Performance estimation of optimization methods
A guided tour

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 many collaborators, including Nizar Bousselmi, Julien Hendrickx, Yassine Kamri, Ion Necoara, Panos Patrinos, Teodor Rotaru, Adrien Taylor and Moslem Zamani

Funding: FNRS/FRIA, KUL GlobalParternship, EU MSCA

Workshop on Nonsmooth Optimization and Applications
in Honor of the 75th Birthday of Boris Mordukhovich
NOPTA 2024 – University of Antwerp – April 8, 2024

Performance estimation of an optimization method

Goal of a PEP (Performance Estimation Problem):

compute the worst-case behavior

- of a given optimization method
  (considering a fixed number of iterations)
- applied to any function belonging to a given class
  (e.g. convex functions, possibly with some additional regularity property such as smoothness)
- from any starting point
  (possibly satisfying some condition w.r.t. a minimizer)
- for a given performance criteria
  (e.g. objective accuracy, distance to solution, gradient norm)

Example: after computing N steps of the (unconstrained) gradient method with fixed step-size $\frac{1}{L}$ applied to a convex function f with an L-Lipschitz gradient and minimizer x^* from an initial iterate satisfying $\|x_0 - x_*\| \leq R$, what is the worst (largest) possible objective function accuracy for the last iterate $f(x_N) - f(x_*)$?

Output of a performance estimation problem

For a given PEP (Performance Estimation Problem) we will

- compute the exact value of the performance criteria's worst-case = optimal value of PEP problem
- identify an explicit function (and starting point) achieving this worst-case value = primal solution of PEP problem + interpolation
- obtain an independently-checkable proof that this worst-case value is a valid (upper) bound on the performance criteria = dual multiplier of PEP problem
- all three steps can be done either numerically or analytically

For a large class of first-order methods applied to convex composite optimization problems, these can be computed exactly using a semidefinite programming (SDP) problem.

Two ingredients to formulate PEP as convex/SDP prob.

1. explicit interpolation inequalities for studied class of functions.

   For example, smooth convex interpolation:

   Set $\{(x_i, g_i, f_i)\}_{i \in S}$ is interpolable by a function $f \in \mathcal{F}_{0,L}$

   $\Leftrightarrow$ there exists a convex function f with L-Lipschitz gradient such that $f(x_i) = f_i$ and $\nabla f(x_i) = g_i$ for all $i \in S$

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

2. optimization method defined as constraint over $\{(x_i, g_i, f_i)\}_{i \in S}$

   For example, gradient step with constant stepsize :

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

   $\Leftrightarrow$ $\|x_{k+1} - (x_k - \frac{h}{L}g_k)\|^2 = 0$

3. All conditions above must be convex in variables f_i and products between x_i and g_i (elements of their Gram matrix)
   Ideally they must also be SDP-representable

What you obtain when you solve a PEP

1. In all cases:
   numerical value for worst-case performance
   numerical description of worst-case function (interpolate primal)
   numerical proof of worst-case performance (dual multipliers)

2. If you are lucky/clever: explicit formulas for some/all of above
   $\rightarrow$ this requires fitting/guessing algebraic expressions

3. In the best case: a rigorous mathematical proof
   $\rightarrow$ this requires a proof with (often tedious) reformulation of worst-case rate as inequality involving sums-of-squares

   Note: by theorem, all valid proofs must be writable as sum-of-squares inequalities based on combinations of necessary and sufficient interpolation inequalities
   (hence any use of another inequality, while also leading to an upper bound on rate, may destroy its tightness!)

Very brief history of performance estimation

- Concept of performance estimation introduced by Drori and Teboulle (2012) who solve a relaxation of a nonconvex formulation, providing bounds shown to be tight for some algorithms (using matching worst-case functions)
- Taylor, Hendrickx and G. show SDP formulation is exact (2015), using necessary and sufficient interpolation conditions
- Other approaches similar in spirit:
  integral quadratic constraints by Lessard, Recht, Packard (2014); Lyapunov/potential functions by Taylor, Bach (2019)

- A growing subfield of algorithmic optimization (above three seminal papers total 1000+ citations)
- This talk mostly focuses on results obtained in UCLouvain, but there are many other recent exciting results

Outline

Obtaining a SDP formulation: a worked example

We explain all steps to obtain an exact SDP formulation on the following example, dealing with the gradient method with the simplest so-called 'optimal' $\frac{1}{L}$ steps:

> What is the worst possible objective function accuracy after applying N steps of the gradient method with fixed step-size $\frac{1}{L}$ to a convex function f with L-Lipschitz gradient and minimizer x_*, from an initial iterate satisfying $\|x_0 - x_*\| \leq R$?

Formally, this is an infinite-dimensional problem over functions f:

$$\max_{f, x_*, x_0, x_1, \ldots, x_N} f(x_N) - f(x_*) \text{ s.t. } \begin{cases} f \text{ is proper and convex} \\ \text{f has an } L\text{-Lipschitz gradient} \\ x_* \text{ is a minimizer of } f \\ \|x_0 - x_*\| \leq R \\ x_{i+1} = x_i - \frac{1}{L} g_i \text{ for every } 0 \leq i < N \\ (\text{where } g_i = \nabla f(x_i) \text{ is the gradient}) \end{cases}$$

Step 1: use the black-box property

Performance estimation problem is infinite dimensional but the gradient method is black-box

(i.e. next iterate only depends on previously obtained oracle information, consisting of the function value and an (arbitrary) subgradient for each of the previous iterates)

In general, starting from initial x_0, a black-box method computes

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

Only depends on x_0 and the finite list of outputs from the oracle ($\mathcal{O}_f(x_i) = \{f(x_i), \nabla f(x_i)\}$)

Step 1: finite-dimensional reformulation using black-box

Hence infinite-dimensional variable f can be replaced by the list of oracle outputs, each defining a variable

$$f_i = f(x_i) \text{ and } g_i = \nabla f(x_i) \text{ for all } i = 0, 1, \dots N \text{ as well as } i = *$$

and performance estimation problem becomes

$$\begin{cases} \max \\ x_*, x_0, x_1, \dots, x_N \\ f_*, f_0, f_1, \dots, f_N \\ g_*, g_0, g_1, \dots, g_N \end{cases} f_N - f_* \text{ s.t.} \begin{cases} \text{there exists proper and convex } f \text{ with} \\ \text{an } L\text{-Lipschitz gradient such that} \\ \qquad f(x_i) = f_i \text{ and } g_i = \nabla f(x_i) \\ \text{for all } i \in \{*, 0, 1, \dots N\} \\ g_* = 0 \\ \|x_0 - x_*\| \leq R \\ x_{i+1} = x_i - \frac{1}{L} g_i \text{ for every } 0 \leq i < N \end{cases}$$

which is finite-dimensional (minimizer condition became $g_* = 0$).
Remaining issue: imposing existence of suitable f compatible with f_i and g_i requires necessary and sufficient interpolating conditions

Step 2: introduce interpolating conditions

We need to express in our problem the fact that

> there exists proper and convex f with an L-Lipschitz gradient such that $f(x_i) = f_i$ and $g_i = \nabla f(x_i)$ for all $i \in I = \{*, 0, 1, \dots N\}$

i.e. given values of x_i, f_i and g_i we must guarantee existence of f

$\rightarrow$ we need necessary and sufficient conditions for interpolation

We use the following equivalence [Taylor,Hendrickx,G 2017]

there exists proper and convex f satisfying

$$f(x_i) = f_i \text{ and } g_i = \nabla f(x_i) \text{ for every } i \in I$$

$$\Leftrightarrow$$

$$f_j \geq f_i + g_i(x_j - x_i) + \frac{1}{2L}\|g_i - g_j\|^2 \text{ for every pair } i \in I, j \in I$$

(can be extended to deal with the smooth strongly convex case, and the smooth nonconvex/weakly convex cases)

Step 2 (cont.): introduce interpolating conditions

Performance estimation problem becomes
(with $I = \{*, 0, 1, \dots, N\}$)

$$\max_{\{x_i, f_i, g_i\}_{i \in I}} f_N - f_* \text{ s.t. } \begin{cases} f_j \geq f_i + g_i^T (x_j - x_i) + \frac{1}{2L} \|g_i - g_j\|^2 \ \forall i \neq j \in I \\ x_* = 0, f_* = 0, g_* = 0 \\ \|x_0 - x_*\| \leq R \\ x_{i+1} = x_i - \frac{1}{L} g_i \text{ for every } 0 \leq i < N \end{cases}$$

(note that constraints $f_* = 0$ and $x* = 0$ can be added w.l.o.g.)
Worst-case is computed exactly with a finite explicit formulation ;
idea first introduced in Drori and Teboulle (2014)

Unfortunately, problem is nonconvex (inner products $g_i^T x_j$)

Step 3: convexify using a Gram matrix

$$\max_{\{x_i,f_i,g_i\}_{i\in I}} f_N - f_* \text{ s.t. } \begin{cases} f_j \geq f_i + g_i^T(x_j - x_i) + \frac{1}{2L}\|g_i - g_j\|^2 \ \forall i \neq j \in I \\ x_* = 0, f_* = 0, g_* = 0 \\ \|x_0 - x_*\|^2 \leq R^2 \\ \left\|x_{i+1} - x_i + \frac{1}{L}g_i\right\|^2 = 0 \text{ for every } 0 \leq i < N \end{cases}$$

All terms are quadratic in x_i and g_i, and linear in f_i
$\rightarrow$ introduce a Gram matrix for variables f_i and g_i !

$$\text{Example } N = 1 : \qquad P = (x_0 \mid x_1 \mid g_0 \mid g_1)$$

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

Step 3: convexify using a Gram matrix

Problem becomes linear in f_i and elements of the positive semidefinite Gram matrix $\rightarrow$ a semidefinite optimization problem
Convex problem, can be solved globally and efficiently with IPM

Formulation is a priori exact if the Gram matrix reformulation is equivalent, which happens if dimension of vectors x_i and g_i (i.e. dimension of f) is large enough (larger than dimension of G)

$\rightarrow$ an exact dimension-free formulation for the large-scale case (for results valid for small-dimensional f, impose rank constraint on G ; worst-case can only deteriorate as dimension of f increases)

From $G \succeq 0$, we can recover values of x_i and g_i corresponding to compute worst-case function ; combined with values f_i this allows to identify an explicit function achieving the worst-case, using interpolation

Rank of G determines dimension of worst-case f (at most $2N + 2$)

Step 4: recovering outputs

- Exact worst-case value computed by the SDP formulation (a priori under the large-scale assumption)
- Independently-checkable proof is derived from dual solution: combining interpolating inequalities with the corresponding dual multipliers provides a proof for the worst-case bound
- Explicit worst-case function derived by interpolation from optimal solution, with dimension equal to rank of G
- If G is rank deficient, worst-case result valid for all dimensions larger than that rank (in particular: if G has rank one, worst-case unconditionally valid, which is quite frequent)

Additional step: use homogeneity

Our performance estimation problem is parameterized by

- Lipschitz constant L for gradient
- Maximum distance R between starting point and a minimizer

To avoid having to solve for every value of L and R use homogeneity properties

- if $L \to \lambda L$ with $\lambda > 0$, worst-case is also multiplied by λ (proof: scale $f \to \lambda f$)
- if $R \to \lambda R$ with $\lambda > 0$, worst-case is also multiplied by λ^2 (proof: $f \to \lambda^2 \lambda f(\cdot/\lambda)$)

hence worst-case $w_*(L, R, N)$ is proportional to L and R^2

$\to$ solve problem for $B = R = 1$, find worst-case value $w_*(1, 1, N)$ so that for general L and R we will have

$$f(x_N) - f(x_*) \leq w_*(L, R, N) = \frac{LR^2}{2} \cdot w_*(1, 1, N)$$

Only remaining parameter is N, number of steps

Worst-case gradient method with constant $\frac{1}{L}$ stepsize

Apply PEP to N steps of gradient method with $\frac{1}{L}$ stepsize:

1. Notice numerical values of worst-case match formula

$$w^*(1,1,N) = \frac{1}{2N+1}$$

(easy to see for small N, then verified for large N)

2. Guess from (factor of) Gram matrix the values of gradients g_i
Matrix is rank one, interpolating values leads to univariate piecewise linear-quadratic function f_ρ (Huber) with $\rho = \frac{1}{2N+1}$:

$$f_\rho(x) = \begin{cases} \rho|x| - \frac{\rho^2}{2} & \text{when } |x| \geq \rho \\ \frac{x^2}{2} & \text{when } |x| \geq \rho \end{cases}$$

Easy to check performance of method on f_ρ (starts with $x_0 = 1$, then $x_i = 1 - i\rho$), which establishes $w^* \geq \frac{1}{2N+1}$

Worst-case gradient method with constant $\frac{1}{L}$ stepsize

3. To prove upper bound on w^*, it is sufficient to exhibit a dual feasible solution with same value
   To describe a dual solution it is actually sufficient to give multipliers $\lambda_{i,j}$ for each interpolating inequalities $i, j \in I$
   Again guess from numerical values ; e.g. for $N = 1$:

$$\lambda_{0,*} = \lambda_{1,*} = \lambda_{1,0} = \frac{1}{2} \quad \text{and all other multipliers zero}$$

4. Finally one needs to prove (analytically) that the above set of multipliers is feasible (hardest part is proving the slack matrix is positive semidefinite), leading finally to $w^* \leq \frac{1}{2N+1}$

(this result was already obtained $\forall N$ in Drori and Teboulle, 2014)

Interpretation of proof with dual multipliers

Actually SDP formulation/machinery is not needed to derive a proof of the upper bound: one can alternatively and equivalently multiply the interpolating inequalities by the dual multipliers and simplify the resulting expression

For example when $N = 1$, one can check that the following identity holds when $x_1 = x_0 - \frac{1}{L}\nabla f(x_0)$:

$$
\begin{aligned}
& f(x_1) - f(x_*) \\
= \; & \tfrac{L}{6}\|x_0 - x_*\|^2 \\
- \; & \tfrac{1}{2}\left[f(x_0) - f(x_1) - \nabla f(x_1)^\top (x_0 - x_1) - \tfrac{1}{2L}\|\nabla f(x_0) - \nabla f(x_1)\|^2\right] \\
- \; & \tfrac{1}{2}\left[f(x_*) - f(x_0) - \nabla f(x_0)^\top (x_* - x_0) - \tfrac{1}{2L}\|\nabla f(x_0) - \nabla f(x_*)\|^2\right] \\
- \; & \tfrac{1}{2}\left[f(x_*) - f(x_1) - \nabla f(x_1)^\top (x_* - x_1) - \tfrac{1}{2L}\|\nabla f(x_1) - \nabla f(x_*)\|^2\right] \\
- \; & \tfrac{L}{6}\left\|x_0 - x_* - \tfrac{3}{2L}\nabla f(x_0) - \tfrac{3}{2L}\nabla f(x_1)\right\|^2 - \tfrac{L}{8}\|\nabla f(x_0) - \nabla f(x_1)\|^2
\end{aligned}
$$

(hardest part is grouping of slack terms into squares, equivalent to proving semidefiniteness of slack ; proof is not necessarily unique!)

Outline

Gradient method, L-smooth function, constant step-size $\frac{h}{L}$

Worst-case rate for final iterate accuracy, for any $h \in [0, 2]$

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

We actually know

- analytical expression for worst-case performance
- analytical expression of worst-case function (Huber/quadratic)
- analytical expression of dual multipliers

but rigorous sum-of-squares proof currently known only for $h \leq 1.5$ (Drori and Teboulle 2014, Teboulle and Vaisbourd 2023)

Dealing with smooth strongly convex functions

Strongly convex functions $\Leftrightarrow$ lower bound $\mu > 0$ on curvature

To tackle smooth strongly convex functions (class $\mathcal{F}_{\mu,L}$) we only need one new ingredient: suitable interpolation conditions

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 $\mathcal{F}_{\mu,L}$-interpolable if and only

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

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

(generalizes conditions for smooth convex interpolation)

Gradient method for smooth strongly convex functions

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

Linear convergence for all performance criteria, with same rate $\rho = \max\{(1-Lh)^2, (1-\mu h)^2\}$ [Taylor,Hendrickx,G 2018]

$$\begin{aligned} \|x_k - x_*\|^2 &\leq \rho^k \|x_0 - x_*\|^2 \\ \|\nabla f(x_k)\|^2 &\leq \rho^k \|\nabla f(x_0)\|^2 \\ f(x_k) - f(x_*) &\leq \rho^k \big(f(x_0) - f(x_*)\big) \end{aligned}$$

- All results with fully rigorous PEP-type mathematical proofs (pure linear rates $\rightarrow$ sufficient to prove them for one step)
- Optimal steplength is $h^* = \frac{2}{L+\mu}$ with $\rho^* = \left(\frac{L-\mu}{L+\mu}\right)^2$
- Worst-case functions are 1D quadratics $\frac{\mu}{2}x^2$ and $\frac{L}{2}x^2$
- Same rates also hold for projected/proximal gradient
- However linear rate become void when $\rho = 0$ (giving $\rho = 1$) and such rates do not (cannot) work in nonconvex case $\rightarrow$ we move to another setup

Dealing with smooth nonconvex functions

Smooth nonconvex actually quite similar to smooth convex:

smooth nonconvex functions $\Leftrightarrow$ curvature must belong to $[-L, L]$

To tackle smooth nonconvex functions (class $\mathcal{F}_{-L,L}$) we only need one new ingredient: suitable interpolation conditions

It turns out that interpolation conditions for smooth strongly convex functions in class $\mathcal{F}_{\mu,L}$ also work, with exactly the same expressions, when μ is negative !

Hence smooth nonconvex interpolations conditions are obtained simply by taking $\mu = -L$

This is used in [Abbaszadehpeivasti,de Klerk,Zamani 2022] to derive tight rates for gradient method on smooth nonconvex objectives with stepsize $h \leq \frac{\sqrt{3}}{L}$

Dealing with nonconvex and hypoconvex functions

We can even interpolate smoothly between smooth convex ($\mathcal{F}_{0,L}$) and smooth nonconvex ($\mathcal{F}_{-L,L}$) with all values of $\mu \in]-L, 0[$

This leads to the class of hypoconvex functions $\mathcal{F}_{\mu,L}$ for any $\mu < 0$ (also known as weakly convex)

Necessary and sufficient interpolation conditions can be found in [Taylor 2017][Rotaru,G,Patrinos 2022][Abbaszadehpeivasti,de Klerk,Zamani 2022]

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

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

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

(again generalizes conditions for smooth convex and smooth strongly convex interpolation)

Gradient method for smooth functions

The following rates are all

- exact, with explicit analytical expressions,
- established using mathematical proofs (PEP-inspired),
- cover all constant stepsizes schedules in the standard (converging) range $[0, \frac{2}{L}]$,
- cover convex, strongly convex, nonconvex and hypoconvex objective functions in a unified way
- hence performance criteria we pick is gradient norm (for last iterate, or smallest among all iterates)

preprint soon to be updated on arxiv [Rotaru,G,Patrinos 2024]

Exact rates for gradient method in (strongly) convex case

Gradient method with constant stepsize γ

Theorem (Convex)

$f \in \mathcal{F}_{0,L}$ (L-smooth and convex):

$$\frac{1}{2L}\|\nabla f(x_N)\|^2 \leq \frac{f(x_0) - f(x_*)}{1 + \gamma L \min\left\{2N\,,\, \dfrac{-1 + (1-\gamma L)^{-2N}}{\gamma L}\right\}}$$

Theorem (Strongly convex)

$f \in \mathcal{F}_{\mu,L}$, with $\mu \in (0, L]$ (L-smooth and μ-strongly convex):

$$\frac{1}{2L}\|\nabla f(x_N)\|^2 \leq \frac{f(x_0) - f(x_*)}{1 + \gamma L \min\left\{\dfrac{-1 + (1-\gamma\mu)^{-2N}}{\gamma\mu}\,,\, \dfrac{-1 + (1-\gamma L)^{-2N}}{\gamma L}\right\}}$$

Similar rates, correspond to worst of two simple (quadratic/Huber) functions (but proof is *not* simple)

Exact rates for gradient method in nonconvex case

Theorem (Weakly convex (hypoconvex))

$f \in \mathcal{F}_{\mu,L}$, with $\mu \in (-\infty, 0)$:

$$\frac{1}{2L} \min_{0 \le i \le N} \left\{ \|\nabla f(x_i)\|^2 \right\} \le \frac{f(x_0) - f(x_*)}{1 + \gamma L \min \left\{ P_N(\gamma L, \gamma \mu)\,,\ \dfrac{-1 + (1 - \gamma L)^{-2N}}{\gamma L} \right\}}$$

with $P_N(\gamma L, \gamma \mu)$ given by

$$\begin{cases} \left(2 - \dfrac{-\gamma\mu\,\gamma L}{\gamma L - \gamma \mu}\right) N, & \gamma L \in (0, 1]; \\ \dfrac{(2 - \gamma L)(2 - \gamma \mu)}{2 - \gamma L - \gamma \mu} \left(N - \dfrac{-\gamma\mu\,\gamma L}{\gamma L - \gamma \mu} \sum_{k=0}^{N} \{ [T_k(\gamma L, \kappa)]_+ \right), & \gamma L \in [1, 2). \end{cases}$$

Stepsize thresholds

For any $\kappa \in (-\infty, 1)$ and integers k, quantities

$$T_k(\gamma L, \kappa) := \frac{1 - (1 - \gamma L)^{-2k}}{\gamma L} - \frac{1 - (1 - \gamma \mu)^{-2k}}{\gamma \mu}$$

determine distinct regimes in all theorems,
which are delimited by stepsize thresholds:

$$\gamma L_k(\kappa) = \{\gamma L \in (1, 2) \mid T_k(\gamma L, \kappa) = 0\}$$

Comments on exact rates for gradient method

- First proof of a tight convergence rate for the gradient method covering the full range of constant stepsizes (i.e. $h \in [0, \frac{2}{L}]$)

- In (strongly) convex case: simple expression for rate
  (worst among two simple functions)
  but proof is unexpectedly difficult
  (reason: hidden distinct regimes behind a single rate)
- In nonconvex/hypoconvex case: very complicated rates
  (interpolate continuously between convex and nonconvex)

- To cover full range need N distinct proofs/regimes
- Key element for tightness: use of distance-two inequalities
  (same expected for analysis of very recent *long-step* method,
  see Parrilo-Altschuler, Grimmer et at.)

Go see Teodor Rotaru's poster on Friday!

Outline

Nonsmooth optimization: subgradient method

First-order methods can deal with nonsmooth convex functions using the concept of subgradient

$$g \in \partial f(x) \quad \Leftrightarrow \quad f(y) \geq f(x) + g^T(y - x) \text{ for all } y$$

Subgradient method startsfrom x_0 and performs for all $i \geq 0$

$$x_{i+1} = x_i - h_i g_i \text{ for some } g_i \in \partial f(x_i)$$

for some stepsize schedule $\{h_i\}_{i \geq 0}$

Worst-case rates on objective accuracy require

- Distance R from initial iterate x_0 to minimizer x^*
- Bound B on maximum norm of any subgradient $g \in \partial f(x)$

Interpolation conditions for nonsmooth convex functions

We need explicit conditions for the following

there exists proper and convex f with B-bounded subgradients such that $f(x_i) = f_i$ and $g_i \in \partial f(x_i)$ for all $i \in I = \{*, 0, 1, \dots N\}$

i.e. given values of x_i, f_i and g_i we need to guarantee existence of f

We use the well-known (and easy to show) equivalence

there exists proper and convex f satisfying

$$f(x_i) = f_i \text{ and } g_i \in \partial f(x_i) \text{ for every } i \in I$$

$$\Leftrightarrow$$

$$f_j \geq f_i + g_i(x_j - x_i) \text{ for every } i, j \in I$$

which can be extended to deal with B-bounded subgradients We use the well-known (and easy to show) equivalence

there exists proper and convex f with B-bounded subgradients s.t.
$f(x_i) = f_i$ and $g_i \in \partial f(x_i)$ for every $i \in I$

$$\Leftrightarrow$$

$$f_i > f_i + g_i^T(x_i - x_i) \text{ for every } i, j \in I$$

Results: average iterate

Worst-case for fixed-step subgradient method

$$x_{i+1} = x_i - h(\tfrac{R}{B})g_i$$

applied to convex function with B-bounded subgradients

- For average value of iterates $\hat{f}_N = \frac{f(x_0)+f(x_1)+\ldots+f(x_N)}{N+1}$, tight worst-case is

$$\hat{f}_N - f(x_*) \leq \begin{cases} BR\left(\frac{1}{2}h + \frac{1}{2(N+1)}\frac{1}{h}\right) & \text{when } h \geq \frac{1}{N+1} \\ BR\left(1 - \frac{N}{2}h\right) & \text{when } h \leq \frac{1}{N+1} \end{cases}$$

  (recovers a well-known result for large h)

- Optimal constant step-size is then $h^* = \frac{1}{\sqrt{N+1}}$ (belongs to "large step" case) leading to tight worst-case

$$\hat{f}_N - f(x_*) \leq \frac{BR}{\sqrt{N+1}}$$

Results: last iterate

- Define sequence $\{s_N\}_{N\geq 0} = \{1, 2, \frac{5}{2}, \frac{29}{10}, \ldots\}$ with

$$s_0 = 1, s_{i+1} = s_i + \frac{1}{s_i} \text{ for all } i \geq 0$$

- No closed form, s_N grows like $\sqrt{2(N+1) + \frac{1}{2}\log(N)}$, also appears in [Nesterov 2009] for primal-dual subgradient

- For value of last iterate $f(x_N)$, tight worst-case is

$$f(x_N) - f(x_*) \leq \begin{cases} BR\left[(\frac{1}{2}s_N^2 - N)h + \frac{1}{2s_N^2}\frac{1}{h}\right] & \text{when } h \geq \frac{1}{s_N^2} \\ BR(1 - Nh) & \text{when } h \leq \frac{1}{s_N^2} \end{cases}$$

- No previous result with correct asymptotic rate for last iterate
- [Harvey,Liaw,Plan,Randhawa 2019] prove a $\frac{\log N}{32\sqrt{N}}$ lower bound when $B = 1$ with stepsize $h_i = \frac{1}{\sqrt{i}}$, and prove a high probability $\mathcal{O}(\frac{\log N}{\sqrt{N}})$ upper bound in stochastic case

Results: optimal stepsize and variants

- To perform N subgradient iterations, optimal stepsize is then

$$h^* = \frac{1}{\sqrt{s_N^2(s_N^2 - 2N)}}$$

and corresponding worst-case value satisfies

$$f(x_N) - f(x_*) \leq BR\sqrt{1 - \frac{2N}{s_N^2}} \lesssim BR \cdot \sqrt{\frac{1 + \frac{1}{4}\log(N)}{N+1}}$$

- Using suboptimal $h^\dagger = \frac{1}{\sqrt{N+1}}$ leads to slightly worse

$$f(x_N) - f(x_*) \leq BR \cdot \left(\frac{\frac{5}{4} + \frac{1}{4}\log(N)}{\sqrt{N+1}}\right)$$

- All results also hold for normalized stepsize $\frac{h}{R}\frac{g_i}{\|g_i\|}$

Proof technique: PEP no longer needed!

All results are based on the following key Lemma, whose proof does not require PEP-style interpolation inequalities manipulations, only Jensen's inequality (based on tracking the distance between the current iterate and a different reference point at each iteration)

Lemma ([Zamani,G 2023])

Suppose that $\hat{x} \in X$, $h_{N+1} > 0$ *and weights* v_k *satisfy* $0 < v_0 \leq v_1 \leq \cdots \leq v_N \leq v_{N+1}$ *Then iterates of the subgradient methods with starting point* $x^1 \in X$ *generating* $\{(x^k, g^k)\}$ *satisfy*

$$\sum_{k=1}^{N+1} \left(h_k v_k^2 - (v_k - v_{k-1}) \sum_{i=k}^{N+1} h_i v_i \right) \left(f(x^k) - f(\hat{x}) \right)$$

$$\leq \frac{v_0^2}{2} \underbrace{\left\| x^1 - \hat{x} \right\|^2}_{R} + \frac{1}{2} \sum_{k=1}^{N+1} h_k^2 v_k^2 \underbrace{\left\| g^k \right\|^2}_{B}$$

Still credit for the inspiration of the proof goes to PEP!

Optimal last-iterate subgradient method

- Using the following new linearly decreasing stepsize schedule

$$x_{k+1} = x_k - \frac{R}{B}\frac{(N+1-k)}{(N+1)^{3/2}}$$

leads the optimal rate for the last iterate [Zamani,G 2023]

$$f(x_N) - f(x_*) \leq \frac{BR}{\sqrt{N+1}}$$

(improves $\frac{15BD}{\sqrt{N+1}}$ [Jain,Nagaraj,Netrapalli 2021] for diameter D)

- Same proof technique, using key lemma with other weights v_k
- Schedule dependence on N is forced for optimal method (already impossible to find fixed stepsizes h_1 and h_2 that are optimal for both $N = 1$ and $N = 2$)
- Existence of a last-iterate optimal method with momentum?

Go see Moslem Zamani's poster on Friday!

Outline

Toolboxes

Please visit https://github.com/PerformanceEstimation for toolboxes: PESTO (in MATLAB) [Taylor,Hendrickx,G] and PEPit (in Python) [Goujaud,Moucer,G,Hendrickx,Taylor]

PerformanceEstimation

This organization regroups works/packages/toolboxes related to performance estimation problems.

If you find some content useful, please don't hesitate to give feedbacks and or to star the content!

Current packages

- **PESTO**: allows a quick access to performance estimation problems in Matlab.
- **PEPit**: allows a quick access to performance estimation problems in Python.

Education

- Informal introduction to PEPs: **here**.
- Practical exercises: **here**.

Events

If you organize a PEP-event, we would be happy to list your event below.

Upcoming events:

- February 2023: **PEP-talks**.

Example: subgradient method with $h_k = \frac{1}{\sqrt{N+1}}$

```
P = pep();

param.R        = 1;
F              = P.DeclareFunction(
                   'ConvexBoundedGradient',param);
[xstar,fstar] = F.OptimalPoint();

x0 = P.StartingPoint();
P.InitialCondition((x0-xstar)^2<=1);

N=5;
x=x0;
for i=1:N
    [g,f] = F.oracle(x);
    x     = x - 1/sqrt(N+1)*g;
end

xN = x;
fN=F.value(xN);
P.PerformanceMetric(fN-fstar);

P.solve()
disp(double(fN - fstar))
```

Contains numerous introductory examples

(tries to keep up with literature, current count is 80+)

- 1. Unconstrained convex minimization
  - 1.1. Gradient descent
  - 1.2. Subgradient method
  - 1.3. Subgradient method under restricted secant inequality and error bound
  - 1.4. Gradient descent with exact line search
  - 1.5. Conjugate gradient
  - 1.6. Heavy Ball momentum
  - 1.7. Accelerated gradient for convex objective
  - 1.8. Accelerated gradient for strongly convex objective
  - 1.9. Optimized gradient
  - 1.10. Optimized gradient for gradient
  - 1.11. Robust momentum
  - 1.12. Triple momentum
  - 1.13. Information theoretic exact method
  - 1.14. Cyclic coordinate descent
  - 1.15. Proximal point
  - 1.16. Accelerated proximal point
  - 1.17. Inexact gradient descent
  - 1.18. Inexact gradient descent with exact line search
  - 1.19. Inexact accelerated gradient
  - 1.20. Epsilon-subgradient method
  - 1.21. Gradient descent for quadratically upper bounded convex objective
  - 1.22. Gradient descent with decreasing step sizes for quadratically upper bounded convex objective
  - 1.23. Conjugate gradient for quadratically upper bounded convex objective
  - 1.24. Heavy Ball momentum for quadratically upper bounded convex objective

Outline

Projected gradient and proximal methods

We can actually handle with little extra effort

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

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

- proximal algorithms i.e. express proximal steps

$$x_{k+1} = \operatorname{prox}_L f(x_k) = \arg\min_u \left(f(u) + \frac{1}{2L}\|u - x_k\|^2\right)$$

- composite minimization: $\min f(x) + h(x)$ where f is smooth and h is proximable, using proximal gradient method

$$x_{k+1} = \operatorname{prox}_L h(x_k - \tfrac{1}{L}\nabla f(x_k))$$

Projected gradient and proximal methods

$$x_+ = \mathrm{prox}_L f(x) = \arg\min_u \left(f(u) + \frac{1}{2L}\|u - x\|^2 \right)$$

- Key idea: proximal steps can be formulated as

  $$x_+ = \mathrm{prox}_L f(x) \quad \Leftrightarrow \quad x_+ - \tfrac{1}{L} g_+ = x \text{ and } g_+ \in \partial f(x_+)$$

  which is a linear condition involving iterates and oracle outputs

- Proximal gradient for composite optimization $\min f(x) + h(x)$ can be decomposed in two successive, independent steps: $x_+ = \mathrm{prox}_L h(x - \frac{1}{L}\nabla f(x))$ is equivalent

  $$y = x - \tfrac{1}{L}\nabla f(x) \quad \text{then} \quad x_+ = \mathrm{prox}_L h(y)$$

- Projected gradient = proximal gradient using indicator function of set C for nonsmooth term h
- Requires corresponding interpolation conditions (e.g. for indicator function $\mathbb{I}_C(x)$ of a convex set)
- Linear rates unchanged in smooth strongly convex case!

Methods using inexact gradient

Instead of computing $x_{k+1} = x_k - \frac{1}{L}\nabla f(x_k)$ with exact gradient assume gradient is computed inexactly with bounded error

$$\tilde{g}_k \approx \nabla f(x_k) \text{ such that } \|\tilde{g}_k - \nabla f(x_k)\| \leq \Delta$$

Key technique: rewrite step with inexact gradient

$$x_{k+1} = x_k - \tilde{g}_k$$

as

$$x_{k+1} = x_k - \nabla f(x_k) - \left(\tilde{g}_k - \nabla f(x_k)\right)$$

and then observe it is equivalent to

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

which can be written using the assumption on the error as

$$L\|x_k - \nabla f(x_k) - x_{k+1}\| \leq \Delta$$

Leads to a convex SDP formulation (after squaring both sides)

Methods involving linear mappings

[Bousselmi, Hendrickx,G 2023]

We want to minimize $g(Mx)$ (alone/in composite objective) where M is a linear mapping with some characteristics

- M symmetric and constraints on minimum/maximum eigenvalues $[\mu, L]$
- M non-symmetric (possibly rectangular) with maximum singular value S

A gradient step on $F(x) = g(Mx)$ requires gradient

$$\nabla F(x) = M^T \nabla g(Mx)$$

which can be decomposed as three successive operations:

$$\begin{aligned} y &= Mx \\ u &= \nabla g(y), \\ v &= M^T u = \nabla F(x). \end{aligned}$$

Gradient of function composed with linear mapping

In order to compute worst-case for methods involving

$$
\begin{aligned}
y &= Mx \\
u &= \nabla g(y), \\
v &= M^T u = \nabla F(x).
\end{aligned}
$$

we need to interpolate the following

$$
\begin{aligned}
y_i &= Mx_i, \\
u_i &= \nabla g(y_i), \\
v_i &= M^T u_i = \nabla F(x_i).
\end{aligned}
$$

New expressions: $y_i = Mx_i$ and $v_i = M^T u_i$

Requires interpolability of two sequences $\{x_i, y_i\}$ and $\{u_i, v_i\}$ by a linear mapping M and its transpose M^T

Interpolation theorem for linear mappings

For simplicity of notation we represent sequence $\{x_i\}$ as a matrix X (columns are x_i), same for $\{y_i\}$, $\{u_i\}$, $\{v_i\}$

Let $X \in \mathbb{R}^{m\times N_1}$, $Y \in \mathbb{R}^{n\times N_1}$, $U \in \mathbb{R}^{m\times N_2}$ and $V \in \mathbb{R}^{n\times N_2}$.
(X, Y, U, V) is $\mathbb{R}_S^{m\times n}$-matrix-interpolable if, and only if,

$$\begin{cases} X^T V = Y^T U, \\ Y^T Y \preceq S^2 X^T X, \\ V^T V \preceq S^2 U^T U. \end{cases}$$

(where $\preceq$ denotes the Lowner $=$ positive semidefinite order)

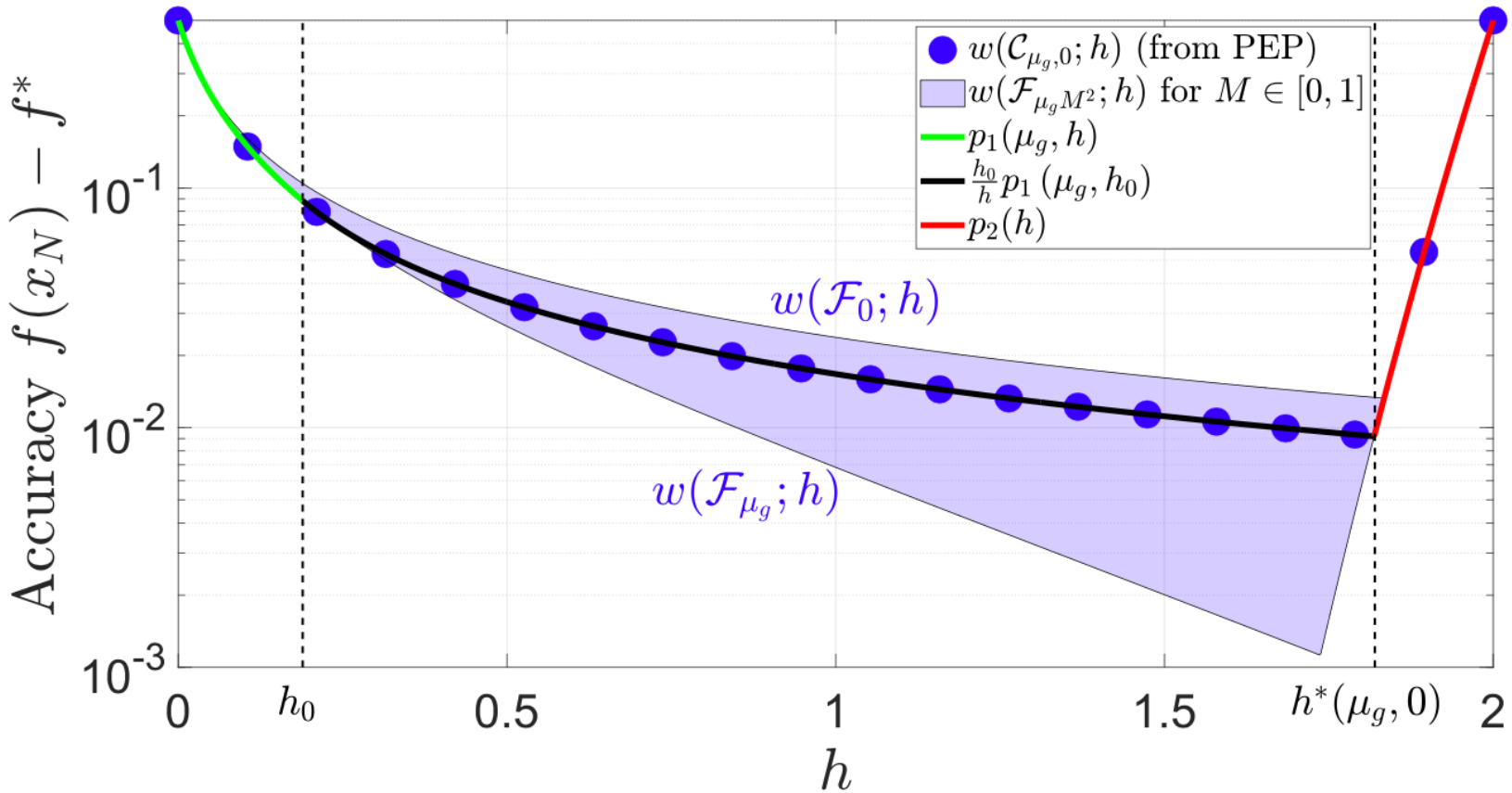
Moreover, if $U = X$ and $V = Y$ (resp. $V = -Y$), the interpolant matrix can be chosen symmetric (resp. skew-symmetric).

[Bousselmi,Hendrickx,G 2023]

Result: gradient method

$$\min g(Ax) \quad \text{vs.} \quad \min f(x)$$

when g is strongly convex, but $f = g \circ A$ is not



Result: Chambolle-Pock method

Let f and g convex and $||M|| \leq L_M$. If $\tau\sigma \leq \frac{1}{L_M^2}$, then after $N \geq 1$ iterations of the Chambolle-Pock algorithm started from x_0 and u_0 we have, for any x and u, that the primal-dual gap satisfies

$$\mathcal{L}(\bar{x}_N, u) - \mathcal{L}(x, \bar{u}_N) \leq \frac{\frac{1}{\tau}\|x-x_0\|^2 + \frac{1}{\sigma}\|u-u_0\|^2 - 2(u-u_0)^T M(x-x_0)}{2(N+1)}$$

Tight result with analytical proof
(slight improvement over previous proof showing $\frac{1}{2N}$)

Many variants can be analyzed, e.g.

$$\mathcal{L}(\bar{x}_N, u) - \mathcal{L}(x, \bar{u}_N) \leq \frac{\frac{1}{\tau}\|x - x_0\|^2 + \frac{1}{\sigma}\|u - u_0\|^2}{N+1}$$

and others for which we have no analytical rate yet

Go see Nizar Bousselmi's poster on Tuesday!

Outline

Reflections

- Automated procedure to compute worst-case rates
  But: is a PEP proof the final goal?

- More than once, these steps actually happened
  1. Numerical rate computed, analytical expression guessed/identified/confirmed
  2. Using insight provided by worst-case/proof, a closer look at rate/proof provides further intuition and new ideas
  3. Result can now be derived in standard way
     (and its "PEP" origin becomes unnoticeable!)

- Even when such simplifications are not found, shouldn't the goal of mathematical optimization theory first and foremost to increase our understanding?

To conclude: a few open questions

- Some rates are known explicitly (including multipliers) but no proof available

  $$\max f(x_N) - f^* = \frac{LR^2}{2} \max\left(\frac{1}{2Nh+1}, (1-h)^{2N}\right) \text{ for } h > 1.5$$

  (should not underestimate difficulty, even with explicit expressions: original proof by Drori and Teboulle for gradient method with $h \leq 1$ required six pages of matrix analysis/algebra)

- Can we design first-order methods using PEP?

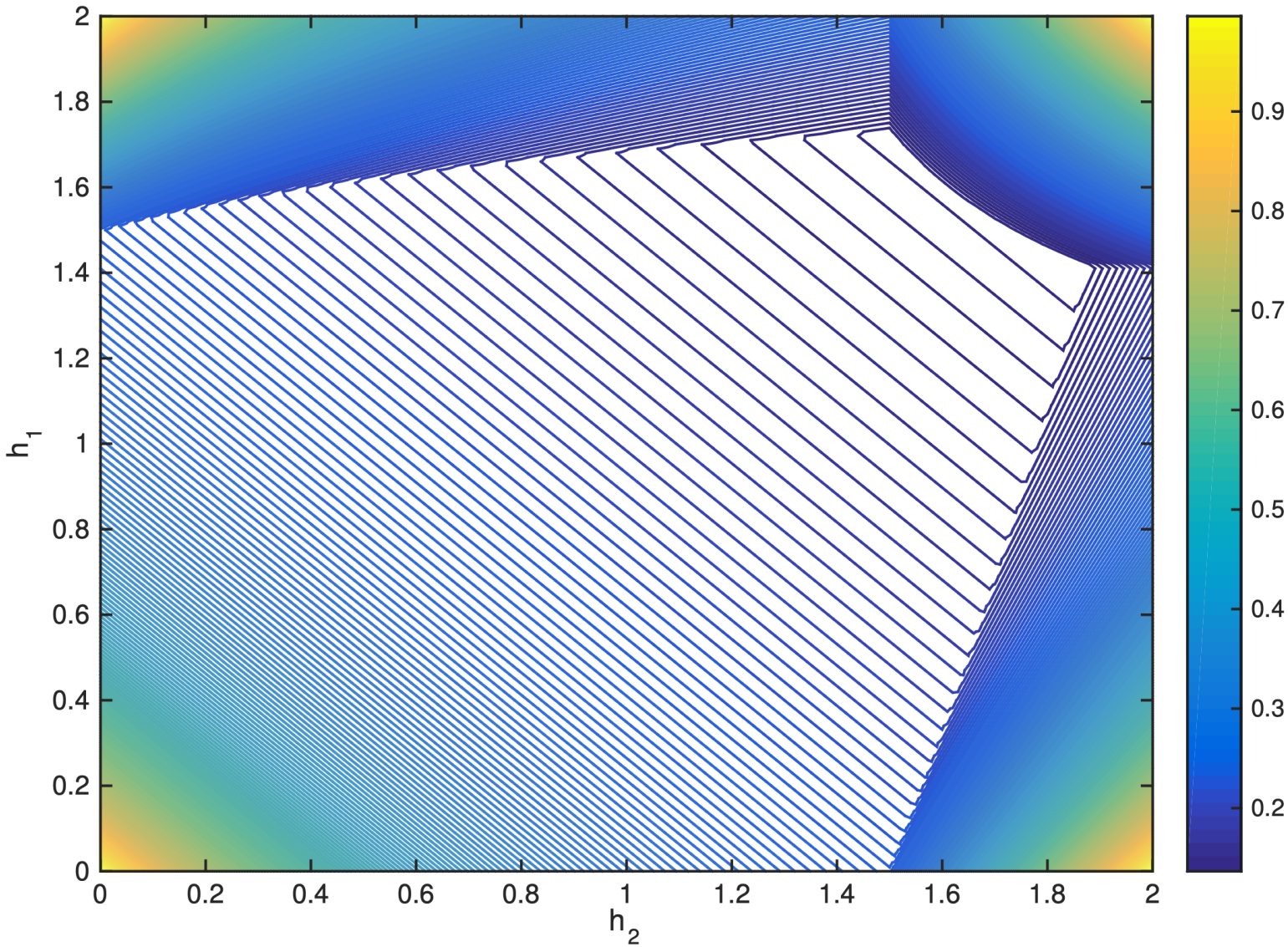
  Main difficulty: considering method coefficients to be variable (e.g. stepsizes h_k) destroys convexity of PEP formulation

  Attempts to (and succeeds in) solving resulting nonconvex SDP [Das Gupta,Van Parys,Ryu 2022]

  Is there a non obvious convex formulation for method design?

Convergence of $N = 2$ steps of gradient method

[Daccache 2019] Stepsizes (h_1, h_2), convex objective, $L = 1$-smooth, initial distance $\|x_0 - x_*\| \leq 1$; contour plot of $2(f(x_2) - f_*)$



To conclude: a few open questions

- Some classes lack necessary and sufficient interpolation conditions

  Example: convex functions with coordinate smoothness
  More generally: intersections of two or more classes
  (grouping conditions is necessary, but not always sufficient)

- How come worst-case functions are univariate/low-dimensional for so many (but not all) methods ?
  Is there a fundamental reason for that?

- Can we go beyond first-order methods with fixed steps?
  Nonlinear stepsize rules: some recent success for nonlinear conjugate gradient [Das Gupta,Freund,Sun,Taylor 2023]

  What about second-order methods? Interior-point methods?
  Higher order/tensor method?

Thank you again for your attention!

Python toolbox PEPit and Matlab toolbox PESTO available at

`https://github.com/PerformanceEstimation`

Interested in gradient method for nonconvex?
Go see Teodor Rotaru's poster on Friday!

Interested in ADMM convergence?
Go see Moslem Zamani's poster on Friday!

Interested in methods using linear operations?
Go see Nizar Bousselmi's poster on Tuesday!