Nonconvex optimization algorithms with complexity guarantees

Clément W. Royer

Soutenance d'Habilitation à Diriger des Recherches

September 22, 2025

Background

- **PhD** (2016)
  Université de Toulouse.
- **Postdoc** (2016 - 2019)
  University of Wisconsin-Madison.
- **Faculty** (2019 -)
  Université Paris Dauphine-PSL.

Collaborators since the PhD

Co-authors (including **junior researchers** and **group members**)

- **North America** Albert S. Berahas, Kwassi Joseph Dzahini, Michael J. O'Neill, Akwum Onwunta, Daniel P. Robinson, Oumaima Sohab, Stephen J. Wright (USA); Youssef Diouane, Warren Hare, Gabriel Jarry-Bolduc (Canada).
- **Dauphine** Rémi Chan–Renous-Legoubin, Yann Chevaleyre, Denis Cornaz, Florentin Goyens, Sébastien Kerleau, Laurent Meunier.
- **France** Jérémy Rapin, Olivier Teytaud.
- **Europe** Vladimir Kunc, Vyacheslav Kungurtsev (Czech Republic); Francesco Rinaldi, Damiano Zeffiro (Italy).
- **Africa** El Houcine Bergou (Morocco).
- **Oceania** Lindon Roberts (Australia).

Collaborators since the PhD

Co-authors (including **junior researchers** and **group members**)

- **North America** Albert S. Berahas, Kwassi Joseph Dzahini, Michael J. O'Neill, Akwum Onwunta, Daniel P. Robinson, Oumaima Sohab, Stephen J. Wright (USA); Youssef Diouane, Warren Hare, Gabriel Jarry-Bolduc (Canada).
- **Dauphine** Rémi Chan–Renous-Legoubin, Yann Chevaleyre, Denis Cornaz, Florentin Goyens, Sébastien Kerleau, Laurent Meunier.
- **France** Jérémy Rapin, Olivier Teytaud.
- **Europe** Vladimir Kunc, Vyacheslav Kungurtsev (Czech Republic); Francesco Rinaldi, Damiano Zeffiro (Italy).
- **Africa** El Houcine Bergou (Morocco).
- **Oceania** Lindon Roberts (Australia).

Other collaborations

Alexandre Allauzen, Antonin Chambolle, Irène Waldspurger, Florian Yger.

Special people

The group @ EUROPT 2025

- Bastien Cavarretta (PhD 2024 -)
- Sébastien Kerleau (PhD $\to$ November 2025)
- Iskander Legheraba (PhD $\to$ September 2025)
- Annette Dumas (Postdoc $\to$ September 2025).

Special people

The group @ EUROPT 2025

- Bastien Cavarretta (PhD 2024 -)
- Sébastien Kerleau
  (PhD → November 2025)
- Iskander Legheraba
  (PhD → September 2025)
- Annette Dumas
  (Postdoc → September 2025).

Other group members

- Gaetano Agazzotti (Master, 2025 -)
- Florentin Goyens (Postdoc, 2022 - 2024)
- Rémi Chan–Renous Legoubin (Master, 2021).
- Thomas Georges, Marc Kaspar, Christian Kayo, Eloi Martin, Luca Solbiati (Master, [2021,2024]).

Funding sources

cnrs SCIENCES INFORMATIQUES

DARPA

DEPARTMENT OF ENERGY · UNITED STATES OF AMERICA ·

ffCRfCRf
FONDS FRANCE CANADA POUR LA RECHERCHE
FRANCE CANADA RESEARCH FUND

FACE FOUNDATION

PGMO
Programme Gaspard Monge pour l'optimisation, la recherche opérationnelle et leurs interactions avec les sciences des données

FRANCE 2030 PROGRAMME DE RECHERCHE
INTELLIGENCE ARTIFICIELLE

PR[AI]RIE
PaRis Artificial Intelligence Research InstitutE

My research

Numerical optimization

- Design algorithms to minimize a (continuous) function.

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}}\, f(\boldsymbol{x}).$$

- Prove theoretical guarantees.
- Validate practical performance.

Numerical optimization

- Design algorithms to minimize a (continuous) function.

$$\operatorname*{minimize}_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}).$$

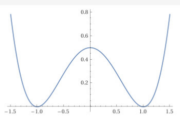
- Prove theoretical guarantees.
- Validate practical performance.

Recurring theme: Complexity analysis in nonconvex settings.

Main problem

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\text{minimize}} \, f(\boldsymbol{x})$$

f smooth ($\mathcal{C}^1$ or $\mathcal{C}^2$) and nonconvex.

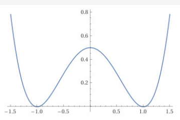


Main problem

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}}\; f(\boldsymbol{x})$$

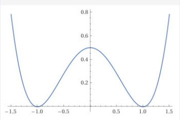
f smooth ($\mathcal{C}^1$ or $\mathcal{C}^2$) and nonconvex.

- NP-hard to solve, even locally.
- Settle to find approximate stationary points.



Main problem

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}}\; f(\boldsymbol{x})$$



f smooth ($\mathcal{C}^1$ or $\mathcal{C}^2$) and nonconvex.

- NP-hard to solve, even locally.
- Settle to find approximate stationary points.

Approximate stationary points

First-order For $\epsilon \in (0,1)$,

$$\|\nabla f(\boldsymbol{x})\| \le \epsilon.$$

Second-order For $\epsilon, \epsilon_H \in (0,1)^2$,

$$\|\nabla f(\boldsymbol{x})\| \le \epsilon, \qquad \nabla^2 f(\boldsymbol{x}) \succeq -\epsilon_H \boldsymbol{I}.$$

Complexity analysis

Problem minimize$_{\boldsymbol{x}\in\mathbb{R}^n}$ $f(\boldsymbol{x})$.

Solving the problem

- **Algorithm** Iterative process $\{\boldsymbol{x}_0, \boldsymbol{x}_1, \dots\}$.
- **Cost** Iterations, evaluations of f and derivatives, etc.

Complexity analysis

Problem minimize$_{\boldsymbol{x} \in \mathbb{R}^n}$ $f(\boldsymbol{x})$.

Solving the problem

- **Algorithm** Iterative process $\{\boldsymbol{x}_0, \boldsymbol{x}_1, \dots\}$.
- **Cost** Iterations, evaluations of f and derivatives, etc.

Complexity

Given ϵ, ϵ_H and an algorithm $\{\boldsymbol{x}_j\}_j$, find worst-case cost of the algorithm to reach $\boldsymbol{x}_J$ such that

1. $\|\nabla f(\boldsymbol{x}_J)\| \le \epsilon$, or
2. $\|\nabla f(\boldsymbol{x}_J)\| \le \epsilon, \quad \nabla^2 f(\boldsymbol{x}_J) \succeq -\epsilon_H \boldsymbol{I}$.

Canonical example: Gradient descent

Input $x_0 \in \mathbb{R}^n$.
For $j = 0, 1, 2, \ldots$

- Compute $\alpha_j > 0$.
- Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j - \alpha_j \nabla f(\boldsymbol{x}_j)$.

Input $x_0 \in \mathbb{R}^n$.
For $j = 0, 1, 2, \ldots$

- Compute $\alpha_j > 0$.
- Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j - \alpha_j \nabla f(\boldsymbol{x}_j)$.

- Stepsize α_j: Chosen constant or with line search.
- **Cost** Iterations/Calls to ∇f + Calls to f (Line search).

Complexity of gradient descent

Gradient descent $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j - \alpha_j \nabla f(\boldsymbol{x}_j)$.

Goal $\|\nabla f(\boldsymbol{x})\| \leq \epsilon$.

Complexity theorem

Assume ∇f Lipschitz continuous, f bounded below. Then the method runs in at most

$$\mathcal{O}(\epsilon^{-2})$$

iterations/calls to ∇f.

Complexity of gradient descent

Gradient descent $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j - \alpha_j \nabla f(\boldsymbol{x}_j)$.

Goal $\|\nabla f(\boldsymbol{x})\| \leq \epsilon$.

Complexity theorem

Assume ∇f Lipschitz continuous, f bounded below. Then the method runs in at most

$$\mathcal{O}(\epsilon^{-2})$$

iterations/calls to ∇f.

- Holds for fixed/adaptive α_j choices.
- With line search, cost in calls to f is also $\mathcal{O}(\epsilon^{-2})$.
- Sharp bounds in terms of ϵ.

Three axes of my research

- **Newton-type methods**
  - Obtain sharp complexity with Newton steps.
  - Study special classes of nonconvex problems.

Three axes of my research

- **Newton-type methods**
  - Obtain sharp complexity with Newton steps.
  - Study special classes of nonconvex problems.
- **Conjugate gradient methods**
  - Restart conditions in nonlinear case.
  - Checks to detect nonconvex behavior.

Three axes of my research

- **Newton-type methods**
  - Obtain sharp complexity with Newton steps.
  - Study special classes of nonconvex problems.
- **Conjugate gradient methods**
  - Restart conditions in nonlinear case.
  - Checks to detect nonconvex behavior.
- **Derivative-free methods**
  - Improve complexity with randomness.
  - Use geometry to handle difficult settings.

Three axes of my research

- **Newton-type methods**
  - Obtain sharp complexity with Newton steps.
  - Study special classes of nonconvex problems.
- **Conjugate gradient methods**
  - Restart conditions in nonlinear case.
  - Checks to detect nonconvex behavior.
- **Derivative-free methods**
  - Improve complexity with randomness.
  - Use geometry to handle difficult settings.

This presentation: Focus on contributions with students/postdocs.

Outline

Outline

Problem setup

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}}\, f(\boldsymbol{x})$$

$f \in \mathcal{C}^2$ bounded below and nonconvex.

Problem setup

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}}\, f(\boldsymbol{x})$$

$f \in \mathcal{C}^2$ bounded below and nonconvex.

Goal: Reach (ϵ, ϵ_H)-point

$$\|\nabla f(\boldsymbol{x})\| \le \epsilon \quad \text{and} \quad \nabla^2 f(\boldsymbol{x}) \succeq -\epsilon_H \boldsymbol{I}.$$

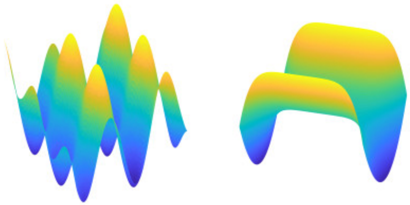
- For convex functions: Second condition always true$\Rightarrow$ Close to a global minimum!
- For nonconvex functions: Depends on the function landscape.

Goal Reach (ϵ, ϵ_H)-points

Bad instances

(ϵ, ϵ_H)-points can be close to

- Local, non-global minima.
- High-order saddle points.



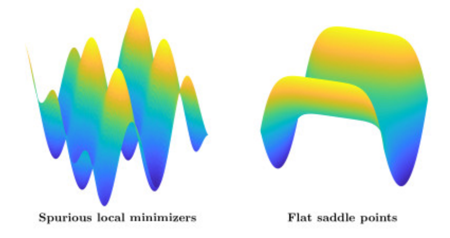
Spurious local minimizers　　Flat saddle points

Goal Reach (ϵ, ϵ_H)-points

Bad instances

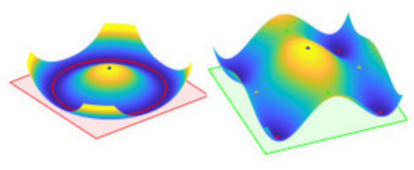
(ϵ, ϵ_H)-points can be close to

- Local, non-global minima.
- High-order saddle points.



Nice instances

- (ϵ, ϵ_H)-points close to local minima.
- All local minima are global.



Figures: J. Wright and Y. Ma, *High-Dimensional Data Analysis with Low-Dimensional Models*, 2022.

Nicest case in optimization: Strongly convex functions

- $f : \mathbb{R}^n \to \mathbb{R}$ $\mathcal{C}^2$ is μ-strongly convex if

$$\nabla^2 f(\boldsymbol{x}) \succeq \mu \, \boldsymbol{I}.$$

- Unique minimum $\boldsymbol{x}^* \in \mathbb{R}^n$.

One consequence of μ-strong convexity

For any $\gamma > 0$,

$$\|\nabla f(\boldsymbol{x})\| \leq \gamma \quad \Rightarrow \quad \|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \frac{2\gamma}{\mu}.$$

Strict saddle functions

Definition (Adapted from Ge et al '17)

f is $(\gamma, \lambda, \mu, \delta)$-strict saddle with $\gamma, \lambda, \mu, \delta > 0$ if at any $\boldsymbol{x} \in \mathbb{R}^n$, one of the above holds:

1. $\|\nabla f(\boldsymbol{x})\| \geq \gamma$;
2. $\lambda_{\min}\left(\nabla^2 f(\boldsymbol{x})\right) \leq -\beta$;
3. There exists x^* local minimum of f such that

$$\|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \delta \quad \text{and} \quad \nabla^2 f(\boldsymbol{y}) \succeq \gamma \boldsymbol{I} \succ 0 \quad \forall \boldsymbol{y}, \ \|\boldsymbol{y} - \boldsymbol{x}^*\| \leq 2\delta.$$

Strict saddle functions

Definition (Adapted from Ge et al '17)

f is $(\gamma, \lambda, \mu, \delta)$-strict saddle with $\gamma, \lambda, \mu, \delta > 0$ if at any $\boldsymbol{x} \in \mathbb{R}^n$, one of the above holds:

1. $\|\nabla f(\boldsymbol{x})\| \geq \gamma$;
2. $\lambda_{\min}\left(\nabla^2 f(\boldsymbol{x})\right) \leq -\beta$;
3. There exists x^* local minimum of f such that

$$\|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \delta \quad \text{and} \quad \nabla^2 f(\boldsymbol{y}) \succeq \gamma \boldsymbol{I} \succ 0 \quad \forall \boldsymbol{y},\ \|\boldsymbol{y} - \boldsymbol{x}^*\| \leq 2\delta.$$

- Similar in spirit to phase/regional complexity frameworks.
- Includes strongly convex as a special case.

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\Delta_0 > 0$, $\eta > 0$.

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

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\Delta_0 > 0$, $\eta > 0$.

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

1. Define $m_j(\boldsymbol{x}_j + \boldsymbol{s}) := \nabla f(\boldsymbol{x}_j)^{\mathrm{T}}\boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^{\mathrm{T}}\nabla^2 f(\boldsymbol{x}_j)\boldsymbol{s}$ and compute

$$\boldsymbol{s}_j \in \operatorname*{argmin}_{\|\boldsymbol{s}\|\leq\Delta_j} m_j(\boldsymbol{x}_j + s).$$

Trust region for minimize$_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x})$

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\Delta_0 > 0$, $\eta > 0$.

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

1. Define $m_j(\boldsymbol{x}_j + \boldsymbol{s}) := \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{s} + \frac{1}{2} \boldsymbol{s}^{\mathrm{T}} \nabla^2 f(\boldsymbol{x}_j) \boldsymbol{s}$ and compute
$$\boldsymbol{s}_j \in \operatorname*{argmin}_{\|\boldsymbol{s}\| \le \Delta_j} m_j(\boldsymbol{x}_j + s).$$
2. Compute $\rho_j = \frac{f(\boldsymbol{x}_j) - f(\boldsymbol{x}_j + \boldsymbol{s}_j)}{m_j(\boldsymbol{x}_j) - m_j(\boldsymbol{x}_j + \boldsymbol{s}_j)}$.

Trust region for minimize$_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x})$

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\Delta_0 > 0$, $\eta > 0$.

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

1. Define $m_j(\boldsymbol{x}_j + \boldsymbol{s}) := \nabla f(\boldsymbol{x}_j)^{\mathrm{T}}\boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^{\mathrm{T}}\nabla^2 f(\boldsymbol{x}_j)\boldsymbol{s}$ and compute
$$\boldsymbol{s}_j \in \operatorname*{argmin}_{\|\boldsymbol{s}\|\leq\Delta_j} m_j(\boldsymbol{x}_j + s).$$
2. Compute $\rho_j = \frac{f(\boldsymbol{x}_j) - f(\boldsymbol{x}_j + \boldsymbol{s}_j)}{m_j(\boldsymbol{x}_j) - m_j(\boldsymbol{x}_j + \boldsymbol{s}_j)}$.
3. If $\rho_j \geq \eta$, set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \boldsymbol{s}_j$ and $\Delta_{j+1} = 2\Delta_j$.
4. Otherwise, set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j$ and $\Delta_{j+1} = 0.5\Delta_j$.

Trust region for minimize$_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x})$

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\Delta_0 > 0$, $\eta > 0$.

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

1. Define $m_j(\boldsymbol{x}_j + \boldsymbol{s}) := \nabla f(\boldsymbol{x}_j)^{\mathrm{T}}\boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^{\mathrm{T}}\nabla^2 f(\boldsymbol{x}_j)\boldsymbol{s}$ and compute
$$\boldsymbol{s}_j \in \operatorname*{argmin}_{\|\boldsymbol{s}\|\le\Delta_j} m_j(\boldsymbol{x}_j + s).$$
2. Compute $\rho_j = \frac{f(\boldsymbol{x}_j) - f(\boldsymbol{x}_j + \boldsymbol{s}_j)}{m_j(\boldsymbol{x}_j) - m_j(\boldsymbol{x}_j + \boldsymbol{s}_j)}$.
3. If $\rho_j \ge \eta$, set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \boldsymbol{s}_j$ and $\Delta_{j+1} = 2\Delta_j$.
4. Otherwise, set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j$ and $\Delta_{j+1} = 0.5\Delta_j$.

- No Hessian term in m_j: Gradient descent.
- **Cost** Iterations.

Complexity results for trust region

Goal $\|\nabla f(\boldsymbol{x}_J)\| \leq \epsilon$ and $\nabla^2 f(\boldsymbol{x}_J) \succeq -\epsilon_H \boldsymbol{I}$.

General nonconvex f

$$J = \mathcal{O}\left(\max\{\epsilon^{-2}, \epsilon_H^{-3}\}\right).$$

Strict saddle f (w/ Florentin Goyens)

If f is $(\gamma, \lambda, \mu, \delta)$-strict saddle,

$$J = J_f + J_\epsilon, \qquad \begin{cases} J_f &= \mathcal{O}\left(\max\left\{\gamma^{-2}\lambda^{-1}, \gamma^{-2}\mu^{-1}, \lambda^{-3}, \mu^{-3}, \mu^{-2}\delta^{-1}\right\}\right) \\ J_\epsilon &= \log\log\left[\mathcal{O}\left(\mu\epsilon^{-1}\right)\right]. \end{cases}$$

Complexity results for trust region

Goal $\|\nabla f(\boldsymbol{x}_J)\| \le \epsilon$ and $\nabla^2 f(\boldsymbol{x}_J) \succeq -\epsilon_H \boldsymbol{I}$.

General nonconvex f

$$J = \mathcal{O}\left(\max\{\epsilon^{-2}, \epsilon_H^{-3}\}\right).$$

Strict saddle f (w/ Florentin Goyens)
If f is $(\gamma, \lambda, \mu, \delta)$-strict saddle,

$$J = J_f + J_\epsilon, \qquad \left\{ \begin{array}{rcl} J_f & = & \mathcal{O}\left(\max\left\{\gamma^{-2}\lambda^{-1}, \gamma^{-2}\mu^{-1}, \lambda^{-3}, \mu^{-3}, \mu^{-2}\delta^{-1}\right\}\right) \\ J_\epsilon & = & \log\log\left[\mathcal{O}\left(\mu\epsilon^{-1}\right)\right]. \end{array} \right.$$

Complexity results for trust region

Goal $\|\nabla f(\boldsymbol{x}_J)\| \le \epsilon$ and $\nabla^2 f(\boldsymbol{x}_J) \succeq -\epsilon_H \boldsymbol{I}$.

General nonconvex f

$$J = \mathcal{O}\left(\max\{\epsilon^{-2}, \epsilon_H^{-3}\}\right).$$

Strict saddle f (w/ Florentin Goyens)

If f is $(\gamma, \lambda, \mu, \delta)$-strict saddle,

$$J = J_f + J_\epsilon, \qquad \begin{cases} J_f &= \mathcal{O}\left(\max\left\{\gamma^{-2}\lambda^{-1}, \gamma^{-2}\mu^{-1}, \lambda^{-3}, \mu^{-3}, \mu^{-2}\delta^{-1}\right\}\right) \\ J_\epsilon &= \log\log\left[\mathcal{O}\left(\mu\epsilon^{-1}\right)\right]. \end{cases}$$

- Second term vanishes for large ϵ.
- $\log\log$ dependency in ϵ not on ϵ_H ($\sim$ strongly convex case)!

Improved complexity using Newton steps

More contributions on Newton-type methods

Improved complexity using Newton steps

- For generic nonconvex f, regularize trust-region models!
  (w/ F. E. Curtis, D. P. Robinson, S. J. Wright).

$$\mathcal{O}\left(\max\{\epsilon^{-2}, \epsilon_H^{-3}\}\right) \quad \longrightarrow \quad \mathcal{O}\left(\max\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\}\right).$$

More contributions on Newton-type methods

Improved complexity using Newton steps

- For generic nonconvex f, regularize trust-region models!
  (w/ F. E. Curtis, D. P. Robinson, S. J. Wright).

$$\mathcal{O}\left(\max\{\epsilon^{-2}, \epsilon_H^{-3}\}\right) \quad \longrightarrow \quad \mathcal{O}\left(\max\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\}\right).$$

- Line-search variants *(w/ M. O'Neill, S. J. Wright)*.

More contributions on Newton-type methods

Improved complexity using Newton steps

- For generic nonconvex f, regularize trust-region models! *(w/ F. E. Curtis, D. P. Robinson, S. J. Wright)*.

$$\mathcal{O}\left(\max\{\epsilon^{-2}, \epsilon_H^{-3}\}\right) \quad \longrightarrow \quad \mathcal{O}\left(\max\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\}\right).$$

- Line-search variants *(w/ M. O'Neill, S. J. Wright)*.

Structured nonconvex problems

More contributions on Newton-type methods

Improved complexity using Newton steps

- For generic nonconvex f, regularize trust-region models!
  (w/ F. E. Curtis, D. P. Robinson, S. J. Wright).

$$\mathcal{O}\left(\max\{\epsilon^{-2}, \epsilon_H^{-3}\}\right) \quad \longrightarrow \quad \mathcal{O}\left(\max\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\}\right).$$

- Line-search variants *(w/ M. O'Neill, S. J. Wright)*.

Structured nonconvex problems

- Manifold constraints: minimize$_{\boldsymbol{x} \in \mathcal{M}} f(\boldsymbol{x})$, $\quad \mathcal{M} = \mathbb{S}^{n-1}, \mathbb{C}^n, \dots$
  (w/ Florentin Goyens)

More contributions on Newton-type methods

Improved complexity using Newton steps

- For generic nonconvex f, regularize trust-region models!
  (w/ F. E. Curtis, D. P. Robinson, S. J. Wright).

$$\mathcal{O}\left(\max\{\epsilon^{-2}, \epsilon_H^{-3}\}\right) \quad \longrightarrow \quad \mathcal{O}\left(\max\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\}\right).$$

- Line-search variants *(w/ M. O'Neill, S. J. Wright)*.

Structured nonconvex problems

- Manifold constraints: minimize$_{\boldsymbol{x} \in \mathcal{M}} f(\boldsymbol{x})$, $\quad \mathcal{M} = \mathbb{S}^{n-1}, \mathbb{C}^n, \dots$
  (w/ Florentin Goyens)
- Least-squares problems: minimize$_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{r}(\boldsymbol{x})\|^2$.
  (PhD Iskander Legheraba).

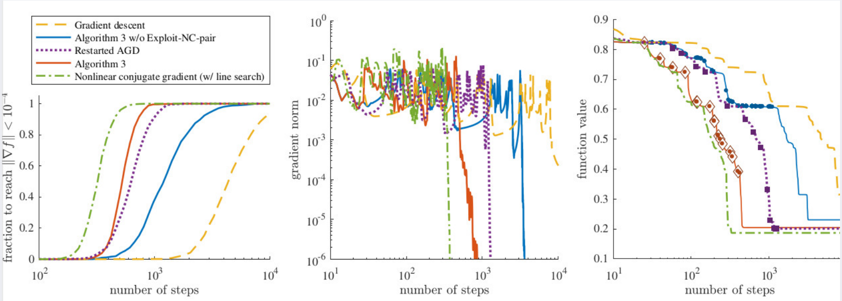
Outline

Problem minimize$_{\boldsymbol{x}\in\mathbb{R}^n}$ $f(\boldsymbol{x})$, f $\mathcal{C}^1$ nonconvex. **Goal** $\|\nabla f(\boldsymbol{x})\| \leq \epsilon$.

Setup

Problem minimize$_{\boldsymbol{x}\in\mathbb{R}^n}$ $f(\boldsymbol{x})$, f $\mathcal{C}^1$ nonconvex. **Goal** $\|\nabla f(\boldsymbol{x})\| \le \epsilon$.

An observation (from Carmon et al '17)



- Gradient descent (for f $\mathcal{C}^1$): $\mathcal{O}(\epsilon^{-2})$ calls to ∇f.
- Algorithm 3/AGD (for f $\mathcal{C}^2$): $\tilde{\mathcal{O}}(\epsilon^{-7/4})$ calls to ∇f.
- **All outperformed by standard nonlinear CG on a nonconvex regression task!**

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$.

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

Basic nonlinear CG framework

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$.

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

1. Compute $\alpha_j \in \{1, \theta, \theta^2, \dots\}$ such that

$$f(\boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j) < f(\boldsymbol{x}_j) + \eta \alpha_j \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{d}_j.$$

Basic nonlinear CG framework

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$.

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

1. Compute $\alpha_j \in \{1, \theta, \theta^2, \ldots\}$ such that

$$f(\boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j) < f(\boldsymbol{x}_j) + \eta \alpha_j \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{d}_j.$$

2. Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$.

Basic nonlinear CG framework

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$.

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

1. Compute $\alpha_j \in \{1, \theta, \theta^2, \ldots\}$ such that

$$f(\boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j) < f(\boldsymbol{x}_j) + \eta \alpha_j \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{d}_j.$$

2. Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$.
3. Set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1}) + \beta_{j+1} \boldsymbol{d}_j$.

Basic nonlinear CG framework

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$.

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

1. Compute $\alpha_j \in \{1, \theta, \theta^2, \dots\}$ such that
$$f(\boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j) < f(\boldsymbol{x}_j) + \eta \alpha_j \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{d}_j.$$
2. Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$.
3. Set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1}) + \beta_{j+1} \boldsymbol{d}_j$.
4. If $\nabla f(\boldsymbol{x}_{j+1})^{\mathrm{T}} \boldsymbol{d}_{j+1} \geq 0$, set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1})$.

Basic nonlinear CG framework

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$.

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

1. Compute $\alpha_j \in \{1, \theta, \theta^2, \dots\}$ such that
$$f(\boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j) < f(\boldsymbol{x}_j) + \eta \alpha_j \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{d}_j.$$
2. Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$.
3. Set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1}) + \beta_{j+1} \boldsymbol{d}_j$.
4. If $\nabla f(\boldsymbol{x}_{j+1})^{\mathrm{T}} \boldsymbol{d}_{j+1} \geq 0$, set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1})$.

- β_{j+1} chosen using standard formulas (PRP+), restart condition practical.
  No complexity guarantees!

Basic nonlinear CG framework

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$.

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

1. Compute $\alpha_j \in \{1, \theta, \theta^2, \dots\}$ such that

$$f(\boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j) < f(\boldsymbol{x}_j) + \eta \alpha_j \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{d}_j.$$

2. Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$.
3. Set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1}) + \beta_{j+1} \boldsymbol{d}_j$.
4. If $\nabla f(\boldsymbol{x}_{j+1})^{\mathrm{T}} \boldsymbol{d}_{j+1} \geq 0$, set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1})$.

- β_{j+1} chosen using standard formulas (PRP+), restart condition practical.
  No complexity guarantees!
- Our approach Change restart condition!

Nonlinear CG with restart conditions

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$.

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

1. Compute $\alpha_j \in \{1, \theta, \theta^2, \ldots\}$ such that

$$f(\boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j) < f(\boldsymbol{x}_j) + \eta \alpha_j \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{d}_j.$$

2. Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$.
3. Set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1}) + \beta_{j+1} \boldsymbol{d}_j$.

Nonlinear CG with restart conditions

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$, $p \geq 0$, $\kappa \in (0,1)$.

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

1. Compute $\alpha_j \in \{1, \theta, \theta^2, \dots\}$ such that
$$f(\boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j) < f(\boldsymbol{x}_j) + \eta \alpha_j \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{d}_j.$$
2. Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$.
3. Set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1}) + \beta_{j+1} \boldsymbol{d}_j$.
4. If $\nabla f(\boldsymbol{x}_{j+1})^{\mathrm{T}} \boldsymbol{d}_{j+1} \geq -\kappa \|\nabla f(\boldsymbol{x}_{j+1})\|^{1+p}$ or $\|\boldsymbol{d}_{j+1}\| \geq \kappa \|\nabla f(\boldsymbol{x}_{j+1})\|^{\frac{1+p}{2}}$, set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1})$.

Nonlinear CG with restart conditions

Init $\boldsymbol{x}_0 \in \mathbb{R}^n$, $(\eta, \theta) \in (0,1)^2$, $\boldsymbol{d}_0 = -\nabla f(\boldsymbol{x}_0)$, $p \geq 0$, $\kappa \in (0,1)$.

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

1. Compute $\alpha_j \in \{1, \theta, \theta^2, \ldots\}$ such that
$$f(\boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j) < f(\boldsymbol{x}_j) + \eta \alpha_j \nabla f(\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{d}_j.$$
2. Set $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$.
3. Set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1}) + \beta_{j+1} \boldsymbol{d}_j$.
4. If $\nabla f(\boldsymbol{x}_{j+1})^{\mathrm{T}} \boldsymbol{d}_{j+1} \geq -\kappa \|\nabla f(\boldsymbol{x}_{j+1})\|^{1+p}$ or $\|\boldsymbol{d}_{j+1}\| \geq \kappa \|\nabla f(\boldsymbol{x}_{j+1})\|^{\frac{1+p}{2}}$, set $\boldsymbol{d}_{j+1} = -\nabla f(\boldsymbol{x}_{j+1})$.

- Restarted iterations: Take gradient steps.
- Non-restarted: Guarantees on step quality.

Main result

Goal $\|\nabla f(\boldsymbol{x}_J)\| \leq \epsilon$.

Theorem (w/ Rémi Chan–Renous-Legoubin)

For restarted NCG, J is at most

$$\underbrace{\mathcal{O}(\epsilon^{-2})}_{\text{restarted}} + \underbrace{\mathcal{O}(\epsilon^{-(1+p)})}_{\text{non restarted}}$$

Main result

Goal $\|\nabla f(\boldsymbol{x}_J)\| \leq \epsilon$.

Theorem (w/ Rémi Chan–Renous-Legoubin)

For restarted NCG, J is at most

$$\underbrace{\mathcal{O}(\epsilon^{-2})}_{\text{restarted}} + \underbrace{\mathcal{O}(\epsilon^{-(1+p)})}_{\text{non restarted}}$$

- The bound is still $\mathcal{O}(\epsilon^{-2})$ (like gradient descent)
- But better results for **non-restarted iterations** for $p \in [0, 1)$!

Main result

Goal $\|\nabla f(\boldsymbol{x}_J)\| \leq \epsilon$.

Theorem (w/ Rémi Chan–Renous-Legoubin)

For restarted NCG, J is at most

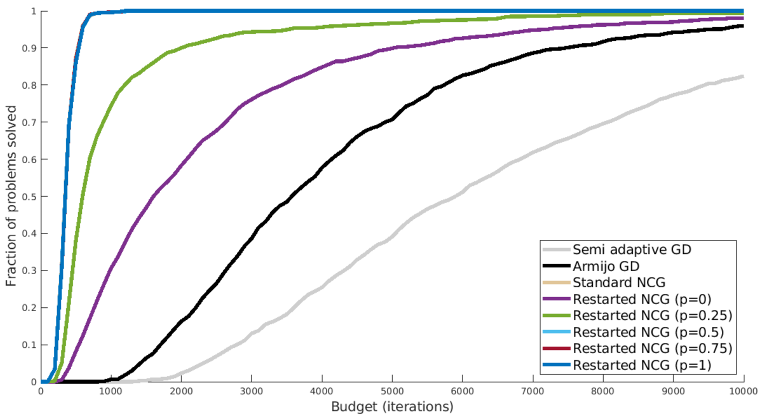
$$\underbrace{\mathcal{O}(\epsilon^{-2})}_{\text{restarted}} + \underbrace{\mathcal{O}(\epsilon^{-(1+p)})}_{\text{non restarted}}$$

- The bound is still $\mathcal{O}(\epsilon^{-2})$ (like gradient descent)
- But better results for **non-restarted iterations** for $p \in [0, 1)$!

How do restarted methods behave in practice?

Comparison

- Two gradient-based methods, Standard linear CG, restarted variants.
- Nonconvex regression instances from (Carmon et al '17).



More contributions on conjugate gradient methods

Restarted methodology *(w/ A. S. Berahas, M. O'Neill)*

- Applied to quasi-Newton schemes.
- Adapted to noisy optimization.

$\rightarrow$ *Restarting can be applied to yield complexity guarantees without harming the practical performance!*

More contributions on conjugate gradient methods

Restarted methodology *(w/ A. S. Berahas, M. O'Neill)*

- Applied to quasi-Newton schemes.
- Adapted to noisy optimization.

$\rightarrow$ *Restarting can be applied to yield complexity guarantees without harming the practical performance!*

Special case: quadratic functions *(w/ M. O'Neill, S. J. Wright)*

- Nonconvex quadratics from Newton-type methods.
- Can get complexity for **linear** conjugate gradient applied to those problems.

$\rightarrow$ *Extra checks in algorithms allow to detect and exploit nonconvexity!*

Outline

Derivative-free paradigm

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad f(\boldsymbol{x}), \qquad f \ \mathcal{C}^1 \quad \text{nonconvex.}$$

Blackbox/Derivative-free optimization

- **Derivatives unavailable for algorithmic use.**
- Only access to values of f.
- Those can take a long time to compute!

Derivative-free paradigm

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad f(\boldsymbol{x}), \qquad f\ \mathcal{C}^1 \quad \text{nonconvex.}$$

Blackbox/Derivative-free optimization

- **Derivatives unavailable for algorithmic use.**
- Only access to values of f.
- Those can take a long time to compute!

Complexity aspects

- **Goal** Find $\boldsymbol{x}_k$ such that $\|\nabla f(\boldsymbol{x}_k)\| \le \epsilon$.
- **Cost** Number of function evaluations.

Basic direct-search framework

Inputs $x_0 \in \mathbb{R}^n$, $\alpha_0 > 0$.

For $j = 0, 1, \dots$

- Choose $\mathcal{D}_j \subset \mathbb{R}^n$.

Basic direct-search framework

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\alpha_0 > 0$.

For $j = 0, 1, \ldots$

- Choose $\mathcal{D}_j \subset \mathbb{R}^n$.
- If $\exists\ \boldsymbol{d}_j \in \mathcal{D}_j$ such that

$$f(\boldsymbol{x}_j + \alpha_j\, \boldsymbol{d}_j) < f(\boldsymbol{x}_j) - \alpha_j^2 \|\boldsymbol{d}_j\|^2$$

  set $\boldsymbol{x}_{j+1} := \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$, $\alpha_{j+1} := 2\alpha_j$.

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\alpha_0 > 0$.

For $j = 0, 1, \dots$

- Choose $\mathcal{D}_j \subset \mathbb{R}^n$.
- If $\exists\ \boldsymbol{d}_j \in \mathcal{D}_j$ such that

$$f(\boldsymbol{x}_j + \alpha_j \, \boldsymbol{d}_j) < f(\boldsymbol{x}_j) - \alpha_j^2 \|\boldsymbol{d}_j\|^2$$

  set $\boldsymbol{x}_{j+1} := \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$, $\alpha_{j+1} := 2\alpha_j$.
- Otherwise, set $\boldsymbol{x}_{j+1} := \boldsymbol{x}_j$, $\alpha_{j+1} := 0.5\alpha_j$.

Basic direct-search framework

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\alpha_0 > 0$.

For $j = 0, 1, \dots$

- **Choose a PSS $\mathcal{D}_j \subset \mathbb{R}^n$.**
- If $\exists\ \boldsymbol{d}_j \in \mathcal{D}_j$ such that

$$f(\boldsymbol{x}_j + \alpha_j\, \boldsymbol{d}_j) < f(\boldsymbol{x}_j) - \alpha_j^2 \|\boldsymbol{d}_j\|^2$$

  set $\boldsymbol{x}_{j+1} := \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$, $\alpha_{j+1} := 2\alpha_j$.
- Otherwise, set $\boldsymbol{x}_{j+1} := \boldsymbol{x}_j$, $\alpha_{j+1} := 0.5\alpha_j$.

Positive Spanning Set (PSS)

$$\mathcal{D} \subset \mathbb{R}^n \text{ PSS} \qquad \Longleftrightarrow \qquad \mathrm{cm}(\mathcal{D}) = \max_{\boldsymbol{d} \in \mathcal{D}} \min_{\|\boldsymbol{v}\|=1} \frac{\boldsymbol{d}^{\mathrm{T}} \boldsymbol{v}}{\|\boldsymbol{d}\|} > 0.$$

$\mathrm{cm}(\mathcal{D})$: Cosine measure.

Goal $\|\nabla f(\boldsymbol{x}_J)\| \leq \epsilon$.

Theorem

If $\mathcal{D}_j = \mathcal{D} \ \forall j$, the method takes at most

$$\mathcal{O}\left(\ |\mathcal{D}| \ \mathrm{cm}(\mathcal{D})^{-2} \, \epsilon^{-2} \ \right)$$

calls to f.

Goal $\|\nabla f(\boldsymbol{x}_J)\| \leq \epsilon$.

Theorem

If $\mathcal{D}_j = \mathcal{D}\ \forall j$, the method takes at most

$$\mathcal{O}\left(\ |\mathcal{D}|\ \mathrm{cm}(\mathcal{D})^{-2}\,\epsilon^{-2}\ \right)$$

calls to f.

- $|\mathcal{D}|$ and $\mathrm{cm}(\mathcal{D})$ depend on dimension n!
- Best known bound: $\mathcal{O}(n^2\,\epsilon^{-2})$.

Complexity of deterministic direct search

Goal $\|\nabla f(\boldsymbol{x}_J)\| \leq \epsilon$.

Theorem

If $\mathcal{D}_j = \mathcal{D}\ \forall j$, the method takes at most

$$\mathcal{O}\left(\ |\mathcal{D}|\ \mathrm{cm}(\mathcal{D})^{-2}\,\epsilon^{-2}\ \right)$$

calls to f.

- $|\mathcal{D}|$ and $\mathrm{cm}(\mathcal{D})$ depend on dimension n!
- Best known bound: $\mathcal{O}(n^2\,\epsilon^{-2})$.

What if certain calls to f take too long to compute?

Direct search with stragglers

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\alpha_0 > 0$.

For $j = 0, 1, \dots$

- Choose $\mathcal{D}_j \subset \mathbb{R}^n$.
- If $\exists\ \boldsymbol{d}_j \in \mathcal{D}_j \setminus \mathcal{S}_j$ such that

$$f(\boldsymbol{x}_j + \alpha_j\, \boldsymbol{d}_j) < f(\boldsymbol{x}_j) - \alpha_j^2 \|\boldsymbol{d}_j\|^2$$

  set $\boldsymbol{x}_{j+1} := \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$, $\alpha_{j+1} := 2\alpha_j$.
- Otherwise, set $\boldsymbol{x}_{j+1} := \boldsymbol{x}_j$, $\alpha_{j+1} := 0.5\alpha_j$.

- $\mathcal{S}_j$ Unknown straggler evaluations (at most k).

Direct search with stragglers

Inputs $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\alpha_0 > 0$.

For $j = 0, 1, \ldots$

- **Choose a PkSS $\mathcal{D}_j \subset \mathbb{R}^n$.**
- If $\exists\ \boldsymbol{d}_j \in \mathcal{D}_j \setminus \mathcal{S}_j$ such that

$$f(\boldsymbol{x}_j + \alpha_j\, \boldsymbol{d}_j) < f(\boldsymbol{x}_j) - \alpha_j^2 \|\boldsymbol{d}_j\|^2$$

  set $\boldsymbol{x}_{j+1} := \boldsymbol{x}_j + \alpha_j \boldsymbol{d}_j$, $\alpha_{j+1} := 2\alpha_j$.
- Otherwise, set $\boldsymbol{x}_{j+1} := \boldsymbol{x}_j$, $\alpha_{j+1} := 0.5\alpha_j$.

- $\mathcal{S}_j$ Unknown straggler evaluations (at most k).
- Fix: Use richer $\mathcal{D}_j$s!

Definition

$\mathcal{D} \subset \mathbb{R}^n$ is a PkSS with $k \geq 1$ if

- Any $\mathcal{N} \subset \mathcal{D}$ with $|\mathcal{N}| = |\mathcal{D}| - k + 1$ is a PSS.
- Removing $k - 1$ vectors from $\mathcal{D}$ does not change its PSS nature.

Positive k-spanning sets (PkSS)

Definition

$\mathcal{D} \subset \mathbb{R}^n$ is a PkSS with $k \geq 1$ if

- Any $\mathcal{N} \subset \mathcal{D}$ with $|\mathcal{N}| = |\mathcal{D}| - k + 1$ is a PSS.
- Removing $k - 1$ vectors from $\mathcal{D}$ does not change its PSS nature.

The k-cosine measure *(w/ W. Hare, G. Jarry-Bolduc, Sébastien Kerleau)*

- For any $\mathcal{D} \subset \mathbb{R}^n$, the k-cosine measure of $\mathcal{D}$ is

$$\mathrm{cm}_k(\mathcal{D}) = \min_{\substack{\mathcal{N} \subset \mathcal{D} \\ |\mathcal{N}| = |\mathcal{D}| - k + 1}} \mathrm{cm}(\mathcal{N}).$$

- $\mathcal{D}$ PkSS $\iff \mathrm{cm}_k(\mathcal{D}) > 0$.

Complexity of direct search with PkSSs

Goal $\|\nabla f(\boldsymbol{x}_J)\| \le \epsilon$.

Theorem *(PhD Sébastien Kerleau)*

Suppose $\mathcal{D}_j = \mathcal{D}$ PkSS for all j. Then, the method takes at most

$$\mathcal{O}\left(|\mathcal{D}|\ \mathrm{cm}_k(\mathcal{D})^{-2}\epsilon^{-2}\right)$$

calls to f.

Complexity of direct search with PkSSs

Goal $\|\nabla f(\boldsymbol{x}_J)\| \le \epsilon$.

Theorem *(PhD Sébastien Kerleau)*

Suppose $\mathcal{D}_j = \mathcal{D}$ PkSS for all j. Then, the method takes at most

$$\mathcal{O}\left(|\mathcal{D}| \, \mathrm{cm}_k(\mathcal{D})^{-2} \epsilon^{-2}\right)$$

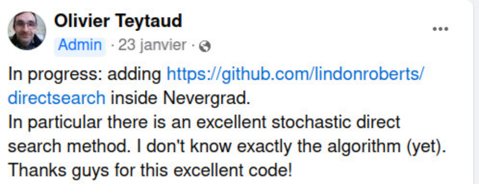
calls to f.

- $|\mathcal{D}|$ and $\mathrm{cm}_k(\mathcal{D})$ depend on k and n.
- Best found bound: $\mathcal{O}(kn^2\epsilon^{-2})$.

More contributions on direct-search methods

Complexity and direct search

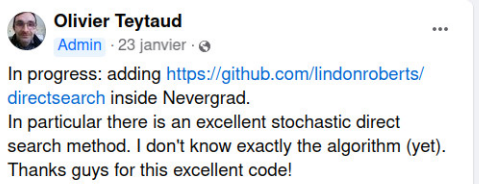
- Randomized subspace variant (Roberts & Royer ’23)



More contributions on direct-search methods

Complexity and direct search

- Randomized subspace variant (Roberts & Royer '23)


Olivier Teytaud
Admin · 23 janvier ·

In progress: adding https://github.com/lindonroberts/directsearch inside Nevergrad.
In particular there is an excellent stochastic direct search method. I don't know exactly the algorithm (yet).
Thanks guys for this excellent code!

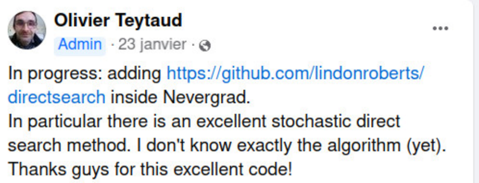
- Surveyed the area (Dzahini et al '25)

$\rightarrow$ Dimension dependencies can be improved through randomization!

More contributions on direct-search methods

Complexity and direct search

- Randomized subspace variant (Roberts & Royer '23)



- Surveyed the area (Dzahini et al '25)

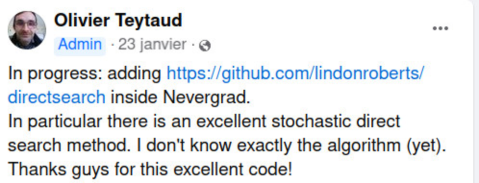
$\rightarrow$ Dimension dependencies can be improved through randomization!

Geometrical structures in direct search

More contributions on direct-search methods

Complexity and direct search

- Randomized subspace variant (Roberts & Royer '23)



- Surveyed the area (Dzahini et al '25)

$\rightarrow$ Dimension dependencies can be improved through randomization!

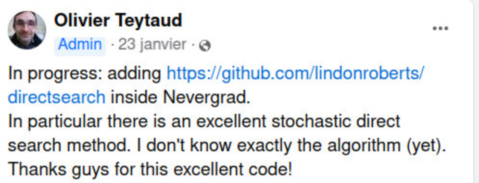
Geometrical structures in direct search

- Decomposition of PSSs along subspaces through graph theory arguments (Cornaz et al '25).

More contributions on direct-search methods

Complexity and direct search

- Randomized subspace variant (Roberts & Royer '23)



- Surveyed the area (Dzahini et al '25)

$\rightarrow$ Dimension dependencies can be improved through randomization!

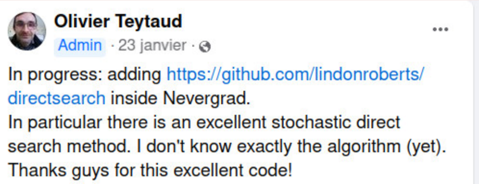
Geometrical structures in direct search

- Decomposition of PSSs along subspaces through graph theory arguments (Cornaz et al '25).
- Generation of PkSSs through polytopes (Kerleau '25).

More contributions on direct-search methods

Complexity and direct search

- Randomized subspace variant (Roberts & Royer '23)



- Surveyed the area (Dzahini et al '25)

$\rightarrow$ Dimension dependencies can be improved through randomization!

Geometrical structures in direct search

- Decomposition of PSSs along subspaces through graph theory arguments (Cornaz et al '25).
- Generation of PkSSs through polytopes (Kerleau '25).

$\rightarrow$ Discrete structures matter for direct search!

Outline

Complexity $\longrightarrow$ **Structures**

Research overview

Complexity $\longrightarrow$ Structures

- **Newton-type methods**
  - Obtain sharp complexity with Newton steps.
  - Study special classes of nonconvex problems.
- **Conjugate gradient methods**
  - Restart conditions in nonlinear case.
  - Checks to detect nonconvex behavior.
- **Derivative-free methods**
  - Improve complexity with randomness.
  - Use geometry to handle difficult settings.

Research overview

Complexity $\longrightarrow$ Structures

- **Newton-type methods**
  - Obtain sharp complexity with Newton steps.
    Optimal bounds within the class of second-order methods
  - Study special classes of nonconvex problems.
    Structured nonconvexity
- **Conjugate gradient methods**
  - Restart conditions in nonlinear case.
  - Checks to detect nonconvex behavior.
- **Derivative-free methods**
  - Improve complexity with randomness.
  - Use geometry to handle difficult settings.

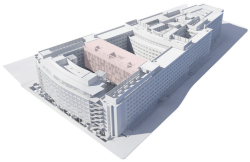
Complexity $\longrightarrow$ Structures

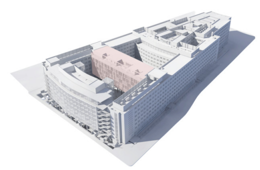
- **Newton-type methods**
  - Obtain sharp complexity with Newton steps.
    Optimal bounds within the class of second-order methods
  - Study special classes of nonconvex problems.
    Structured nonconvexity
- **Conjugate gradient methods**
  - Restart conditions in nonlinear case.
    Complexity obtained without hurting performance
  - Checks to detect nonconvex behavior.
    Leverage algorithmic structure
- **Derivative-free methods**
  - Improve complexity with randomness.
  - Use geometry to handle difficult settings.

Complexity $\longrightarrow$ Structures

- **Newton-type methods**
  - Obtain sharp complexity with Newton steps.
    Optimal bounds within the class of second-order methods
  - Study special classes of nonconvex problems.
    Structured nonconvexity
- **Conjugate gradient methods**
  - Restart conditions in nonlinear case.
    Complexity obtained without hurting performance
  - Checks to detect nonconvex behavior.
    Leverage algorithmic structure
- **Derivative-free methods**
  - Improve complexity with randomness.
    Subspace variants with complexity
  - Use geometry to handle difficult settings.
    Graph/Polytope structures

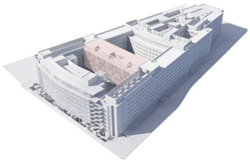
Research structures





Goal: Optimize moves during renovation

(with S. Airiau, L. Galand, J. Lang, S. Toubaline)

Goal: Optimize moves during renovation
(with S. Airiau, L. Galand, J. Lang, S. Toubaline)

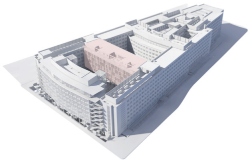
- Linear Program (simplicity)→ Other models are nonconvex!

Goal: Optimize moves during renovation

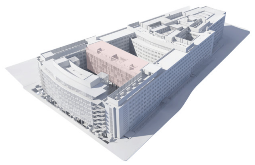
(with S. Airiau, L. Galand, J. Lang, S. Toubaline)

- Linear Program (simplicity) $\rightarrow$ Other models are nonconvex!
- Continuous LP solvers struggle $\rightarrow$ Revisit algorithms!

Goal: Optimize moves during renovation
(with S. Airiau, L. Galand, J. Lang, S. Toubaline)

- Linear Program (simplicity)$\rightarrow$ Other models are nonconvex!
- Continuous LP solvers struggle$\rightarrow$ Revisit algorithms!
- Model parameters to tune$\rightarrow$ Derivative-free methods!

Goal: Optimize moves during renovation

(with S. Airiau, L. Galand, J. Lang, S. Toubaline)

- Linear Program (simplicity)$\rightarrow$ Other models are nonconvex!
- Continuous LP solvers struggle$\rightarrow$ Revisit algorithms!
- Model parameters to tune$\rightarrow$ Derivative-free methods!

Complexity and **structure** key to practical efficiency

Thank you for your attention!