Full-low evaluation methods for bound and linearly constrained derivative-free optimization

Clément W. Royer

ISMP - July 26, 2024

Academic family business

Oumaima Sohab (Lehigh⇒Meta AI)

Luis Nunes Vicente (Lehigh)

Academic family business

Oumaima Sohab (Lehigh⇒Meta AI)

Luis Nunes Vicente (Lehigh)

Story

- Extend unconstrained method to linear constraints;
- Nice numerical findings!
- Some theoretical guarantees.

Outline

Problem setup

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\, f(x) \quad \text{s.t.} \quad x \in \mathcal{F} := \{x \in \mathbb{R}^n \mid A\,x = b, \ell \leq A_{\mathcal{I}}\,x \leq u\}\,.$$

with $A \in \mathbb{R}^{m \times n}$ full row-rank, $\ell, u \in \overline{\mathbb{R}}^{m_{\mathcal{I}}},\ \ell < u$.

Problem setup

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\, f(x) \quad \text{s.t.} \quad x \in \mathcal{F} := \{x \in \mathbb{R}^n \mid A\,x = b, \ell \leq A_{\mathcal{I}}\,x \leq u\}.$$

with $A \in \mathbb{R}^{m \times n}$ full row-rank, $\ell, u \in \overline{\mathbb{R}}^{m_{\mathcal{I}}}$, $\ell < u$.

DFO setup

- f may have a derivative...
- ...but we cannot use it in algorithms!

Problem setup

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\, f(x) \quad \text{s.t.} \quad x \in \mathcal{F} := \{x \in \mathbb{R}^n \mid A\,x = b, \ell \leq A_{\mathcal{I}}\,x \leq u\}\,.$$

with $A \in \mathbb{R}^{m \times n}$ full row-rank, $\ell, u \in \overline{\mathbb{R}}^{m_{\mathcal{I}}}$, $\ell < u$.

DFO setup

- f may have a derivative...
- ...but we cannot use it in algorithms!

Our constraints

- QUAK constraints (Le Digabel, Wild '24).
- Unrelaxable$\Rightarrow$ Feasible methods!

DFO and bound/linear constraints

Sample (biased) bibliography

- Model-based methods (Powell '09, Gratton et al '11, Curtis et al '24).
- Derivative-free line search (Lucidi/Sciandrone '02, Lucidi/Sciandrone/Tseng '02, Brilli et al '24).
- Direct search (Abramson et al '08, Audet/Le Digabel/Peyrega '15, Kolda et al '03, Kolda et al '06, Gratton et al '19, Dzahini et al '24).
- More in (Audet, Hare '17, Larson/Menickelly/Wild '22)!

Sample (biased) bibliography

- Model-based methods (Powell '09, Gratton et al '11, Curtis et al '24).
- Derivative-free line search (Lucidi/Sciandrone '02, Lucidi/Sciandrone/Tseng '02, Brilli et al '24).
- Direct search (Abramson et al '08, Audet/Le Digabel/Peyrega '15, Kolda et al '03, Kolda et al '06, Gratton et al '19, Dzahini et al '24).
- More in (Audet, Hare '17, Larson/Menickelly/Wild '22)!

Our approach:

Extend the Full-Low Eval framework to handle linear constraints.

A. S. Berahas, O. Sohab, L. N. Vicente, *Full-low evaluation methods for derivative-free optimization*, Optim. Methods Softw., 2022.

The full-low evaluation framework

Inputs: $x_0 \in \mathbb{R}^n$, $\alpha_0 > 0$, $t_0 \in \{\texttt{Full-Eval}, \texttt{Low-Eval}\}$.

For $k = 0, 1, 2, \dots$

- If $t_k = \texttt{Full-Eval}$, compute $(x_{k+1}, \alpha_{k+1}, t_{k+1})$ through a `Full-Eval` iteration.
- Otherwise, compute $(x_{k+1}, \alpha_{k+1}, t_{k+1})$ through a `Low-Eval` iteration.

The full-low evaluation framework

Inputs: $x_0 \in \mathbb{R}^n$, $\alpha_0 > 0$, $t_0 \in \{\texttt{Full-Eval}, \texttt{Low-Eval}\}$.

For $k = 0, 1, 2, \dots$

- If $t_k = \texttt{Full-Eval}$, compute $(x_{k+1}, \alpha_{k+1}, t_{k+1})$ through a `Full-Eval` iteration.
- Otherwise, compute $(x_{k+1}, \alpha_{k+1}, t_{k+1})$ through a `Low-Eval` iteration.

- `Full-Eval`: Most expensive procedure, typically better for smooth problems.
- `Low-Eval`: Least expensive procedure, well-suited for nonsmoothness/noise.

The full-low evaluation framework

Inputs: $x_0 \in \mathcal{F}$, $\alpha_0 > 0$, $t_0 \in \{\texttt{Full-Eval}, \texttt{Low-Eval}\}$.
For $k = 0, 1, 2, \dots$

- If $t_k = \texttt{Full-Eval}$, compute $(x_{k+1}, \alpha_{k+1}, t_{k+1})$ through a `Full-Eval` iteration.
- Otherwise, compute $(x_{k+1}, \alpha_{k+1}, t_{k+1})$ through a `Low-Eval` iteration.

- `Full-Eval`: Most expensive procedure, typically better for smooth problems.
- `Low-Eval`: Least expensive procedure, well-suited for nonsmoothness/noise.

Feasible version

- Feasible starting point.
- Both steps must preserve feasibility.

A `Full-Eval` iteration: Projected gradient-type iteration

- g_k: Finite-difference gradient approximation;
- p_k: Gradient-related direction.

A `Full-Eval` iteration: Projected gradient-type iteration

- g_k: Finite-difference gradient approximation;
- p_k: Gradient-related direction.

`Full-Eval` **iteration**(x_k, α_k):

- Compute $\bar{x}_k = P_{\mathcal{F}}\left[x_k + p_k\right]$.
- Find $\beta_k \in \{1, \frac{1}{2}, \frac{1}{4}, \dots\}$ such that

$$f(x_k + \beta_k(\bar{x}_k - x_k)) \le f(x_k) + c\,\beta_k g_k^{\mathrm{T}}(\bar{x}_k - x_k).$$

  with $c \in (0, 1)$.
- Set $x_{k+1} = x_k + \beta_k(\bar{x}_k - x_k)$.

A `Low-Eval` iteration: Direct search

- Forcing function ρ (typically $\rho(\alpha) = \alpha^2$.
- Generators of tangent cones for $\mathcal{F}$.

A `Low-Eval` iteration: Direct search

- Forcing function ρ (typically $\rho(\alpha) = \alpha^2$.
- Generators of tangent cones for $\mathcal{F}$.

`Low-Eval` **iteration**(x_k, α_k):

- Compute $D_k \subset \mathbb{R}^n$ feasible directions.
- If $\exists d_k \in D_k$ such that $x_k + \alpha_k d_k \in \mathcal{F}$

$$f(x_k + \alpha_k d_k) \leq f(x_k) - \rho(\alpha_k),$$

  set $x_{k+1} = x_k + \alpha_k d_k$ and $\alpha_{k+1} = \min(2\alpha_k, \alpha_{\max})$.
- Otherwise set $x_{k+1} = x_k$ and $\alpha_k = \alpha_k/2$.

Detour: Tangent cones

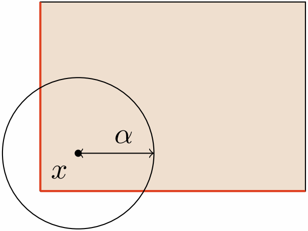
- Simplify: $\mathcal{F} = \{l \le x \le u\}$.

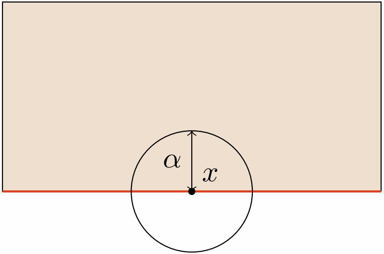
Nearby constraints

The indexes

$$
\begin{aligned}
I_u(x,\alpha) &= \{i : |u_i - [x]_i| \le \alpha\} \\
I_l(x,\alpha) &= \{i : |l_i - [x]_i| \le \alpha\}
\end{aligned}
$$

define the nearby constraints at $x \in \mathcal{F}$ given $\alpha > 0$.



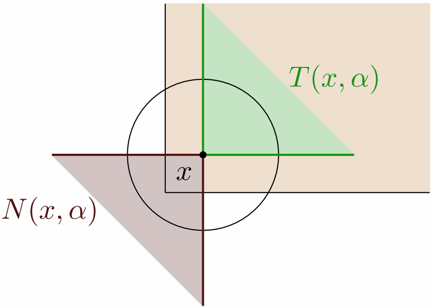


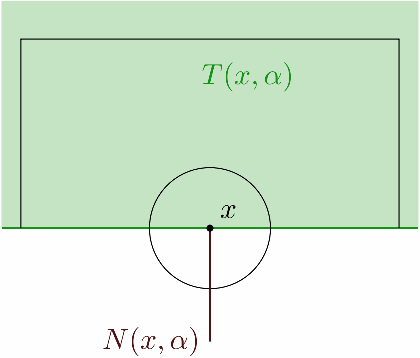
Detour: tangent cones (2)

- **Approximate normal cone $N(x,\alpha)$:** Positive span of

$$\{e_i\}_{i \in I_u(x,\alpha)} \cup \{-e_i\}_{i \in I_l(x,\alpha)}.$$

- **Approximate tangent cone $T(x,\alpha)$:** polar of $N(x,\alpha)$.





Algorithm

Inputs: $x_0 \in \mathcal{F}$, $\alpha_0 > 0$, $t_0 = \texttt{Full-Eval}$, $\gamma \in [0, \infty]$.

For $k = 0, 1, 2, \ldots$

- $t_k = \texttt{Full-Eval}$:

- $t_k = \texttt{Low-Eval}$:

Algorithm

Inputs: $x_0 \in \mathcal{F}$, $\alpha_0 > 0$, $t_0 = \texttt{Full-Eval}$, $\gamma \in [0, \infty]$.

For $k = 0, 1, 2, \dots$

- $t_k = \texttt{Full-Eval}$:
  - Compute $\bar{x}_k = P_{\mathcal{F}}\left[x_k + p_k\right]$.
  - Find $\beta_k \in \{1, \frac{1}{2}, \frac{1}{4}, \dots\}$ such that

$$f(x_k + \beta_k(\bar{x}_k - x_k)) \leq f(x_k) + c\,\beta_k g_k^{\mathrm{T}}(\bar{x}_k - x_k).$$

  - If $\beta_k \geq \gamma\alpha_k$, set $x_{k+1} = x_k + \beta_k(\bar{x}_k - x_k)$, $\alpha_{k+1} = \alpha_k$ and $t_{k+1} = \texttt{Full-Eval}$. Otherwise set $x_{k+1} = x_k$, $\alpha_{k+1} = \alpha_k$ and $t_{k+1} = \texttt{Low-Eval}$.
- $t_k = \texttt{Low-Eval}$:

Algorithm

Inputs: $x_0 \in \mathcal{F}$, $\alpha_0 > 0$, $t_0 = \texttt{Full-Eval}$, $\gamma \in [0, \infty]$.

For $k = 0, 1, 2, \dots$

- $t_k = \texttt{Full-Eval}$:
  - Compute $\bar{x}_k = P_{\mathcal{F}}\left[x_k + p_k\right]$.
  - Find $\beta_k \in \{1, \frac{1}{2}, \frac{1}{4}, \dots\}$ such that

$$f(x_k + \beta_k(\bar{x}_k - x_k)) \leq f(x_k) + c\,\beta_k g_k^{\mathrm{T}}(\bar{x}_k - x_k).$$

  - If $\beta_k \geq \gamma\alpha_k$, set $x_{k+1} = x_k + \beta_k(\bar{x}_k - x_k)$, $\alpha_{k+1} = \alpha_k$ and $t_{k+1} = \texttt{Full-Eval}$. Otherwise set $x_{k+1} = x_k$, $\alpha_{k+1} = \alpha_k$ and $t_{k+1} = \texttt{Low-Eval}$.
- $t_k = \texttt{Low-Eval}$:
  - Compute $D_k \subset \mathbb{R}^n$ feasible directions.
  - If $\exists d_k \in D_k$ such that $x_k + \alpha_k d_k \in \mathcal{F}$ and

$$f(x_k + \alpha_k d_k) \leq f(x_k) - \rho(\alpha_k),$$

    set $x_{k+1} = x_k + \alpha_k d_k$, $\alpha_{k+1} = 2\alpha_k$, $t_{k+1} = \texttt{Low-Eval}$.
  - Otherwise set $x_{k+1} = x_k$, $\alpha_k = \alpha_k/2$.
    Choose t_{k+1} depending on $t_{k-1}, \dots, t_{k-\log_{1/2}(\beta_k)}$.

Switching procedure

From `Full-Eval` to `Low-Eval`

- Switch to `Low-Eval` when $\beta_k < \gamma \alpha_k$.
- $\gamma = 0$: Only `Full-Eval`.
- $\gamma = \infty$: Only `Low-Eval`.

From `Low-Eval` to `Full-Eval`

- $\log_{1/2}(\beta_k)$: Number of backtracks in the last `Full-Eval` iteration.
- Switch to `Full-Eval` after $\log_{1/2}(\beta_k)$ unsuccessful `Low-Eval` iterations.
- $\gamma = 0$: No `Low-Eval` steps.
- $\gamma = \infty$: Regular `Low-Eval` algorithm.

Outline

Implementation

`Full-Eval` step

- Use finite-difference BFGS direction: $p_k = -H_k\, g_k$.
- Line-search condition:

$$f(x_k+\beta_k(P_{\mathcal{F}}[x_k+p_k]-x_k)) \le f(x_k)+10^{-8}\beta_k\, g_k^{\mathrm{T}}(P_{\mathcal{F}}[x_k+p_k]-x_k).$$

Per-step cost: $n-m$ evaluations ($Ax = b \in \mathbb{R}^m$) + line search.

`Low-Eval` step

- Accept first point that satisfies

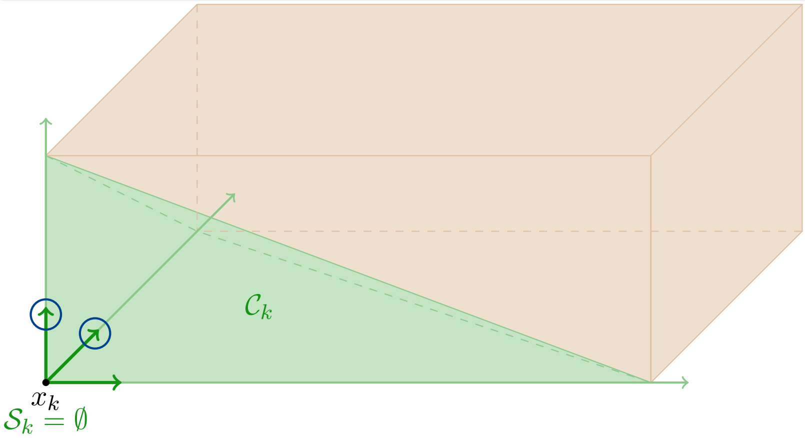
$$f(x_k+\alpha_k d_k) \le f(x_k) - \min\{10^{-5}, 10^{-5}\alpha_k^2\}.$$

- Probabilistic feasible descent:
  - Use random directions in unconstrained subspaces!
  - Use random subsets of tangent cone generators otherwise.

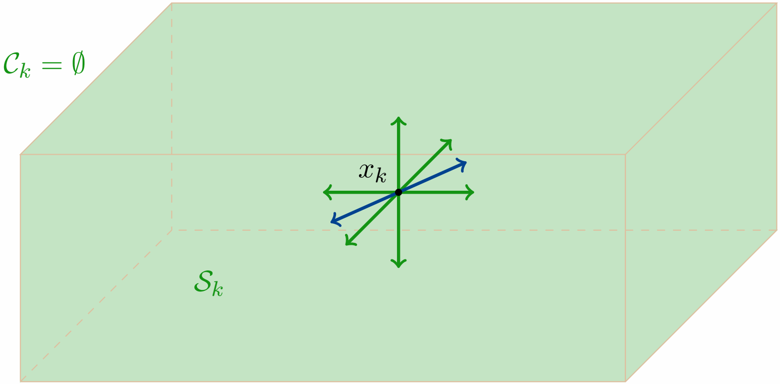
Per-step cost: Number of generators.

Low-Eval directions: Illustration for bound constraints

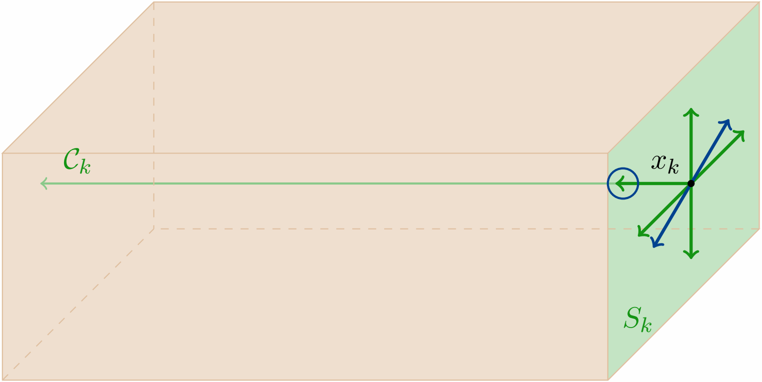
- In $\mathcal{C}_k$: Random subset of generators.



- In $\mathcal{S}_k$: Random one-dimensional subspace $[\boldsymbol{d} \ \ -\boldsymbol{d}]$.



- In $\mathcal{C}_k$: Random subset of generators.
- In $\mathcal{S}_k$: Random one-dimensional subspace $[\boldsymbol{d} \ -\boldsymbol{d}]$.



Comparison (MATLAB)

Us

- `ConstFLE`: Full-Low framework with $\gamma = 1$.
- `ConstBFGS`: Full-Eval steps only ($\gamma = 0$).
- `dspfd`: Low-Eval steps only ($\gamma = \infty$).

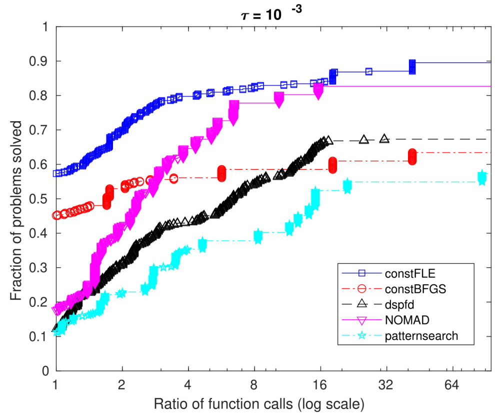
Them

- `NOMAD` (Montréal team!): MATLAB implementation, no search step, progressive barrier for non-bound constraints.
- `patternsearch`: Toolbox function, uses tangent cone generators.

- **Budget:** $100(n+1)$ evaluations.
- **Criterion:** $f(x_0) - f(x_k) \geq \tau(f(x_0) - f_{best})$ ($\tau = 10^{-3}$).

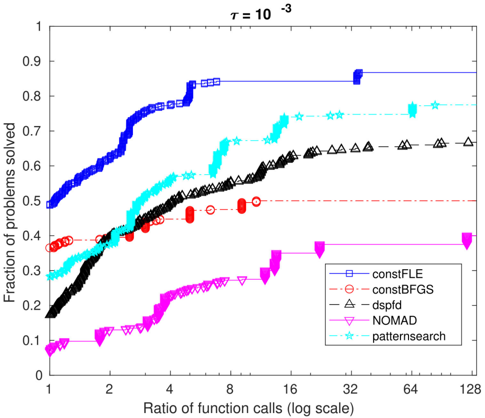
Smooth bound-constrained problems

- 41 CUTEst problems with bounds.
- Dimensions: $2 \leq n \leq 20$.



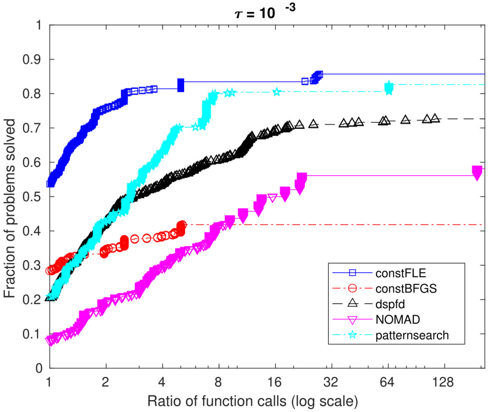
Smooth linearly-constrained problems

- 40 CUTEst problems with at least one linear inequality constraint.
- Dimensions: $2 \leq n \leq 15$, $1 \leq m_I \leq 2000$.



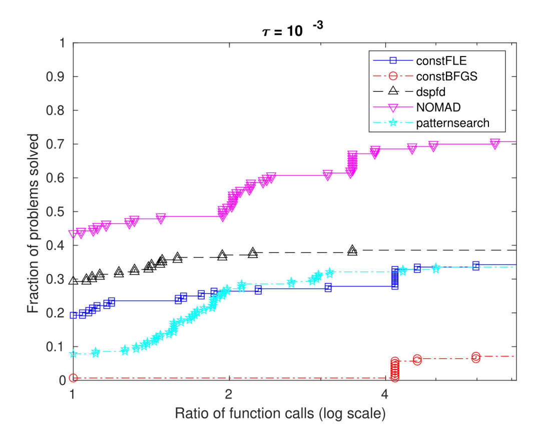
Nonsmooth linearly-constrained problems

- 52 nonsmooth problems with linear inequality constraints.
- CUTEst problems+nonsmooth penalty terms for some constraints.
- Dimensions: $2 \leq n \leq 20$, $1 \leq m_I \leq 15$.



Nonsmooth problems and linear inequalities

- 22 nonsmooth problems (Lukšan, Vleck '00) with at least one linear inequality constraint.
- Dimensions: $2 \leq n \leq 20$, $1 \leq m_I \leq 15$.



Takeaways

Unconstrained takeaways (Berahas et al '22)

- `Full-Eval` steps good for smooth problems.
- `Low-Eval` steps good for nonsmooth problems.

Linearly constrained takeaways

- `Full-Eval` steps good for bounds (and linear equalities).
- `Low-Eval` steps good for linear inequalities.
- Nonsmoothness: We should have used structure!

Outline

Algorithm (again)

Inputs: $x_0 \in \mathcal{F}$, $\alpha_0 > 0$, $t_0 = \texttt{Full-Eval}$, $\gamma \in [0, \infty]$.

For $k = 0, 1, 2, \dots$

- $t_k = \texttt{Full-Eval}$:
  - Compute $\bar{x}_k = P_{\mathcal{F}}\left[x_k + p_k\right]$.
  - Find $\beta_k \in \{1, \frac{1}{2}, \frac{1}{4}, \dots\}$ such that

$$f(x_k + \beta_k(\bar{x}_k - x_k)) \leq f(x_k) + c\,\beta_k g_k^{\mathrm{T}}(\bar{x}_k - x_k).$$

  - If $\beta_k \geq \gamma\alpha_k$, set $x_{k+1} = x_k + \beta_k(\bar{x}_k - x_k)$, $\alpha_{k+1} = \alpha_k$ and $t_{k+1} = \texttt{Full-Eval}$. Otherwise set $x_{k+1} = x_k$, $\alpha_{k+1} = \alpha_k$ and $t_{k+1} = \texttt{Low-Eval}$.
- $t_k = \texttt{Low-Eval}$:
  - Compute $D_k \subset \mathbb{R}^n$ feasible directions.
  - If $\exists d_k \in D_k$ such that $x_k + \alpha_k d_k \in \mathcal{F}$

$$f(x_k + \alpha_k d_k) \leq f(x_k) - \rho(\alpha_k),$$

    set $x_{k+1} = x_k + \alpha_k d_k$, $\alpha_{k+1} = 2\alpha_k$, $t_{k+1} = \texttt{Low-Eval}$.
  - Otherwise set $x_{k+1} = x_k$, $\alpha_k = \alpha_k/2$.
    Choose t_{k+1} depending on $t_{k-1}, \dots, t_{k-\log_{1/2}(\beta_k)}$.

Theory: Smooth setting

Assumptions (problem)

- f bounded below.
- ∇f Lipschitz continuous.

Convergence metric: $\|q(x)\|$, $\quad q(x) := P_{\mathcal{F}}\left[x - \nabla f(x)\right] - x.$

Theory: Smooth setting

Assumptions (problem)

- f bounded below.
- ∇f Lipschitz continuous.

Convergence metric: $\|q(x)\|, \qquad q(x) := P_{\mathcal{F}}\left[x - \nabla f(x)\right] - x.$

Assumptions (algorithm)

- Accurate gradient estimate g_k

$$\|g_k - \nabla f(x_k)\| \le u_g \|q_k^g\|, \qquad q_k^g := P_{\mathcal{F}}\left[x_k - g_k\right] - x_k.$$

Can be satisfied in finite time.

- Descent-type direction:

$$p_k = -g_k \quad \Rightarrow \quad -g_k^{\mathrm{T}} q_k^g \ge \|q_k^g\|^2.$$

More general conditions possible.

Convergence in the smooth setting

Theorem

The method reaches x_K such that

$$\|q(x_K)\| = \|P_{\mathcal{F}}\left[x_k - \nabla q(x_k)\right] - x_k\| \le \epsilon$$

in at most $\mathcal{O}(\epsilon^{-2})$ successful `Full-Eval` steps.

Convergence in the smooth setting

Theorem

The method reaches x_K such that

$$\|q(x_K)\| = \|P_{\mathcal{F}}\left[x_k - \nabla q(x_k)\right] - x_k\| \leq \epsilon$$

in at most $\mathcal{O}(\epsilon^{-2})$ successful `Full-Eval` steps.

- Proof: Classical backtracking line search (if $\gamma = 0$, projected gradient proof!).
- Limitations: No function evaluation count.
- On par with unconstrained case (Berahas et al '22).

Theory: nonsmooth setting

Assumptions (problem)

- f bounded below.
- f locally Lipschitz continuous.

Convergence metric: $f^\circ(x;d) = \limsup_{\substack{y \to x,\ y \in \mathcal{F} \\ t \downarrow 0,\ y+td \in \mathcal{F}}} \frac{f(y+td)-f(y)}{t}.$

Theory: nonsmooth setting

Assumptions (problem)

- f bounded below.
- f locally Lipschitz continuous.

Convergence metric: $f^\circ(x;d) = \limsup_{\substack{y \to x,\ y \in \mathcal{F} \\ t \downarrow 0,\ y+td \in \mathcal{F}}} \frac{f(y+td)-f(y)}{t}$.

Assumptions (algorithm)

- `Full-Eval` steps satisfy $\|q_k^g\| \geq \epsilon_g > 0$.
- $\{x_k\}$ bounded.

Convergence in the nonsmooth setting

Theorem

There exists a subsequence of **unsuccessful** `Low-Eval` **iterations** $\mathcal{K}$ such that

- $\lim_{k \in \mathcal{K}} x_k = x_* \in \mathcal{F}$.
- $f^\circ(x_*; d) \geq 0$ for any refining direction d

$$d \in \mathcal{R} = \left\{ \lim_{k \in \mathcal{K}} \frac{d_k}{\|d_k\|}, d_k \in D_k \forall k \in \mathcal{K} \right\}.$$

Convergence in the nonsmooth setting

Theorem

There exists a subsequence of **unsuccessful `Low-Eval` iterations** $\mathcal{K}$ such that

- $\lim_{k \in \mathcal{K}} x_k = x_* \in \mathcal{F}$.
- $f^\circ(x_*; d) \geq 0$ for any refining direction d

$$d \in \mathcal{R} = \left\{ \lim_{k \in \mathcal{K}} \frac{d_k}{\|d_k\|}, d_k \in D_k \forall k \in \mathcal{K} \right\}.$$

- Get stationarity with density assumptions on $\mathcal{R}$.
- Results purely based on `Low-Eval` steps behavior (if $\gamma = \infty$, direct-search proof!)
- Again matches the unconstrained setting.

Summary

Full-low framework⇒linear constraints

- Full steps for smoothness⇒ bound constraints?
- Low-eval steps for nonsmoothness/noise ⇒ linear inequalities?

Our results and more

- One implementation⇒ Many possible variants!
- Good numerics⇒ Still room for improvement.
- Theoretical support⇒ Stronger guarantees based on switching.

That's it!

References

- *Full-low evaluation methods for bound and linearly constrained derivative-free optimization*
  C. W. Royer, O. Sohab, L. N. Vicente
  Accepted in *Computational Optimization and Applications* last week!
- *Full-low evaluation methods for derivative-free optimization*
  O. Sohab, PhD Thesis, defended July 9, 2024.
- Code: `https://github.com/sohaboumaima/FLE`

That's it!

References

- *Full-low evaluation methods for bound and linearly constrained derivative-free optimization*
  C. W. Royer, O. Sohab, L. N. Vicente
  Accepted in *Computational Optimization and Applications* last week!
- *Full-low evaluation methods for derivative-free optimization*
  O. Sohab, PhD Thesis, defended July 9, 2024.
- Code: https://github.com/sohaboumaima/FLE

Thank you!

clement.royer@lamsade.dauphine.fr