A derivative-free algorithm for continuous submodular optimization

Clément Royer

Joint work with Marc Kaspar

COCANA Seminar - July 29, 2025

Dauphine | PSL UNIVERSITÉ PARIS

PR[AI]RIE PaRis Artificial Intelligence Research InstitutE

Story of an internship in Dauphine

Timeline

- **Fall 2023**: Marc follows my Master 1 course.
- **May 2024**: Starts an internship with me.
- **August 2024**: The internship ends.
- **July 2025**: This talk!

Story of an internship in Dauphine

Timeline

- **Fall 2023**: Marc follows my Master 1 course.
- **May 2024**: Starts an internship with me.
- **August 2024**: The internship ends.
- **July 2025**: This talk!

The topic: Submodular optimization

- Popular in ML recently (part of my department).
- Originally a discrete maths concept (the other part of my department).

$\rightarrow$ Goal: Apply what I do (derivative-free optimization) to this setting!

Outline

Outline

Submodular functions

Definition (Edmonds '70?)

$f : 2^n \to \mathbb{R}$ is submodular if for every $A \subset B \subset \{1, \dots, n\}$ and $\boldsymbol{v} \notin B$, one has

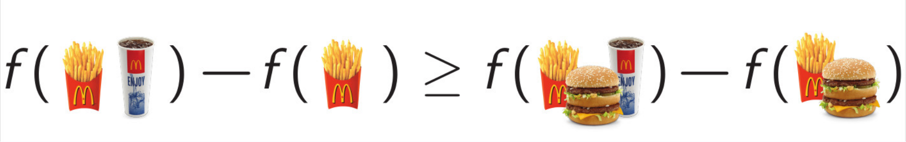
$$f(A \cup \{\boldsymbol{v}\}) - f(A) \geq f(B \cup \{\boldsymbol{v}\}) - f(B).$$

Submodular functions

Definition (Edmonds '70?)

$f : 2^n \to \mathbb{R}$ is submodular if for every $A \subset B \subset \{1, \dots, n\}$ and $\boldsymbol{v} \notin B$, one has

$$f(A \cup \{\boldsymbol{v}\}) - f(A) \geq f(B \cup \{\boldsymbol{v}\}) - f(B).$$



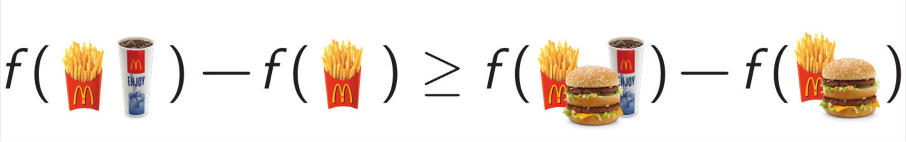
J. Bilmes, *Submodular Functions, Optimization and Machine Learning (2020).*

Submodular functions

Definition (Edmonds '70?)

$f : 2^n \to \mathbb{R}$ is submodular if for every $A \subset B \subset \{1, \dots, n\}$ and $\boldsymbol{v} \notin B$, one has

$$f(A \cup \{\boldsymbol{v}\}) - f(A) \geq f(B \cup \{\boldsymbol{v}\}) - f(B).$$



J. Bilmes, *Submodular Functions, Optimization and Machine Learning (2020).*

- Discrete concept for set-valued functions.
- Sometimes called diminishing returns (DR).
- Arises in economics, network/graph theory, etc.

Discrete submodular optimization

A submodular optimization problem (for today)

$$\underset{X \subset \{1,\dots,n\}}{\text{maximize}} \; f(X) \quad \text{s.t.} \quad X \in \mathcal{C}$$

- f submodular function.
- $\mathcal{C} \subset 2^n$ constraint set.

$\rightarrow$ NP-hard problem in general!

$\rightarrow$ Optimal value can be approximated up to a certain factor.

Discrete submodular optimization

A submodular optimization problem (for today)

$$\underset{X \subset \{1,\dots,n\}}{\text{maximize}} \; f(X) \quad \text{s.t.} \quad X \in \mathcal{C}$$

- f submodular function.
- $\mathcal{C} \subset 2^n$ constraint set.

$\rightarrow$ NP-hard problem in general!

$\rightarrow$ Optimal value can be approximated up to a certain factor.

Example: Cardinality constraint

- f nonnegative monotone, $\mathcal{C} = \{|X| \le k\}$, $k < n$.
- Can compute X_k such that $f(X_k) \ge \left(1 - \frac{1}{e}\right) \max_{|X| \le k} f(X)$ in polynomial time!

Continuous submodular functions

Continuous submodularity

- Discrete submodularity ($f : 2^n \to \mathbb{R}$)

$$\forall (X, Y) \in 2^n, \qquad f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y).$$

Continuous submodular functions

Continuous submodularity

- Discrete submodularity ($f : 2^n \to \mathbb{R}$)

$$\forall (X, Y) \in 2^n, \qquad f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y).$$

- **Continuous submodularity** ($f : [0, 1]^n \to \mathbb{R}$)

$$\forall (\boldsymbol{x}, \boldsymbol{y}) \in [0, 1]^n, \quad f(\boldsymbol{x}) + f(\boldsymbol{y}) \geq f(\boldsymbol{x} \vee \boldsymbol{y}) + f(\boldsymbol{x} \wedge \boldsymbol{y}),$$

$\vee / \wedge$ componentwise max/min.

Continuous submodular functions

Continuous submodularity

- Discrete submodularity ($f : 2^n \to \mathbb{R}$)

$$\forall (X, Y) \in 2^n, \qquad f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y).$$

- **Continuous submodularity** ($f : [0,1]^n \to \mathbb{R}$)

$$\forall (\boldsymbol{x}, \boldsymbol{y}) \in [0,1]^n, \quad f(\boldsymbol{x}) + f(\boldsymbol{y}) \geq f(\boldsymbol{x} \vee \boldsymbol{y}) + f(\boldsymbol{x} \wedge \boldsymbol{y}),$$

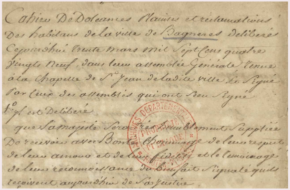
$\vee / \wedge$ componentwise max/min.

- Several other concepts of submodularity exist, with connections to concavity (An Bian et al '17, Bilmes '22).
- Applications in information theory and natural language processing.

Case study: Topic modeling and summarization

Topic modeling

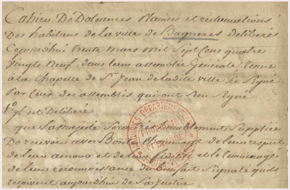
- Data: Text documents.
- Goal: Identify topics through occurrences of certain words.



Example: Grievances from France's 1789 *cahiers!*

Topic modeling

- Data: Text documents.
- Goal: Identify topics through occurrences of certain words.



Example: Grievances from France's 1789 *cahiers!*

Topic summarization (Lin & Bilmes '11)

- Input: Documents and probabilistic topic model for each document.
- Goal: Select a subset of documents to maximize the probabilistic coverage of topics.
  - Discrete: Select a subset of documents.
  - Continuous: Select document i with probability $x_i \in [0, 1]$.

$\rightarrow$ **Submodular maximization problem!**

From submodular to a internship

- **Continuous submodular optimization**
  - Continuous submodular replaces discrete submodular.
  - Cool applications in natural language processing.

From submodular to a internship

- **Continuous submodular optimization**
  - Continuous submodular replaces discrete submodular.
  - Cool applications in natural language processing.
- **Solving those problems**
  - Algorithms with various approximation guarantees and **complexity** (An Bian et al '17).
  - **Derivative-free** optimization techniques (Chen et al '20).

From submodular to a internship

- **Continuous submodular optimization**
  - Continuous submodular replaces discrete submodular.
  - Cool applications in natural language processing.
- **Solving those problems**
  - Algorithms with various approximation guarantees and **complexity** (An Bian et al '17).
  - **Derivative-free** optimization techniques (Chen et al '20).

My internship proposal

Given a submodular optimization problem and my favorite (derivative-free) algorithm,

- What can we prove in theory?
- Does it work in practice?

Outline

1. Continuous submodular optimization
2. Derivative-free optimization and direct search
3. Submodular optimization with direct search
4. Conclusion

Derivative-free optimization

Today's problem class

$$\begin{cases} \text{maximize}_{\boldsymbol{x} \in \mathbb{R}^n} & f(\boldsymbol{x}) \\ \text{subject to} & \boldsymbol{x} \in \mathcal{F} = \{\boldsymbol{\ell} \le \boldsymbol{x} \le \boldsymbol{u}, \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}\}. \end{cases}$$

with $f : \mathbb{R}^n \to \mathbb{R}$, $\mathcal{F}$ polytope.

Derivative-free optimization

Today's problem class

$$\begin{cases} \text{maximize}_{\boldsymbol{x} \in \mathbb{R}^n} & f(\boldsymbol{x}) \\ \text{subject to} & \boldsymbol{x} \in \mathcal{F} = \{\boldsymbol{\ell} \le \boldsymbol{x} \le \boldsymbol{u}, \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}\}. \end{cases}$$

with $f : \mathbb{R}^n \to \mathbb{R}$, $\mathcal{F}$ polytope.

Derivative-free/Black-box optimization setup

- Derivatives of f not available for algorithmic purposes.
- Algorithm must use only function values.

Derivative-free optimization

Today's problem class

$$\begin{cases} \text{maximize}_{\boldsymbol{x} \in \mathbb{R}^n} & f(\boldsymbol{x}) \\ \text{subject to} & \boldsymbol{x} \in \mathcal{F} = \{\boldsymbol{\ell} \le \boldsymbol{x} \le \boldsymbol{u}, \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}\}. \end{cases}$$

with $f : \mathbb{R}^n \to \mathbb{R}$, $\mathcal{F}$ polytope.

Derivative-free/Black-box optimization setup

- Derivatives of f not available for algorithmic purposes.
- Algorithm must use only function values.

Two algorithmic paradigms in derivative-free optimization

- Build a model of f.
- Explore the space through directions$\rightarrow$Direct search.

Direct search for maximization

Problem maximize$_{\boldsymbol{x}\in\mathbb{R}^n}$ $f(\boldsymbol{x})$ s.t. $\boldsymbol{x} \in \mathcal{F} = \{\boldsymbol{\ell} \leq \boldsymbol{x} \leq \boldsymbol{u},\ \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}$.

Inputs: $\boldsymbol{x}_0 \in \mathcal{F}$, $\alpha_0 > 0$.
Iteration k: Given $(\boldsymbol{x}_k, \alpha_k)$,

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

$$\boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k \in \mathcal{F} \qquad \text{and} \qquad f(\boldsymbol{x}_k + \alpha_k\, \boldsymbol{d}_k) > f(\boldsymbol{x}_k) + \alpha_k^2$$

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

Direct search for maximization

Problem maximize$_{\boldsymbol{x}\in\mathbb{R}^n}$ $f(\boldsymbol{x})$ s.t. $\boldsymbol{x} \in \mathcal{F} = \{\boldsymbol{\ell} \le \boldsymbol{x} \le \boldsymbol{u},\ \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}\}$.

Inputs: $\boldsymbol{x}_0 \in \mathcal{F}$, $\alpha_0 > 0$.
Iteration k: Given $(\boldsymbol{x}_k, \alpha_k)$,

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

$$\boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k \in \mathcal{F} \qquad \text{and} \qquad f(\boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k) > f(\boldsymbol{x}_k) + \alpha_k^2$$

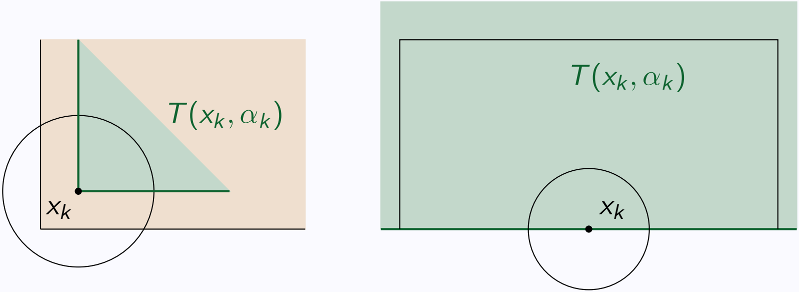
  set $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k$, $\alpha_{k+1} := 2\alpha_k$.
- Otherwise, set $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k$, $\alpha_{k+1} := \alpha_k/2$.

Key for theory and pratice: Choice of $\mathcal{D}_k$.

Assumptions on directions $\mathcal{D}_k$

- $\mathcal{D}_k$ consists of m unit vectors.
- $\mathcal{D}_k$ is a κ-descent set for the constraints

$$\min_{\substack{\boldsymbol{v} \in T(\boldsymbol{x}_k, \alpha_k) \\ \boldsymbol{v} \neq 0}} \max_{\boldsymbol{d} \in \mathcal{D}_k} \frac{\boldsymbol{v}^{\mathrm{T}} \boldsymbol{d}}{\|\boldsymbol{v}\|} \geq \kappa.$$



Complexity result for maximize$_{\boldsymbol{x} \in \mathcal{F}} f(\boldsymbol{x})$

Assumptions

- f is $\mathcal{C}^1$, ∇f Lipschitz continuous.
- f is **concave**, has a maximum f^*.
- Distance to maxima is bounded (technical condition).
- $\mathcal{D}_k$ κ-descent, $|\mathcal{D}_k| = m \ \forall k$.

Theorem (from Dodangeh & Vicente '14, Gratton et al '19)

Direct search reaches $\boldsymbol{x}_k$ such that $f^* - f(\boldsymbol{x}_k) \le \epsilon$ in at most

- $\mathcal{O}(\kappa^{-2}\epsilon^{-1})$ iterations.
- $\mathcal{O}(m\kappa^{-2}\epsilon^{-1})$ evaluations of f.

Complexity result for maximize$_{\mathbf{x} \in \mathcal{F}} f(\mathbf{x})$

Assumptions

- f is $\mathcal{C}^1$, ∇f Lipschitz continuous.
- f is **concave**, has a maximum f^*.
- Distance to maxima is bounded (technical condition).
- $\mathcal{D}_k$ κ-descent, $|\mathcal{D}_k| = m \ \forall k$.

Theorem (from Dodangeh & Vicente '14, Gratton et al '19)

Direct search reaches $\mathbf{x}_k$ such that $f^* - f(\mathbf{x}_k) \leq \epsilon$ in at most

- $\mathcal{O}(\kappa^{-2}\epsilon^{-1})$ iterations.
- $\mathcal{O}(m\kappa^{-2}\epsilon^{-1})$ evaluations of f.

Complexity result for maximize$_{\mathbf{x}\in\mathcal{F}} f(\mathbf{x})$

Assumptions

- f is $\mathcal{C}^1$, ∇f Lipschitz continuous.
- f is concave, has a maximum f^*.
- Distance to maxima is bounded (technical condition).
- $\mathcal{D}_k$ κ-descent, $|\mathcal{D}_k| = m \ \forall k$.

Theorem (from Dodangeh & Vicente '14, Gratton et al '19)

Direct search reaches $\mathbf{x}_k$ such that $f^* - f(\mathbf{x}_k) \le \epsilon$ in at most

- $\mathcal{O}(\kappa^{-2}\epsilon^{-1})$ iterations.
- $\mathcal{O}(m\kappa^{-2}\epsilon^{-1})$ evaluations of f.

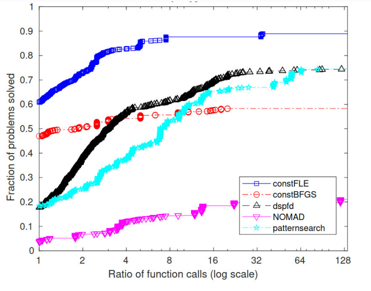
$\rightarrow$ ϵ^{-1}: on par with derivative-based methods.

$\rightarrow$ $m\,\kappa^{-2}$: cost of being derivative-free.

What about practice?

My code: `dspfd`

- Deterministic and randomized techniques for choosing directions in linearly-constrained problems.
- MATLAB code from 2017, still works off the shelf!
- Still beats MATLAB's *patternsearch* (and sometimes Polytechnique Montréal's *nomad*).



Experiments on CUTEst linearly-constrained problems (Royer et al '24).

Outline

Same problem, different assumptions

$$\begin{cases} \text{maximize}_{\boldsymbol{x} \in \mathbb{R}^n} & f(\boldsymbol{x}) \\ \text{subject to} & \boldsymbol{x} \in \mathcal{F} = \{\boldsymbol{\ell} \le \boldsymbol{x} \le \boldsymbol{u}, \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}\}. \end{cases}$$

Key: Suppose f is (DR)-submodular!

Same problem, different assumptions

$$\begin{cases} \text{maximize}_{\boldsymbol{x} \in \mathbb{R}^n} & f(\boldsymbol{x}) \\ \text{subject to} & \boldsymbol{x} \in \mathcal{F} = \{\boldsymbol{\ell} \leq \boldsymbol{x} \leq \boldsymbol{u}, \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}. \end{cases}$$

Key: Suppose f is (DR)-submodular!

Definition

f is **DR-submodular** if

$$\forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{F}^2, \quad \boldsymbol{x} \geq \boldsymbol{y} \Rightarrow \nabla f(\boldsymbol{x}) \geq \nabla f(\boldsymbol{y}).$$

$\boldsymbol{x} \geq \boldsymbol{y} \Leftrightarrow x_i \geq y_i \ \forall i = 1, \dots, n.$

Same problem, different assumptions

$$\begin{cases} \text{maximize}_{\boldsymbol{x} \in \mathbb{R}^n} & f(\boldsymbol{x}) \\ \text{subject to} & \boldsymbol{x} \in \mathcal{F} = \{\boldsymbol{\ell} \le \boldsymbol{x} \le \boldsymbol{u}, \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}\}. \end{cases}$$

Key: Suppose f is (DR)-submodular!

Definition

f is **DR-submodular** if

$$\forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{F}^2, \quad \boldsymbol{x} \ge \boldsymbol{y} \Rightarrow \nabla f(\boldsymbol{x}) \ge \nabla f(\boldsymbol{y}).$$

$\boldsymbol{x} \ge \boldsymbol{y} \Leftrightarrow x_i \ge y_i \ \forall i = 1, \dots, n.$

- Continuous submodularity+concavity along positive directions.
- Example Nonconvex quadratics

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{x}, \quad A_{ij} \le 0 \ \forall (i,j).$$

Assumptions on f and previous complexity

Assume f is

- DR-submodular.
- Monotone: $f(\boldsymbol{x}) \geq f(\boldsymbol{y})$ if $\boldsymbol{x} \geq \boldsymbol{y}$.

Goal: Approach the value $(1-\frac{1}{\mathrm{e}})f^*$.

Derivative-based Frank-Wolfe method (Bian et al '17)

After K iterations, get $\boldsymbol{x}_K^{FW}$ such that

$$f(\boldsymbol{x}_K^{FW}) \geq (1-\mathrm{e}^{-\delta})f^* - \frac{L}{2}\sum_{k=0}^{K-1}\gamma_k^2 + \mathrm{e}^{-\delta} f(\boldsymbol{x}_0).$$

- γ_k: Stepsize, predefined (constant).
- $\delta \in (0,1)$.
- Translates in complexity $\mathcal{O}(\epsilon^{-1})$ to get within ϵ of $\left(1-\frac{1}{\mathrm{e}}\right) f^*$.

Direct search for maximization on $[\boldsymbol{\ell}, \boldsymbol{u}]$

Inputs: $\boldsymbol{x}_0 \in \mathcal{F}$, $\alpha_0 > 0$.

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

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

$$\boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k \in \mathcal{F} \quad \text{and} \quad f(\boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k) > f(\boldsymbol{x}_k) + \alpha_k^2$$

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

Direct search for maximization on $[\boldsymbol{\ell}, \boldsymbol{u}]$

Inputs: $\boldsymbol{x}_0 \in \mathcal{F}$, $\alpha_0 > 0$.

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

- Choose a set $\mathcal{D}_k \subset \mathbb{R}^n$ of m vectors.
- If $\boldsymbol{d}_k = \mathrm{argmax}_{\boldsymbol{d} \in \mathcal{D}_k} f(\boldsymbol{x}_k + \alpha_k \boldsymbol{d})$ satisfies

$$\boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k \in \mathcal{F} \quad \text{and} \quad f(\boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k) > f(\boldsymbol{x}_k) + \alpha_k^2$$

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

- Difference from before: complete polling.
- Needed for the analysis, not in practice.

Direction quality

Classical assumption: $\mathcal{D}_k$ has m unit vectors.

Main assumption on $\mathcal{D}_k$

At every iteration k,

$$\max_{\boldsymbol{d} \in D_k} \boldsymbol{d}^{\mathrm{T}} \nabla f(\boldsymbol{x}_k) \geq \kappa \max_{\boldsymbol{v} \in \mathcal{F}, \|\boldsymbol{v}\| \leq 1} \boldsymbol{v}^{\mathrm{T}} \nabla f(\boldsymbol{x}_k) \quad \text{where} \quad \kappa \in (0,1].$$

- **Stronger** than the assumption from the concave case.
- Link to Frank-Wolfe requirements.

Direction quality

Classical assumption: $\mathcal{D}_k$ has m unit vectors.

Main assumption on $\mathcal{D}_k$

At every iteration k,

$$\max_{\boldsymbol{d} \in D_k} \boldsymbol{d}^{\mathrm{T}} \nabla f(\boldsymbol{x}_k) \geq \kappa \max_{\boldsymbol{v} \in \mathcal{F}, \|\boldsymbol{v}\| \leq 1} \boldsymbol{v}^{\mathrm{T}} \nabla f(\boldsymbol{x}_k) \quad \text{where} \quad \kappa \in (0,1].$$

- **Stronger** than the assumption from the concave case.
- Link to Frank-Wolfe requirements.

The case $\mathcal{F} = \{0 \leq \boldsymbol{x} \leq 1\}$

Natural choice: $\mathcal{D}_k = [\boldsymbol{I}_n \ -\boldsymbol{I}_n]$.

- Concave case: κ-descent with $\kappa = \frac{1}{\sqrt{n}}$.
- Submodular case: $\kappa = \frac{1}{n}$ (worse!).

Results in the submodular case

Theorem (Kaspar, R. '24-25)

After $K \geq 1$ successful iterations $j_1, \dots, j_K$, the method satisfies

$$f(x_{j_K}) \;\geq\; f^* - \left(1 - \mathrm{e}^{-\mathcal{O}(\kappa^2)\sum_{i=1}^K \alpha_{j_i}^2}\right)(f^* - f(\boldsymbol{x}_0)) - \frac{L}{2}\sum_{i=1}^K \alpha_{j_i}^2.$$

Results in the submodular case

Theorem (Kaspar, R. '24-25)

After $K \geq 1$ successful iterations $j_1, \dots, j_K$, the method satisfies

$$f(x_{j_K}) \;\geq\; f^* - \left(1 - \mathrm{e}^{-\mathcal{O}(\kappa^2)\sum_{i=1}^K \alpha_{j_i}^2}\right)(f^* - f(\mathbf{x}_0)) - \frac{L}{2}\sum_{i=1}^K \alpha_{j_i}^2.$$

Corollary

Reaches $\mathbf{x}_k$ such that $f(\mathbf{x}_k) \geq (1 - \frac{1}{e})f^* + \epsilon$ in at most

- $\tilde{\mathcal{O}}(\kappa^{-2}\,\epsilon^{-1})$ iterations.
- $\tilde{\mathcal{O}}(m\,\kappa^{-2}\,\epsilon^{-1})$ iterations.

$\rightarrow \epsilon^{-1}$: On par with (derivative-based) Frank-Wolfe approach

$\rightarrow m\,\kappa^{-2}$: Similar to concave maximization (but values of m/κ may differ!)

Other stepsize choices

- Previous work: Predefined stepsizes (fixed, adaptive).
- Variant of direct search without distinction between success/failure.

Other stepsize choices

- Previous work: Predefined stepsizes (fixed, adaptive).
- Variant of direct search without distinction between success/failure.

α_k	Guarantee after K iterations	Complexity (its/evals)
$\frac{1}{K}$	$f(\boldsymbol{x}_K) \geq (1-\mathrm{e}^{-\kappa}) f(\boldsymbol{x}^*) - \frac{L}{2K} + \mathrm{e}^{-\kappa} f(\boldsymbol{x}_0)$	$\mathcal{O}(\kappa^{-1}\epsilon^{-1})/\mathcal{O}(m\kappa^{-1}\epsilon^{-1})$
$\frac{1}{\kappa K}$	$f(\boldsymbol{x}_K) \geq (1-\mathrm{e}^{-1}) f(\boldsymbol{x}^*) - \frac{L\kappa}{2K} + \mathrm{e}^{-1} f(\boldsymbol{x}_0)$	$\mathcal{O}(\kappa^{-1}\epsilon^{-1})/\mathcal{O}(m\kappa^{-1}\epsilon^{-1})$.

Other stepsize choices

- Previous work: Predefined stepsizes (fixed, adaptive).
- Variant of direct search without distinction between success/failure.

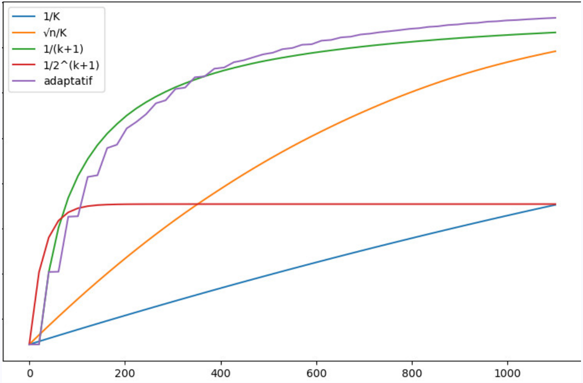
α_k	Guarantee after K iterations	Complexity (its/evals)
$\frac{1}{K}$	$f(\boldsymbol{x}_K) \geq (1-\mathrm{e}^{-\kappa})f(\boldsymbol{x}^*) - \frac{L}{2K} + \mathrm{e}^{-\kappa} f(\boldsymbol{x}_0)$	$\mathcal{O}(\kappa^{-1}\epsilon^{-1})/\mathcal{O}(m\kappa^{-1}\epsilon^{-1})$
$\frac{1}{\kappa K}$	$f(\boldsymbol{x}_K) \geq (1-\mathrm{e}^{-1})f(\boldsymbol{x}^*) - \frac{L\kappa}{2K} + \mathrm{e}^{-1} f(\boldsymbol{x}_0)$	$\mathcal{O}(\kappa^{-1}\epsilon^{-1})/\mathcal{O}(m\kappa^{-1}\epsilon^{-1})$.

- Better complexities than before (factor κ^{-1}).
- Should we use those instead?

The best stepsize choice

Two variants of direct search

- Fixed/Decreasing step sizes.
- *adaptatif* (pardon my French): Classical updating rule.



Run on a submodular quadratic over $[0, 1]^{10}$ using 100 simplex gradients.

A topic summarization problem

Data

- 40 lectures that I gave on optimization for machine learning.
- 4 courses (1-8, 9-16, 17-24, 25-40).
- Discrete probability distribution of the lectures around 4 topics (derivatives/convexity/algorithms/applications).

$\rightarrow$ A matrix of topic probabilities $T \in [0,1]^{40 \times 1}$.

A topic summarization problem

Data

- 40 lectures that I gave on optimization for machine learning.
- 4 courses (1-8, 9-16, 17-24, 25-40).
- Discrete probability distribution of the lectures around 4 topics (derivatives/convexity/algorithms/applications).

$\rightarrow$ A matrix of topic probabilities $T \in [0,1]^{40 \times 1}$.

Topic summarization in this setting

- Find a subset of lectures that covers the four topics as best as possible.
- Constraints: At most two lectures from the first three courses, four from the last course.

A topic summarization problem ('ed)

The problem

$$\begin{cases} \text{maximize}_{\boldsymbol{x} \in \mathbb{R}^{40}} & \frac{1}{4}\sum_{t=1}^{4}\left(1 - \prod_{i=1}^{40}(1 - p_i(t)x_i)\right) \\ \text{s.t.} & \sum_{i=1}^{8} x_i \le 2, \sum_{i=9}^{16} x_i \le 2 \\ & \sum_{i=17}^{24} x_i \le 2, \sum_{i=25}^{40} x_i \le 4 \\ & 0 \le \boldsymbol{x} \le 1. \end{cases}$$

- Continuous submodular optimization problem!
- Probabilities explicit here, could result from a blackbox process.

A topic summarization problem ('ed)

The problem

$$
\begin{cases}
\text{maximize}_{\boldsymbol{x}\in\mathbb{R}^{40}} & \frac{1}{4}\sum_{t=1}^{4}\left(1-\prod_{i=1}^{40}(1-p_i(t)x_i)\right) \\
\text{s.t.} & \sum_{i=1}^{8} x_i \le 2, \sum_{i=9}^{16} x_i \le 2 \\
 & \sum_{i=17}^{24} x_i \le 2, \sum_{i=25}^{40} x_i \le 4 \\
 & 0 \le \boldsymbol{x} \le 1.
\end{cases}
$$

- Continuous submodular optimization problem!
- Probabilities explicit here, could result from a blackbox process.

Comparison

- Deterministic and randomized direct-search variants (`dspfd`).
- Budget: $200n$ evaluations ($n = 40$).

Results: Deterministic VS Randomized approach

- Best function value: Deterministic (0.96 VS 0.94).
- Sparser solution: Randomized (10 nonzero VS 25).
  $\Rightarrow$ Randomized better at finding integer solutions!

Results: Deterministic VS Randomized approach

- Best function value: Deterministic (0.96 VS 0.94).
- Sparser solution: Randomized (10 nonzero VS 25).
  $\Rightarrow$ Randomized better at finding integer solutions!

Lectures selected by randomized approach

- Course 1: Basics of Optimization, Gradient descent.
- Course 2: Basics of Optimization, Last year's exam.
- Course 3: Basics of Optimization, Lab gradient descent.
- Course 4: Optimality conditions, Advanced gradient descent, Stochastic gradient, Course homework.

Good coverage of the four topics (derivatives/convexity/algorithms/applications).

Summing up

Submodular optimization

- Discrete and continuous concepts!
- Applications in machine learning.

Summing up

Submodular optimization

- Discrete and continuous concepts!
- Applications in machine learning.

Direct search and concave maximization

- Existing algorithms for linearly constrained problems.
- Guarantees for concave and non-concave maximization.

Summing up

Submodular optimization

- Discrete and continuous concepts!
- Applications in machine learning.

Direct search and concave maximization

- Existing algorithms for linearly constrained problems.
- Guarantees for concave and non-concave maximization.

Our method for submodular optimization

- Guarantees even with adaptive stepsizes (under complete polling?)
- Encouraging behavior of randomized variants.

References

- C. Audet and W. Hare, *Derivative-Free and Blackbox Optimization*, Second Edition, Springer, 2025.
- A. An Bian, B. Mirzasoleiman, J. M. Buhmann and A. Krause, *Guaranteeed Non-convex Optimization: Submodular Maximization over Continuous Domains*, AISTATS, 2017.
- J. Bilmes, *Submodularity in Machine Learning and Artificial Intelligence*, arXiv:2202.00132, 2022.
- L. Chen, M. Zhang, H. Hassani and A. Karbasi, *Black Box Submodular Maximization: Discrete and Continuous Settings*, AISTATS, 2020.
- M. Dodangeh and L. N. Vicente, *Worst case complexity of direct search under convexity*, Mathematical Programming, 2015.
- J. Edmonds, *Submodular Functions, Matroids, and Certain Polyhedra*, 1970.
- S. Gratton, C. W. Royer, L. N. Vicente and Z. Zhang, *Direct search based on probabilistic feasible descent for bound and linearly constrained problems*, Computational Optimization and Applications, 2019.
- M. Kaspar and C. W. Royer, *A direct-search method for continuous submodular optimization*, 202X.
- A. Krause and D. Golovin, *Submodular Function Maximization*, Tractability, 2014.
- H. Lin and J. Bilmes, *A class of submodular functions for document summarization*, ACL, 2011.
- C. W. Royer, O. Sohab and L. N. Vicente, *Full-Low Evaluation Methods For Bound and Linearly Constrained Derivative-Free Optimization*, Computational Optimization and Applications, 2024.

References

- C. Audet and W. Hare, *Derivative-Free and Blackbox Optimization*, Second Edition, Springer, 2025.
- A. An Bian, B. Mirzasoleiman, J. M. Buhmann and A. Krause, *Guaranteeed Non-convex Optimization: Submodular Maximization over Continuous Domains*, AISTATS, 2017.
- J. Bilmes, *Submodularity in Machine Learning and Artificial Intelligence*, arXiv:2202.00132, 2022.
- L. Chen, M. Zhang, H. Hassani and A. Karbasi, *Black Box Submodular Maximization: Discrete and Continuous Settings*, AISTATS, 2020.
- M. Dodangeh and L. N. Vicente, *Worst case complexity of direct search under convexity*, Mathematical Programming, 2015.
- J. Edmonds, *Submodular Functions, Matroids, and Certain Polyhedra*, 1970.
- S. Gratton, C. W. Royer, L. N. Vicente and Z. Zhang, *Direct search based on probabilistic feasible descent for bound and linearly constrained problems*, Computational Optimization and Applications, 2019.
- M. Kaspar and C. W. Royer, *A direct-search method for continuous submodular optimization*, 202X.
- A. Krause and D. Golovin, *Submodular Function Maximization*, Tractability, 2014.
- H. Lin and J. Bilmes, *A class of submodular functions for document summarization*, ACL, 2011.
- C. W. Royer, O. Sohab and L. N. Vicente, *Full-Low Evaluation Methods For Bound and Linearly Constrained Derivative-Free Optimization*, Computational Optimization and Applications, 2024.

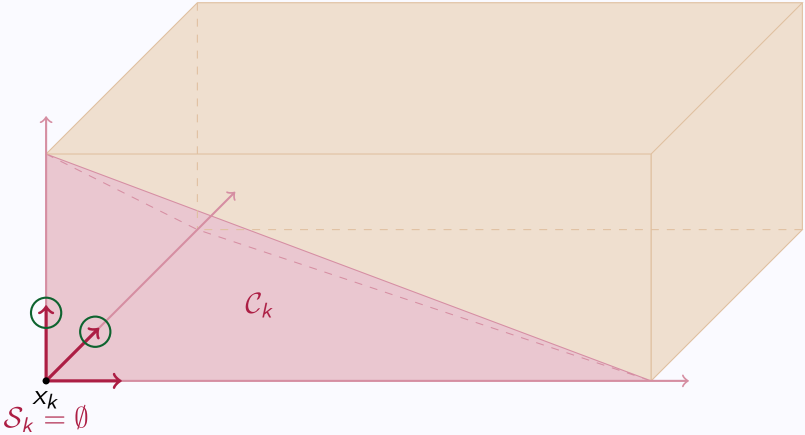
Thank you!

clement.royer@lamsade.dauphine.fr

Bonus: Randomized strategy for direct search

Decomposition $T(\boldsymbol{x}_k, \alpha_k) = \mathcal{C}_k + \mathcal{S}_k$

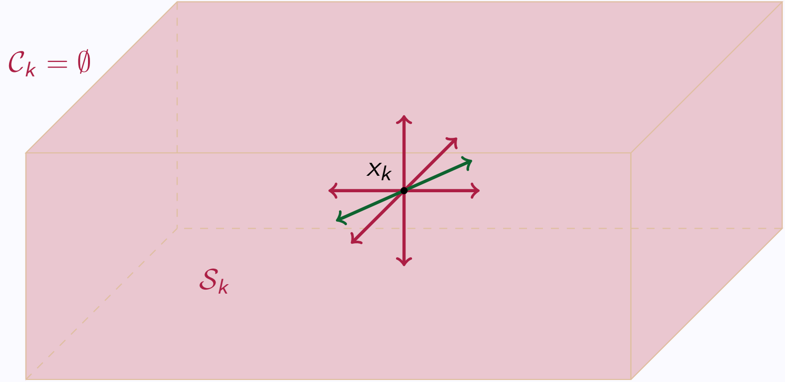
- In $\mathcal{C}_k$: Random subset of generators.



Bonus: Randomized strategy for direct search

Decomposition $T(\boldsymbol{x}_k, \alpha_k) = \mathcal{C}_k + \mathcal{S}_k$

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



Bonus: Randomized strategy for direct search

Decomposition $T(\boldsymbol{x}_k, \alpha_k) = \mathcal{C}_k + \mathcal{S}_k$

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