Newton-type methods with complexity guarantees for nonconvex data science

Clément W. Royer

Centre Automatique et Systèmes - February 2, 2023

Collaborators

- **Trust-region Newton-CG with strong second-order complexity guarantees for nonconvex optimization**,
  F. E. Curtis, D. P. Robinson, C. W. Royer and S. J. Wright, *SIAM Journal on Optimization*, 2021.
- **Newton-type methods for strict saddle problems**
  F. Goyens and C. W. Royer, in preparation.

The plan

Our interests

- Nonconvex data science tasks.
- Algorithms with complexity guarantees.

The plan

Our interests

- Nonconvex data science tasks.
- Algorithms with complexity guarantees.

Our framework

- Newton-Conjugate Gradient + trust region, revisited.
- Complexity results + numerical relevance.

The plan

Our interests

- Nonconvex data science tasks.
- Algorithms with complexity guarantees.

Our framework

- Newton-Conjugate Gradient + trust region, revisited.
- Complexity results + numerical relevance.

Our latest

- Manifold optimization.
- Strict saddle problems.

Outline

Outline

1. Nonconvex problems and algorithms
   - Nonconvexity in data science
   - Complexity bounds
2. Newton-type framework
3. Extensions

Nonconvex optimization

Nonconvex ?

- Many data science problems are convex: linear classification, logistic regression,...
- **Nonconvex** instances: Deep/shallow neural networks, nonconvex regularization (SCAD,MDP),...

Nonconvex optimization

Nonconvex ?

- Many data science problems are convex: linear classification, logistic regression,...
- **Nonconvex** instances: Deep/shallow neural networks, nonconvex regularization (SCAD,MDP),...

Optimization ?

- Those problems often come with structure.
- In many cases, **global minima** can be characterized (and found) in polynomial time!

Benign nonconvexity

Definition (S. Wright, 2023)

A nonconvex optimization problem has ***benign nonconvexity*** *if useful solutions (even global minima) can be found by optimization methods.*

Benign nonconvexity

Definition (S. Wright, 2023)

A nonconvex optimization problem has ***benign nonconvexity*** *if useful solutions (even global minima) can be found by optimization methods.*

Typical properties

- All local minima are global.
- All saddle points (zero derivative but not local minima) are strict.
- Algorithms can start close to a global minimum.

Examples of benignly nonconvex problems (1/2)

Nonconvex factored matrix problems

- With two matrix variables:

$$\min_{U \in \mathbb{R}^{n \times r}, V \in \mathbb{R}^{m \times r}} f(U V^\top) \quad f \text{ smooth.}$$

$\Rightarrow$ **Nonconvex in U and V** even when f convex, but second-order stationary points typically **global minima** (or close in function value).

Examples of benignly nonconvex problems (1/2)

Nonconvex factored matrix problems

- With two matrix variables:

$$\min_{U \in \mathbb{R}^{n \times r}, V \in \mathbb{R}^{m \times r}} f(UV^\top) \quad f \text{ smooth.}$$

$\Rightarrow$ **Nonconvex in U and V** even when f convex, but second-order stationary points typically **global minima** (or close in function value).

- Similar results holds using multiple matrices!

Examples of benignly nonconvex problems (1/2)

Nonconvex factored matrix problems

- With two matrix variables:

$$\min_{U \in \mathbb{R}^{n \times r}, V \in \mathbb{R}^{m \times r}} f(UV^\top) \quad f \text{ smooth.}$$

$\Rightarrow$ **Nonconvex in U and V** even when f convex, but second-order stationary points typically **global minima** (or close in function value).

- Similar results holds using multiple matrices!

Examples (Ge et al '17,Eftekhari '20)

- Low-rank matrix sensing :

$$f(UV^\top) = \frac{1}{2s} \sum_{i=1}^{s} \left(\langle UV^\top, A_i \rangle - b_i \right)^2, \quad M \in \mathbb{R}^{m \times n}$$

- Deep linear networks : $f(U_1, \dots, U_r) = \frac{1}{2} \| U_r \cdots U_1 A - B \|_F^2$.

Phase retrieval

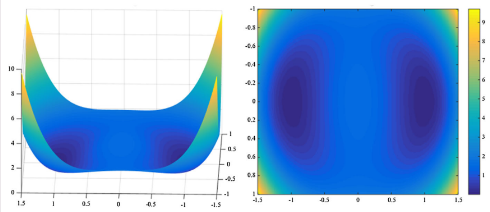
Given $A = [a_i]_{i=1}^m \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, find $x \in \mathbb{C}^n$ such that

$$|a_i^* x| = b_i \quad \forall i = 1, \ldots, n.$$

Nonconvex optimization problem (Sun et al '18)

$$\min_{x \in \mathbb{C}^n} \frac{1}{2m} \sum_{i=1}^m (b_i^2 - |a_i^* x|^2)^2$$

- All local minima are global.
- Saddle points are strict.



Solving these nonconvex instances

What we have

- Classes of structured nonconvex problems.
- Characterization of their solutions using second-order derivatives.

Solving these nonconvex instances

What we have

- Classes of structured nonconvex problems.
- Characterization of their solutions using second-order derivatives.

What we want

- **Efficient** algorithms to reach second-order necessary points;
- Efficiency measured by **complexity**, akin to theoretical CS/**convex** optimization.

Outline

1. Nonconvex problems and algorithms
   - Nonconvexity in data science
   - Complexity bounds
2. Newton-type framework
3. Extensions

General problem and definitions

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

with $f \in \mathcal{C}^2(\mathbb{R}^n)$ bounded below and nonconvex.

General problem and definitions

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

with $f \in \mathcal{C}^2(\mathbb{R}^n)$ bounded below and nonconvex.

Definitions in smooth nonconvex minimization

- *First-order stationary point*: $\|\nabla f(x)\| = 0$;
- *Second-order stationary point*: $\|\nabla f(x)\| = 0, \nabla^2 f(x) \succeq 0$[a].

[a] $A \succeq \beta I \Leftrightarrow \lambda_{\min}(A) \geq \beta$.

General problem and definitions

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

with $f \in \mathcal{C}^2(\mathbb{R}^n)$ bounded below and nonconvex.

Definitions in smooth nonconvex minimization

- *First-order stationary point*: $\|\nabla f(x)\| = 0$;
- *Second-order stationary point*: $\|\nabla f(x)\| = 0, \nabla^2 f(x) \succeq 0$[a].

If x does not satisfy these conditions, $\exists\ d$ such that

1. $d^\top \nabla f(x) < 0$: **gradient-related direction**.
   and/or
2. $d^\top \nabla^2 f(x) d < 0$: **negative curvature direction** $\Rightarrow$ **specific to nonconvex problems**.

[a] $A \succeq \beta I \Leftrightarrow \lambda_{\min}(A) \geq \beta$.

Complexity in nonconvex optimization

Setup: Sequence of points $\{x_k\}$ generated by an algorithm applied to $\min_{x \in \mathbb{R}^n} f(x)$.

Complexity in nonconvex optimization

Setup: Sequence of points $\{x_k\}$ generated by an algorithm applied to $\min_{x \in \mathbb{R}^n} f(x)$.

First-order complexity result

Given $\epsilon \in (0,1)$:

- **Worst-case cost** to obtain an ϵ-point x_K such that $\|\nabla f(x_K)\| \leq \epsilon$.
- Focus: **Dependency on ϵ**.

Setup: Sequence of points $\{x_k\}$ generated by an algorithm applied to $\min_{x \in \mathbb{R}^n} f(x)$.

First-order complexity result

Given $\epsilon \in (0,1)$:

- **Worst-case cost** to obtain an ϵ-point x_K such that $\|\nabla f(x_K)\| \le \epsilon$.
- Focus: **Dependency on** ϵ.

Second-order complexity result

Given $\epsilon, \epsilon_H \in (0,1)$:

- **Worst-case cost** to obtain an (ϵ, ϵ_H)**-point** x_K such that

$$\|\nabla f(x_K)\| \le \epsilon, \qquad \nabla^2 f(x_K) \succeq -\epsilon_H.$$

- Focus: **Dependencies on** ϵ, ϵ_H.

Complexity results

From nonconvex optimization (2006-)

- Cost measure: Number of iterations (but those may be expensive);
- Two types of guarantees:
  1. $\|\nabla f(x)\| \leq \epsilon$;
  2. $\|\nabla f(x)\| \leq \epsilon$ and $\nabla^2 f(x) \succeq -\epsilon_H I$.
- Best methods: Second-order methods, deterministic variations on Newton's iteration involving Hessians.

Complexity results

From nonconvex optimization (2006-)

- Cost measure: Number of iterations (but those may be expensive);
- Two types of guarantees:
  1. $\|\nabla f(x)\| \leq \epsilon$;
  2. $\|\nabla f(x)\| \leq \epsilon$ and $\nabla^2 f(x) \succeq -\epsilon_H I$.
- Best methods: Second-order methods, deterministic variations on Newton's iteration involving Hessians.

Influenced by convex optimization/learning (2016-)

- Cost measure: gradient evaluations+Hessian-vector products.
- Two types of guarantees:
  1. $\|\nabla f(x)\| \leq \epsilon$
  2. $\|\nabla f(x)\| \leq \epsilon$ and $\nabla^2 f(x) \succeq -\epsilon^{1/2} I$.
- Best methods: developed from accelerated gradient, assume knowledge of Lipschitz constants.

Complexity results (2)

Methods with good complexity

- Designed to get good guarantees;
- Sensitive to parameter choices;
- Not necessarily efficient in practice.

Practical methods

- Efficient without convexity;
- Often scalable (e.g. matrix-free);
- No complexity guarantees.

Outline

General problem and definitions

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

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

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

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

Goal: Find approximate stationary points

Given $\epsilon, \epsilon_H \in (0,1)$,

- x is an (ϵ, ϵ_H)-point if

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

- Complexity: Given an algorithm, bound the cost of the method to find an (ϵ, ϵ_H)-point.

Some complexity results

Goal: Find x such that $\|\nabla f(x)\| \leq \epsilon$, $\nabla^2 f(x) \succeq -\epsilon_H I$.

Gradient-based line search/trust region (Cartis et al '12)

- Cost: Iterations, calls to f,∇f,$\nabla^2 f$;
- Order: $\max\{\epsilon^{-2}\epsilon_H^{-1}, \epsilon_H^{-3}\}$;
- Newton steps not used/leveraged.

Some complexity results

Goal: Find x such that $\|\nabla f(x)\| \leq \epsilon$, $\nabla^2 f(x) \succeq -\epsilon_H I$.

Gradient-based line search/trust region (Cartis et al '12)

- Cost: Iterations, calls to $f, \nabla f, \nabla^2 f$;
- Order: $\max\{\epsilon^{-2}\epsilon_H^{-1}, \epsilon_H^{-3}\}$;
- Newton steps not used/leveraged.

Optimal Newton-type methods (Cartis et al '19, Curtis et al '17)

- Cost: Iterations, calls to $f, \nabla f, \nabla^2 f$;
- Bound: $\max\{\epsilon^{-3/2}, \epsilon_H^{-3}\} \Rightarrow \epsilon^{-3/2}$ when $\epsilon_H = \sqrt{\epsilon}$;
- Optimal iteration complexity but expensive Newton steps.

Our goal

Newton-type methods

- Compute a Newton step or use negative curvature;
- Provide **decrease guarantees** (for complexity);
- Use **inexact steps** (for practicality).

Specific features

- Trust region for globalization;
- Conjugate gradient (inexact version).

Trust-region Newton-type method

Inputs: $x_0 \in \mathbb{R}^n$, $\delta_0 > 0$, $\eta > 0$.

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

1. Define $m_k(x_k + s) := \nabla f(x_k)^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} \nabla^2 f(x_k) s$ and compute

$$s_k \in \operatorname*{argmin}_{\substack{s \in \mathbb{R}^n \\ \|s\| \le \delta_k}} m_k(x_k + s) \qquad .$$

2. Compute $\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$.
3. If $\rho_k \ge \eta$, set $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = 2\delta_k$.
4. Otherwise, set $x_{k+1} = x_k$ and $\delta_{k+1} = 0.5\delta_k$.

Trust-region Newton-type method

Inputs: $x_0 \in \mathbb{R}^n$, $\delta_0 > 0$, $\eta > 0$.

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

1. Define $m_k(x_k + s) := \nabla f(x_k)^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} \nabla^2 f(x_k) s$ and compute

$$s_k \in \operatorname*{argmin}_{\substack{s \in \mathbb{R}^n \\ \|s\| \le \delta_k}} m_k(x_k + s) \quad .$$

2. Compute $\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$.
3. If $\rho_k \ge \eta$, set $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = 2\delta_k$.
4. Otherwise, set $x_{k+1} = x_k$ and $\delta_{k+1} = 0.5\delta_k$.

- Standard version: Can get (suboptimal) iteration complexity.

Trust-region Newton-type method

Inputs: $x_0 \in \mathbb{R}^n$, $\delta_0 > 0$, $\eta > 0$, $\epsilon_H \in (0,1)$.

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

1. Define $m_k(x_k + s) := \nabla f(x_k)^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} \nabla^2 f(x_k) s$ and compute
$$s_k \in \operatorname*{argmin}_{\substack{s \in \mathbb{R}^n \\ \|s\| \le \delta_k}} m_k(x_k + s) + \frac{\epsilon_H}{2}\|s\|^2.$$
2. Compute $\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$.
3. If $\rho_k \ge \eta$, set $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = 2\delta_k$.
4. Otherwise, set $x_{k+1} = x_k$ and $\delta_{k+1} = 0.5\delta_k$.

- Standard version: Can get (suboptimal) iteration complexity.
- **Our version: Regularization to improve complexity.**

Analysis of the exact method

Goal: Compute x_k such that $\|\nabla f(x_k)\| \leq \epsilon$ and $\nabla^2 f(x_k) \succeq -\epsilon_H I$.

As long as x_k is not an (ϵ, ϵ_H)-point:

- $m_k(x_k) - m_k(x_k + s_k) \geq \frac{\epsilon_H}{2}\|s_k\|^2$;
- $\delta_k \geq \mathcal{O}(\epsilon_H)$.

Analysis of the exact method

Goal: Compute x_k such that $\|\nabla f(x_k)\| \le \epsilon$ and $\nabla^2 f(x_k) \succeq -\epsilon_H I$.

As long as x_k is not an (ϵ, ϵ_H)-point:

- $m_k(x_k) - m_k(x_k + s_k) \ge \frac{\epsilon_H}{2}\|s_k\|^2$;
- $\delta_k \ge \mathcal{O}(\epsilon_H)$.

For any successful iteration $(x_{k+1} = x_k + s_k)$,

- If $\|s_k\| = \delta_k$,

$$f(x_k) - f(x_{k+1}) \ge \frac{\eta}{2}\epsilon_H \delta_k^2 \ge \mathcal{O}(\epsilon_H^3)$$

- If $\|s_k\| < \delta_k$,

$$f(x_k) - f(x_{k+1}) \ge \mathcal{O}\left(\min\left\{\|\nabla f(x_{k+1})\|^2 \epsilon_H^{-1}, \epsilon_H^3\right\}\right)$$

Complexity of the exact method

Theorem

The trust-region algorithm reaches an (ϵ, ϵ_H)-point in at most

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

successful iterations/calls to $\nabla f / \nabla^2 f$ and

$$\mathcal{O}\left(\log(\epsilon_H^{-1}) \max\left\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\right\}\right) = \tilde{\mathcal{O}}\left(\max\left\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\right\}\right)$$

total iterations/calls to f.

Complexity of the exact method

Theorem

The trust-region algorithm reaches an (ϵ, ϵ_H)-point in at most

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

successful iterations/calls to $\nabla f / \nabla^2 f$ and

$$\mathcal{O}\left(\log(\epsilon_H^{-1}) \max\left\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\right\}\right) = \tilde{\mathcal{O}}\left(\max\left\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\right\}\right)$$

total iterations/calls to f.

- Order for classical method: $\max\left\{\epsilon^{-2}\epsilon_H^{-1}, \epsilon_H^{-3}\right\}$.
- $\epsilon_H = \epsilon^{1/2}$ gives optimal $\mathcal{O}(\epsilon^{-3/2})$ complexity.

Outline

Inexact trust-region Newton-type method

Inputs: $x_0 \in \mathbb{R}^n$, $\delta_0 > 0$, $\zeta > 0$, $\eta > 0$.

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

1. Define $m_k(x_k + s) := \nabla f(x_k)^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} \nabla^2 f(x_k) s$ and compute
$$s_k \approx \operatorname*{argmin}_{\substack{s \in \mathbb{R}^n \\ \|s\| \le \delta_k}} m_k(x_k + s) \qquad .$$
2. Compute $\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$.
3. If $\rho_k \ge \eta$, set $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = 2\delta_k$.
4. Otherwise, set $x_{k+1} = x_k$ and $\delta_{k+1} = 0.5\delta_k$.

Inexact trust-region Newton-type method

Inputs: $x_0 \in \mathbb{R}^n$, $\delta_0 > 0$, $\zeta > 0$, $\eta > 0$.

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

1. Define $m_k(x_k + s) := \nabla f(x_k)^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} \nabla^2 f(x_k) s$ and compute

$$s_k \approx \operatorname*{argmin}_{\substack{s \in \mathbb{R}^n \\ \|s\| \le \delta_k}} m_k(x_k + s) \qquad .$$

2. Compute $\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$.
3. If $\rho_k \ge \eta$, set $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = 2\delta_k$.
4. Otherwise, set $x_{k+1} = x_k$ and $\delta_{k+1} = 0.5\delta_k$.

- Standard version: Solve subproblem via Conjugate Gradient (CG);

Inexact trust-region Newton-type method

Inputs: $x_0 \in \mathbb{R}^n$, $\delta_0 > 0$, $\zeta > 0$, $\eta > 0$, $\epsilon_H \in (0,1)$.

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

1. Define $m_k(x_k + s) := \nabla f(x_k)^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} \nabla^2 f(x_k) s$ and compute

$$s_k \approx \operatorname*{argmin}_{\substack{s \in \mathbb{R}^n \\ \|s\| \le \delta_k}} m_k(x_k + s) + \epsilon_H \|s\|^2.$$

2. Compute $\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$.
3. If $\rho_k \ge \eta$, set $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = 2\delta_k$.
4. Otherwise, set $x_{k+1} = x_k$ and $\delta_{k+1} = 0.5\delta_k$.

- Standard version: Solve subproblem via Conjugate Gradient (CG);
- **Our approach:**
  - Regularization tailored to inexact setting.
  - Extra stopping criteria on CG for complexity.

Linear Conjugate Gradient (CG)

Goal: Solve $Hs = -g$ with H symmetric matrix and $g \in \mathbb{R}^n$.

Linear CG

Init: Set $s_0 = 0_{\mathbb{R}^n},\ r_0 = g,\ p_0 = -g,\ j = 0,\ \xi \geq 0$.

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

- Compute $s_{j+1} = s_j + \frac{\|r_j\|^2}{p_j^{\mathrm{T}} H p_j} p_j$ and $r_{j+1} = Hs_{j+1} + g$.
- Set $p_{j+1} = -r_{j+1} + \frac{\|r_{j+1}\|^2}{\|r_j\|^2} p_j$.
- Set $j = j + 1$; terminate if $\|Hs_j + g\| \leq \xi \|g\|$.

Linear Conjugate Gradient (CG)

Goal: Solve $Hs = -g$ with H symmetric matrix and $g \in \mathbb{R}^n$.

Linear CG

Init: Set $s_0 = 0_{\mathbb{R}^n},\ r_0 = g,\ p_0 = -g,\ j = 0,\ \xi \geq 0.$

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

- Compute $s_{j+1} = s_j + \frac{\|r_j\|^2}{p_j^{\mathrm{T}} H p_j} p_j$ and $r_{j+1} = H s_{j+1} + g$.
- Set $p_{j+1} = -r_{j+1} + \frac{\|r_{j+1}\|^2}{\|r_j\|^2} p_j$.
- Set $j = j + 1$; terminate if $\|H s_j + g\| \leq \xi \|g\|$.

- Only requires $v \mapsto Hv$ ("matrix-free");
- Terminates in at most n iterations **when** $H \succ 0$.

The Steihaug-Toint approach

TR subproblem

$$\min_{s\in\mathbb{R}^n} g^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} H s \quad \text{s.t.} \quad \|s\| \le \delta, \qquad H = H^T.$$

- Apply conjugate gradient (CG) to the linear system $Hs = -g$;
- Stop when residual small enough $\|Hs + g\| \le \zeta \|g\|$ or the boundary is reached;
- For $H \not\succeq 0$: if **negative curvature** is encountered in H, take a negative curvature step towards the boundary.

The Steihaug-Toint approach

TR subproblem

$$\min_{s\in\mathbb{R}^n} g^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} H s \quad \text{s.t.} \quad \|s\| \le \delta, \qquad H = H^T.$$

- Apply conjugate gradient (CG) to the linear system $Hs = -g$;
- Stop when residual small enough $\|Hs + g\| \le \zeta \|g\|$ or the boundary is reached;
- For $H \not\succeq 0$: if **negative curvature** is encountered in H, take a negative curvature step towards the boundary.

Steihaug's approach within TR

- Optimal iteration complexity?
- Cost: Number of Hessian-vector products?

Goal: $\min_{s \in \mathbb{R}^n} g^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}}(H + 2\epsilon_H I)s \quad \text{s.t.} \quad \|s\| \leq \delta.$

Conjugate gradient method with explicit cap

Goal: $\min_{s \in \mathbb{R}^n} g^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} (H + 2\epsilon_H I) s \quad \text{s.t.} \quad \|s\| \le \delta.$

Key differences

Stop after J iterations of CG if one of the following conditions holds:

1. Convergence: $\|(H + 2\epsilon_H I)s + g\| \le \zeta \min\{\|g\|, \epsilon_H \|s\|\}$;
2. Boundary reached;

Conjugate gradient method with explicit cap

Goal: $\min_{s\in\mathbb{R}^n} g^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}}(H + 2\epsilon_H I)s \quad \text{s.t.} \quad \|s\| \leq \delta.$

Key differences

Stop after J iterations of CG if one of the following conditions holds:

1. Convergence: $\|(H + 2\epsilon_H I)s + g\| \leq \zeta \min\{\|g\|, \epsilon_H \|s\|\}$;
2. Boundary reached;
3. Small curvature: A vector u is found such that

$$u^{\mathrm{T}}(H + 2\epsilon_H I)u \leq \epsilon_H \|u\|^2 \quad \Rightarrow \quad u^{\mathrm{T}} H u \leq -\epsilon_H \|u\|^2.$$

4. Explicit iteration cap: $J \leq \hat{J} := \min\{n, \tilde{\mathcal{O}}(\epsilon_H^{-1/2})\}$ iterations
   If $H + 2\epsilon_H I \succeq \epsilon_H I$, convergence (case 1) occurs in less than $\hat{J}$ iterations!

Towards second-order guarantees

CG with explicit cap

- Good steps when converged and $\|\nabla f(x_k)\| \geq \epsilon$;
- Or when negative curvature is detected;
- But **may not converge/miss negative curvature information**!

Towards second-order guarantees

CG with explicit cap

- Good steps when converged and $\|\nabla f(x_k)\| \geq \epsilon$;
- Or when negative curvature is detected;
- But **may not converge/miss negative curvature information**!

Our approach

At iteration x_k,

1. Run CG on the regularized problem first;
2. If the cap is triggered $(\hat{J})$ or $\|\nabla f(x_k)\| \leq \epsilon$ and the convergence criterion is met, call a **minimum eigenvalue oracle** to check whether $\nabla^2 f(x_k) \succeq -\epsilon_H I$.

Minimum eigenvalue oracle (MEO)

Given $H = H^{\mathrm{T}} \in \mathbb{R}^{n \times n}$, $\epsilon_H \in (0,1)$, *and* $\xi \in (0,1)$, output

1. A vector s such that
$$s^{\mathrm{T}} H s \le -\frac{\epsilon_H}{2} \|s\|^2.$$
2. **OR** a certificate that $H \succeq -\epsilon_H I$, valid with probability $1 - \xi$.

Minimum eigenvalue oracle (MEO)

Given $H = H^{\mathrm{T}} \in \mathbb{R}^{n \times n}$, $\epsilon_H \in (0,1)$, *and* $\xi \in (0,1)$, output

1. A vector s such that
$$s^{\mathrm{T}} H s \le -\frac{\epsilon_H}{2} \|s\|^2.$$
2. **OR** a certificate that $H \succeq -\epsilon_H I$, valid with probability $1-\xi$.

An example of MEO

Run CG on $Hs = b$, b uniform on the unit sphere.

- Produces output in $\min\{n, \tilde{\mathcal{O}}(\epsilon_H^{-1/2})\}$ iterations;
- Same order than the cap $\hat{J}$ on CG earlier!

Analysis of the *inexact* method

Goal: Compute x_k such that $\|\nabla f(x_k)\| \le \epsilon$ and $\nabla^2 f(x_k) \succeq -\epsilon_H I$.

For any realization, as long as x_k is not an (ϵ, ϵ_H)-point:

- $m_k(x_k) - m_k(x_k + s_k) \ge \frac{\epsilon_H}{4}\|s_k\|^2$;
- $\delta_k \ge \mathcal{O}(\epsilon_H)$.

For any realization and any successful iteration $(x_{k+1} = x_k + s_k)$,

- If $\|s_k\| = \delta_k$,

$$f(x_k) - f(x_{k+1}) \ge \frac{\eta}{4}\epsilon_H \delta_k^2 \ge \mathcal{O}(\epsilon_H^3)$$

- If $\|s_k\| < \delta_k$,

$$f(x_k) - f(x_{k+1}) \ge \mathcal{O}\left(\min\left\{\|\nabla f(x_{k+1})\|^2 \epsilon_H^{-1}, \epsilon_H^3\right\}\right)$$

Iteration complexity of the inexact method (1/2)

Theorem

The trust-region algorithm reaches an (ϵ, ϵ_H)-point in at most

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

successful iterations/calls to ∇f and

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

total iterations/calls to f with probability $(1-\xi)^{\mathcal{O}\left(\max\left\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\right\}\right)}$.

Iteration complexity of the inexact method (1/2)

Theorem

The trust-region algorithm reaches an (ϵ, ϵ_H)-point in at most

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

successful iterations/calls to ∇f and

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

total iterations/calls to f with probability $(1-\xi)^{\mathcal{O}\left(\max\left\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\right\}\right)}$.

- Same order of complexity than before;
- **With small probability,** the method terminates at x_k where $\|\nabla f(x_k)\| \le \epsilon$ but $\nabla^2 f(x_k) \prec -\epsilon_H I$.

Computational complexity of the inexact method (2/2)

Matrix-free variant: Can we quantify the cost of computing the trust-region step?

Theorem

For any realization of the inexact algorithm, the number of Hessian-vector products used in CG+MEO is

$$\tilde{\mathcal{O}}\left(\min\{n, \epsilon_H^{-1/2}\} \times \max\left\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\right\}\right).$$

Computational complexity of the inexact method (2/2)

Matrix-free variant: Can we quantify the cost of computing the trust-region step?

Theorem

For any realization of the inexact algorithm, the number of Hessian-vector products used in CG+MEO is

$$\tilde{\mathcal{O}}\left(\min\{n, \epsilon_H^{-1/2}\} \times \max\left\{\epsilon^{-2}\epsilon_H, \epsilon_H^{-3}\right\}\right).$$

- Deterministic result (covers early termination).
- $\epsilon_H = \epsilon^{1/2}$ and large n gives best known $\tilde{\mathcal{O}}(\epsilon^{-7/4})$ complexity.

Outline

Setup

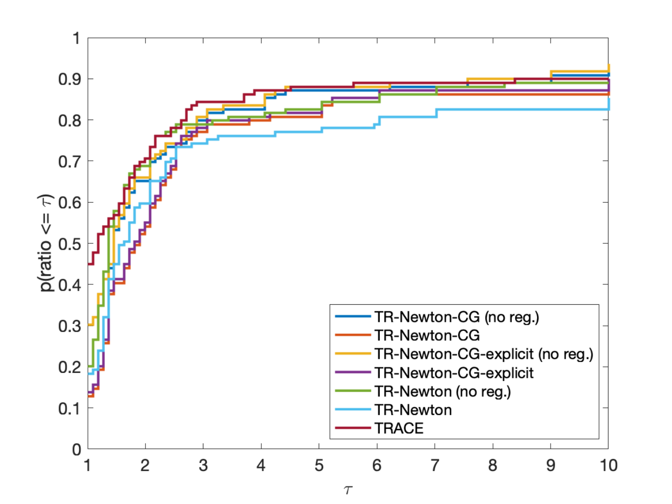
Test problems

- CUTEst smooth unconstrained problems with $n \geq 100$ (109 problems);
- Performance profiles for $\epsilon_H = \epsilon^{1/2}$, $\epsilon = 10^{-5}$.

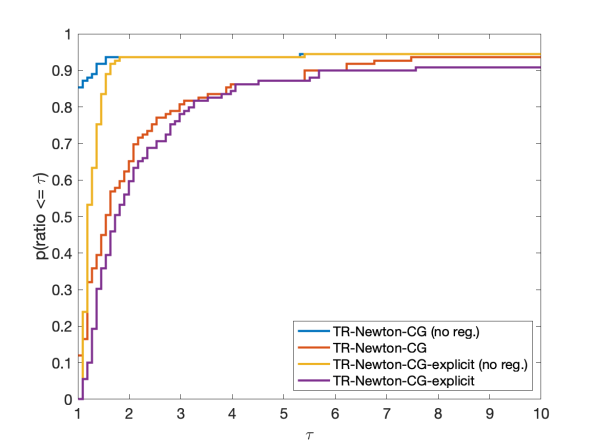
Algorithms (trust-region type)

- TRACE (Curtis, Robinson, Samadi '17);
- TR-Newton (Moré, Sorensen '83);
- TR-Newton-CG (Steihaug '83);
- TR-Newton-CG-explicit (ours with capped CG+MEO).

TR-Newton methods tested with/without regularization.


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
p(ratio <= τ)
1
2
3
4
5
6
7
8
9
10
τ
TR-Newton-CG (no reg.)
TR-Newton-CG
TR-Newton-CG-explicit (no reg.)
TR-Newton-CG-explicit
TR-Newton (no reg.)
TR-Newton
TRACE

Performance profile: Hessian-vector products



The matrix completion example

Matrix completion

$$\min_{X \in \mathbb{R}^{n \times m}, \mathrm{rank}(X) = r} \|\mathcal{P}_\Omega(X - M)\|_F^2, \quad M \in \mathbb{R}^{n \times m},\ \Omega \subset [n] \times [m].$$

The matrix completion example

Matrix completion

$$\min_{X \in \mathbb{R}^{n \times m}, \mathrm{rank}(X)=r} \|\mathcal{P}_\Omega(X - M)\|_F^2, \quad M \in \mathbb{R}^{n \times m}, \ \Omega \subset [n] \times [m].$$

Nonconvex factored reformulation (Burer & Monteiro, '03)

$$\min_{U \in \mathbb{R}^{n \times r}, V \in \mathbb{R}^{m \times r}} \left\|\mathcal{P}_\Omega(U V^\top - M)\right\|_F^2,$$

$\Rightarrow$ **Nonconvex in U and V.**

Numerical illustration

Matrix problem

$$\min_{U,V} \frac{1}{2} \left\| P_\Omega(UV^\top - M) \right\|_F^2,$$

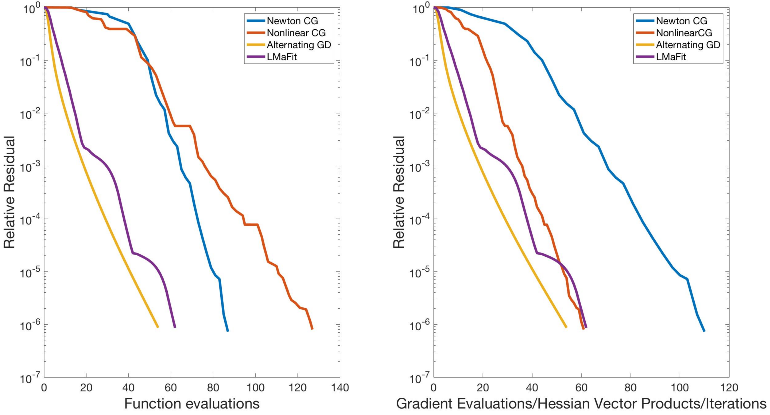
with $M \in \mathbb{R}^{m \times n}$, $U \in \mathbb{R}^{m \times r}$, $V \in \mathbb{R}^{n \times r}$, $|\Omega| \approx \{5\%, 15\%\} \times mn$.

- Synthetic data: $(n, m) = (500, 499)$.

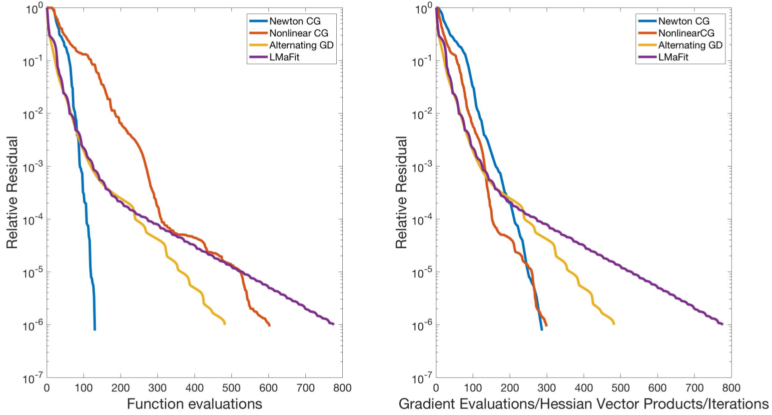
Comparison

- Our Newton+Conjugate Gradient (CG) technique;
- Nonlinear CG (Polak-Ribière);
- Dedicated solvers (Alternating methods):
  - Alternated gradient descent (Tanner and Wei 2016);
  - LMaFit (Wen et al. 2012).

Matrix completion (synthetic data, rank 5)



Matrix completion (synthetic data, rank 15)



In short: Newton-Capped Conjugate Gradient

Our changes to Steihaug's method

- Regularization to get decrease guarantees;
- MEO to get second-order probabilistic results;
- Extra checks in (linear) conjugate gradient.

In short: Newton-Capped Conjugate Gradient

Our changes to Steihaug's method

- Regularization to get decrease guarantees;
- MEO to get second-order probabilistic results;
- Extra checks in (linear) conjugate gradient.

The (typical) cost of complexity

- More iterations of Conjugate Gradient;
- Eigenvalue oracle typically triggered once!

Outline

Motivation

Our problems of interest

- Could involve complex variables (e.g. phase retrieval).
- Matrix completion/factorization: Variables naturally in matrix form.
- Additional constraints: Orthogonal columns, e.g. in phase retrieval.

Motivation

Our problems of interest

- Could involve complex variables (e.g. phase retrieval).
- Matrix completion/factorization: Variables naturally in matrix form.
- Additional constraints: Orthogonal columns, e.g. in phase retrieval.

Manifold optimization

- Solve problems on a Riemannian manifold, i.e. a space that can be mapped to $\mathbb{R}^n$.
- Preserve feasibility throughout.
- Examples:
  1. Vectors : $\mathbb{R}^n$, $\mathbb{C}^n$, $\S^{n-1}$;
  2. Matrices : $\mathbb{R}^{n \times m}$, Grassmann (subspaces), Stiefel (orthogonal matrices).

Manifold optimization

Problem: $\min_{x \in \mathcal{M}} f(x)$, $\mathcal{M}$ Riemannian manifold.

Algorithmic blocks

- Riemannian gradient and Hessian :
  - Counterparts of gradient and Hessian in Euclidean ($\mathbb{R}^n$) setting.
  - Formulas depending on $\mathcal{M}, \nabla f(x), \nabla^2 f(x)$ can be derived by hand or using toolboxes (`Manopt`).

Manifold optimization

Problem: $\min_{x \in \mathcal{M}} f(x)$, $\mathcal{M}$ Riemannian manifold.

Algorithmic blocks

- Riemannian gradient and Hessian :
  - Counterparts of gradient and Hessian in Euclidean ($\mathbb{R}^n$) setting.
  - Formulas depending on $\mathcal{M}, \nabla f(x), \nabla^2 f(x)$ can be derived by hand or using toolboxes (`Manopt`).
- Retraction :
  - Operator that "projects" back onto the manifold.
  - Depends solely on $\mathcal{M}$, formulas available in toolboxes.

Manifold optimization

Problem: $\min_{x \in \mathcal{M}} f(x)$, $\mathcal{M}$ Riemannian manifold.

Algorithmic blocks

- Riemannian gradient and Hessian :
  - Counterparts of gradient and Hessian in Euclidean ($\mathbb{R}^n$) setting.
  - Formulas depending on $\mathcal{M}, \nabla f(x), \nabla^2 f(x)$ can be derived by hand or using toolboxes (`Manopt`).
- Retraction :
  - Operator that "projects" back onto the manifold.
  - Depends solely on $\mathcal{M}$, formulas available in toolboxes.

- With these operations, can adapt most algorithms to the Riemannian setting.
- Complexity guarantees are preserved but now apply to finding Riemannian stationary points.

Illustration: Trust-Region Newton

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

Inputs: $x_0 \in \mathbb{R}^n$, $\delta_0 > 0$, $\eta > 0$.

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

1. Define $m_k(x_k + s) := \nabla f(x_k)^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} \nabla^2 f(x_k) s$ and compute
$$s_k \in \underset{\substack{s \in \mathbb{R}^n \\ \|s\| \le \delta_k}}{\operatorname{argmin}} \, m_k(x_k + s) + \frac{\epsilon_H}{2} \|s\|^2.$$
2. Compute $\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$.
3. If $\rho_k \ge \eta$, set $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = 2\delta_k$.
4. Otherwise, set $x_{k+1} = x_k$ and $\delta_{k+1} = 0.5\delta_k$.

Illustration: Trust-Region Newton

Problem: $\min_{x \in \mathcal{M}} f(x)$, $\mathcal{M}$ Riemannian manifold.

Inputs: $x_0 \in \mathcal{M}$, $\delta_0 > 0$, $\eta > 0$, $\epsilon_H \in (0,1)$.

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

1. **Compute the Riemannian gradient $g_{f,\mathcal{M}}(x_k)$ and Riemannian Hessian $H_{f,\mathcal{M}}(x_k)$.**
2. Define $m_k(x_k + s) := g_{f,\mathcal{M}}(x_k)^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} H_{f,\mathcal{M}}(x_k) s$ and compute
$$s_k \in \operatorname*{argmin}_{\substack{s \in \mathbb{R}^n \\ \|s\| \le \delta_k}} m_k(x_k + s) + \frac{\epsilon_H}{2} \|s\|^2.$$
3. **Define $x_k^{\mathcal{M}}$ as the retraction of $x_k + s_k$ onto $\mathcal{M}$.**
4. Compute $\rho_k = \frac{f(x_k) - f(x_k^{\mathcal{M}})}{m_k(x_k) - m_k(x_k^{\mathcal{M}})}$.
5. If $\rho_k \ge \eta$, set $x_{k+1} = x_k^{\mathcal{M}}$ and $\delta_{k+1} = 2\delta_k$.
6. Otherwise, set $x_{k+1} = x_k$ and $\delta_{k+1} = 0.5\delta_k$.

Outline

Strict saddle functions

What are those?

- Special nonconvex functions;
- Various definitions exist.

Strict saddle functions

What are those?

- Special nonconvex functions;
- Various definitions exist.

An informal definition

Given $\alpha > 0$,$\beta > 0$,$\gamma > 0$, a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is (α, β, γ)-strict saddle if for any $x \in \mathbb{R}^n$, one of these properties holds:

1. $\|\nabla f(x)\| \geq \alpha$;
2. $\nabla^2 f(x) \not\succeq -\beta I$;
3. There exists $x^* \in \operatorname{argmin}_x f(x)$ such that $\|x - x^*\| \leq \gamma$.

Strict saddle functions (2)

Why are strict saddle functions interesting?

- Second-order methods will converge near a global minimum.
- Convergence will be driven by **problem-dependent quantities** (α, β, γ).

Strict saddle functions (2)

Why are strict saddle functions interesting?

- Second-order methods will converge near a global minimum.
- Convergence will be driven by **problem-dependent quantities** (α, β, γ).

Phase retrieval (Sun et al '18)

$$\min_{x \in \mathbb{C}^n} \frac{1}{2m} \sum_{i=1}^{m} (b_i^2 - |a_i^* x|^2)^2.$$

- Manifold optimization problem ($\mathbb{C}^n$).
- Under certain assumptions and for m large enough, the objective is $(c, \frac{c}{n \log(m)}, \frac{c}{n \log(m)})$-strict saddle for some absolute constant $c > 0$.

Our approach

Algorithm

- Newton trust-region (+manifold if needed).
- Assuming (α, β, γ) are known, take different steps at every iteration.
- Promote Newton steps, especially for the third case (close to global minimum).

Our approach

Algorithm

- Newton trust-region (+manifold if needed).
- Assuming (α, β, γ) are known, take different steps at every iteration.
- Promote Newton steps, especially for the third case (close to global minimum).

Complexity (Goyens and R., '23)

The method reaches an (ϵ, ϵ_H)-point in

$$\mathcal{O}\left(\max\left\{\alpha^{-2}, \beta^{-3}\right\}\right) + \log_2\log_2\left[\mathcal{O}\left(\max\{\epsilon^{-1}, \epsilon_H^{-1}\}\right)\right]$$

- Dependencies in ϵ/ϵ_H are "log-log" thanks to Newton.
- Improves over existing results (O'Neill and Wright '23).
- Key: Dependencies in α/β!

Summary

Nonconvex optimization problems

- Tractable formulations ubiquitous in data science.
- Interest in fast algorithms (in a complexity sense).

Our approach

- Revisit popular frameworks in nonlinear optimization (Newton-CG);
- Get optimal complexity + good numerical performance.

Going further

- Handle constraints/matrix variables using manifold optimization.
- Tailor the method to specific structures (strict saddle).

References

- S. Burer and R. D. C. Monteiro, *A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization*, Mathematical Programming, 2003.
- C. Cartis, N. I. M. Gould and Ph. L. Toint, *Complexity bounds for second-order optimality in unconstrained optimization*, Journal of Complexity, 2012.
- C. Cartis, N. I. M. Gould and Ph. L. Toint,*Worst-case evaluation complexity and optimality of second-order methods for nonconvex smooth optimization*, International Congress of Mathematicians (ICM), 2019.
- F. E. Curtis, D. P. Robinson and M. Samadi, *A trust region Algorithm with a worst-case iteration complexity of* $\mathcal{O}\left(\epsilon^{-3/2}\right)$ *for nonconvex optimization*, Mathematical Programming, 2017.
- F. E. Curtis, D. P. Robinson, C. W. Royer and S. J. Wright. *Trust-region Newton-CG with strong second-order complexity guarantees for nonconvex optimization*, SIAM Journal on Optimization, 2021.
- A. Eftekhari, *Training linear neural networks: Non-local convergence and complexity results*, International Conference on Machine Learning, 2020.
- R. Ge, C. Jin and Y. Zheng, *No spurious local minima in nonconvex low rank problems: A unified geometric analysis*, International Conference on Machine Learning, 2017.
- R. Ge and T. Ma, *On the optimization landscape of tensor decompositions*, Advances in Neural Information Processing Systems, 2017.
- F. Goyens and C. W. Royer. *Newton-type methods for strict saddle problems*, in preparation, 2023.
- J. J. Moré and D. C. Sorensen, *Computing a trust-region step*, SIAM Journal on Scientific Computing, 1983.
- M. O'Neill and S. J. Wright. *A line-search descent algorithm for strict saddle functions with complexity guarantees*, Journal of Machine Learning Research, 2023.
- T. Steihaug. *The conjugate gradient method and trust regions in large scale optimization*, SIAM Journal on Numerical Analysis, 1983.
- J. Sun, Q. Qu and J. Wright. *A geometric analysis of phase retrieval*, Found. Comput. Math., 2018.

References

- S. Burer and R. D. C. Monteiro, *A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization*, Mathematical Programming, 2003.
- C. Cartis, N. I. M. Gould and Ph. L. Toint, *Complexity bounds for second-order optimality in unconstrained optimization*, Journal of Complexity, 2012.
- C. Cartis, N. I. M. Gould and Ph. L. Toint,*Worst-case evaluation complexity and optimality of second-order methods for nonconvex smooth optimization*, International Congress of Mathematicians (ICM), 2019.
- F. E. Curtis, D. P. Robinson and M. Samadi, *A trust region Algorithm with a worst-case iteration complexity of* $O\left(\epsilon^{-3/2}\right)$ *for nonconvex optimization*, Mathematical Programming, 2017.
- F. E. Curtis, D. P. Robinson, C. W. Royer and S. J. Wright. *Trust-region Newton-CG with strong second-order complexity guarantees for nonconvex optimization*, SIAM Journal on Optimization, 2021.
- A. Eftekhari, *Training linear neural networks: Non-local convergence and complexity results*, International Conference on Machine Learning, 2020.
- R. Ge, C. Jin and Y. Zheng, *No spurious local minima in nonconvex low rank problems: A unified geometric analysis*, International Conference on Machine Learning, 2017.
- R. Ge and T. Ma, *On the optimization landscape of tensor decompositions*, Advances in Neural Information Processing Systems, 2017.
- F. Goyens and C. W. Royer. *Newton-type methods for strict saddle problems*, in preparation, 2023.
- J. J. Moré and D. C. Sorensen, *Computing a trust-region step*, SIAM Journal on Scientific Computing, 1983.
- M. O'Neill and S. J. Wright. *A line-search descent algorithm for strict saddle functions with complexity guarantees*, Journal of Machine Learning Research, 2023.
- T. Steihaug. *The conjugate gradient method and trust regions in large scale optimization*, SIAM Journal on Numerical Analysis, 1983.
- J. Sun, Q. Qu and J. Wright. *A geometric analysis of phase retrieval*, Found. Comput. Math., 2018.

Thank you!

clement.royer@lamsade.dauphine.fr