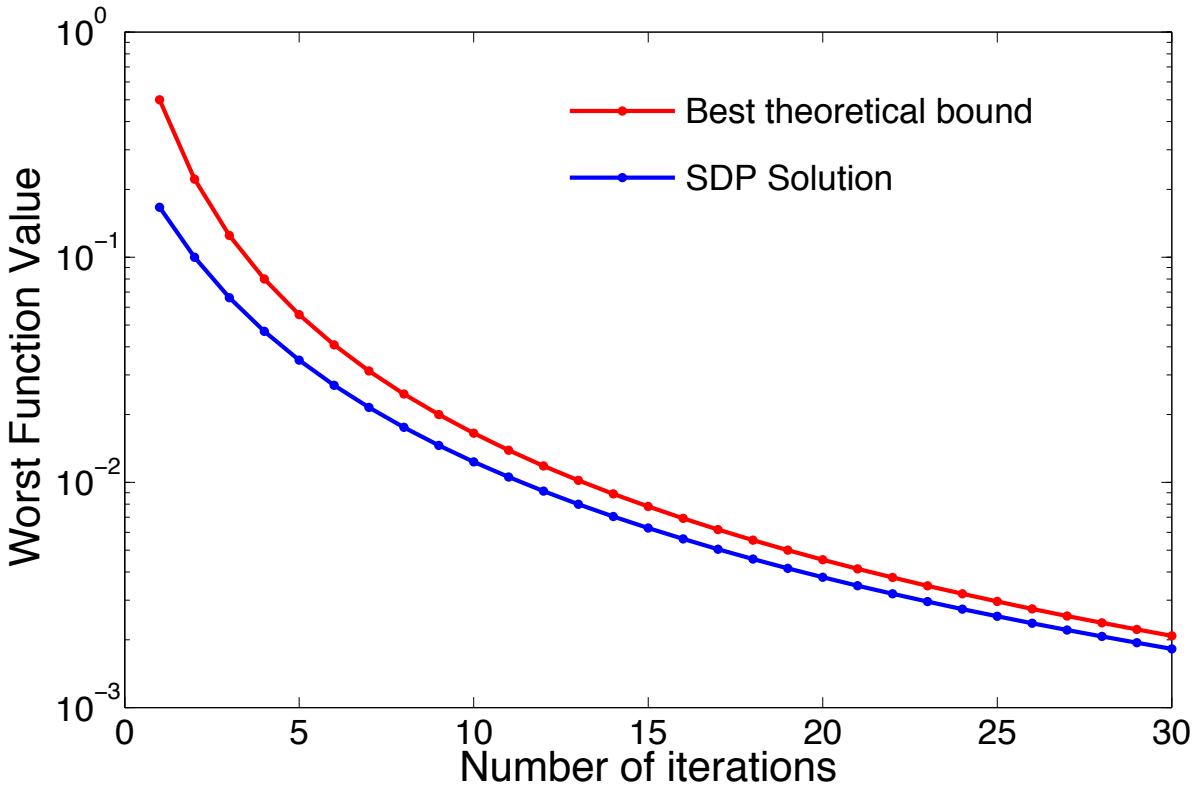
$$x_{i+1} = x_i - \left(1 + \frac{t_i - 1}{t_{i+1}}\right) \frac{g_i}{L} + \frac{t_i - 1}{t_{i+1}} (x_i - x_{i-1}) + \frac{t_i - 1}{t_{i+1}} \frac{g_{i-1}}{L}$$

Termination: $x_N = x_{N-1} - \frac{g_{N-1}}{L}$

(nonstandard description, obtained from standard two-sequence algorithm)

Nesterov's accelerated gradient method

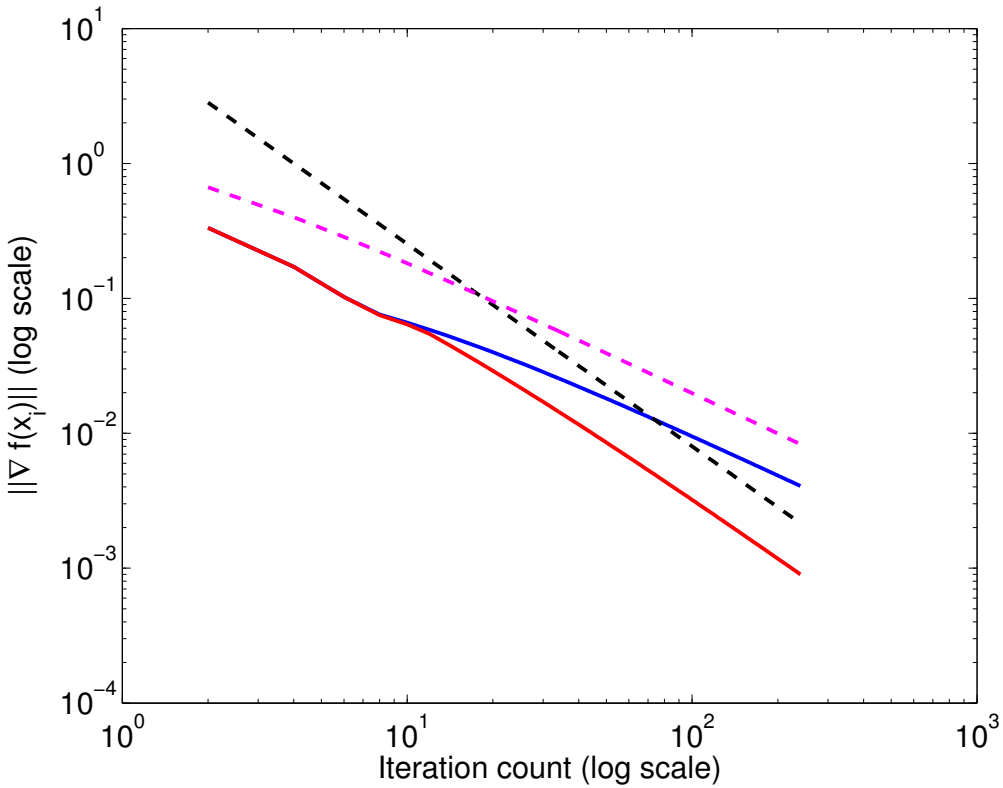
Accelerated gradient with $L = 1$, $R = 1$ and $\mu = 0$
Known theoretical bound: $f_N - f^* \leq \frac{2LR^2}{(N+1)^2}$



Relatively modest improvement ($\approx 15\%$ better)

Smallest gradient norm among all iterates

- Dual methods care about gradient norm ($\leftarrow$ primal feasibility)
- For accelerated gradient methods, objective function accuracy is $\mathcal{O}(\frac{1}{k^2})$ but norm of residual gradient $\|\nabla f(x_N)\|$ is only $\mathcal{O}(\frac{1}{k})$
- However this norm is not necessarily decreasing monotonically!
- We can compute worst-case performance for $\min_{0\leq i\leq N} ||\nabla f(x_i)||_2^2$ which is still semidefinite-representable



Suggests $\mathcal{O}(\frac{1}{k^{3/2}})$ rate for accelerated gradient (red) (previously known only for modified ad hoc method)

(dashed: standard bound ; blue: exact last iterate ; red: exact best iter.)

Performance estimation: take-home messages

- Worst-case behaviour of first-order methods can be computed exactly using semidefinite optimization
- For any fixed-coefficient first-order method after any given number of iterations on the class of smooth convex objective with given parameters
- Methodology provides easy-to-check (but not very intuitive) proofs and explicit examples of worst-case functions
- Very flexible: choice of performance criteria (e.g. objective value, gradient norm); can be extended to many settings such as constrained, nonsmooth, proximal, non-convex, stochastic, block-coordinate, etc. (see tomorrow's talk)

Thank you for your attention!

See you tomorrow morning for (many) extensions and recent results and a presentation of two software toolboxes: PESTO and PEPit
https://github.com/PerformanceEstimation

References (unconstrained case):

Performance of first-order methods for smooth convex minimization: a novel approach, Yoel Drori and Marc Teboulle., Math.Prog.vol.145 issue 1 (June 2014)

Smooth strongly convex interpolation and exact worst-case performance of first-order methods, Adrien B. Taylor, Julien M. Hendrickx, François Glineur, Math. Prog. vol. 161 issue 1 (Jan. 2017)

Thank you for your attention!

See you tomorrow morning for (many) extensions and recent results and a presentation of two software toolboxes: PESTO and PEPit
https://github.com/PerformanceEstimation

References (alternative approaches):

Analysis and design of optimization algorithms via integral quadratic constraints, Laurent Lessard, Benjamin Recht and Andrew Packard, SIAM J. on Optimization, 26(1), pp.57-95.

Stochastic first-order methods: non-asymptotic and computer-aided analyses via potential functions, Adrien Taylor and Francis Bach. Conf. on Learning Theory, PMLR, 2019, pp.2934-2992.