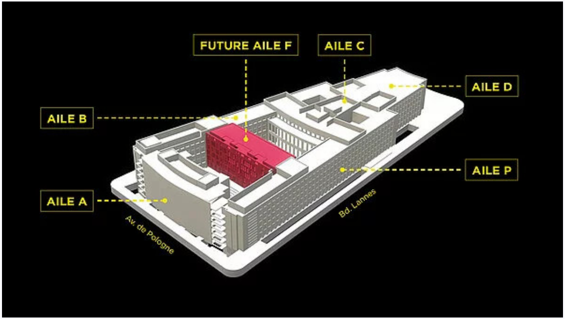
Random subspaces approaches in derivative-free optimization

Clément W. Royer (Université Paris Dauphine-PSL)

Journées Franciliennes de Recherche Opérationnelle

November 26, 2024

Motivation: Dauphine's *Nouveau Campus*



- New wing in construction⇒ 2025.
- Others renovated in order: B, P, C+D, A.
- Expected year of completion: 2028.

Our task: Allocate office space during the renovation process.

Motivation: Dauphine's *Nouveau Campus* ('ed)

Our model for the Dauphine problem

- Huge integer LP, solved via Gurobi.
- ~ 30 hyperparameters defining the model (for now).
- Parallel runs on the department server.

Sub-task: Optimize hyperparameters.

Motivation: Dauphine's *Nouveau Campus* ('ed)

Our model for the Dauphine problem

- Huge integer LP, solved via Gurobi.
- $\sim$ 30 hyperparameters defining the model (for now).
- Parallel runs on the department server.

Sub-task: Optimize hyperparameters.

Problem challenges

- Cannot differentiate (easily) within Gurobi
  $\Rightarrow$ Derivative-free/Blackbox algorithms!

Motivation: Dauphine's *Nouveau Campus* ('ed)

Our model for the Dauphine problem

- Huge integer LP, solved via Gurobi.
- $\sim$ 30 hyperparameters defining the model (for now).
- Parallel runs on the department server.

Sub-task: Optimize hyperparameters.

Problem challenges

- Cannot differentiate (easily) within Gurobi
  $\Rightarrow$ Derivative-free/Blackbox algorithms!
- Solving time depends on hyperparameters
  (3-48 hours to find a feasible point!)
  $\Rightarrow$ Expensive evaluations.

Motivation: Dauphine's *Nouveau Campus* ('ed)

Our model for the Dauphine problem

- Huge integer LP, solved via Gurobi.
- ~ 30 hyperparameters defining the model (for now).
- Parallel runs on the department server.

Sub-task: Optimize hyperparameters.

Problem challenges

- Cannot differentiate (easily) within Gurobi
  $\Rightarrow$ Derivative-free/Blackbox algorithms!
- Solving time depends on hyperparameters
  (3-48 hours to find a feasible point!)
  $\Rightarrow$ Expensive evaluations.
- Feedback on the model $\Rightarrow$ More hyperparameters!
  $\Rightarrow$ Need algorithms that scale.

This talk

Subspace methods

- Help reduce the cost of blackbox optimization.
- Theory: Dimensionality reduction/Sketching.
- Practice: Easy to implement.

Research questions

- How do you use subspaces in an algorithm?
- Can this work? If so, why?

Today

- Focus on direct search.
- Results apply to other settings (model-based).

Roadmap

1. Direct-search algorithm
2. Reduced subspace approach
3. Subspace dimensions

Roadmap

1. Direct-search algorithm
2. Reduced subspace approach
3. Subspace dimensions

Setup

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

Assumptions

- f bounded below;
- f continuously differentiable (for analysis).

Blackbox optimization

- Derivatives unavailable for algorithmic use.
- Only access to values of f.

A (simplified) direct-search framework

Similar to: Local search, (1+1)-ES, ...

Inputs: $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\delta_0 > 0$.
Iteration k**:** Given $(\boldsymbol{x}_k, \delta_k)$,

- Choose a set $\mathcal{D}_k \subset \mathbb{R}^n$ of m vectors.

A (simplified) direct-search framework

Similar to: Local search, (1+1)-ES, ...

Inputs: $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\delta_0 > 0$.
Iteration k**:** Given $(\boldsymbol{x}_k, \delta_k)$,

- Choose a set $\mathcal{D}_k \subset \mathbb{R}^n$ of m vectors.
- If $\exists\ \boldsymbol{d}_k \in \mathcal{D}_k$ such that

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

set $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \delta_k \boldsymbol{d}_k$, $\delta_{k+1} := 2\delta_k$.

A (simplified) direct-search framework

Similar to: Local search, (1+1)-ES, ...

Inputs: $\mathbf{x}_0 \in \mathbb{R}^n$, $\delta_0 > 0$.

Iteration k**:** Given $(\mathbf{x}_k, \delta_k)$,

- Choose a set $\mathcal{D}_k \subset \mathbb{R}^n$ of m vectors.
- If $\exists\ \mathbf{d}_k \in \mathcal{D}_k$ such that

$$f(\mathbf{x}_k + \delta_k\, \mathbf{d}_k) < f(\mathbf{x}_k) - \delta_k^2 \|\mathbf{d}_k\|^2$$

  set $\mathbf{x}_{k+1} := \mathbf{x}_k + \delta_k \mathbf{d}_k$, $\delta_{k+1} := 2\delta_k$.
- Otherwise, set $\mathbf{x}_{k+1} := \mathbf{x}_k$, $\delta_{k+1} := \delta_k/2$.

A (simplified) direct-search framework

Similar to: Local search, (1+1)-ES, ...

Inputs: $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\delta_0 > 0$.
Iteration k**:** Given $(\boldsymbol{x}_k, \delta_k)$,

- Choose a set $\mathcal{D}_k \subset \mathbb{R}^n$ of m vectors.
- If $\exists\ \boldsymbol{d}_k \in \mathcal{D}_k$ such that

$$f(\boldsymbol{x}_k + \delta_k \boldsymbol{d}_k) < f(\boldsymbol{x}_k) - \delta_k^2 \|\boldsymbol{d}_k\|^2$$

  set $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \delta_k \boldsymbol{d}_k$, $\delta_{k+1} := 2\delta_k$.
- Otherwise, set $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k$, $\delta_{k+1} := \delta_k/2$.

A (simplified) direct-search framework

Similar to: Local search, (1+1)-ES, ...

Inputs: $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\delta_0 > 0$.

Iteration k: Given $(\boldsymbol{x}_k, \delta_k)$,

- **Choose a set $\mathcal{D}_k \subset \mathbb{R}^n$ of m vectors.**
- If $\exists\ \boldsymbol{d}_k \in \mathcal{D}_k$ such that

$$f(\boldsymbol{x}_k + \delta_k \boldsymbol{d}_k) < f(\boldsymbol{x}_k) - \delta_k^2 \|\boldsymbol{d}_k\|^2$$

  set $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \delta_k \boldsymbol{d}_k$, $\delta_{k+1} := 2\delta_k$.
- Otherwise, set $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k$, $\delta_{k+1} := \delta_k/2$.

Which vectors should we use?

Choosing $\mathcal{D}_k$

A measure of set quality

The set $\mathcal{D}_k$ is called κ-descent for f at $\boldsymbol{x}_k$ if

$$\max_{\boldsymbol{d} \in \mathcal{D}_k} \frac{-\boldsymbol{d}^{\mathrm{T}} \nabla f(\boldsymbol{x}_k)}{\|\boldsymbol{d}\| \|\nabla f(\boldsymbol{x}_k)\|} \geq \kappa \in (0,1].$$

Choosing $\mathcal{D}_k$

A measure of set quality

The set $\mathcal{D}_k$ is called κ-descent for f at $\boldsymbol{x}_k$ if

$$\max_{\boldsymbol{d}\in\mathcal{D}_k} \frac{-\boldsymbol{d}^{\mathrm{T}}\nabla f(\boldsymbol{x}_k)}{\|\boldsymbol{d}\|\|\nabla f(\boldsymbol{x}_k)\|} \geq \kappa \in (0,1].$$

- Guaranteed when $\mathcal{D}_k$ is a Positive Spanning Set (PSS);
- $\mathcal{D}_k$ PSS $\Rightarrow |\mathcal{D}_k| \geq n+1$;
- Ex) $\mathcal{D}_\oplus := [\boldsymbol{I}_n \;\; -\boldsymbol{I}_n]$ is always $\frac{1}{\sqrt{n}}$-descent.

Complexity of deterministic direct search

Assumption: For every k, $\mathcal{D}_k$ is κ-descent and contains m unit directions.

Theorem (Vicente '12)

Let $\epsilon \in (0,1)$ and N_ϵ be the number of function evaluations needed to reach $\boldsymbol{x}_k$ such that $\|\nabla f(\boldsymbol{x}_k)\| \le \epsilon$. Then,

$$N_\epsilon \ \le \ \mathcal{O}\left(m\,\kappa^{-2}\,\epsilon^{-2}\right).$$

Complexity of deterministic direct search

Assumption: For every k, $\mathcal{D}_k$ is κ-descent and contains m unit directions.

Theorem (Vicente '12)

Let $\epsilon \in (0,1)$ and N_ϵ be the number of function evaluations needed to reach $\boldsymbol{x}_k$ such that $\|\nabla f(\boldsymbol{x}_k)\| \leq \epsilon$. Then,

$$N_\epsilon \ \leq \ \mathcal{O}\left(m\,\kappa^{-2}\,\epsilon^{-2}\right).$$

- Unit norm can be replaced by bounded norm.
- Choosing $\mathcal{D}_k = \mathcal{D}_\oplus$, one has $\kappa = \frac{1}{\sqrt{n}}$, $m = 2n$, and the bound becomes

$$N_\epsilon \ \leq \ \mathcal{O}\left(n^2\,\epsilon^{-2}\right).$$

$\Rightarrow$**Best possible dependency** w.r.t. n for deterministic direct-search algorithms.

Randomizing direct search

Classical direct search

- Set $\mathcal{D}_k \subset \mathbb{R}^n$, $|\mathcal{D}_k| = m$, $\text{cm}(\mathcal{D}_k) \geq \kappa$;
- Complexity:

$$\mathcal{O}(m\kappa^{-2}\,\epsilon^{-2}).$$

  - m depends on n ($m \geq n+1$).
  - κ depends on n (approximate $\nabla f(\boldsymbol{x}_k) \in \mathbb{R}^n$).

Randomizing direct search

Classical direct search

- Set $\mathcal{D}_k \subset \mathbb{R}^n$, $|\mathcal{D}_k| = m$, $\text{cm}(\mathcal{D}_k) \geq \kappa$;
- Complexity:

$$\mathcal{O}(m\kappa^{-2}\,\epsilon^{-2}).$$

  - m depends on n ($m \geq n+1$).
  - κ depends on n (approximate $\nabla f(\boldsymbol{x}_k) \in \mathbb{R}^n$).

My original thought

- Generate directions in random subspaces of $\mathbb{R}^n$;
- Use results from dimensionality reduction;
- Remove all dependencies on n!

Randomizing direct search

Classical direct search

- Set $\mathcal{D}_k \subset \mathbb{R}^n$, $|\mathcal{D}_k| = m$, $\mathrm{cm}(\mathcal{D}_k) \geq \kappa$;
- Complexity:

$$\mathcal{O}(m\kappa^{-2}\,\epsilon^{-2}).$$

  - m depends on n ($m \geq n+1$).
  - κ depends on n (approximate $\nabla f(\boldsymbol{x}_k) \in \mathbb{R}^n$).

My original thought

- Generate directions in random subspaces of $\mathbb{R}^n$;
- Use results from dimensionality reduction;
- Remove all dependencies on n!

Spoiler alert: You can only *reduce* the dependency on n.

What can you do?

Our approach

- Consider a random subspace of dimension $r \leq n$;
- Use a PSS to approximate the projected gradient in the subspace;
- Guarantee sufficient gradient information **in probability**.

What it brings us

- Use random directions.
- Possibly less than n.
- Possibly **unbounded**.

Not the only game in town (1/2)

Probabilistic descent (Gratton et al '15)

- Use directions $[\boldsymbol{d} \;\; -\boldsymbol{d}]$ with $\boldsymbol{d} \sim \mathcal{U}(\mathbb{S}^{n-1})$.
- Complexity improves from $\mathcal{O}(n^2\epsilon^{-2})$ to $\mathcal{O}(n\epsilon^{-2})$ $(m = 2)$.
- Limited to one distribution.

Not the only game in town (1/2)

Probabilistic descent (Gratton et al '15)

- Use directions $[\boldsymbol{d} \ -\boldsymbol{d}]$ with $\boldsymbol{d} \sim \mathcal{U}(\mathbb{S}^{n-1})$.
- Complexity improves from $\mathcal{O}(n^2\epsilon^{-2})$ to $\mathcal{O}(n\epsilon^{-2})$ $(m = 2)$.
- Limited to one distribution.

Gaussian smoothing approach: Draw $\boldsymbol{d} \sim \mathcal{N}(0, \boldsymbol{I})$ and use

$$\frac{f(\boldsymbol{x}+\delta\boldsymbol{d}) - f(\boldsymbol{x})}{\delta}\boldsymbol{d} \quad \text{or} \quad \frac{f(\boldsymbol{x}+\delta\boldsymbol{d}) - f(\boldsymbol{x}-\delta\boldsymbol{d})}{\delta}\boldsymbol{d}.$$

Random gradient-free method (Nesterov and Spokoiny 2017),
Stochastic three-point method (Bergou et al, 2020).

- Also achieve $\mathcal{O}(n\epsilon^{-2})$ bound.
- Use one-dimensional subspace based on Gaussian vectors.
- Use fixed or decreasing stepsizes.

Not the only game in town (2/2)

Zeroth-order (Kozak et al '21, '22)

- Estimate directional derivatives directly.
- Use orthogonal random directions $\boldsymbol{Q} \in \mathbb{R}^{n \times r}$, $\boldsymbol{Q}^{\mathrm{T}} \boldsymbol{Q} = \boldsymbol{I}$.
- Complexity results for convex/PL functions.

Not the only game in town (2/2)

Zeroth-order (Kozak et al '21, '22)

- Estimate directional derivatives directly.
- Use orthogonal random directions $\boldsymbol{Q} \in \mathbb{R}^{n \times r}$, $\boldsymbol{Q}^{\mathrm{T}} \boldsymbol{Q} = \boldsymbol{I}$.
- Complexity results for convex/PL functions.

Our approach

- General, subspace-based framework.
- Inspiration: Model-based methods
  (Cartis and Roberts '23, Dzahini and Wild '22a).

Outline

Algorithm

Inputs: $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\delta_0 > 0$.

Iteration k: Given $(\boldsymbol{x}_k, \delta_k)$,

- Choose $\boldsymbol{P}_k \in \mathbb{R}^{r \times n}$ **at random**.
- Choose $\mathcal{D}_k \subset \mathbb{R}^r$ having m vectors.
- If $\exists\ \boldsymbol{d}_k \in \mathcal{D}_k$ such that

$$f(\boldsymbol{x}_k + \delta_k \boldsymbol{P}_k^{\mathrm{T}} \boldsymbol{d}_k) < f(\boldsymbol{x}_k) - \delta_k^2 \|\boldsymbol{P}_k^{\mathrm{T}} \boldsymbol{d}_k\|^2,$$

  set $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \delta_k \boldsymbol{P}_k^{\mathrm{T}} \boldsymbol{d}_k$, $\delta_{k+1} := 2\delta_k$.
- Otherwise, set $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k$, $\delta_{k+1} := \delta_k / 2$.

New polling sets

$$\left\{ \boldsymbol{P}_k^{\mathrm{T}} \boldsymbol{d} \;\middle|\; \boldsymbol{d} \in \mathcal{D}_k \right\} \subset \mathbb{R}^n.$$

- $\boldsymbol{P}_k \in \mathbb{R}^{r \times n}$: Maps onto r-dimensional subspace;
- $\mathcal{D}_k$: Direction set in $\mathbb{R}^r$.

What do we want?

- Preserve information while applying $\boldsymbol{P}_k / \boldsymbol{P}_k^{\mathrm{T}}$.
- Approximate $-\boldsymbol{P}_k \nabla f(\boldsymbol{x}_k)$ using $\mathcal{D}_k$.

Probabilistic properties for P_k

P_k is $(\eta, \sigma, P_{\max})$-well aligned for (f, x_k) if

$$\begin{cases} \|P_k \nabla f(x_k)\| & \geq & \eta \|\nabla f(x_k)\|, \\ \sigma_{\min}(P_k) & \geq & \sigma, \\ \sigma_{\max}(P_k) & \leq & P_{\max}. \end{cases}$$

Probabilistic properties for $\boldsymbol{P}_k$

$\boldsymbol{P}_k$ is $(\eta, \sigma, P_{\max})$-well aligned for $(f, \boldsymbol{x}_k)$ if

$$\left\{ \begin{array}{rcl} \|\boldsymbol{P}_k \nabla f(\boldsymbol{x}_k)\| & \geq & \eta \|\nabla f(\boldsymbol{x}_k)\|, \\ \sigma_{\min}(\boldsymbol{P}_k) & \geq & \sigma, \\ \sigma_{\max}(\boldsymbol{P}_k) & \leq & P_{\max}. \end{array} \right.$$

Ex) $\boldsymbol{P}_k = \boldsymbol{I}_n \in \mathbb{R}^{n \times n}$ *is* $(1,1,1)$*-well aligned.*

Probabilistic properties for $\boldsymbol{P}_k$

$\boldsymbol{P}_k$ is $(\eta, \sigma, P_{\max})$-well aligned for $(f, \boldsymbol{x}_k)$ if

$$\left\{ \begin{array}{rcl} \|\boldsymbol{P}_k \nabla f(\boldsymbol{x}_k)\| & \geq & \eta \|\nabla f(\boldsymbol{x}_k)\|, \\ \sigma_{\min}(\boldsymbol{P}_k) & \geq & \sigma, \\ \sigma_{\max}(\boldsymbol{P}_k) & \leq & P_{\max}. \end{array} \right.$$

Ex) $\boldsymbol{P}_k = \boldsymbol{I}_n \in \mathbb{R}^{n \times n}$ *is* $(1,1,1)$*-well aligned.*

Probabilistic version

$\{\boldsymbol{P}_k\}$ is $(q, \eta, \sigma, P_{\max})$-well aligned if:

$$\begin{array}{rcl} \mathbb{P}\left(\boldsymbol{P}_0 \ (q, \eta, \sigma, P_{\max})\text{-well aligned }\right) & \geq & q \\ \forall k \geq 1, \quad \mathbb{P}\left((q, \eta, \sigma, P_{\max})\text{-well aligned } \mid \boldsymbol{P}_0, \mathcal{D}_0, \dots, \boldsymbol{P}_{k-1}, \mathcal{D}_{k-1}\right) & \geq & q, \end{array}$$

Probabilistic properties for $\mathcal{D}_k$

Deterministic descent

The set $\mathcal{D}_k$ is $(\kappa, d_{\max})$-descent for $(f, \boldsymbol{x}_k)$ if

$$\begin{cases} \max_{\boldsymbol{d} \in \mathcal{D}_k} \frac{-\boldsymbol{d}^{\mathrm{T}} \boldsymbol{P}_k \nabla f(\boldsymbol{x}_k)}{\|\boldsymbol{d}\| \|\boldsymbol{P}_k \nabla f(\boldsymbol{x}_k)\|} \geq \kappa, \\ \forall \boldsymbol{d} \in \mathcal{D}_k, \quad d_{\max}^{-1} \leq \|\boldsymbol{d}\| \leq d_{\max}. \end{cases}$$

Probabilistic properties for $\mathcal{D}_k$

Deterministic descent

The set $\mathcal{D}_k$ is $(\kappa, d_{\max})$-descent for $(f, \boldsymbol{x}_k)$ if

$$\begin{cases} \max_{\boldsymbol{d} \in \mathcal{D}_k} \frac{-\boldsymbol{d}^{\mathrm{T}} \boldsymbol{P}_k \nabla f(\boldsymbol{x}_k)}{\|\boldsymbol{d}\| \|\boldsymbol{P}_k \nabla f(\boldsymbol{x}_k)\|} \geq \kappa, \\ \forall \boldsymbol{d} \in \mathcal{D}_k, \quad d_{\max}^{-1} \leq \|\boldsymbol{d}\| \leq d_{\max}. \end{cases}$$

Ex) $D_\oplus = \{\boldsymbol{e}_1, \dots, \boldsymbol{e}_n, -\boldsymbol{e}_1, \dots, -\boldsymbol{e}_n\}$ *is* $(\frac{1}{\sqrt{n}}, 1)$*-descent.*

Probabilistic properties for $\mathcal{D}_k$

Deterministic descent

The set $\mathcal{D}_k$ is $(\kappa, d_{\max})$-descent for $(f, \boldsymbol{x}_k)$ if

$$\begin{cases} \max_{\boldsymbol{d}\in\mathcal{D}_k} \frac{-\boldsymbol{d}^{\mathrm{T}}\boldsymbol{P}_k\nabla f(\boldsymbol{x}_k)}{\|\boldsymbol{d}\|\|\boldsymbol{P}_k\nabla f(\boldsymbol{x}_k)\|} \geq \kappa, \\ \forall \boldsymbol{d} \in \mathcal{D}_k, \quad d_{\max}^{-1} \leq \|\boldsymbol{d}\| \leq d_{\max}. \end{cases}$$

Ex) $D_{\oplus} = \{\boldsymbol{e}_1, \dots, \boldsymbol{e}_n, -\boldsymbol{e}_1, \dots, -\boldsymbol{e}_n\}$ *is* $(\frac{1}{\sqrt{n}}, 1)$*-descent.*

Probabilistic descent sets

$\{\mathcal{D}_k\}$ is $(p, \kappa, d_{\max})$-descent if:

$$\mathbb{P}\left(\mathcal{D}_0 \ (\kappa, d_{\max})\text{-descent} \ \middle| \ \boldsymbol{P}_0\right) \geq p$$

$$\forall k \geq 1, \quad \mathbb{P}\left(\mathcal{D}_k \ (\kappa, d_{\max})\text{-descent} \ \middle| \ \boldsymbol{P}_0, \mathcal{D}_0, \dots, \boldsymbol{P}_{k-1}, \mathcal{D}_{k-1}, \boldsymbol{P}_k\right) \geq p,$$

Complexity analysis

Theorem (Roberts, R. '23)

Assume:

- $\{\mathcal{D}_k\}$ $(p, \kappa, d_{\max})$-descent, $|\mathcal{D}_k| = m$;
- $\{\boldsymbol{P}_k\}$ $(q, \eta, \sigma, P_{\max})$-well aligned, $pq > \frac{1}{2}$.

Let N_ϵ the number of function evaluations needed to have $\|\nabla f(\boldsymbol{x}_k)\| \le \epsilon$.

$$\mathbb{P}\left(N_\epsilon \le \mathcal{O}\left(\frac{m\phi\epsilon^{-2}}{2pq-1}\right)\right) \ge 1 - \exp\left(-\mathcal{O}\left(\frac{2pq-1}{pq}\phi\epsilon^{-2}\right)\right).$$

where $\phi = d_{\max}^8 \kappa^{-2} \eta^{-2} \sigma^{-2} P_{\max}^4$.

Complexity analysis

Theorem (Roberts, R. '23)

Assume:

- $\{\mathcal{D}_k\}$ $(p, \kappa, d_{\max})$-descent, $|\mathcal{D}_k| = m$;
- $\{\boldsymbol{P}_k\}$ $(q, \eta, \sigma, P_{\max})$-well aligned, $pq > \frac{1}{2}$.

Let N_ϵ the number of function evaluations needed to have $\|\nabla f(\boldsymbol{x}_k)\| \le \epsilon$.

$$\mathbb{P}\left(N_\epsilon \le \mathcal{O}\left(\frac{m\phi\epsilon^{-2}}{2pq-1}\right)\right) \ge 1 - \exp\left(-\mathcal{O}\left(\frac{2pq-1}{pq}\phi\epsilon^{-2}\right)\right).$$

where $\phi = d_{\max}^8 \kappa^{-2} \eta^{-2} \sigma^{-2} P_{\max}^4$.

How does this bound depend on n?
How can we choose $\mathcal{D}_k$ and $\boldsymbol{P}_k$?

Choosing directions ($\mathcal{D}_k$) and subspaces ($\boldsymbol{P}_k$)

- Deterministic
  - $\mathcal{D}_k = [\boldsymbol{I}_n \ -\boldsymbol{I}_n]$ ($m = 2n$)
  - $\boldsymbol{P}_k = \boldsymbol{I}_n$ (no subspace).

Choosing directions ($\mathcal{D}_k$) and subspaces ($\boldsymbol{P}_k$)

- Deterministic
  - $\mathcal{D}_k = [\boldsymbol{I}_n \ -\boldsymbol{I}_n]$ $(m = 2n)$
  - $\boldsymbol{P}_k = \boldsymbol{I}_n$ (no subspace).
- (Random) Orthogonal
  - $\mathcal{D}_k = [\boldsymbol{I}_r \ -\boldsymbol{I}_r]$ $(m = 2r)$
  - $\boldsymbol{P}_k \in \mathbb{R}^{r \times n}$, $\boldsymbol{P}_k \boldsymbol{P}_k^{\mathrm{T}} = \boldsymbol{I}_r$.
  - Known properties on $\boldsymbol{P}_k$ (Kozak et al '21).

Choosing directions ($\mathcal{D}_k$) and subspaces ($\boldsymbol{P}_k$)

- Deterministic
  - $\mathcal{D}_k = [\boldsymbol{I}_n \ -\boldsymbol{I}_n]$ $(m = 2n)$
  - $\boldsymbol{P}_k = \boldsymbol{I}_n$ (no subspace).
- (Random) Orthogonal
  - $\mathcal{D}_k = [\boldsymbol{I}_r \ -\boldsymbol{I}_r]$ $(m = 2r)$
  - $\boldsymbol{P}_k \in \mathbb{R}^{r \times n}$, $\boldsymbol{P}_k \boldsymbol{P}_k^{\mathrm{T}} = \boldsymbol{I}_r$.
  - Known properties on $\boldsymbol{P}_k$ (Kozak et al '21).
- (Random) Gaussian
  - $\mathcal{D}_k = [\boldsymbol{I}_r \ -\boldsymbol{I}_r]$ $(m = 2r)$
  - $\boldsymbol{P}_k \in \mathbb{R}^{r \times n}$, $[\boldsymbol{P}_k]_{i,j} \sim \mathcal{N}(0, \frac{1}{r})$.
  - Known guarantees on singular values of $\boldsymbol{P}_k$ (2010s).

Choosing directions ($\mathcal{D}_k$) and subspaces ($\boldsymbol{P}_k$)

- Deterministic
  - $\mathcal{D}_k = [\boldsymbol{I}_n \ -\boldsymbol{I}_n]$ $(m = 2n)$
  - $\boldsymbol{P}_k = \boldsymbol{I}_n$ (no subspace).
- (Random) Orthogonal
  - $\mathcal{D}_k = [\boldsymbol{I}_r \ -\boldsymbol{I}_r]$ $(m = 2r)$
  - $\boldsymbol{P}_k \in \mathbb{R}^{r \times n}$, $\boldsymbol{P}_k \boldsymbol{P}_k^{\mathrm{T}} = \boldsymbol{I}_r$.
  - Known properties on $\boldsymbol{P}_k$ (Kozak et al '21).
- (Random) Gaussian
  - $\mathcal{D}_k = [\boldsymbol{I}_r \ -\boldsymbol{I}_r]$ $(m = 2r)$
  - $\boldsymbol{P}_k \in \mathbb{R}^{r \times n}$, $[\boldsymbol{P}_k]_{i,j} \sim \mathcal{N}(0, \frac{1}{r})$.
  - Known guarantees on singular values of $\boldsymbol{P}_k$ (2010s).
- (Random) Hashing
  - $\mathcal{D}_k = [\boldsymbol{I}_r \ -\boldsymbol{I}_r]$ $(m = 2r)$
  - $\boldsymbol{P}_k \in \{\pm \frac{1}{\sqrt{s}}, 0\}^{r \times n}$, s nonzero per columns.
  - New theory motivated by our work (Dzahini, Wild '22)

Analysis in a nutshell

$\boldsymbol{P}_k$	Evals/it	Complexity
Identity	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$
Gaussian	$\mathcal{O}(r)$	$\mathcal{O}(n)$
Orthogonal	$\mathcal{O}(r)$	$\mathcal{O}(n)$
Hashing	$\mathcal{O}(r)$	$\mathcal{O}(r^2\, n)$.

$\boldsymbol{P}_k$	Evals/it	Complexity
Identity	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$
Gaussian	$\mathcal{O}(r)$	$\mathcal{O}(n)$
Orthogonal	$\mathcal{O}(r)$	$\mathcal{O}(n)$
Hashing	$\mathcal{O}(r)$	$\mathcal{O}(r^2\, n)$.

Conclusions

- Can compute steps in r-dim. subspaces, $r = \mathcal{O}(1)$.
- Effectively less evaluations per iteration.
- Complexity: $\mathcal{O}(n^2) \Rightarrow \mathcal{O}(n)$!

Experiments in large dimensions

Benchmark:

- Medium-scale test set (90 CUTEst problems of dimension ≈ 100);
- Large-scale test set (28 CUTEst problems of dimension ≈ 1000).

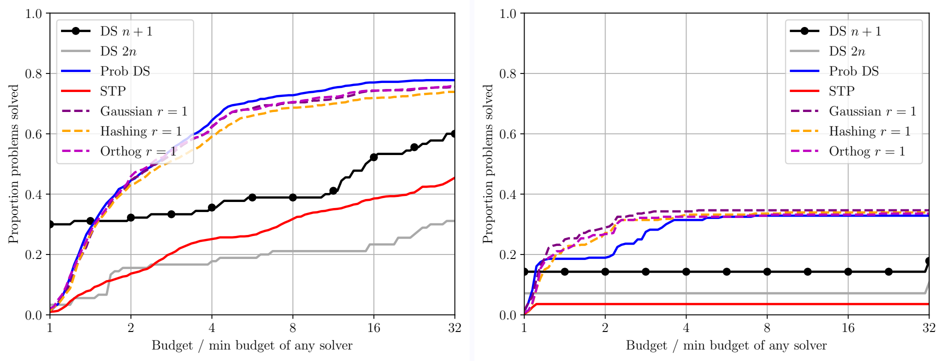
Budget: $200(n+1)$ evaluations.

Comparison:

- Deterministic DS with $\mathcal{D}_k = [\boldsymbol{I}_n \ -\boldsymbol{I}_n]$ or $\mathcal{D}_k = [\boldsymbol{I}_n \ -1_n]$;
- Probabilistic direct search with 2 uniform directions;
- Stochastic Three Point;
- Probabilistic direct search with Gaussian/Hashing/Orthogonal $\boldsymbol{P}_k$ matrices + $r = 1$.

Goal: Satisfy $f(\boldsymbol{x}_k) - f_{opt} \leq 0.1(f(\boldsymbol{x}_0) - f_{opt})$.

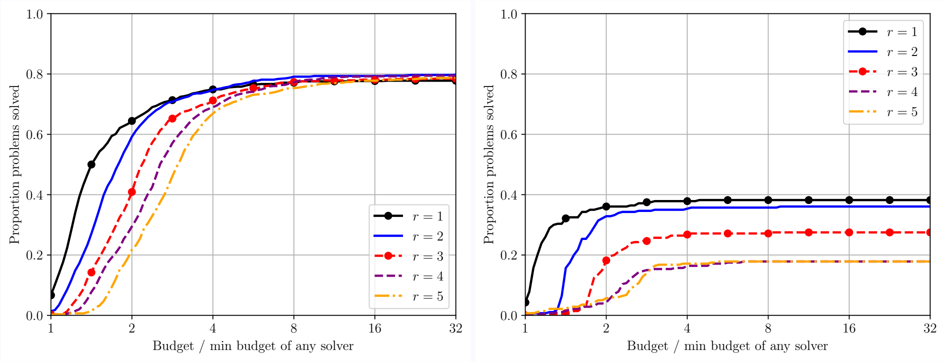
Comparison of all methods



Left: Medium scale; Right: Large scale.

- Challenging examples for (basic) direct search.
- Random subspaces bring improvement!

Gaussian matrices and subspace dimensions



Left: Medium scale; Right: Large scale.

Numerically

- Subspace dimension > 1 may improve performance...
- ...but in general opposite (Gaussian) directions work best!

Towards more numerics...

The package

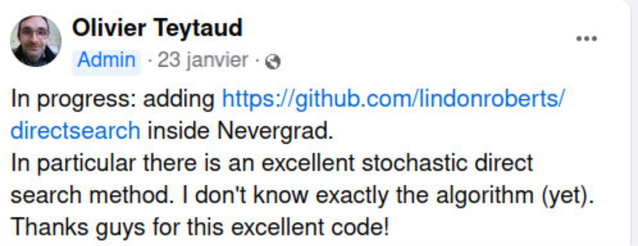
- `https://github.com/lindonroberts/directsearch`
- Python code + paper experiments.
- `pip install directsearch`

Towards more numerics...

The package

- `https://github.com/lindonroberts/directsearch`
- Python code + paper experiments.
- `pip install directsearch`

Recent use at Meta:



Replaced CMA-ES in optimization wizard on smooth problems!

Outline

Our findings

If you want to scale up...

- Can compute steps in r-dim. subspaces, $r = \mathcal{O}(1)$;
- Reduced evaluation cost per iteration;
- Overall complexity: $\mathcal{O}(n^2) \Rightarrow \mathcal{O}(n)$!

Numerically

- Subspaces of dimension $r > 1$ may be good...
- ...but in general opposite Gaussian directions ($r = 1$) are better!

Why do 1-dim. subspaces give best performance?

Key result (Hare, Roberts, R. '22)

Let $\boldsymbol{g} \in \mathbb{S}^{n-1}$, $\boldsymbol{P} \in \mathbb{R}^{r \times n}$ and $\mathcal{D} = [\boldsymbol{I}_r \ -\boldsymbol{I}_r]$.
Then, the expected decrease ratio

$$\frac{\mathbb{E}\left[\min_{\boldsymbol{d} \in \mathcal{D}} \boldsymbol{g}^{\mathrm{T}} \boldsymbol{P}^{\mathrm{T}} \boldsymbol{d}\right]}{2r}$$

is minimized at $r = 1$.

Warren: "But *why* does this work?"

Why do 1-dim. subspaces give best performance?

Key result (Hare, Roberts, R. '22)

Let $\boldsymbol{g} \in \mathbb{S}^{n-1}$, $\boldsymbol{P} \in \mathbb{R}^{r \times n}$ and $\mathcal{D} = [\boldsymbol{I}_r \ -\boldsymbol{I}_r]$.
Then, the expected decrease ratio

$$\frac{\mathbb{E}\left[\min_{\boldsymbol{d} \in \mathcal{D}} \boldsymbol{g}^{\mathrm{T}} \boldsymbol{P}^{\mathrm{T}} \boldsymbol{d}\right]}{2r}$$

is minimized at $r = 1$.

- To decrease $\boldsymbol{x} \mapsto \boldsymbol{g}^{\mathrm{T}} \boldsymbol{x}$, $r = 1$ gives the best "bang for your buck".
- Using Taylor approximation

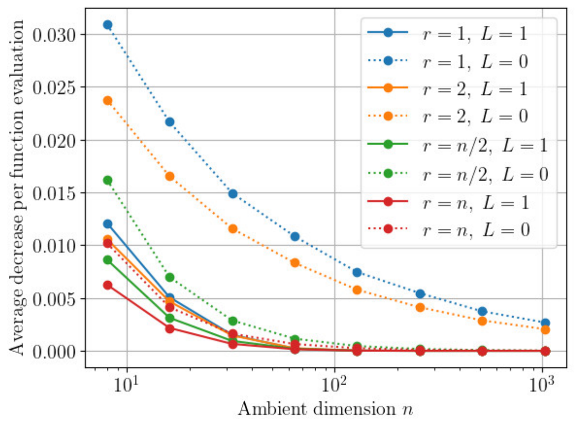
$$f(\boldsymbol{x} + \boldsymbol{v}) - f(\boldsymbol{x}) \approx \nabla f(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{v},$$

explains why this happens beyond linear functions.

Numerical validation

Setup

- Monte-Carlo approximations of expected decrease.
- Quadratic functions with a random linear term $\boldsymbol{x} \mapsto \boldsymbol{g}^{\mathrm{T}}\boldsymbol{x} + \frac{L}{2}\|\boldsymbol{x}\|^2$.
- Normalization by the number of function evaluations.



Summary

Our results...

- Probabilistic analysis/subspace viewpoint.
- Improved complexity backed up by numerics.
- Low dimension provably better on average.

Summary

Our results...

- Probabilistic analysis/subspace viewpoint.
- Improved complexity backed up by numerics.
- Low dimension provably better on average.

...and beyond

- Stochastic setting (Hot topic!).
- Constraints (Ongoing work).
- More numerics (Solvers/Applications).

That's it!

References

- *Direct search based on probabilistic descent in reduced spaces*
  L. Roberts and C. W. Royer, SIAM J. Optim. 33(4):3057-3082, 2023.
- *Expected decrease for derivative-free algorithms using random subspaces*
  W. Hare, L. Roberts and C. W. Royer, Math. Comp., 94:277-304, 2025.

- `https://github.com/lindonroberts/directsearch`

That's it!

References

- *Direct search based on probabilistic descent in reduced spaces*
  L. Roberts and C. W. Royer, SIAM J. Optim. 33(4):3057-3082, 2023.
- *Expected decrease for derivative-free algorithms using random subspaces*
  W. Hare, L. Roberts and C. W. Royer, Math. Comp., 94:277-304, 2025.
- `https://github.com/lindonroberts/directsearch`

Merci!

`clement.royer@lamsade.dauphine.fr`