A polynomial lower bound on the adaptive complexity of submodular optimization

Wenzheng Li, Paul Liu, Jan Vondrák

Stanford University

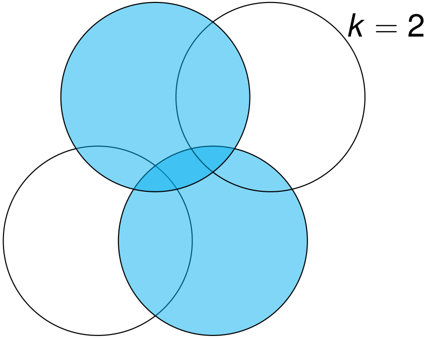
STOC 2020

Part I: Monotone Submodular Optimization

Part II: Non-monotone Submodular Optimization

Background

Our problem: $OPT := \max\{f(S) : |S| \leq k\}$
where f is *monotone* ($S \subset T \Rightarrow f(S) \leq f(T)$)
and *submodular* ($S \subset T \Rightarrow f(S+e) - f(S) \geq f(T+e) - f(T)$).

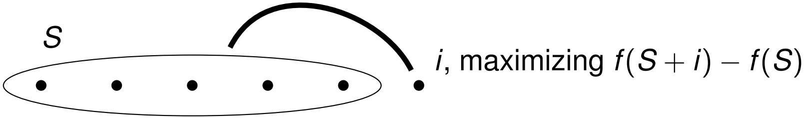


Coverage function (example):
Given $A_1, A_2, \ldots, A_n \subseteq U$, $f(S) = |\cup_{i \in S} A_i|$.

f is monotone submodular.

The Greedy Algorithm [Nemhauser-Wolsey-Fisher '78]

Pick elements one-by-one, maximizing the gain in $f(S)$, while maintaining $|S| \leq k$.

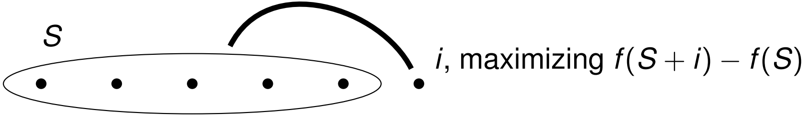


Theorem (Nemhauser-Wolsey-Fisher '78)

GREEDY *finds a solution of value at least* $(1 - 1/e)OPT$.

The Greedy Algorithm [Nemhauser-Wolsey-Fisher '78]

Pick elements one-by-one, maximizing the gain in $f(S)$, while maintaining $|S| \leq k$.



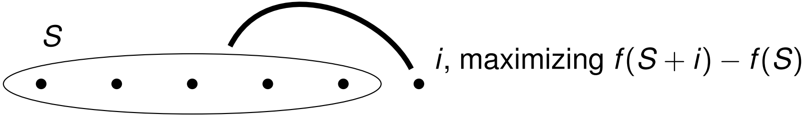
Theorem (Nemhauser-Wolsey-Fisher '78)

GREEDY *finds a solution of value at least* $(1 - 1/e)OPT$.

Optimality: [NW'78] *No algorithm using a polynomial number of queries to f can do better than* $(1 - 1/e)OPT$.

The Greedy Algorithm [Nemhauser-Wolsey-Fisher '78]

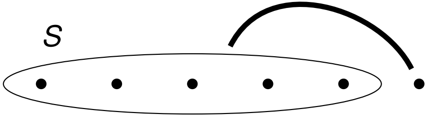
Pick elements one-by-one, maximizing the gain in $f(S)$, while maintaining $|S| \leq k$.



Long chain of k *sequentially dependent* queries.
Can we be more *parallel*?

The Greedy Algorithm [Nemhauser-Wolsey-Fisher '78]

Pick elements one-by-one, maximizing the gain in $f(S)$, while maintaining $|S| \leq k$.



i, maximizing $f(S+i) - f(S)$

Long chain of k *sequentially dependent* queries.
Can we be more *parallel*?

Yes!

The Adaptive Complexity Model

Adaptive Complexity Model [Balkanski-Singer ’18]:

- “Rounds" of polynomially many *parallel* queries.
- Compute cost is the length of the longest sequentially dependent chain.

The Adaptive Complexity Model

Adaptive Complexity Model [Balkanski-Singer '18]:

- "Rounds" of polynomially many *parallel* queries.
- Compute cost is the length of the longest sequentially dependent chain.

Theorem (Balkanski-Rubinstein-Singer '18)

A $(1 - 1/e - \epsilon)$-approximation to OPT can be achieved with $O(\frac{1}{\epsilon^2} \log n)$ rounds of queries.

Theorem (Balkanski-Singer '18)

$\Omega\left(\frac{\log n}{\log \log n}\right)$ *rounds of queries are necessary even for a $\frac{1}{\log n}$-approximation.*

Lower bounds for adaptive complexity

Must the number of rounds blow up as we approach the approximation factor of $1 - 1/e$?
(Recall: GREEDY achieves a clean $1 - 1/e$.)

Lower bounds for adaptive complexity

Must the number of rounds blow up as we approach the approximation factor of $1 - 1/e$? (Recall: GREEDY achieves a clean $1 - 1/e$.)

Yes!

Theorem (Our results, log rounds)

For any $\epsilon > \frac{1}{\log n}$, $\Omega(1/\epsilon)$ rounds are necessary to achieve a $(1 - 1/e - \epsilon)$-approximation to OPT.

Theorem (Our results, *poly* rounds)

For any $\epsilon > \frac{1}{n^c}$, $\Omega(1/\epsilon^{1/3})$ rounds are necessary to achieve a $(1 - 1/e - \epsilon)$-approximation to OPT.

Proof Ideas

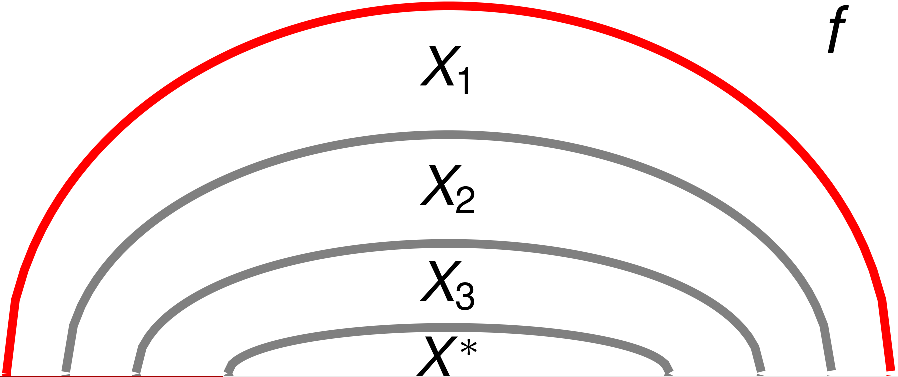
- The **onion-layer** construction inspired by [Balkanski-Singer '18].
- The **symmetry gap** construction [Vondrak '09], originated in [Feige-Mirrokni-V. '07].
- An improved hardness instances for $1 - 1/e$.

The onion layer

- f constructed from r layers $X_1, X_2, \ldots, X_r$, and a core layer $X^\star$.

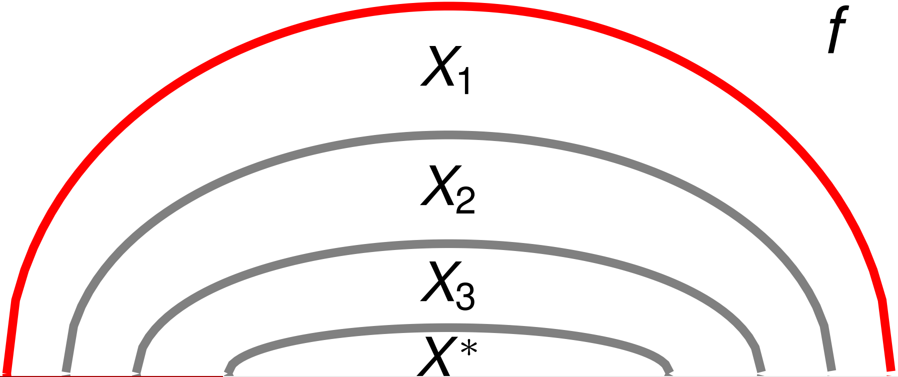
The onion layer

- f constructed from r layers $X_1, X_2, \ldots, X_r$, and a core layer $X^\star$.



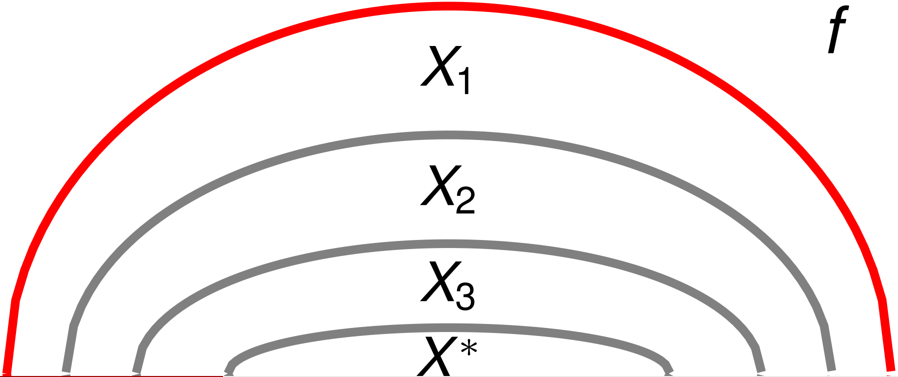
The onion layer

- The layers decrease *geometrically* in size (or slower).



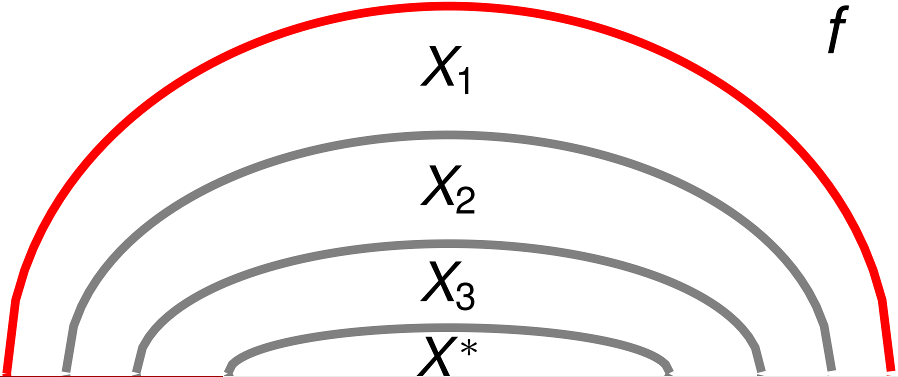
The onion layer

- The layers decrease *geometrically* in size (or slower).
- In the i-th round, no poly. # of queries on f can determine $X_{i+2}, \ldots, X_r, X^\star$.



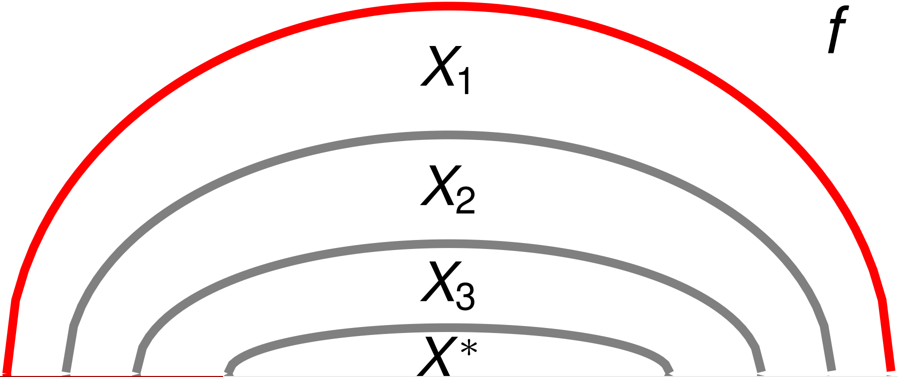
The onion layer

- The layers decrease *geometrically* in size (or slower).
- In the i-th round, no poly. # of queries on f can determine $X_{i+2}, \ldots, X_r, X^\star$.
- Given X_i, only polynomially many queries needed to determine X_{i+1}.



The onion layer

- The layers decrease *geometrically* in size (or slower).
- In the i-th round, no poly. # of queries on f can determine $X_{i+2}, \ldots, X_r, X^\star$.
- Given X_i, only polynomially many queries needed to determine X_{i+1}.
- $X^\star$ contains a $(1 - 1/e)$-hardness instance (thus stopping any algorithm's progress).



Onion layer construction

Let S be our query and $x_i = \frac{1}{k}|S \cap X_i|/|X_i|$ for layers $1, 2, \ldots, r$ and $x_0 = 0$.

Our function takes on the following form:

$$f(S) = 1 - (1 - g(S \cap X^\star)) \prod_{i=0}^{r-1} (1 - h(x_i, x_{i+1})) \text{ where } x_0 = 0.$$

- $x_0 = 0$ allows the first layer to be determined.

Onion layer construction

Let S be our query and $x_i = \frac{1}{k}|S \cap X_i|/|X_i|$ for layers $1, 2, \ldots, r$ and $x_0 = 0$.

Our function takes on the following form:

$$f(S) = 1 - (1 - g(S \cap X^\star)) \prod_{i=0}^{r-1} (1 - h(x_i, x_{i+1})) \text{ where } x_0 = 0.$$

- $x_0 = 0$ allows the first layer to be determined.
- $h(x_{k-1}, x_k)$ does not reveal anything about X_{k-1}, X_k unless we know either x_{k-1} or x_k.

Onion layer construction

Let S be our query and $x_i = \frac{1}{k}|S \cap X_i|/|X_i|$ for layers $1, 2, \ldots, r$ and $x_0 = 0$.

Our function takes on the following form:

$$f(S) = 1 - (1 - g(S \cap X^{\star})) \prod_{i=0}^{r-1} (1 - h(x_i, x_{i+1})) \text{ where } x_0 = 0.$$

- $x_0 = 0$ allows the first layer to be determined.
- $h(x_{k-1}, x_k)$ does not reveal anything about X_{k-1}, X_k unless we know either x_{k-1} or x_k.
- $g(S \cap X^{\star})$ encodes a $(1 - 1/e + o(1))$-hard instance on X_k.

Onion layer construction

Let S be our query and $x_i = \frac{1}{k}|S \cap X_i|/|X_i|$ for layers $1, 2, \ldots, r$ and $x_0 = 0$.

Our function takes on the following form:

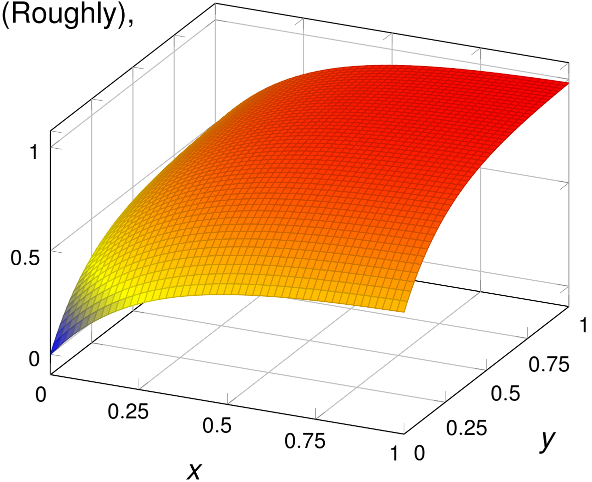
$$f(S) = 1 - (1 - g(S \cap X^\star)) \prod_{i=0}^{r-1} (1 - h(x_i, x_{i+1})) \text{ where } x_0 = 0.$$

- $x_0 = 0$ allows the first layer to be determined.
- $h(x_{k-1}, x_k)$ does not reveal anything about X_{k-1}, X_k unless we know either x_{k-1} or x_k.
- $g(S \cap X^\star)$ encodes a $(1 - 1/e + o(1))$-hard instance on X_k.

$f(S) \approx g(S \cap X^\star)$ best we can do when all parts are “known".

What does h look like?

(Roughly),



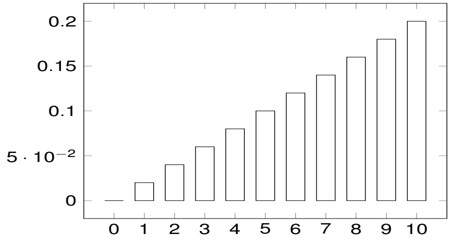
$$h(x, y) = 1 - \frac{1}{2}(e^{-x} + e^{-y})$$

When $x_i \approx x_{i+1}$, $h(x_i, x_{i+1}) \approx 1 - e^{-x_i - x_{i+1}}$, and $f \approx 1 - (1 - g(S)) \exp\left(-\sum_i x_i\right)$ so **none of the X_i can be distinguished. For random S, $x_i \approx x_{i+1}$ for all $i > 1$.**

Analysis

$$h(x, y) = 1 - \frac{1}{2}(e^{-x} + e^{-y}).$$

Solutions where $x_i = x_{i+1}$ are more profitable than those where $x_i \neq x_{i+1}$; $penalty = \Theta((x_i - x_{i+1})^2)$.

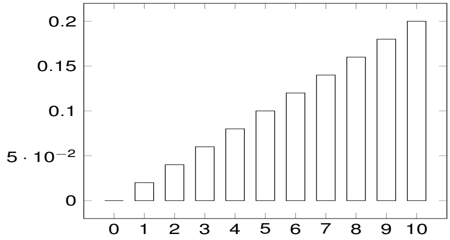


$$penalty = \sum_{i=0}^{k-1}(x_{i+1} - x_i)^2 = \Theta\left(\frac{1}{k^3}\right)$$

Analysis

$$h(x, y) = 1 - \frac{1}{2}(e^{-x} + e^{-y}).$$

Solutions where $x_i = x_{i+1}$ are more profitable than those where $x_i \neq x_{i+1}$;
penalty $= \Theta((x_i - x_{i+1})^2)$.



$$penalty = \sum_{i=0}^{k-1}(x_{i+1} - x_i)^2 = \Theta\left(\frac{1}{k^3}\right)$$

Since $x_0 = 0$, the initial penalty makes the algorithm start below $1 - 1/e$.
Given $k - 1$ rounds, the optimal assignment of variables is $x_i \approx O(i/k^2)$.

Analysis cont.

- Best approx. in k rounds is $1 - 1/e + o(1) - \Omega(1/k^3)$.
- $o(1)$ term from hardness instance on $X^\star$. Previously [Vondrak '09] achieved $o(1) = O\left(\frac{1}{\log(n)}\right)$.
- We need $o(1) = O\left(\frac{1}{poly(n)}\right)$ if $k = O(poly(n))$ (done via a new hardness instance using techniques from [Vondrak '13].)

Part I: Monotone Submodular Optimization

Part II: Non-monotone Submodular Optimization

Switching gears - non-monotone optimization

Our problem: $OPT := \max f(S)$ where f is *submodular*, *non-monotone*, and *unconstrained*.

- A random set R is known to get $OPT/4$ in expectation.

Theorem (Buchbinder-Feldman-Naor-Schwartz '12)

A $1/2$*-approximation can be obtained by the* DOUBLEGREEDY *algorithm in the sequential model.*

Optimality: [Feige-Mirrokni-V. '07] *No algorithm can get better than a* $1/2$*-approximation in a polynomial number of queries.*

Switching gears - non-monotone optimization

Our problem: $OPT := \max f(S)$ where f is *submodular*, *non-monotone*, and *unconstrained*.

- A random set R is known to get $OPT/4$ in expectation.

Theorem (Buchbinder-Feldman-Naor-Schwartz '12)

A $1/2$-approximation can be obtained by the DoubleGreedy *algorithm in the sequential model.*

Optimality: [Feige-Mirrokni-V. '07] *No algorithm can get better than a $1/2$-approximation in a polynomial number of queries.*

Theorem (Chen-Feldman-Karbasi '18, Ene-Nguyen-Vladu '18)

A $(1/2-\epsilon)$-approximation to OPT can be achieved with $O(\frac{1}{\epsilon})$ rounds of queries.

(Through a variant of the double greedy algorithm.)

Hardness for non-monotone maximization?

Recall, our lower bound in the monotone case:

- Started greater than ϵOPT away from $(1 - 1/e)OPT$.
- Never exceeded $(1 - 1/e + o(1))OPT$ even after all its rounds were completed.

Are there similar hardness results in the unconstrained case?

Hardness for non-monotone maximization?

Recall, our lower bound in the monotone case:

- Started greater than ϵOPT away from $(1 - 1/e)OPT$.
- Never exceeded $(1 - 1/e + o(1))OPT$ even after all its rounds were completed.

Are there similar hardness results in the unconstrained case?

No!

Improved non-monotone maximization

Theorem (Our results)

Let R be a uniformly random subset. If $\mathbf{E}[f(R)] \leq (1/2 - \delta)OPT$, then adaptive double greedy achieves value at least $(1/2 + \Omega(\delta^2))OPT$ in $O(1/\delta^2)$ rounds.

$\implies$ Either a random set is already close to $OPT/2$, or the double greedy finds a solution much better than $OPT/2$.

Intuition and Analysis

Continuous double greedy (f is the multilinear extension of the objective)

$\mathbf{x}(0), \mathbf{y}(0) = \mathbf{0}, \mathbf{1}$
While $\mathbf{x}(t) \neq \mathbf{y}(t)$:

- $\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{\nabla f(x)_+}{\nabla f(x)_+ - \nabla f(y)_-}$
- $\frac{\mathrm{d}y}{\mathrm{d}t} = \frac{\nabla f(y)_-}{\nabla f(x)_+ - \nabla f(y)_-}$

Return $\mathbf{x} = \mathbf{y}$ as the solution (ignoring some edge cases).

Intuition and Analysis

Continuous double greedy (f is the multilinear extension of the objective)

$\mathbf{x}(0), \mathbf{y}(0) = \mathbf{0}, \mathbf{1}$
While $\mathbf{x}(t) \neq \mathbf{y}(t)$:

- $\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{\nabla f(x)_+}{\nabla f(x)_+ - \nabla f(y)_-}$
- $\frac{\mathrm{d}y}{\mathrm{d}t} = \frac{\nabla f(y)_-}{\nabla f(x)_+ - \nabla f(y)_-}$

Return $\mathbf{x} = \mathbf{y}$ as the solution (ignoring some edge cases).

Theorem (Ene-Nguyen-Vladu '18)

The returned solution $DG = f(\mathbf{x})$ satisfies

$$DG \geq \frac{OPT}{2} + \frac{1}{4} \int_0^1 \sum_i \frac{(\nabla_i f(\mathbf{x}(t))_+ + \nabla_i f(\mathbf{y}(t))_-)^2}{\nabla_i f(\mathbf{x}(t))_+ - \nabla_i f(\mathbf{y}(t))_-} \mathrm{d}t.$$

Intuition and Analysis cont.

Theorem (Ene-Nguyen-Vladu '18)

The returned solution $DG = f(\mathbf{x})$ satisfies

$$DG \geq \frac{OPT}{2} + \frac{1}{4} \int_0^1 \sum_i \frac{(\nabla_i f(\mathbf{x}(t))_+ + \nabla_i f(\mathbf{y}(t))_-)^2}{\nabla_i f(\mathbf{x}(t))_+ - \nabla_i f(\mathbf{y}(t))_-} \mathrm{d}t.$$

Intuition and Analysis cont.

Theorem (Ene-Nguyen-Vladu '18)

The returned solution $DG = f(\mathbf{x})$ satisfies

$$DG \geq \frac{OPT}{2} + \frac{1}{4}\int_0^1 \sum_i \frac{(\nabla_i f(\mathbf{x}(t))_+ + \nabla_i f(\mathbf{y}(t))_-)^2}{\nabla_i f(\mathbf{x}(t))_+ - \nabla_i f(\mathbf{y}(t))_-} \mathrm{d}t.$$

Lemma

Let R be a uniformly random subset.

$$DG - f(R) = \frac{1}{2}\int_0^1 \sum_i |\nabla_i f(\mathbf{x})_+ + \nabla_i f(\mathbf{y})_-| \, \mathrm{d}t.$$

Judious applications of Cauchy-Scwartz gets our main result.

Open problems

- Does there exist a lower bound for non-monotone optimization?
- Can we improve the $1/\delta^2$ to $1/\delta$ for non-monotone?
- Can we extend the construction of the monotone case smoothly for all ϵ?