Learning and Testing Submodular Functions

Grigory Yaroslavtsev

With Sofya Raskhodnikova (SODA'13)

PENNSTATE

1855

+ Work in progress with Rocco Servedio

Columbia University

October 26, 2012

Submodularity

- Discrete analog of convexity/concavity, law of diminishing returns
- Applications: optimization, algorithmic game theory

Let $\boldsymbol{f}: 2^X \to [0, R]$:

- **Discrete derivative:**

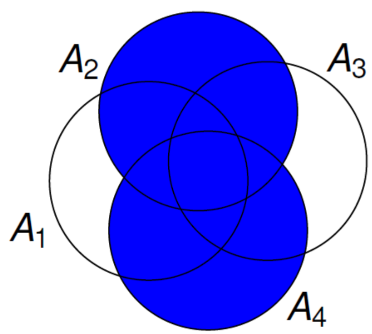
$$\partial_x \boldsymbol{f}(S) = \boldsymbol{f}(S \cup \{x\}) - \boldsymbol{f}(S), \qquad for\ S \subseteq X, x \notin S$$

- **Submodular function:**

$$\partial_x f(S) \geq \partial_x f(T), \quad \forall\, S \subseteq T \subseteq X, x \notin T$$

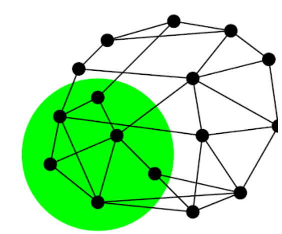
Coverage function:
Given $A_1, \ldots, A_n \subset U$,

$$f(S) = \left| \bigcup_{j \in S} A_j \right|.$$



Cut function:

$$\delta(T) = |e(T, \overline{T})|$$



Exact learning

- **Q:** Reconstruct a submodular $\boldsymbol{f}: 2^X \to R$ with poly(|X|) queries (for all arguments)?
- **A:** Only $\widetilde{\Theta}\left(\sqrt{|X|}\right)$-approximation (multiplicative) possible [Goemans, Harvey, Iwata, Mirrokni, SODA'09]
- **Q:** Only for $(1-\epsilon)$-fraction of points (PAC-style learning **with membership queries** under uniform distribution)?

$$\Pr_{randomness\ of\ \boldsymbol{A}}\left[\Pr_{\boldsymbol{S}\sim U(2^X)}[\boldsymbol{A}(\boldsymbol{S}) = \boldsymbol{f}(\boldsymbol{S})] \geq 1-\epsilon\right] \geq \frac{1}{2}$$

- **A:** Almost as hard [Balcan, Harvey, STOC'11].

Approximate learning

- **PMAC**-learning (**M**ultiplicative), with poly(|X|) queries :

$$\Pr_{randomness\ of\ A}\left[\Pr_{S\sim U(2^X)}[\,f(S)\le A(S)\le \alpha f(S)]\ge 1-\epsilon\right]\ge \frac{1}{2}$$

$$\Omega\left(|X|^{\frac{1}{3}}\right)\le \alpha \le O\left(\sqrt{|X|}\right)$$ (over arbitrary distributions [BH'11])

- **PAAC**-learning (**A**dditive)

$$\Pr_{randomness\ of\ A}\left[\Pr_{S\sim U(2^X)}[\,|f(S)\ -\ A(S)|\le \beta]\ge 1-\epsilon\right]\ge \frac{1}{2}$$

  - Running time: $|X|^{O\left(\frac{|R|}{\beta}\right)^2 \log(\frac{1}{\epsilon})}$ [Gupta, Hardt, Roth, Ullman, STOC'11]
  - Running time: $\mathrm{poly}\left(|X|^{\left(\frac{|R|}{\beta}\right)^2},\ \log\frac{1}{\epsilon}\right)$ [Cheraghchi, Klivans, Kothari, Lee, SODA'12]

Learning $f: 2^X \to [0, R]$

- For all algorithms $\epsilon = const.$

	Goemans, Harvey, Iwata, Mirrokni	Balcan, Harvey	Gupta, Hardt, Roth, Ullman	Cheraghchi, Klivans, Kothari, Lee	Our result with Sofya
Learning	$\tilde{O}\left(\sqrt{\lvert X \rvert}\right)$-approximation Everywhere	P**M**AC **Multiplicative** $\boldsymbol{\alpha}$ $\boldsymbol{\alpha} = O\left(\sqrt{\lvert X \rvert}\right)$	P**A**AC **Additive** $\boldsymbol{\beta}$		**PAC** $\boldsymbol{f: 2^X \to \{0, \ldots, R\}}$ (bounded integral range $R \leq \lvert X \rvert$)
Time	Poly(|X|)	Poly(|X|)	$\lvert X \rvert^{O\left(\frac{\lvert R \rvert}{\beta}\right)^2}$		$\lvert X \rvert^3 \, \lvert R \rvert^{O(\lvert R \rvert \cdot \log \lvert R \rvert)}$
Extra features		Under arbitrary distribution	Tolerant queries	SQ-queries, Agnostic	Agnostic

Learning: Bigger picture

Subadditive

$\supseteq$

XOS = Fractionally subadditive

$\supseteq$

Submodular

$\supseteq$

Gross substitutes

$\supseteq$

OXS

$\supseteq$ $\supseteq$

Additive (linear) | Value demand

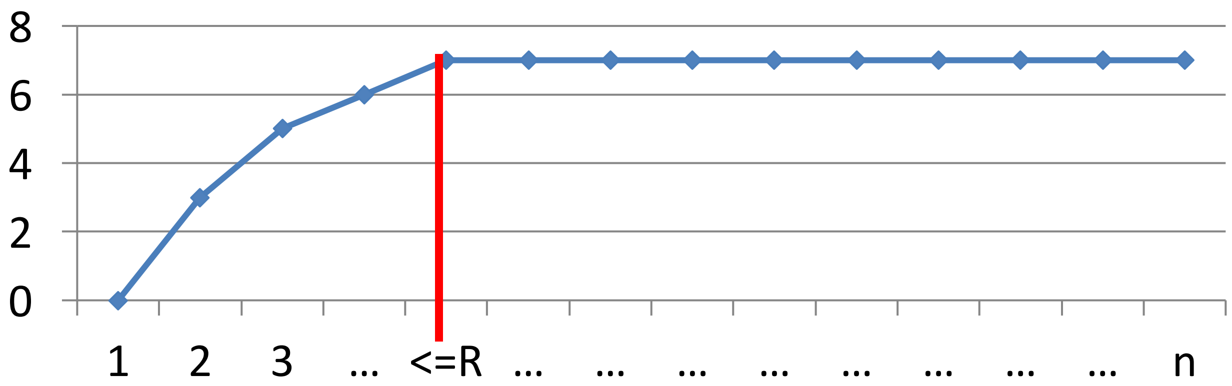
} [Badanidiyuru, Dobzinski, Fu, Kleinberg, Nisan, Roughgarden, SODA'12]

Other positive results:

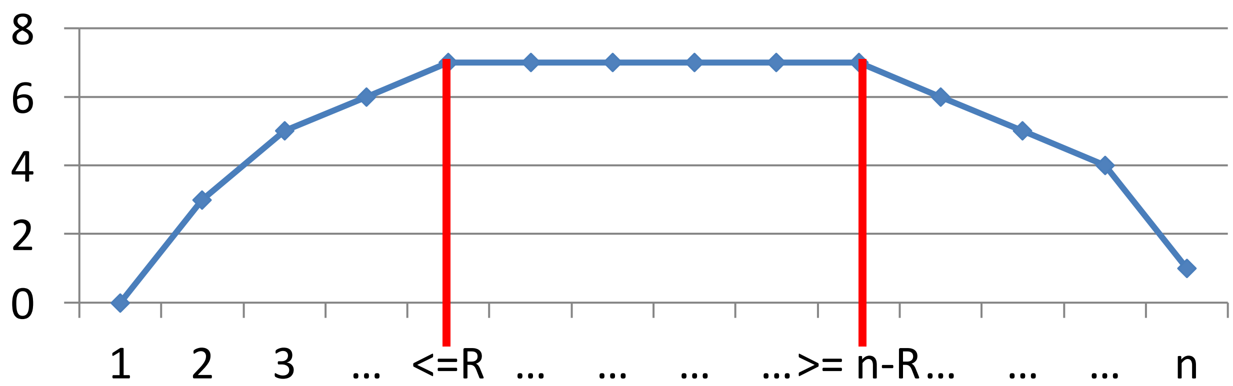
- Learning valuation functions [Balcan, Constantin, Iwata, Wang, COLT'12]
- PMAC-learning (sketching) valuation functions [BDFKNR'12]
- PMAC learning Lipschitz submodular functions [BH'10] (concentration around average via Talagrand)

Discrete convexity

- Monotone convex $f: \{1, \ldots, n\} \rightarrow \{0, \ldots, R\}$



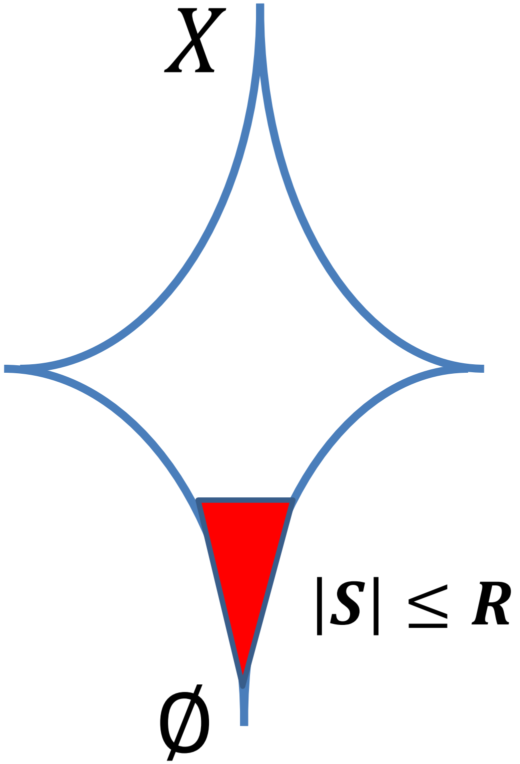
- Convex $f: \{1, \ldots, n\} \rightarrow \{0, \ldots, R\}$



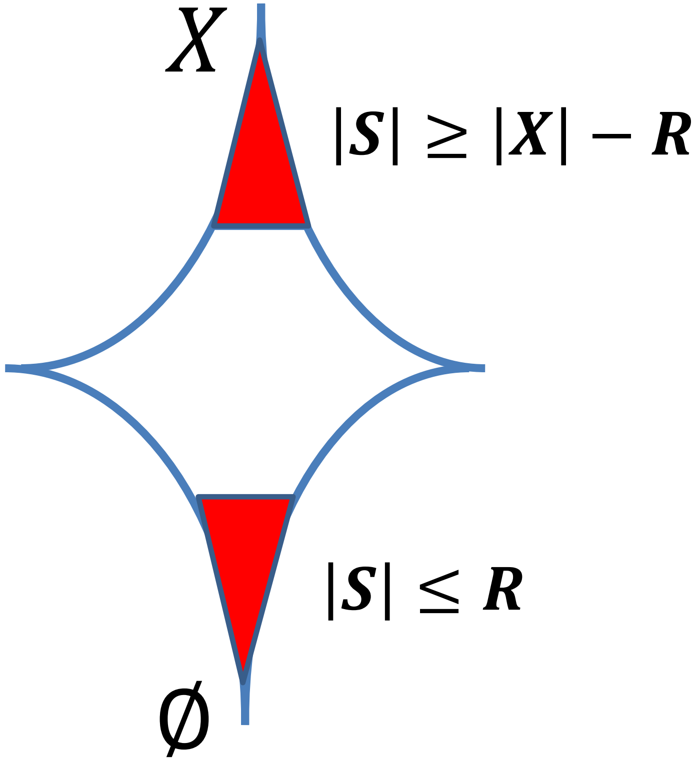
Discrete submodularity $f: 2^X \to \{0, \ldots, R\}$

- **Case study**: R = 1 (Boolean submodular functions $f: \{0,1\}^n \to \{0,1\}$)
  Monotone submodular = $x_{i_1} \vee x_{i_2} \vee \cdots \vee x_{i_a}$ (monomial)
  Submodular = $(x_{i_1} \vee \cdots \vee x_{i_a}) \wedge (\overline{x_{j_1}} \vee \cdots \vee \overline{x_{j_b}})$ (2-term CNF)

- Monotone submodular

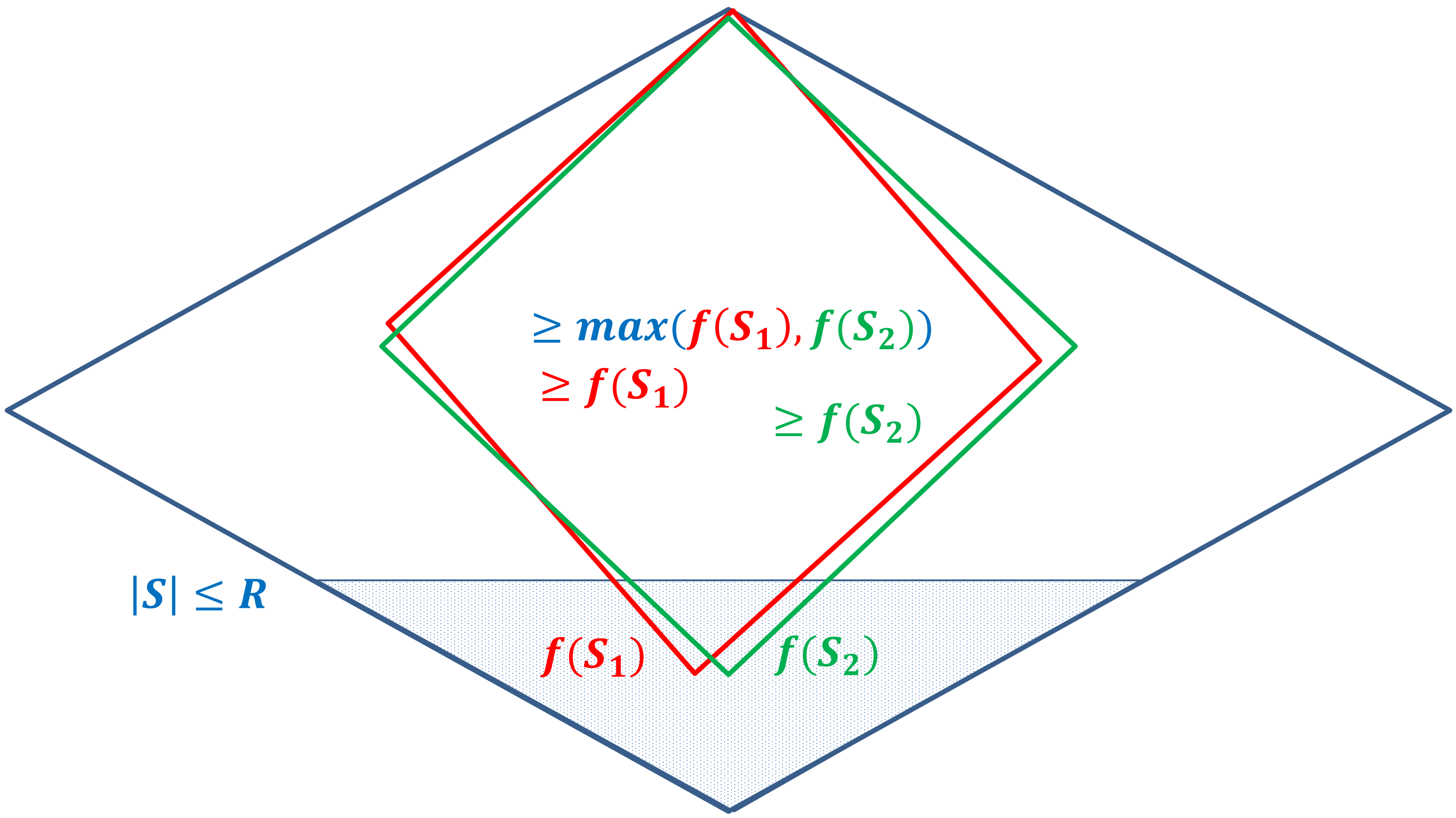


- Submodular



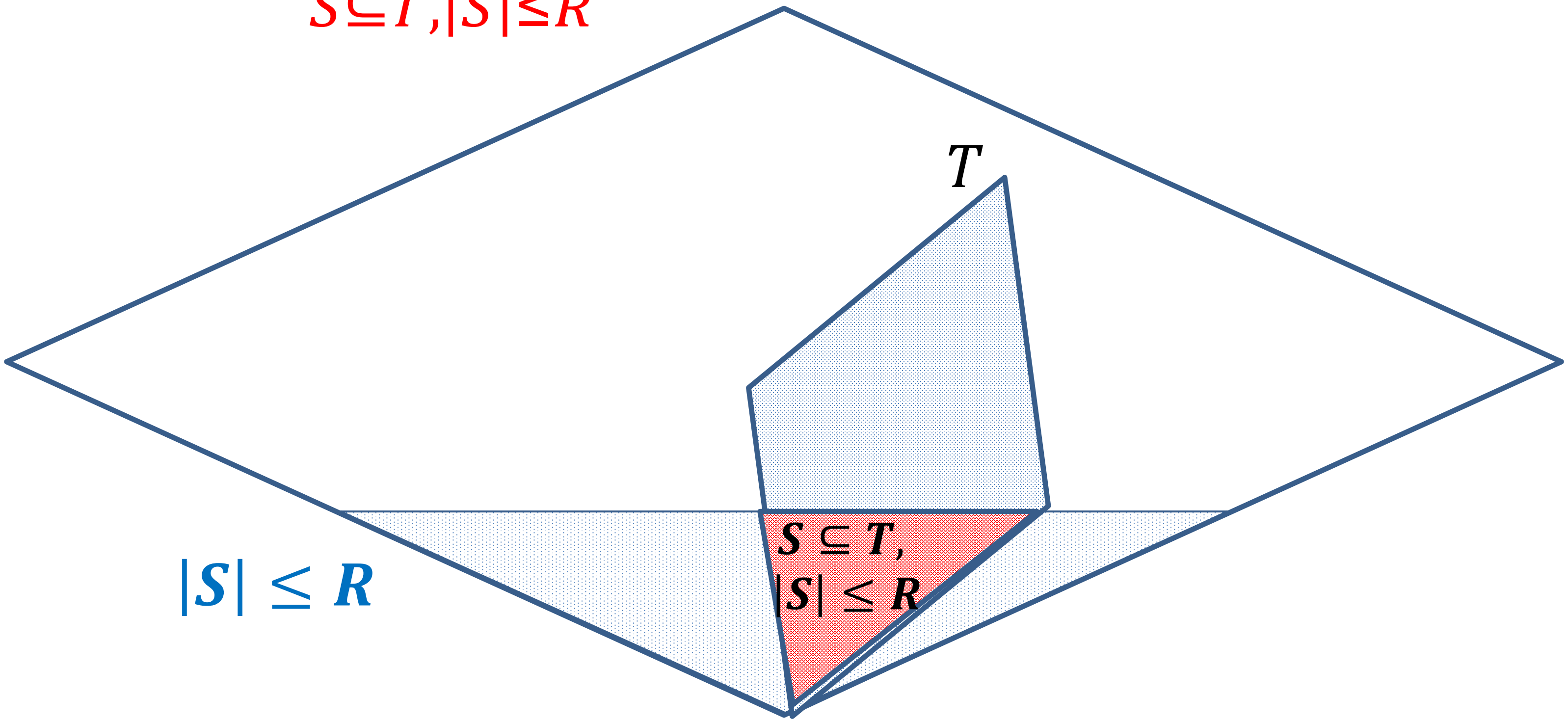
Discrete monotone submodularity

- Monotone submodular $f: 2^X \to \{0, \dots, R\}$



Discrete monotone submodularity

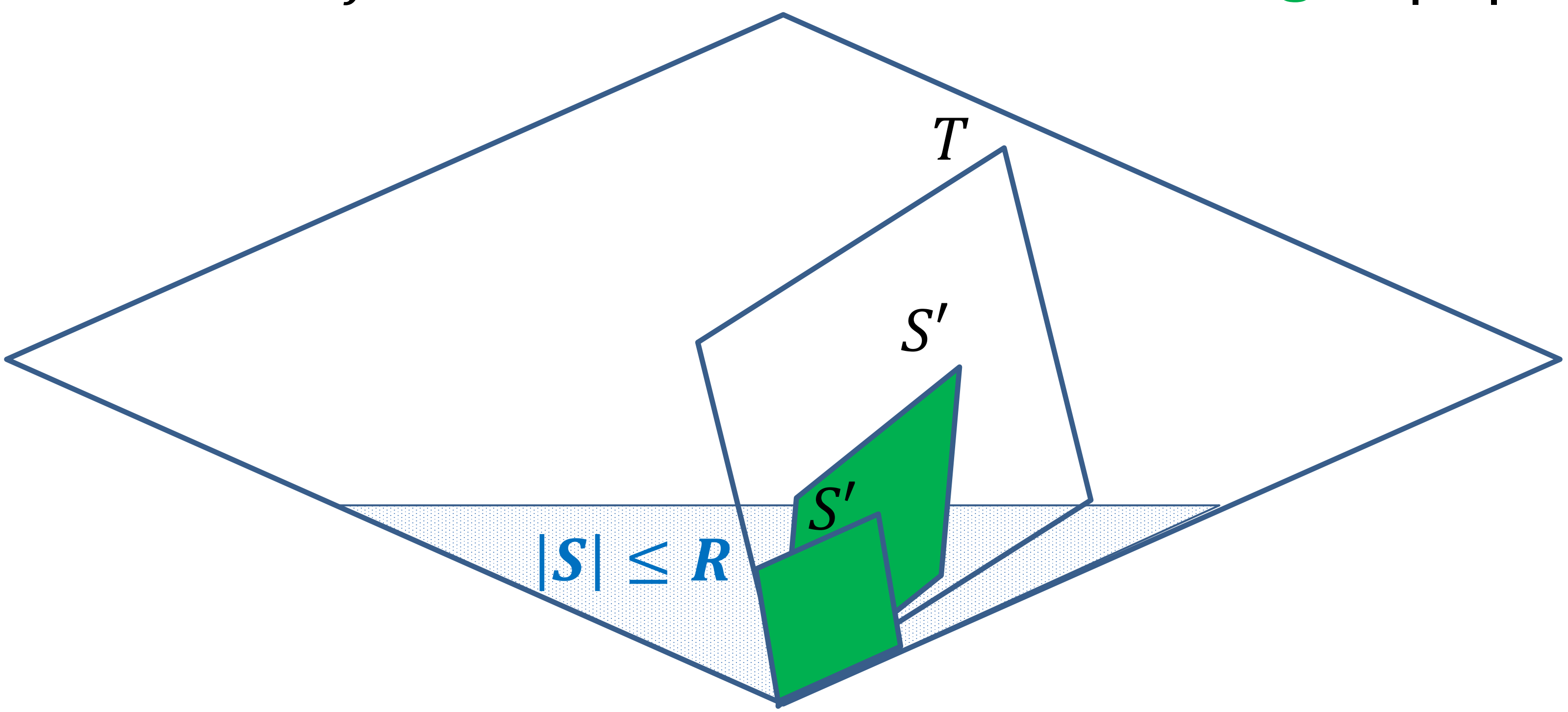
- **Theorem**: for **monotone** submodular $f: 2^X \to \{0, \ldots, R\}$ for all T: $f(T) = \max_{S \subseteq T, |S| \leq R} f(S)$
- $f(T) \geq \max_{S \subseteq T, |S| \leq R} f(S)$ (by monotonicity)



Discrete monotone submodularity

- $f(T) \leq \max_{S \subseteq T, |S| \leq R} f(S)$
- S' $=$ **smallest** subset of T such that $f(T) = f(S')$
- $\forall x \in S'$ we have $\partial_x(S' \setminus \{x\}) > 0$ =>

Restriction of f on $2^{S'}$ is **monotone increasing** =>$|S'| \leq R$



Representation by a formula

- **Theorem**: for **monotone** submodular $f: 2^X \to \{0, \dots, R\}$ for all T:

$$f(T) = \max_{S \subseteq T, |S| \le R} f(S)$$

- Notation switch: $|X| \to n$, $2^X \to (x_1, \dots, x_n)$

- **(Monotone) Pseudo–Boolean k–DNF**

$$(\vee \to \boldsymbol{max}, A_i = 1 \to \boldsymbol{A_i} \in \mathbb{R}):$$

$$\boldsymbol{max}_i \left[\boldsymbol{A_i} \cdot \left(x_{i_1} \wedge \overline{x_{i_2}} \wedge \dots \wedge x_{i_k}\right)\right]$$ (**no negations**)

- **(Monotone)** submodular $f(x_1, \dots, x_n) \to \{0, \dots, \boldsymbol{R}\}$ can be represented as a (**monotone**) pseudo-Boolean **2R**-DNF with constants $A_i \in \{0, \dots, \boldsymbol{R}\}$.

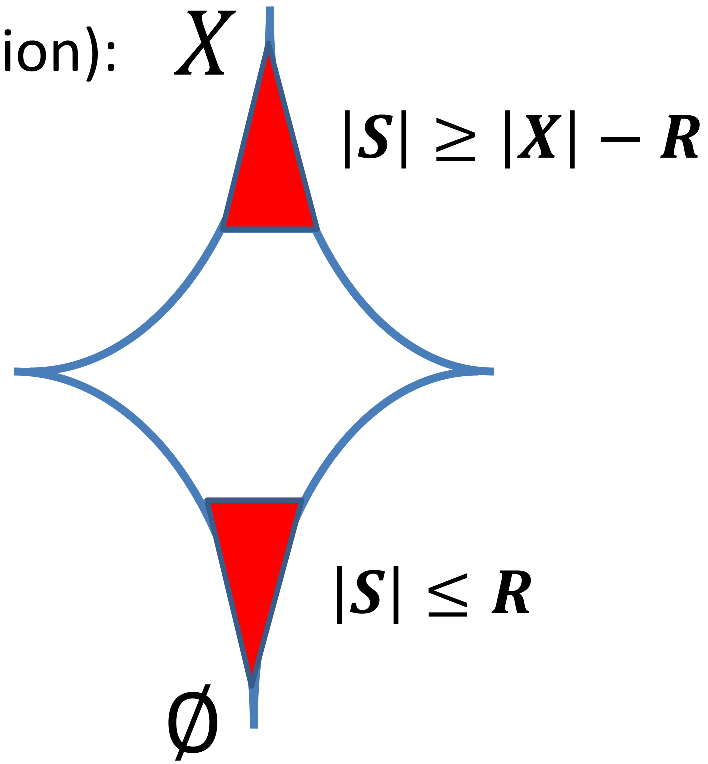
Discrete submodularity

- Submodular $f(x_1, \ldots, x_n) \rightarrow \{0, \ldots, \boldsymbol{R}\}$ can be represented as a pseudo-Boolean **2R**-DNF with constants $A_i \in \{0, \ldots, \boldsymbol{R}\}$.
- Hint [Lovasz] (Submodular monotonization):

Given submodular $\boldsymbol{f}$, define

$$\boldsymbol{f^{mon}(S) = max_{T \subseteq S}\, f(T)}$$

Then $\boldsymbol{f^{mon}}$ is monotone and submodular.



Learning pB-formulas and k-DNF

- $DNF^{\boldsymbol{k},\boldsymbol{R}}$ = class of pB-DNF of width $\boldsymbol{k}$ with $A_i \in \{0, \ldots, \boldsymbol{R}\}$
- **i-slice** $\boldsymbol{f}_{\boldsymbol{i}}(x_1, \ldots, x_n) \rightarrow \{0,1\}$ defined as

$$\boldsymbol{f}_{\boldsymbol{i}}(x_1, \ldots, x_n) = 1 \quad \textbf{iff} \quad \boldsymbol{f}(x_1, \ldots, x_n) \geq i$$

- If $\boldsymbol{f} \in DNF^{\boldsymbol{k},\boldsymbol{R}}$ its **i-slices** $\boldsymbol{f}_{\boldsymbol{i}}$ are $\boldsymbol{k}$-DNF and:

$$\boldsymbol{f}(x_1, \ldots, x_n) = \max_{1 \leq i \leq \boldsymbol{R}} \left(i \cdot \boldsymbol{f}_{\boldsymbol{i}}(x_1, \ldots, x_n)\right)$$

- PAC-learning

$$\Pr_{rand(\boldsymbol{A})}\left[\Pr_{\boldsymbol{S} \sim U(\{0,1\}^n)}[\boldsymbol{A}(\boldsymbol{S}) = \boldsymbol{f}(\boldsymbol{S})] \geq 1 - \epsilon\right] \geq \frac{1}{2}$$

Learning pB-formulas and k-DNF

- Learn every **i-slice** $\boldsymbol{f_i}$ on $1-\epsilon' = (1-\epsilon\ /\ R)$ fraction of arguments
- Learning $\boldsymbol{k}$-DNF ($DNF^{\boldsymbol{k},\boldsymbol{R}}$) (let Fourier sparsity $\boldsymbol{S_F}$= $\boldsymbol{k}^{\boldsymbol{k} \log\left(\frac{\boldsymbol{R}}{\epsilon}\right)}$)
  - Kushilevitz-Mansour (Goldreich-Levin): $poly(n, \boldsymbol{S_F})$ queries/time.
  - ``Attribute efficient learning'': $polylog(n) \cdot poly(\boldsymbol{S_F})$ queries
  - Lower bound: $\Omega(2^{\boldsymbol{k}})$ queries to learn a random $\boldsymbol{k}$-junta ($\in$ $\boldsymbol{k}$-DNF) up to constant precision.
- Optimizations:
  - Slightly better than KM/GL by looking at the Fourier spectrum of $DNF^{\boldsymbol{k},\boldsymbol{R}}$ (see SODA paper: switching lemma => L_1 bound)
  - Do all **R** iterations of KM/GL in parallel by reusing queries

Property testing

- Let **C** be the class of submodular $\boldsymbol{f}: \{0,1\}^n \to \{0, \dots, \boldsymbol{R}\}$
- How to (approximately) test, whether a given $\boldsymbol{f}$ is in **C**?
- Property tester: (Randomized) algorithm for distinguishing:
  1. $f \in \boldsymbol{C}$
  2. ($\boldsymbol{\epsilon}$-far): $\min_{g \in \boldsymbol{C}} |\boldsymbol{f} - \boldsymbol{g}| \geq \boldsymbol{\epsilon}\, 2^n$

- Key idea: $\boldsymbol{k}$-DNFs have small representations:
  - [Gopalan, Meka,Reingold CCC'12] (using quasi-sunflowers [Rossman'10])

  $\forall \epsilon > 0, \forall$ $\boldsymbol{k}$-DNF formula F there exists:

  $\boldsymbol{k}$-DNF formula F' of size $\leq \left(\boldsymbol{k} \log \frac{1}{\boldsymbol{\epsilon}}\right)^{O(\boldsymbol{k})}$ such that $|F - F'| \leq \boldsymbol{\epsilon} 2^n$

Testing by implicit learning

- **Good approximation by juntas => efficient property testing** [surveys: Ron; Servedio]
  - ϵ-approximation by $J(\epsilon)$-junta
  - Good dependence on $\boldsymbol{\epsilon}$: $J(\epsilon) = \left(\boldsymbol{k} \log \frac{1}{\boldsymbol{\epsilon}}\right)^{O(\boldsymbol{k})}$
    - [Blais, Onak] sunflowers for submodular functions $[O\left(\boldsymbol{k} \log \boldsymbol{k} + \log \frac{1}{\boldsymbol{\epsilon}}\right)]^{(\boldsymbol{k}+\mathbf{1})}$
  - Query complexity: $\left(\boldsymbol{k} \log \frac{1}{\boldsymbol{\epsilon}}\right)^{\tilde{O}(\boldsymbol{k})}$ (independent of **n**)
  - Running time: exponential in $J(\epsilon)$ (we think can be reduced it to O($J(\epsilon)$))
  - We have $\Omega(\boldsymbol{k})$ lower bound for testing $\boldsymbol{k}$-DNF (reduction from Gap Set Intersection: distinguishing a random $\boldsymbol{k}$-junta vs $\boldsymbol{k}$ + O(log $\boldsymbol{k}$)-junta requires $\Omega(\boldsymbol{k})$ queries)

Previous work on testing submodularity

$\boldsymbol{f}: \{0,1\}^n \to [0, R]$ [Parnas, Ron, Rubinfeld ‘03, Seshadhri, Vondrak, ICS’11]:

- Upper bound $(\mathbf{1}/\boldsymbol{\epsilon})^{\boldsymbol{O}(\sqrt{\boldsymbol{n}})}$.
- Lower bound: $\boldsymbol{\Omega}(\boldsymbol{n})$

} Gap in query complexity

Special case: coverage functions [Chakrabarty, Huang, ICALP’12].

Thanks!