Smooth Strongly Convex Interpolation and Exact Worst-case Performance 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 and Julien M. Hendrickx

Optimization and Big Data workshop   Edinburgh, May 7th 2015

UCL   CORE

Take-home messages

- Worst-case behaviour of first-order methods can be computed exactly using semidefinite optimization

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
  (smoothness and strong convexity)

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
(smoothness and strong convexity)

- Methodology provides easy-to-interpret proofs
and explicit examples of worst-case functions

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
(smoothness and strong convexity)

- Methodology provides easy-to-interpret proofs
and explicit examples of worst-case functions

- Flexible approach: includes several performance criteria
(objective value, gradient norm, etc.) and can be extended to
constrained, proximal, linear minimization oracle settings

Outline

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

- function $f \in \mathcal{F}_{\mu,L}$: μ-strongly convex, L-Lipschitz gradient,

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

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

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

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

- Methods include for example

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

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

- Methods include for example
  - fixed-step gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

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

- Methods include for example
  - fixed-step gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$
  - accelerated gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k) - \beta_k \nabla f(x_{k-1}) - \gamma_k (x_k - x_{k-1})$

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

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

- Methods include for example
  - fixed-step gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$
  - accelerated gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k) - \beta_k \nabla f(x_{k-1}) - \gamma_k (x_k - x_{k-1})$
- Compute exact worst-case guarantees after N iterations

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

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

- Methods include for example
  - fixed-step gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$
  - accelerated gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k) - \beta_k \nabla f(x_{k-1}) - \gamma_k (x_k - x_{k-1})$
- Compute exact worst-case guarantees after N iterations
- Performance criteria include for example

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

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

- Methods include for example
  - fixed-step gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$
  - accelerated gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k) - \beta_k \nabla f(x_{k-1}) - \gamma_k (x_k - x_{k-1})$
- Compute exact worst-case guarantees after N iterations
- Performance criteria include for example
  - an upper bound on $f(x_N) - f^*$

Context

Goal: study methods designed to solve unconstrained smooth convex minimization

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

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

- Methods include for example
  - fixed-step gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$
  - accelerated gradient: $x_{k+1} = x_k - \alpha_k \nabla f(x_k) - \beta_k \nabla f(x_{k-1}) - \gamma_k (x_k - x_{k-1})$
- Compute exact worst-case guarantees after N iterations
- Performance criteria include for example
  - an upper bound on $f(x_N) - f^*$
  - an upper bound on $\max_{0\leq k\leq N} \|\nabla f(x_k)\|$

Performance estimation problem

Formally, for a given

- problem class $\mathcal{F}$ whose members are equipped with oracle $\mathcal{O}_f$

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

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 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 $\mathcal{P}$

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

$$\begin{aligned} \sup_{f, x_0, \ldots, x_N, x_*} \quad & \mathcal{P}(\mathcal{O}_f, x_0, \ldots, x_N, x_*) \\ \text{s.t.} \quad & f \in \mathcal{F}, x_* \text{ is optimal for } f, \|x_0 - x_*\|_2 \leq R \\ & x_1, \ldots, x_N \text{ is generated by method } \mathcal{M} \text{ starting from } x_0, \end{aligned} \tag{PEP}$$

Variable f is infinite-dimensional
No explicit constraint on dimension of domain of function f

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

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

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

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 and strongly convex functions

Formulation intially introduced by Drori and Teboulle (2014)

Preview of our main result

- A reformulation as a semidefinite optimization problem (dimension proportional to N)

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

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 (series of valid inequalities)

Preview of our main result

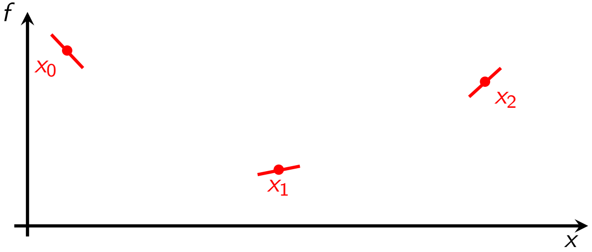
- A reformulation as a semidefinite optimization problem (dimension proportional to N)
- formulation is exact
  $\rightarrow$ optimal value provides the exact worst-case performance
- any dual feasible solution
  $\rightarrow$ upper bound on the worst-case performance
  can be converted into a standard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

Exact worst-case performance of first order methods

Outline

Smooth Strongly 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.

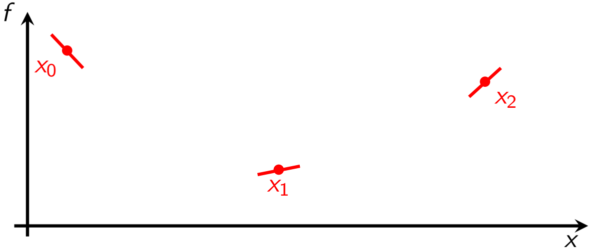


- Is there $f \in \mathcal{F}_{\mu,L}$ (L-Lipschitz gradient, μ-strongly convex) s.t.

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

Smooth Strongly 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 $f \in \mathcal{F}_{\mu,L}$ (L-Lipschitz gradient, μ-strongly convex) s.t.

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

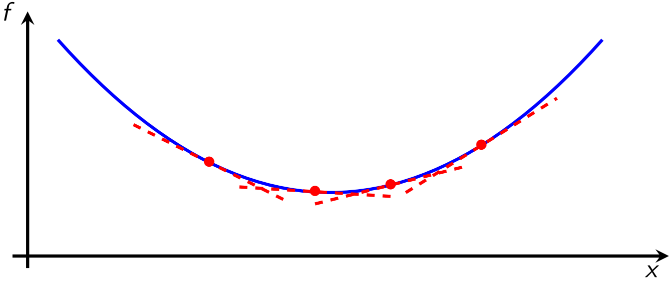
- We want necessary and sufficient conditions for existence of f

Simple case: convex interpolation ($L = \infty, \mu = 0$)

Conditions for $\{(x_i, g_i, f_i)\}_{i \in S}$ to be interpolable by a function $f \in \mathcal{F}_{0,\infty}$ (proper, closed and convex function) ?

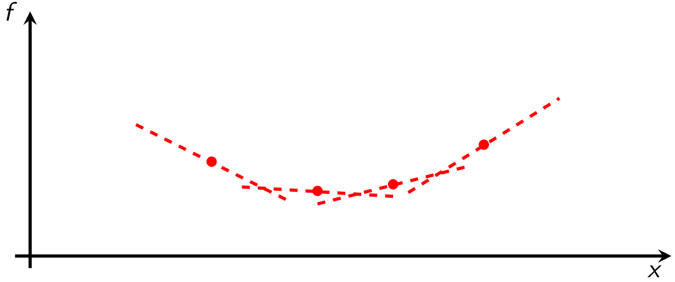
Simple case: convex interpolation ($L = \infty, \mu = 0$)

Conditions for $\{(x_i, g_i, f_i)\}_{i \in S}$ to be interpolable by a function $f \in \mathcal{F}_{0,\infty}$ (proper, closed and convex function) ?



Simple case: convex interpolation $(L = \infty, \mu = 0)$

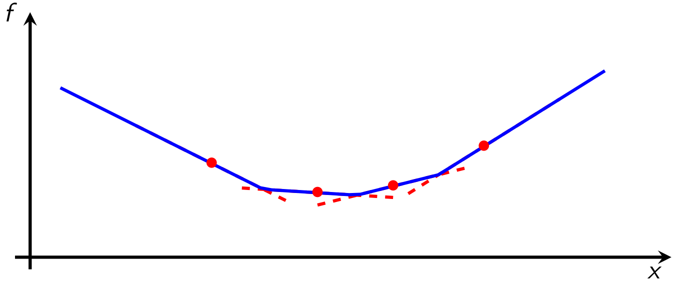
Conditions for $\{(x_i, g_i, f_i)\}_{i\in S}$ to be interpolable by a function $f \in \mathcal{F}_{0,\infty}$ (proper, closed and convex function) ?



Conditions $f_i \geq f_j + g_j^T(x_i - x_j)$ is nec.

Simple case: convex interpolation ($L = \infty, \mu = 0$)

Conditions for $\{(x_i, g_i, f_i)\}_{i \in S}$ to be interpolable by a function $f \in \mathcal{F}_{0,\infty}$ (proper, closed and convex function) ?



Conditions $f_i \geq f_j + g_j^T(x_i - x_j)$ is nec. and suff.

Explicit construction:

$$f(x) = \max_j \left\{ f_j + g_j^T(x - x_j) \right\},$$

Not unique.

Next case: smooth convex interpolation ($L < \infty, \mu = 0$)

Generalization to smooth interpolation ? Interpolation by a function $f \in \mathcal{F}_{0,L}$ (proper, closed and convex function with L-Lipschitz gradient).

Next case: smooth convex interpolation ($L < \infty, \mu = 0$)

Generalization to smooth interpolation ? Interpolation by a function $f \in \mathcal{F}_{0,L}$ (proper, closed and convex function with L-Lipschitz gradient).

First attempt: following set conditions is necessary

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

Next case: smooth convex interpolation ($L < \infty, \mu = 0$)

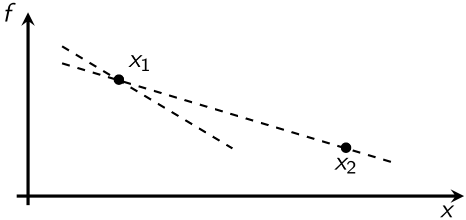
Generalization to smooth interpolation ? Interpolation by a function $f \in \mathcal{F}_{0,L}$ (proper, closed and convex function with L-Lipschitz gradient).

First attempt: following set conditions is necessary

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

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...

Next case: smooth convex interpolation ($L < \infty, \mu = 0$)

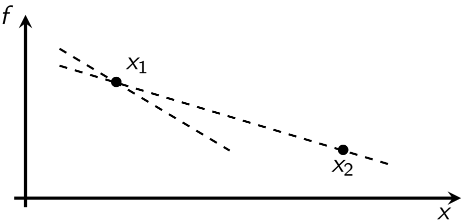
Generalization to smooth interpolation ? Interpolation by a function $f \in \mathcal{F}_{0,L}$ (proper, closed and convex function with L-Lipschitz gradient).

First attempt: following set conditions is necessary

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

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 conditions work if set S is whole domain)

Smooth convex interpolation ($L < \infty, \mu = 0$)

Generalization to smooth interpolation ? Interpolation by a function $f \in \mathcal{F}_{0,L}$ (proper, closed and convex function with L-Lipschitz gradient).

Second attempt: following set conditions is necessary

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

Smooth convex interpolation ($L < \infty, \mu = 0$)

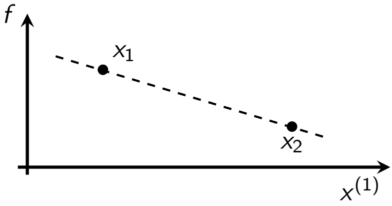
Generalization to smooth interpolation ? Interpolation by a function $f \in \mathcal{F}_{0,L}$ (proper, closed and convex function with L-Lipschitz gradient).

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!

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



Smooth convex interpolation ($L < \infty, \mu = 0$)

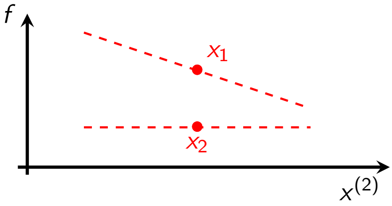
Generalization to smooth interpolation ? Interpolation by a function $f \in \mathcal{F}_{0,L}$ (proper, closed and convex function with L-Lipschitz gradient).

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)...

Smooth convex interpolation ($L < \infty, \mu = 0$)

Generalization to smooth interpolation ? Interpolation by a function $f \in \mathcal{F}_{0,L}$ (proper, closed and convex function with L-Lipschitz gradient).

Second attempt: following set conditions is necessary

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

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.

Basic operations needed in order to transform the problem:

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

  (see later)

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

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

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

Conjugation

Consider 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) $f(x) + f^*(g) = g^T x$,

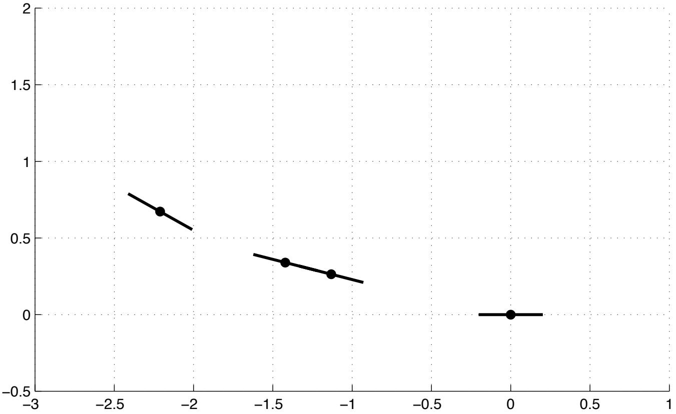
(b) $g \in \partial f(x)$,

(c) $x \in \partial f^*(g)$.

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

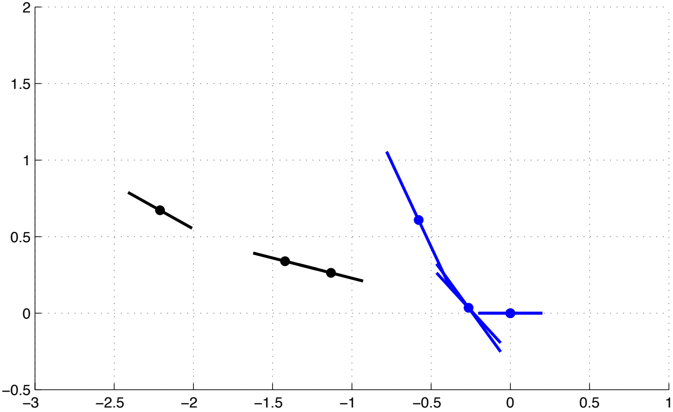
$$(x_i, g_i, f_i)_{i \in S} \text{ is } \mathcal{F}_{0,L}\text{-interpolable}$$
$$\Leftrightarrow (g_i, x_i, x_i^T g_i - f_i)_{i \in S} \text{ is } \mathcal{F}_{1/L,\infty}\text{-interpolable}$$

Example: Smooth Convex Interpolation



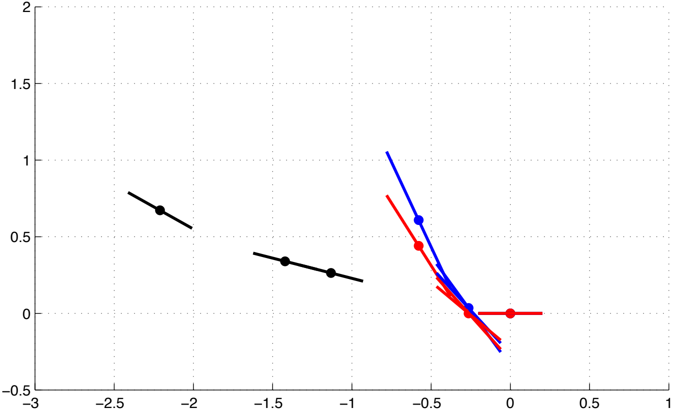
Interpolate $\{(x_i, g_i, f_i)\}_{i \in S}$ by $f \in \mathcal{F}_{0,L}$

Example: Smooth Convex Interpolation



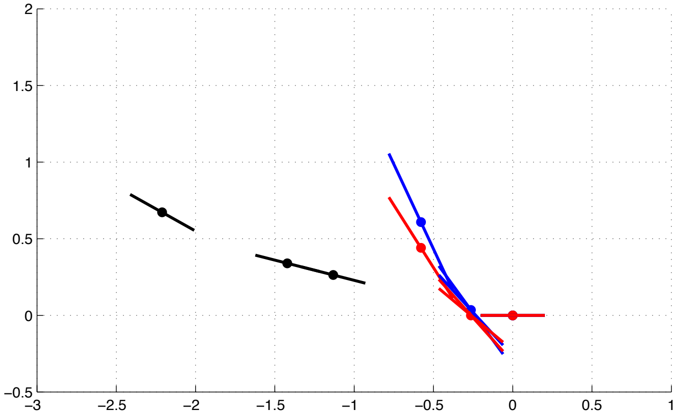
$\Leftrightarrow$ interpolate $\left\{(g_i, x_i, x_i^T g_i - f_i)\right\}_{i \in S}$ by $f^* \in \mathcal{F}_{1/L,\infty}$.

Example: Smooth Convex Interpolation



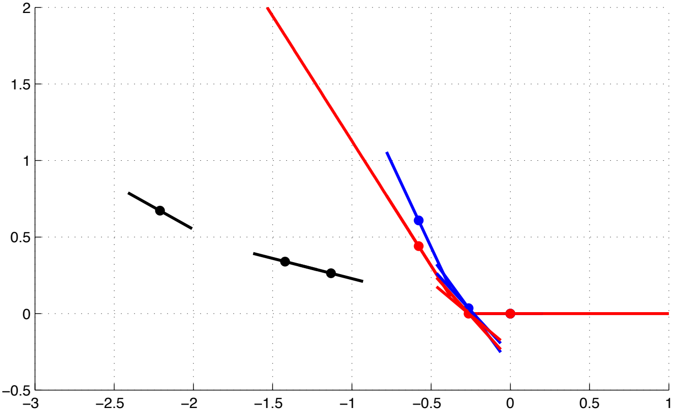
$\Leftrightarrow$ interpolate $\left\{\left(g_i, x_i - \frac{g_i}{L}, x_i^T g_i - f_i - \frac{||g_i||_2^2}{2L}\right)\right\}_{i \in S}$ by $\tilde{f} \in \mathcal{F}_{0,\infty}$.

Example: Smooth Convex Interpolation



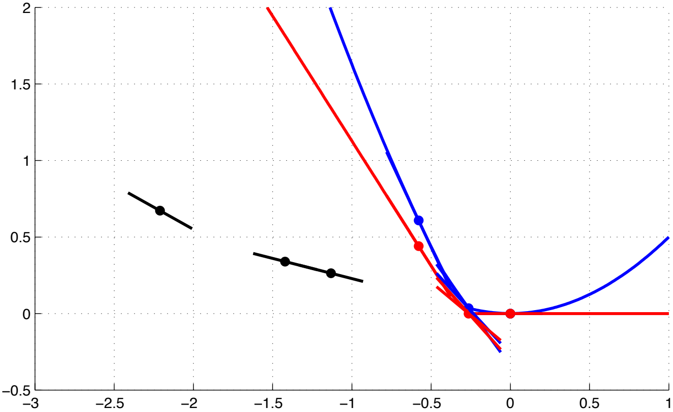
$\Leftrightarrow$ interpolate $\left\{(\tilde{x}_i, \tilde{g}_i, \tilde{f}_i)\right\}_{i\in S}$ by $\tilde{f} \in \mathcal{F}_{0,\infty}$.

Example: Smooth Convex Interpolation



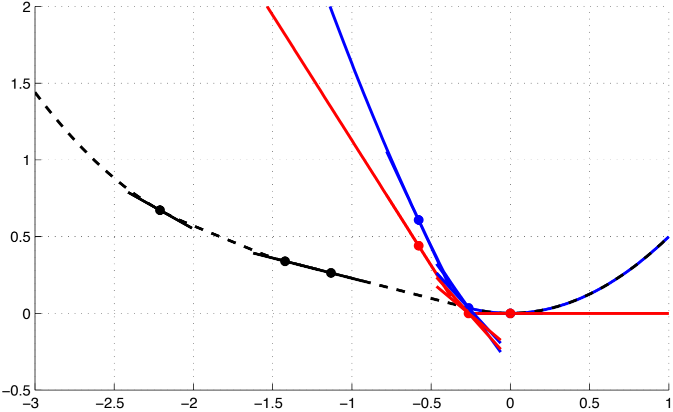
$$\tilde{f}(x) = \max_j \left\{ \tilde{f}_j + \tilde{g}_j^T (x - \tilde{x}_j) \right\}$$

Example: Smooth Convex Interpolation



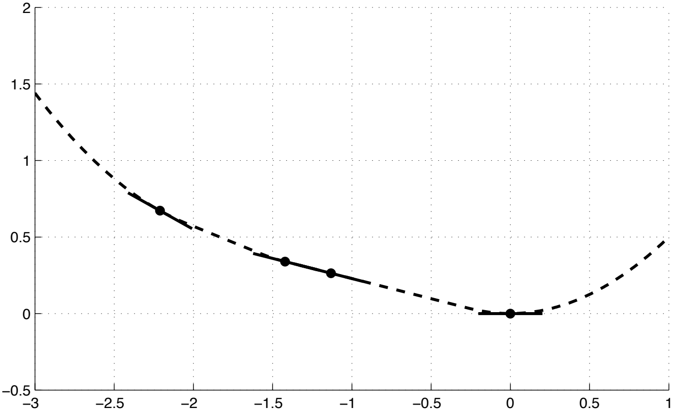
$f^*(x) = \max_j \left\{ \tilde{f}_j + \tilde{g}_j^T (x - \tilde{x}_j) \right\} + \frac{||x||_2^2}{2L}$

Example: Smooth Convex Interpolation



$$f(x) = \left(\max_j \left\{\tilde{f}_j + \tilde{g}_j^T(x - \tilde{x}_j)\right\} + \frac{||x||_2^2}{2L}\right)^*$$

Example: Smooth Convex Interpolation



$$f(x) = \left(\max_j \left\{\tilde{f}_j + \tilde{g}_j^T(x - \tilde{x}_j)\right\} + \frac{||x||_2^2}{2L}\right)^*$$

Necessary and sufficient interpolability conditions

Using the same reasoning:

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

$$f_i - f_j - g_j^\top (x_i - x_j) \geq \frac{1}{2(1-\mu/L)} \left(\frac{1}{L}\|g_i - g_j\|_2^2 \cdots \right.$$
$$\left. + \mu \|x_i - x_j\|_2^2 - 2\frac{\mu}{L}(g_j - g_i)^\top (x_j - x_i) \right)$$

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

Necessary and sufficient interpolability conditions

Using the same reasoning:

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

$$f_i - f_j - g_j^\top (x_i - x_j) \geq \frac{1}{2(1-\mu/L)} \left(\frac{1}{L} \|g_i - g_j\|_2^2 \cdots \right.$$
$$\left. + \mu \|x_i - x_j\|_2^2 - 2\frac{\mu}{L}(g_j - g_i)^\top (x_j - x_i) \right)$$

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

When $\mu = 0$, those conditions reduce to the well-known

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

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

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

$$\begin{aligned} \text{such that } & \{x_i, g_i, f_i\}_{i \in I} \text{ is } \mathcal{F}_{\mu,L}\text{-interpolable}, \\ & x_1, \ldots, x_N \text{ is generated by method } \mathcal{M}, \\ & \|x_0 - x_*\|_2 \leq R. \end{aligned}$$

- Drori and Teboulle (2014) obtain upper bounds with a dual of this nonconvex problem (after relaxation / reformulation)
- We want an exact and convex reformulation
- Need a way to express the interpolability constraints and the method constraints

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

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
- Many classical black-box methods can be reformulated in this way (including methods based on multiple sequences)

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
- Many classical 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)

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

- Many classical 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

Performance Estimation Problems (PEPs)

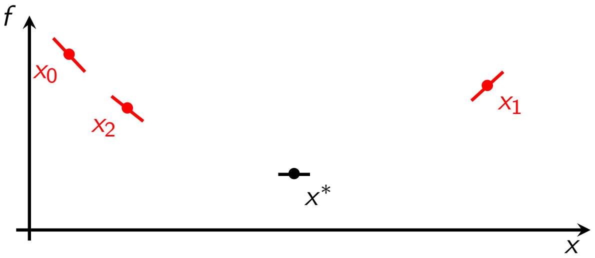
A simple illustrating example

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

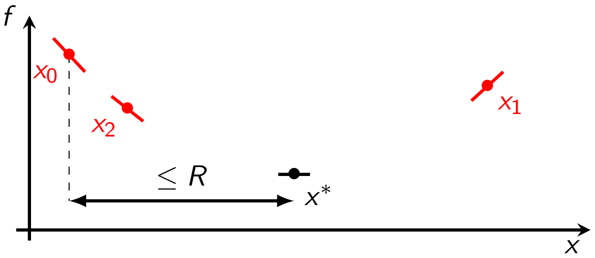
subject to the constraints

$$\begin{aligned}
&f \in \mathcal{F}_{0,L} \text{ i.e. convex and has } L\text{-Lipschitz gradient}\\
&\nabla f(x^*) = 0,\\
&x_{i+1} = x_i - \frac{h}{L} f'(x_i),\\
&||x_0 - x^*||_2^2 \leq R^2.
\end{aligned}$$

PEPs: discrete form

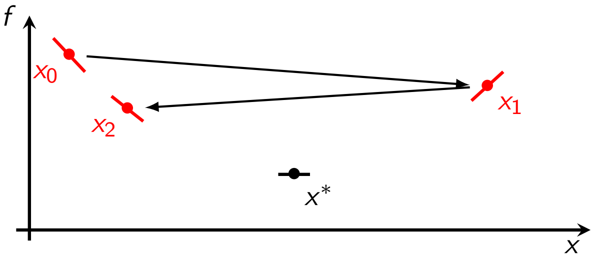


PEPs: discrete form



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

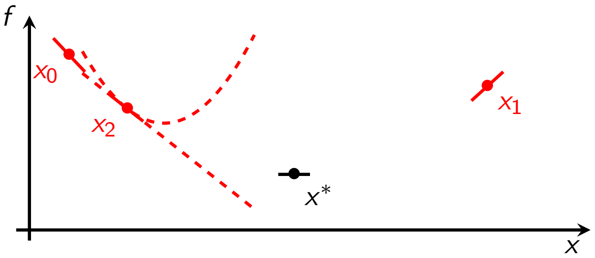
PEPs: discrete form



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

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

PEPs: discrete form

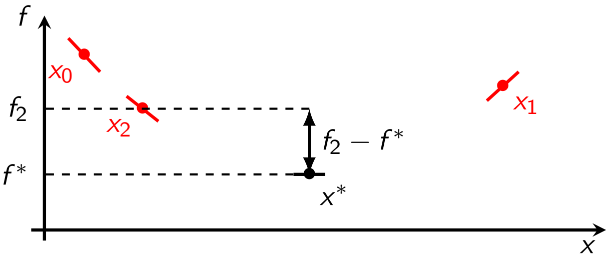


$$||x_0 - x^*||_2^2 \leq R^2$$

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

$$\{(x_i, g_i, f_i)\}_{i \in \{0,...,N-1\}} \cup (x_*, g_* = 0, f_*) \text{ interpolable by a function } f \in \mathcal{F}_{0,L}$$

PEPs: discrete form



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

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

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

$\{(x_i, g_i, f_i)\}_{i \in \{0,...,N-1\}} \cup (x_*, g_* = 0, f_*)$ interpolable by a function $f \in \mathcal{F}_{0,L}$

Formulation for performance optimization

Worst-case estimation problem translates into an exact finite problem

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

$$\begin{aligned}
\text{s.t. } & f_j \geq f_i + {g_i}^T(x_j - x_i) + \frac{1}{2L}||g_i - g_j||_2^2 \qquad && \forall i \neq j \in \{0, ..., N, *\} \\
& x_{i+1} = x_i - \frac{h}{L} g_i && \forall i \in \{0, ..., N-1\} \\
& ||x_0 - x^*||_2^2 \leq R^2 &&
\end{aligned}$$

- Only remaining difficulty: scalar products, i.e. nonconvex quadratic constraints

Formulation for performance optimization

Worst-case estimation problem translates into an exact finite problem

$$\max f_N - f^*$$

$$\begin{aligned} \text{s.t. } & f_j \geq f_i + {g_i}^T(x_j - x_i) + \frac{1}{2L}||g_i - g_j||_2^2 && \forall i \neq j \in \{0, ..., N, *\} \\ & x_{i+1} = x_i - \frac{h}{L} g_i && \forall i \in \{0, ..., N-1\} \\ & ||x_0 - x^*||_2^2 \leq R^2 \end{aligned}$$

- 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

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

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

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

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.

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 $\Leftrightarrow$ $\mathrm{rank}(G) \leq d$
Formulation is exact in large-scale case $N \ll d$ (when $2N + 2 \leq d$)

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

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

- Constant data matrices A_{ij}, and A_R that depend on method $\mathcal{M}$ (i.e. coefficients $h_{i,k}$) and function class paramters L and μ

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

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

- Constant data matrices A_{ij}, and A_R that depend on method $\mathcal{M}$ (i.e. coefficients $h_{i,k}$) and function class paramters L and μ
- Exact formulation, matrix variable G is size $N+2$, has $\mathcal{O}(N^2)$ linear constraints

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 paramters L and μ
- Exact formulation, matrix variable G is size $N+2$, has $\mathcal{O}(N^2)$ linear constraints
- Strictly feasible under some reasonable assumptions ($h_{i,i-1} \neq 0$)

Numerical performance estimation of standard algorithms

- We compute numerical results from our formulation (for values of N up to a few dozens)

Numerical performance estimation of standard algorithms

- We compute numerical results from our formulation (for values of N up to a few dozens)
- However in many cases we can identify the analytical form of a primal solution valid for all N (hence an explicit function f)

Numerical performance estimation of standard algorithms

- We compute numerical results from our formulation (for values of N up to a few dozens)
- However 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

Numerical performance estimation of standard algorithms

- We compute numerical results from our formulation (for values of N up to a few dozens)
- However 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
- We conjecture that these bounds are tight

Numerical performance estimation of standard algorithms

- We compute numerical results from our formulation (for values of N up to a few dozens)
- However 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
- We conjecture that these bounds are tight
- In most cases we also have an explicit (i.e. analytical formula) dual solution with matching objective function for all N
  Proving its feasibility seems too hard for the moment

Numerical performance estimation of standard algorithms

- We compute numerical results from our formulation (for values of N up to a few dozens)
- However 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
- We conjecture that these bounds are tight
- In most cases we also have an explicit (i.e. analytical formula) dual solution with matching objective function for all N
  Proving its feasibility seems too hard for the moment
- These results are (one-sided) conjectures strongly supported by numerical evidence

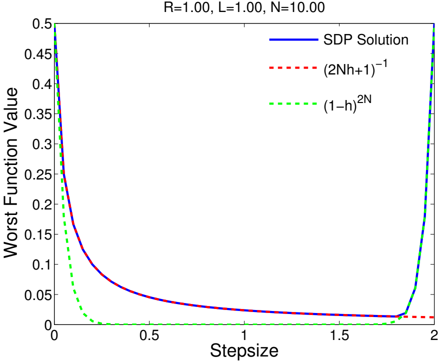
Fixed-step gradient method

W.l.o.g. fix $L = 1$ and $R = 1$ (which factors the common factor LR^2)
Choose $\mu = 0$, criterion $f_N - f^*$ and gradient method with step-size h

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

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

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



Also conjectured by Drori and Teboulle, 2014

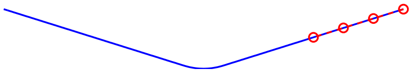
Best theoretical bound from literature is

$$\frac{2LR^2}{N+4}$$

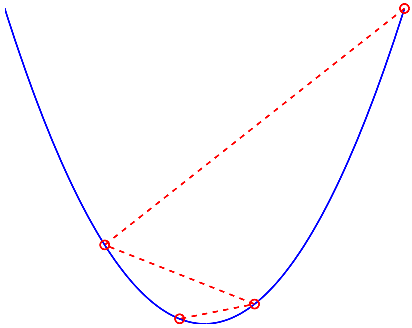
Intuition for each term in the bound

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



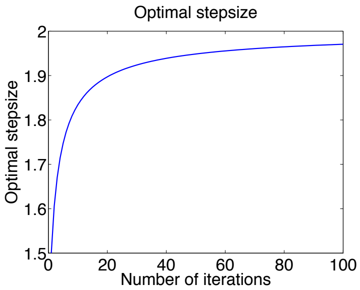
Purely quadratic, controlled overshooting at each iteration

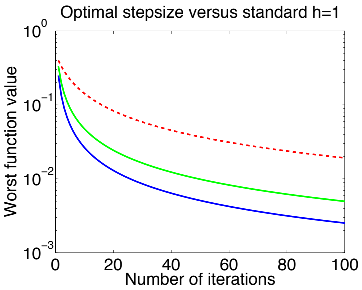
Optimal step-size

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, but no closed-form solution)





Optimal step-size tends quite quickly to 2

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 with $\mu = 0$, worst-case computed with relaxation from Drori and Teboulle and worst-case obtained by exact formulation (SDP-PEP) for the criterion $f(x_N) - f^*$. Error is measured relatively to the conjectured result. Results obtained with MOSEK.

More results for gradient method

- Strongly convex case, with condition number $\kappa = \mu/L$

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

More results for gradient method

- Strongly convex case, with condition number $\kappa = \mu/L$

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

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

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

More results for gradient method

- Strongly convex case, with condition number $\kappa = \mu/L$

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

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

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

- Residual gradient norm, strongly convex case

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

More results for gradient method

- Strongly convex case, with condition number $\kappa = \mu/L$

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

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

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

- Residual gradient norm, strongly convex case

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

- Simple $1D$ piecewise linear-quadratic solutions in all cases (different for each case)

More results for gradient method

- Strongly convex case, with condition number $\kappa = \mu/L$

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

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

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

- Residual gradient norm, strongly convex case

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

- 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

Algorithm:

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

For $i = 1 : N - 2$

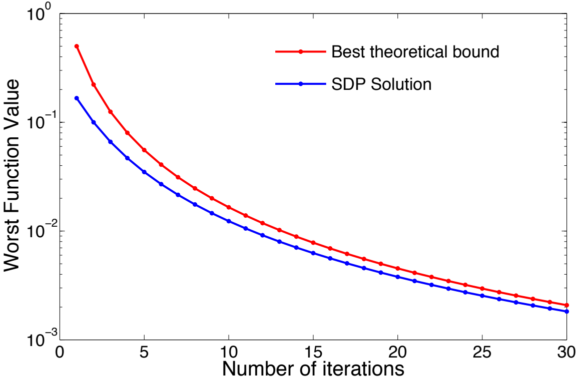
$$t_{i+1} = \frac{1 + \sqrt{1 + 4t_i^2}}{2}$$
$$x_{i+1} = x_i - \left(1 + \frac{t_i - 1}{t_{i+1}}\right)\frac{g_i}{L} + \frac{t_i - 1}{t_{i+1}}(x_i - x_{i-1}) + \frac{t_i - 1}{t_{i+1}}\frac{g_{i-1}}{L}$$

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

Belongs to the class of fixed-step methods

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)
Similar results for the recent optimized method of Kim and Fessler

Smallest gradient norm among all iterates

- Dual methods care about gradient norm ($\leftarrow$ primal feasibility)

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 $\mathcal{O}(\frac{1}{k})$

Smallest gradient norm among all iterates

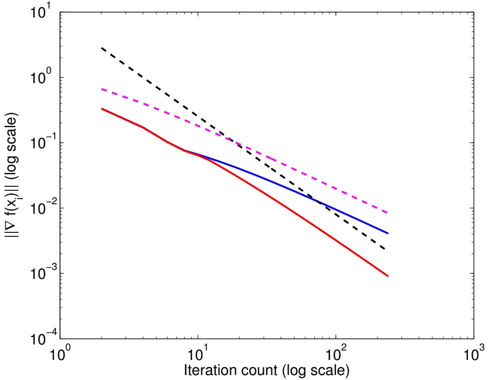
- Dual methods care about gradient norm ($\leftarrow$ primal feasibility)
- For accelerated gradient methods, objective function accuracy is $\mathcal{O}(\frac{1}{k^2})$ but norm of residual gradient $\|\nabla f(x_N)\|$ is only $\mathcal{O}(\frac{1}{k})$
- However this norm is not necessarily decreasing monotonically!

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

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)

Open whether $\mathcal{O}(\frac{1}{k^2})$ achievable by fixed-step method

Recent extensions of our formulation

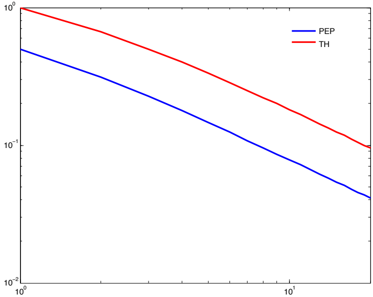
- first-order methods for smooth constrained convex optimization i.e. express projection steps in our formulation

Recent extensions of our formulation

- first-order methods for smooth constrained convex optimization i.e. express projection steps in our formulation
- proximal algorithms i.e. express proximal steps

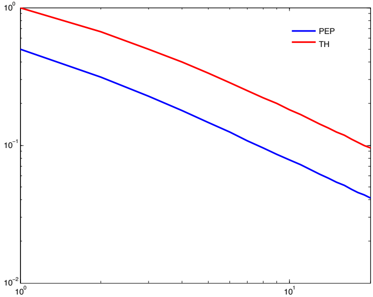
Recent extensions of our formulation

- first-order methods for smooth constrained convex optimization i.e. express projection steps in our formulation
- proximal algorithms i.e. express proximal steps
- linear minimization oracles (i.e. Frank-Wolfe-type methods)



Recent extensions of our formulation

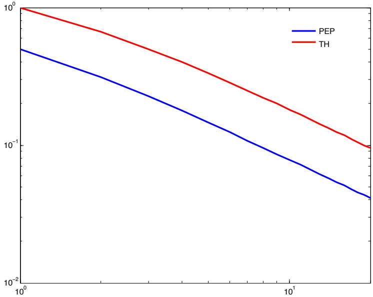
- first-order methods for smooth constrained convex optimization i.e. express projection steps in our formulation
- proximal algorithms i.e. express proximal steps
- linear minimization oracles (i.e. Frank-Wolfe-type methods)



- Computes worst-case over all possible convex feasible domains

Recent extensions of our formulation

- first-order methods for smooth constrained convex optimization i.e. express projection steps in our formulation
- proximal algorithms i.e. express proximal steps
- linear minimization oracles (i.e. Frank-Wolfe-type methods)



- Computes worst-case over all possible convex feasible domains

Key observation: proximal steps can be exactly formulated as

$$x_+ = \mathrm{prox}_L(x) \quad \Leftrightarrow \quad x_+ - g_+ = x \text{ and } g_+ \in \partial f(x_+)$$

Thank you for your attention!

- 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 (smoothness and strong convexity)

- Methodology provides easy-to-interpret proofs and explicit examples of worst-case functions

- Flexible approach: includes several performance criteria (objective value, gradient norm, etc.) and can be extended to constrained, proximal, linear minimization oracle settings

Thank you for your attention

Preprint can dowloaded from ArXiv at
http://arxiv.org/abs/1502.05666