Minimization

Maximization.

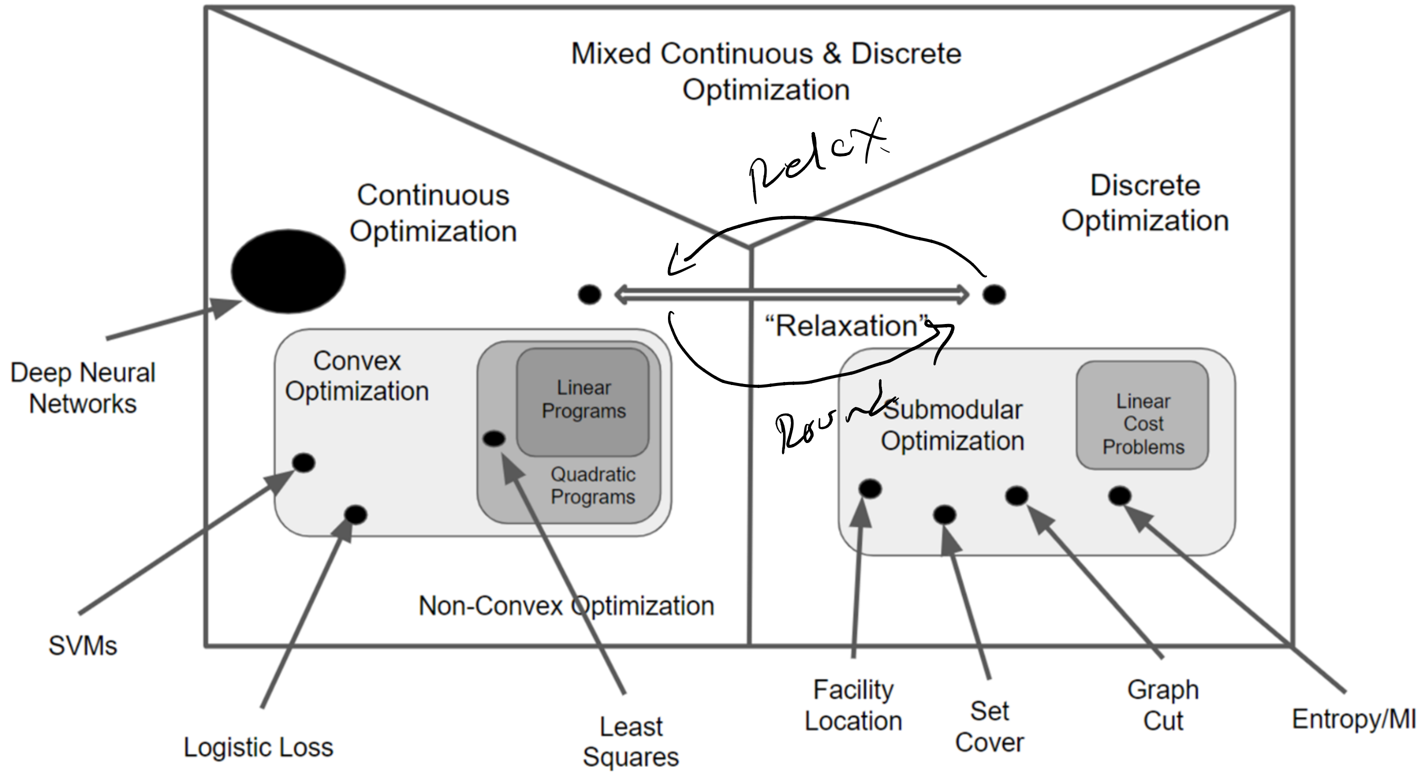
Submodular Function Optimization

Lecture 8

Advanced Topics in Optimization For Machine Learning (CS 7301)

Instructor: Rishabh Iyer

Big Picture: Continuous and Discrete Optimization



Acknowledgements

Slides borrowed from several sources:

1. Submodular Optimization course at UW from Jeff Bilmes
2. Tutorial on Submodular Optimization by Stefanie Jegelka, Andreas Krause and Jeff Bilmes at ICML and NIPS
3. Some of my own tutorials at WACV, IJCAI, ECAI, ICPM etc.

Useful Material

- Fujishige, “Submodular Functions and Optimization”, 2005
- Narayanan, “Submodular Functions and Electrical Networks”, 1997
- Welsh, “Matroid Theory”, 1975
- Oxley, “Matroid Theory”, 1992 (and 2011).
- Lawler, “Combinatorial Optimization: Networks and Matroids”, 1976.
- Schrijver, “Combinatorial Optimization”, 2003
- Gruenbaum, “Convex Polytopes, 2nd Ed”, 2003.
- Additional readings that will be announced here.

"Bach, Francis 2013 Monograph" $\rightarrow$ Submod Min & Convexity

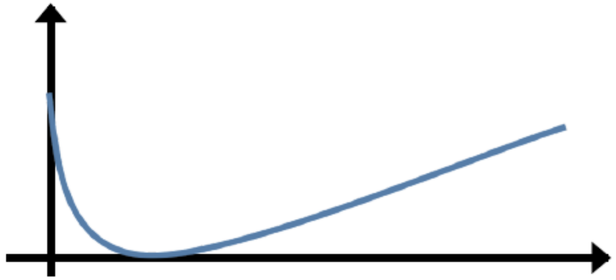
Andreas Krause 2014 $\longrightarrow$ Submod Maximization

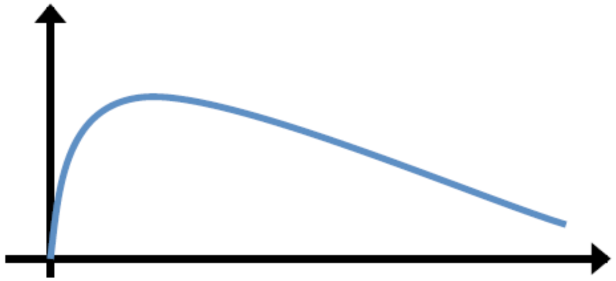
Useful material

- Jeff's Class: http://j.ee.washington.edu/~bilmes/classes/ee596a_fall_2014/
- Stefanie Jegelka & Andreas Krause's 2013 ICML tutorial: http://techtalks.tv/talks/submodularity-in-machine-learning-new-directions-part-i/58125/
- Jeff's NIPS, 2013 tutorial on submodularity: http://melodi.ee.washington.edu/~bilmes/pgs/b2hd-bilmes2013-nips-tutorial.html and http://youtu.be/c4rBof38nKQ
- Andreas Krause's web page: http://submodularity.org
- Francis Bach's updated 2013 text: http://hal.archives-ouvertes.fr/docs/00/87/06/09/PDF/submodular_fot_revisd_hal.pdf
- My WACV 2019 Tutorial: https://sites.google.com/view/wacv2019summarization/home
- Tom McCormick's overview paper on submodular minimization: http://people.commerce.ubc.ca/faculty/mccormick/sfmchap8a.pdf

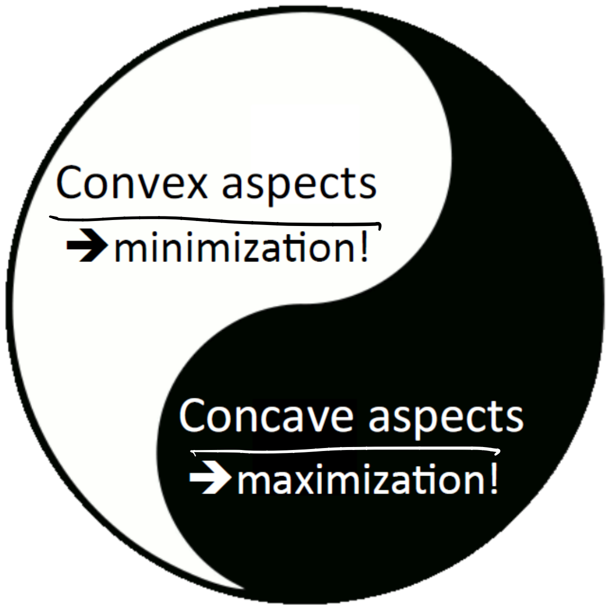
Submodularity, Convexity & Concavity

discrete convexity ….





… or concavity?



Submodularity is related to both convexity and concavity!

2015: Polyhedral aspects of Submodularity, Convexity & Concavity

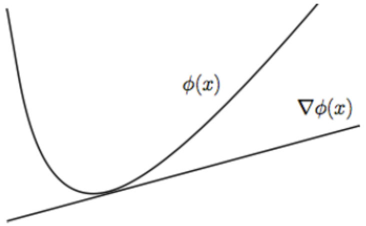
2020: Concave Aspects of Submodular Fns: ISIT 2020

Submodularity, Convexity & Concavity

Edmonds 1983

Convex aspects (Fujishige (1984, 2005), Frank (1982))

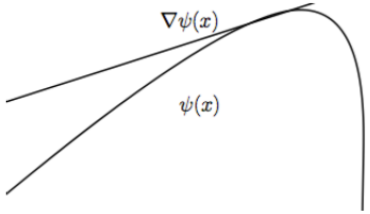
- Minimization: Poly-time.
- Convex continuous extension - Lovász extension.
- Subgradients and Subdifferential.
- Convex duality, discrete seperation etc.

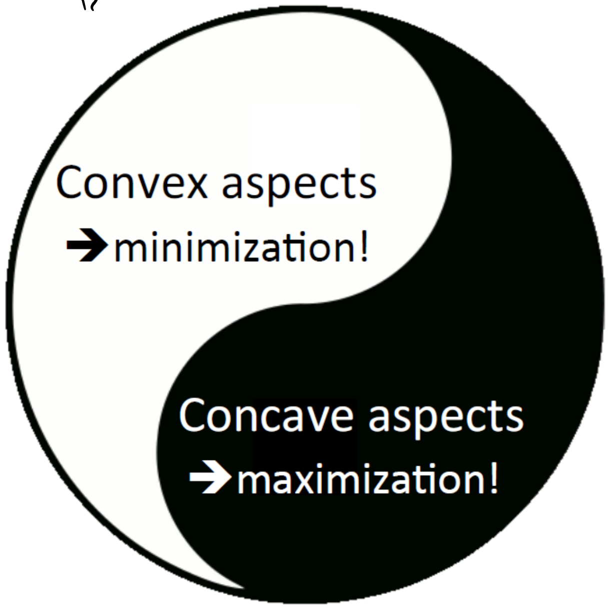


Concave aspects (Vondrak (2007), I-Bilmes (2015))

- Max: constant-factor approx!
- Multilinear extension - concave in a direction.
- Supergradients and Superdifferential.
- Under restricted settings, duality, separation etc.

Submod Max is NP hard!





Submodularity is related to both convexity and concavity!

$V = \{1, 2, 3, \ldots, n\}$

2^n numbers, 2^V Power set

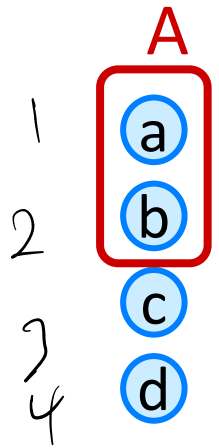
Set Functions and Boolean Functions

any set function with $|V| = n$.

$$F : 2^V \to \mathbb{R}$$

$A = \{a, b\}$

1, 2



$V = \{a, b, c, d\}$

Equivalent

$\hat{=}$

... is a function on binary vectors!

$$F : \{0,1\}^n \to \mathbb{R}$$

$$x = e_A$$

1	a
1	b
0	c
0	d

pseudo-boolean function

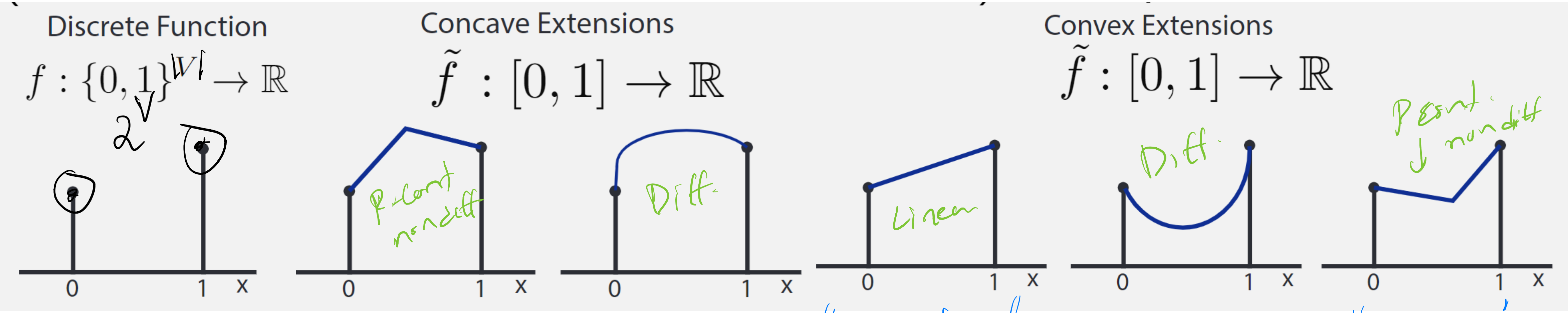
boolean: $F : \{0,1\}^n \to \{0,1\}$

$f : \{0,1\}^n \to \mathbb{R}$

$h = f - g$

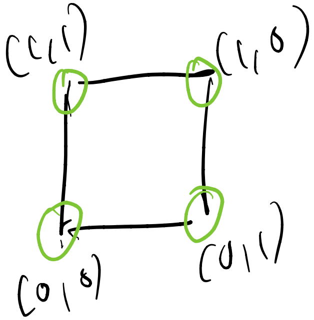
any set fn

What are Continuous Extensions?



"Modular"

"Lovasz"



Different kinds of Continuous Extensions of Discrete Functions

$x(b) \le F(b)$
$x(a) \le F(a)$
$x(a) + x(b) \le F(a,b)$

$|V| = n.$

$x \in P_{a,b},\ a_i^T x \le b_i,\ \forall\ i = 1 \ldots K.$

The Submodular Polyhedron

$$P_F = \{x \in \mathbb{R}^n : x(A) \le F(A) \text{ for all } A \subseteq V\}$$

$$x(A) = \sum_{i \in A} x_i$$

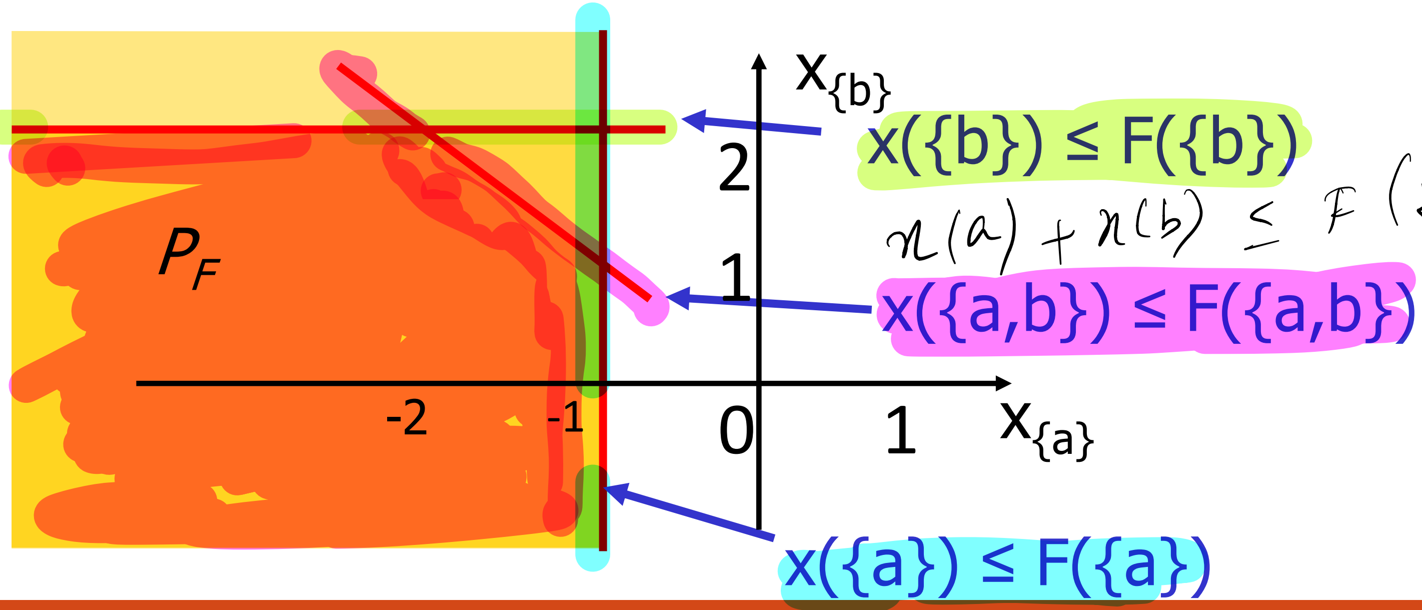
$2^{|V|} = 2^n$

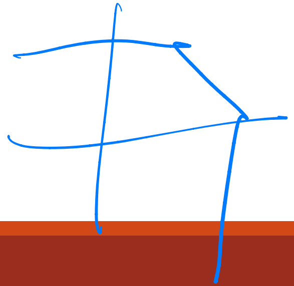
$V = \{a, b\}$

Example: V = {a,b}

A	F(A)
{}	0
{a}	-1
{b}	2
{a,b}	0

$x \in P_A,\ Ax \le b.$





$$P_F = \{x : x(A) \le F(A), \ \forall A\}$$

Q: Can we $\tilde{f}(x) = \max_{y \in P_F} x^T y$? does this work for any set fn ?

$$\hat{P}_F = \{(x,b) \in \mathbb{R}^{n+1} : x(A) + b \le F(A), \ \forall A \subseteq V\}$$

if F is submodular, $\max_{y \in P_F} x^T y \equiv \max_{(y,b) \in \hat{P}_F} x^T y + b$

And for any arbitrary set fn F,

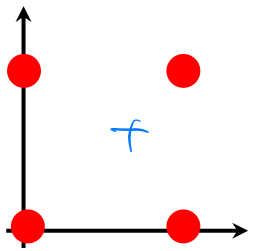
$$\tilde{f}(x) = \max_{(y,b) \in \hat{P}_F} x^T y + b \quad \text{is convex!}$$

Polyhedral aspects of submod, Convexity & concavity 2015.

Convex Extensions of Submodular Functions

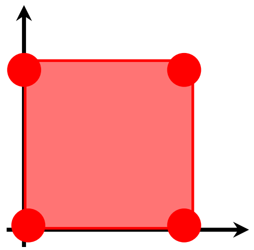
$F : \{0,1\}^n \to \mathbb{R}$ $\longrightarrow$ $f : [0,1]^n \to \mathbb{R}$

extension



$$\min_{x \in [0,1]^n} f(x) = \min_{X \subseteq V} F(X)$$

convex minimization



Lovász extension

$$f(x) = \max_{y \in P_F} x^T \cdot y$$

convex

Lovász, 1982

- minimum of *f* is a minimum of F
- submodular minimization as convex minimization: polynomial time!

Grötschel, Lovász, Schrijver 1981

Minimizing Submodular fns using the Lovasz Extension

Given V, $f: 2^V \to \mathbb{R}$

Output: $X \subseteq V$

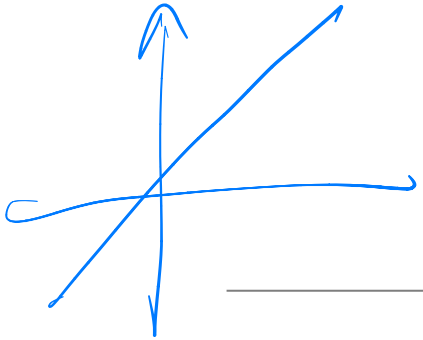
1. Compute the Lovasz Extension

2. Minimize $\tilde{f}(x)$ $\leftarrow$ $O\left(\frac{1}{\varepsilon^2}\right)$
   $x \in [0,1]^n$ Projected Gradient Descent

   $0 \leq x_i \leq 1$

3. Once we obtain $\hat{x}$ from 2
   we can round $\hat{x}$ to obtain
   $\hat{X}$

$O\left(\frac{1}{\varepsilon^2}\right)$ for Lipschitz cont fns



$x_{\sigma_1} \geq x_{\sigma_2} \geq x_{\sigma_3} \geq \cdots \geq x_{\sigma_n}$

$\{\sigma_1, \sigma_2, \ldots, \sigma_n\}$

Evaluating the Lovasz Extension

$$P_F = \{x \in \mathbb{R}^n : x(A) \leq F(A) \text{ for all } A \subseteq V\}$$

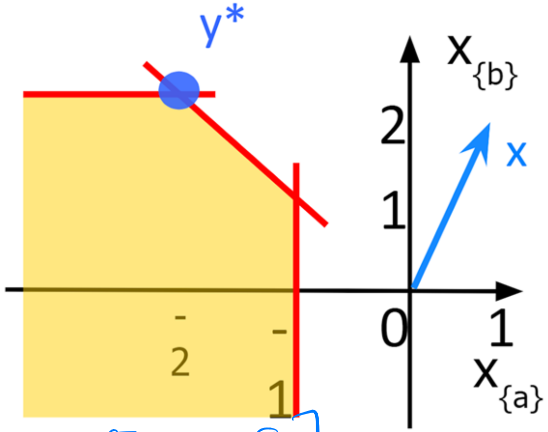
Linear maximization over P_F

$$f(x) = \max_{y \in P_F} x \cdot y$$

Exponentially many constraints!!! ☹

Computable in O(n log n) time ☺

[Edmonds '70]



greedy algorithm:

- sort x
- order defines sets $S_i = \{\ldots\}$
- $y_{\sigma_i} = F(S_i) - F(S_{i-1})$

$S_i = \{\sigma_1, \ldots, \sigma_i\}$

$\hat{y}_x[\sigma_i] = f(S_i) - f(S_{i-1})$

$n = 5$

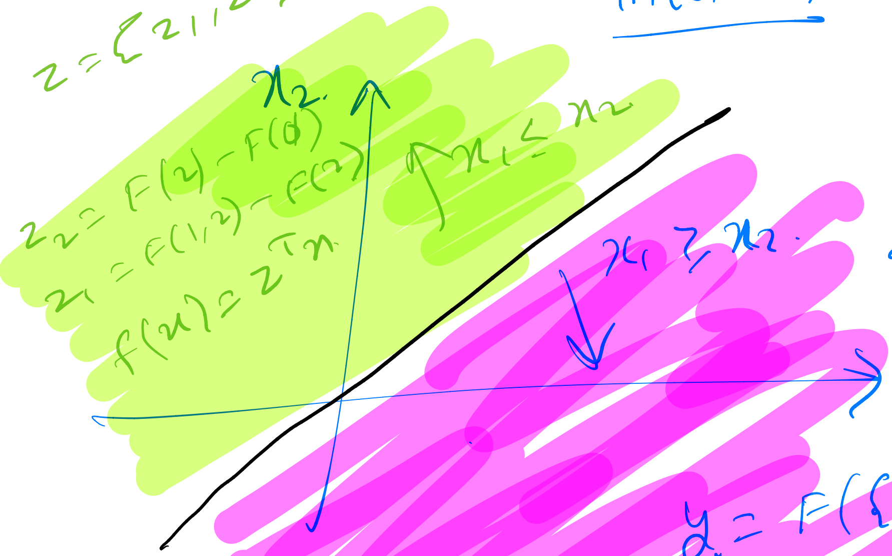
$x = [0.1, 0.4, 0.2, 0.05, 0.9]$

$\{5, 2, 3, 1, 4\}$

The Lovasz Extension is a non-differentiable Convex Function!

$f(x) = x^T \hat{y}_x$

Intuition



$z = \{z_1, z_2\}$

$z_2 = F(2) - F(\emptyset)$

$z_1 = F(1,2) - F(2)$

$f(x) = z^T x$

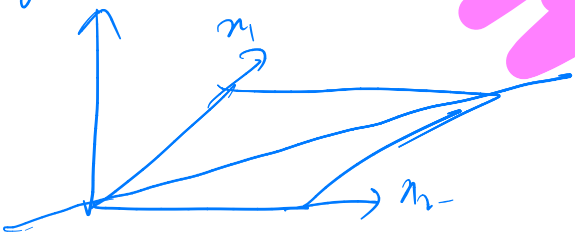
$\{1, 2\}$

$y_1 = F(\{1\}) - F(\emptyset)$

$y_2 = F(\{1,2\}) - F(\{1\})$

$y = [y_1, y_2]$

$f(x) = y^T_{\text{fixed}} x$



$n!$ number of orderings

$f(x) = x^T y_{\sigma x}$

$d_j(x, \sigma)$

$h = f + g$
$\tilde{h} = \tilde{f} + \tilde{g}$

$h = \lambda f$
$\tilde{h} = \lambda \tilde{f}$

Properties of the Lovasz Extension

Theorem 15.6.1

Let $f, g : 2^E \to \mathbb{R}$ *be normalized* $(f(\emptyset) = g(\emptyset) = 0)$. *Then*

1. *Superposition of LE operator: Given* f *and* g *with Lovász extensions* $\tilde{f}$ *and* $\tilde{g}$ *then* $\tilde{f} + \tilde{g}$ *is the Lovász extension of* $f + g$ *and* $\lambda \tilde{f}$ *is the Lovász extension of* λf *for* $\lambda \in \mathbb{R}$.
2. *If* $w \in \mathbb{R}_+^E$ *then* $\tilde{f}(w) = \int_0^{+\infty} f(\{w \geq \alpha\}) d\alpha$.
3. *For* $w \in \mathbb{R}^E$, *and* $\alpha \in \mathbb{R}$, *we have* $\tilde{f}(w + \alpha \mathbf{1}_E) = \tilde{f}(w) + \alpha f(E)$.
4. *Positive homogeneity: I.e.,* $\tilde{f}(\alpha w) = \alpha \tilde{f}(w)$ *for* $\alpha \geq 0$.
5. *For all* $A \subseteq E$, $\tilde{f}(\mathbf{1}_A) = f(A)$. $x = 1_A$
6. f *symmetric as in* $f(A) = f(V \setminus A), \forall A$, *then* $\tilde{f}(w) = \tilde{f}(-w)$ *(*$\tilde{f}$ *is even).*
7. *Given partition* $E^1 \cup E^2 \cup \cdots \cup E^k$ *of* E *and* $w = \sum_{i=1}^k \gamma_i \mathbf{1}_{E_k}$ *with* $\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_k$, *and with* $E^{1:i} = E^1 \cup E^2 \cup \cdots \cup E^i$, *then* $\tilde{f}(w) = \sum_{i=1}^k \gamma_i f(E^i | E^{1:i-1}) = \sum_{i=1}^{k-1} f(E^{1:i})(\gamma_i - \gamma_{i+1}) + f(E)\gamma_k$.

Francis Bach 2012 - 2014.

$\boxed{R(w)} = \tilde{f}(|w|)$ → L1 norm
→ l_∞ norm

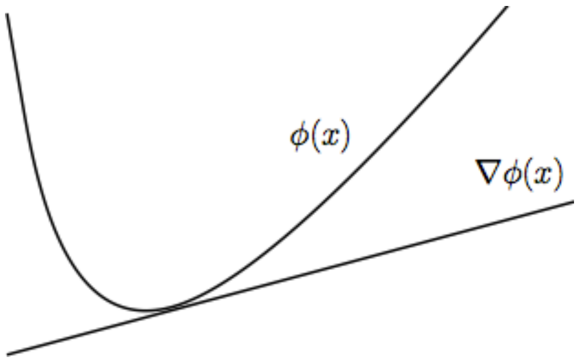
Examples of Lovasz Extension

Some additional submodular functions and their Lovász extensions, where $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m) \geq 0$. Let $W_k \triangleq \sum_{i=1}^{k} w(e_i)$.

$f(A)$	$\tilde{f}(w)$
$\lvert A\rvert$	$\lVert w\rVert_1$
$\min(\lvert A\rvert, 1)$	$\lVert w\rVert_\infty$
$\min(\lvert A\rvert, 1) - \max(\lvert A\rvert - m + 1, 0)$	$\lVert w\rVert_\infty - \min_i w_i$
$\min(\lvert A\rvert, k)$	W_k
$\min(\lvert A\rvert, k) - \max(\lvert A\rvert - (n-k) + 1, 1)$	$2W_k - W_m$
$\min(\lvert A\rvert, \lvert E \setminus A\rvert)$	$2W_{\lfloor m/2 \rfloor} - W_m$

(thanks to K. Narayanan).

Submodular Semi-gradients

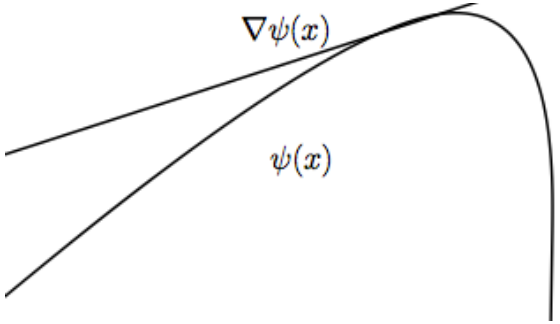


Convex Function

$$\phi(y) \geq \phi(x) + \langle \nabla\phi(x), y - x \rangle$$

$$\phi(y) \geq \phi(x) \Downarrow + \langle h_x^{\phi}, y - x \rangle$$

Linear Lower Bound



Concave Function

$$\psi(y) \leq \psi(x) + \langle \nabla\psi(x), y - x \rangle$$

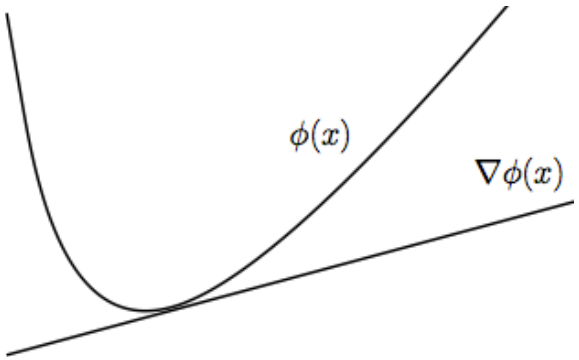
$$\psi(y) \leq \psi(x) \Downarrow + \langle g_x^{\psi}, y - x \rangle$$

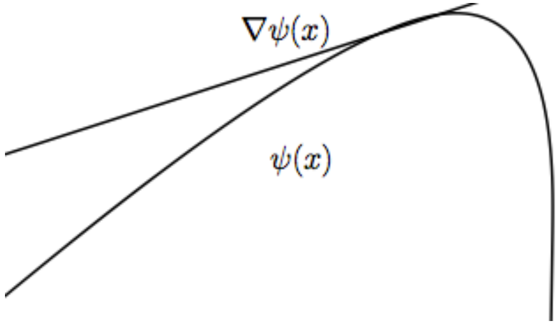
Linear Upper Bound

Can we do something similar for submodular functions?

$$f(Y) \geq f(X) + h_X^f(Y) - h_X^f(X) \qquad f(Y) \leq f(X) + g_X^f(Y) - g_X^f(X)$$

Submodular Semi-gradients





Convex Function

$\phi(y) \geq \phi(x) + \langle \nabla\phi(x), y - x\rangle$

Linear Lower Bound

Concave Function

$\psi(y) \leq \psi(x) + \langle \nabla\psi(x), y - x\rangle$

Linear Upper Bound

We can define both Sub-gradients and Super-gradients for Submodular Functions!

Can we do something similar for submodular functions?

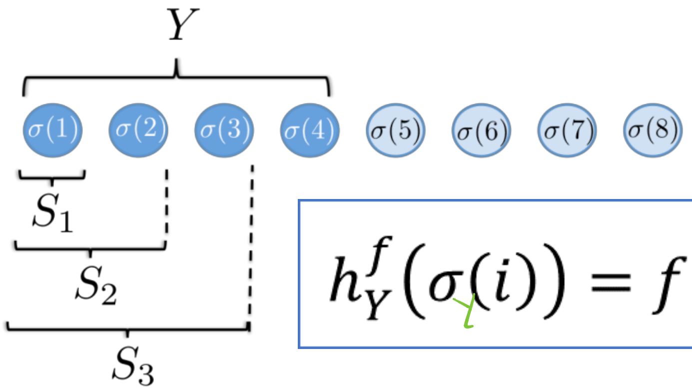
$f(Y) \geq f(X) + h_X(Y) - h_X(X)$

$f(Y) \leq f(X) + g_X(Y) - g_X(X)$

Submodular Semigradients

$\partial f(Y)$

- Define a Sub-gradient h_Y^f at set Y

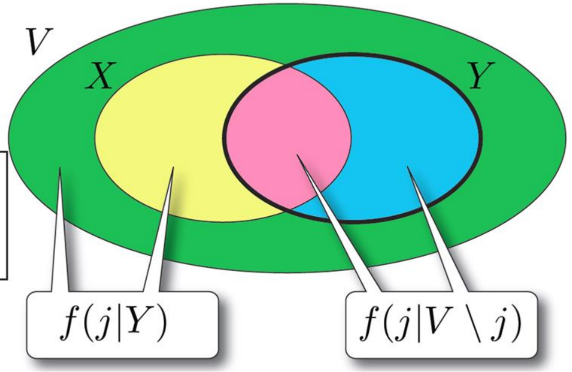


$$\boxed{h_Y^f(\sigma_Y(i)) = f(S_i) - f(S_{i-1})}$$

- Define a Sup-gradient g_Y^f at set Y

Grow:

$$\hat{g}_Y(j) = \begin{cases} f(j|Y) & \text{for } j \notin Y \\ f(j|V\setminus\{j\}) & \text{for } j \in Y \end{cases}$$



- Modular Lower bound:

$$m_Y(X) = f(Y) + h_Y^f(X) - h_Y^f(Y) \leq f(X)$$

"Fujishige 1984"

(Fujishige 2005)

- Modular Upper bound:

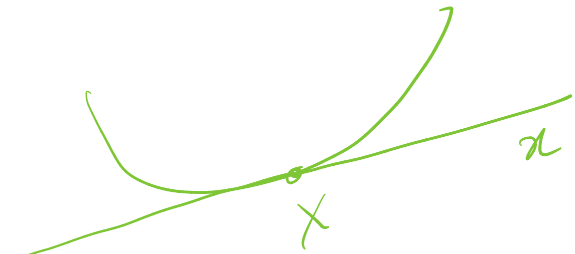
$$m^Y(X) = f(Y) + g_Y^f(X) - g_Y^f(Y) \geq f(X)$$

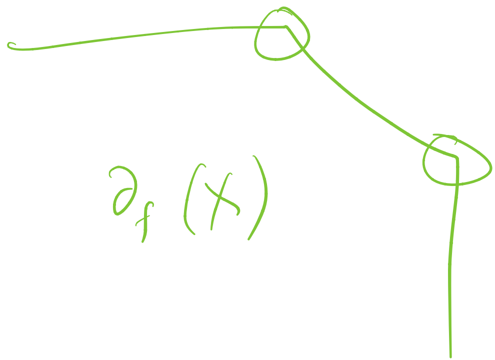
(I-Jegelka-Bilmes ICML2013)

$$\partial_f(X) = \{x \in \mathbb{R}^n : \underline{f(Y) \geq f(X) + x(Y) - x(X)} \ \forall\, Y \subseteq V\}$$

Sub-differential. [Polyhedron]

Modular lower bound of f tight at X



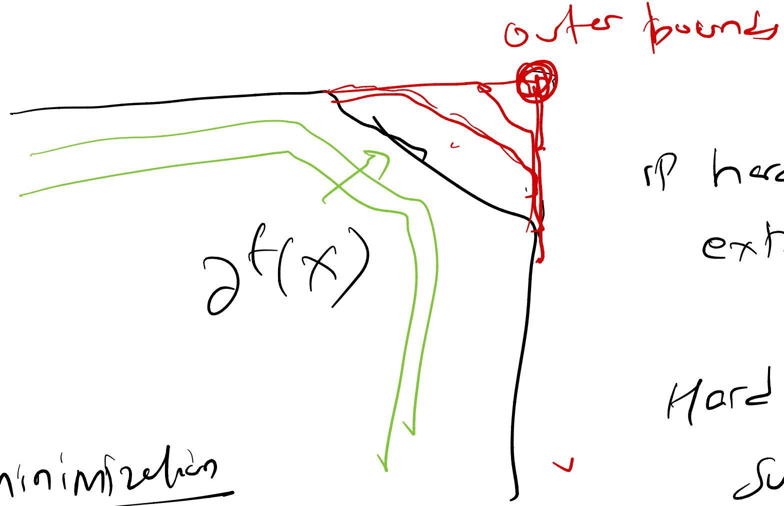


$\partial_f(X)$

Super-differentials

$$\partial^f(X) = \{x \in \mathbb{R}^n : f(Y) \leq f(X) + x(Y) - x(X)\}$$

"Computing extreme points is NP hard"



is hard to compute extreme points

Hardness of computing super differentials is related to the hardness of submod maximization

Submod minimization

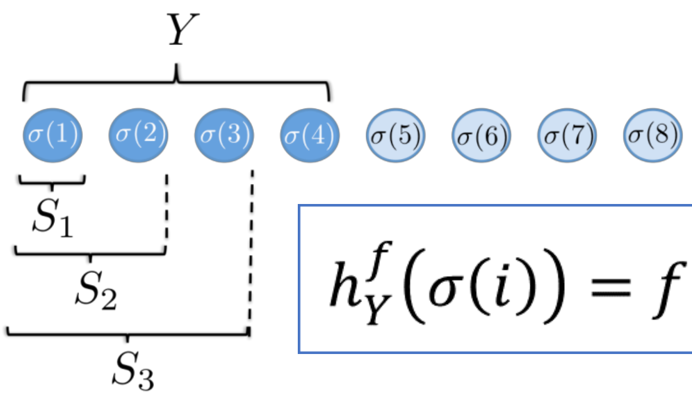
X^* is minima of a submod f iff $0 \in \partial_f(X)$

Submod maxima

X^* is maxima of a submodf f iff $0 \in \partial^f(X)$ $\Rightarrow$ Idea: $\hat{X}$ s.t. $0 \in \hat{\partial}^f(\hat{X})$

Submodular Semigradients

- Define a Sub-gradient h_Y^f at set Y



$$\boxed{h_Y^f(\sigma(i)) = f(S_i) - f(S_{i-1})}$$

- Modular Lower bound:

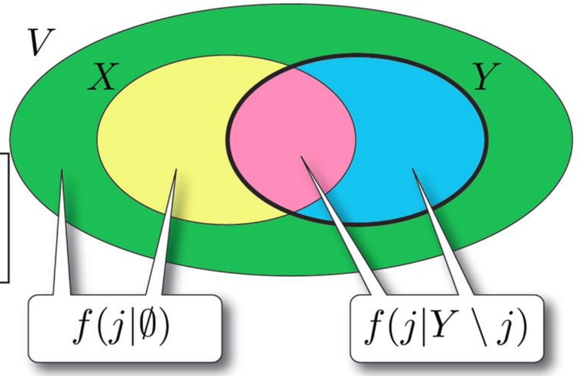
$$m_Y(X) = f(Y) + h_Y^f(X) - h_Y^f(Y) \leq f(X)$$

(Fujishige 2005)

- Define a Sup-gradient g_Y^f at set Y

Shrink:

$$\breve{g}_Y(j) = \begin{cases} f(j|\emptyset) & \text{for } j \notin Y \\ f(j|Y\backslash\{j\}) & \text{for } j \in Y \end{cases}$$



- Modular Upper bound:

$$m^Y(X) = f(Y) + g_Y^f(X) - g_Y^f(Y) \geq f(X)$$

(I-Jegelka-Bilmes ICML2013)

Discrete vs Continuous Optimization

Continuous Functions

Convex Functions: Improved Bounds

- Strong Convexity
- Lipschitz Continuous Gradients
- …

Discrete Functions

Submodular Functions: Improved Bounds

?

Discrete vs Continuous Optimization

$\kappa_f(X) \le \kappa_f(V) = \kappa_f$

Continuous Functions

Convex Functions: Improved Bounds

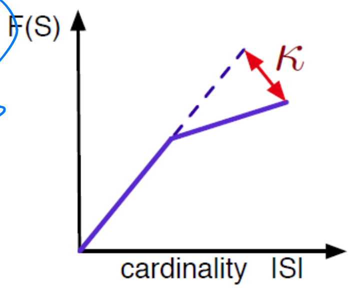
$(1 - \kappa_f(X^*))$

- Strong Convexity
- Lipschitz Continuous Gradients
- …

$\kappa_f(X) = 1 - \min_{j \in X} \frac{f(j|X \setminus j)}{f(j)}$

Discrete Functions

Submodular Functions: Improved Bounds



if $f(j|V \setminus j) = 0$

$$\kappa_f = 1 - \min_j \frac{f(j|V \setminus j)}{f(j)} = 1$$

$\min_{X} f(X|Q) \leftarrow$ Submodular Span.

Minimization vs Maximization

Query Rel.

Cooperation/Complexity

⇩

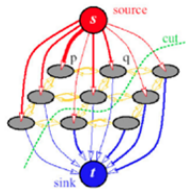
Minimize f

Coverage/ Diversity

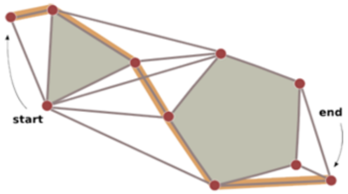
⇩

Maximize g

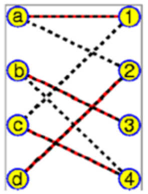
Often Combinatorial Constraints...



cuts



paths



matchings

$\max g(X)$

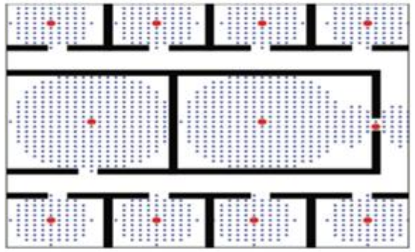
s.t. $f(X|Q) \leq \epsilon$

Simultaneously minimizing f while maximizing g?

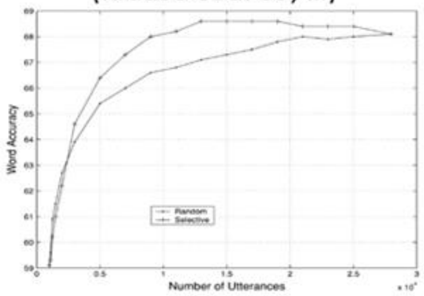
Minimizing Cooperative Costs/Complexity while Simultaneously maximizing coverage/diversity!

Submodular Maximization

Data Summarization
(Lin-Bilmes 11, Tschiatchek-Iyer et al 14, ...)



Sensor Placement
(Krause et al 09, ...)



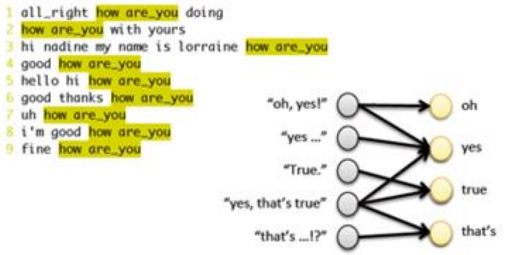
Data Subset Selection
(Wei-Iyer-Bilmes 2015, ...)

Submodular Minimization

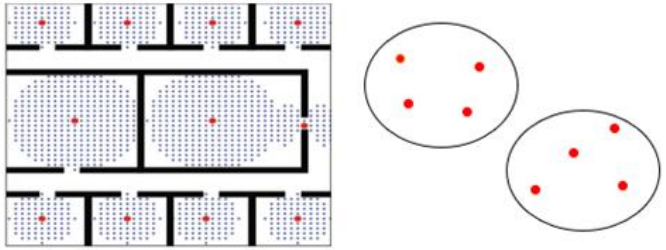
Cooperative Matching
(Iyer-Bilmes 14)

Cooperative Cuts
(Jegelka-Bilmes 11, ...)

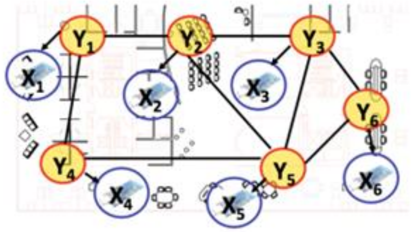
Simultaneous Min & Max



Limited Vocab. Speech Selection
(Liu-Iyer-et al 15, ...)



Sensor Placement+ Cooperative costs
(Iyer-Bilmes 14)



Feature Selection
(Iyer-Bilmes 12, ...)

Submodular Optimization Problems

$f \rightarrow$ cooperative cost or algorithmic complexity.
$g \rightarrow$ coverage/ diversity or information.

- Submodular Minimization:

$$\min_{X \in \mathcal{C}} f(X) \quad \Longrightarrow$$

Minimizing Cooperative Costs

- Submodular Maximization:

$$\max_{X \in \mathcal{C}} g(X) \quad \Longrightarrow$$

Maximizing Coverage/ Diversity

Focus of this Lecture

- Difference of Submodular (DS) optimization:

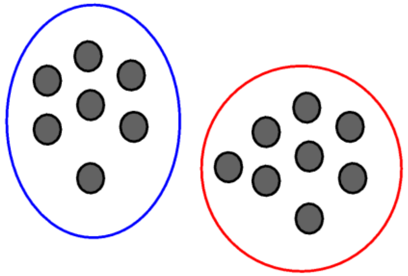
$$\min_{X} f(X) - \lambda g(X)$$

- Submodular optimization under submodular constraints,

$$\min\{f(X) : g(X) \geq c\}, \quad \max\{g(X) : f(X) \leq b\}$$

Minimizing cooperative costs and Maximizing Coverage/ Diversity

Submodular Minimization Applications

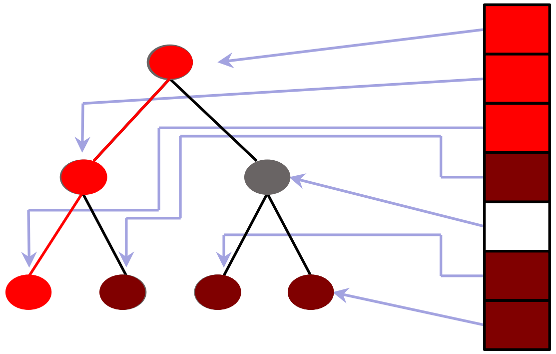


clustering

$S, \underline{V \setminus S}$

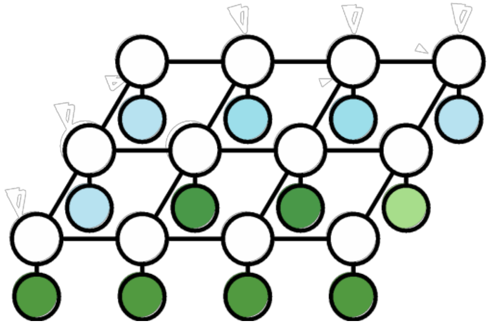
$$\min_{S \subseteq V} \underbrace{F(S) + F(V \setminus S)}_{g(S)}$$

$$\min_{S \subseteq V} F(S)$$

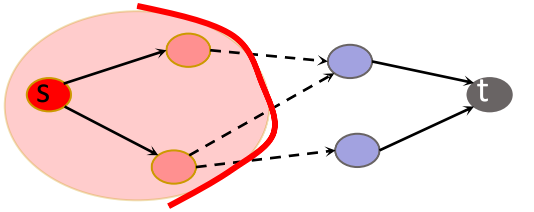


structured sparsity regularization

$\tilde{f}(|x|)$



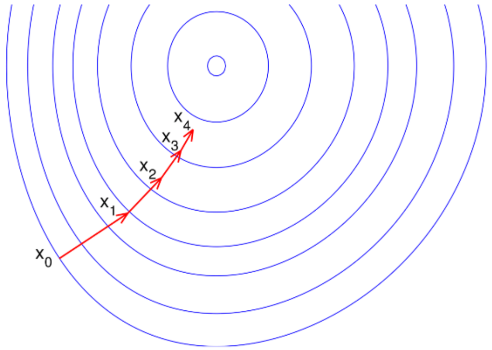
MAP inference



minimum cut

Discrete vs Continuous Optimization

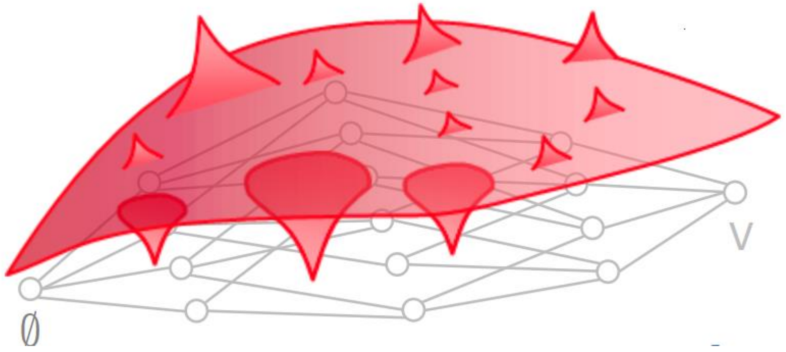
Continuous Functions



Convex Functions

Gradient Descent Framework

Discrete Functions



Submodular Functions

?

Unconstrained Submodular Minimization Algorithms

minimize convex extension

$O(n^3)$

ellipsoid algorithm
[Grötschel et al. `81]

subgradient method,
smoothing [Stobbe & Krause `10]

duality: minimum norm point algorithm
[Fujishige & Isotani '11]

$$\min_{A \subset V} F(A)$$

Yin Tat Lee.

2016-2019

$n^3 - n^4$

combinatorial algorithms

Fulkerson prize
Iwata, Fujishige, Fleischer '01 & Schrijver '00

state of the art:
$O(n^4T + n^5\log M)$ [Iwata '03]
$O(n^6 + n^5T)$ [Orlin '09]

Max singleton value

T = time for evaluating *F*

Unconstrained Submodular Minimization is Polynomial time Solvable!

The minimum-norm-point algorithm

A	F(A)
{}	0
{a}	-1
{b}	2
{a,b}	0

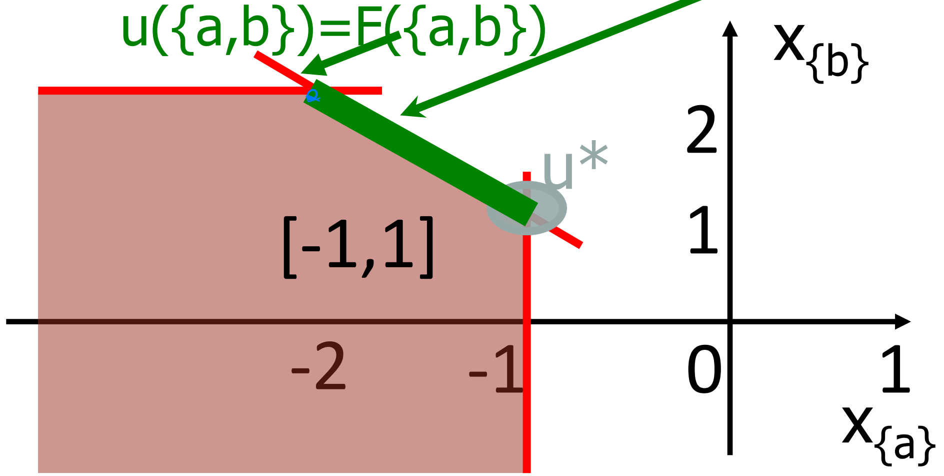
$B_F = P_F \cap \{x : x(V) = F(V)\}$

dual: minimum norm problem

$$u^* = \operatorname*{argmin}_{u \in B_F} \frac{1}{2}\|u\|^2$$

Example: V = {a,b}

Base polytope B_F



$$A^* = \{i \mid u^*(i) \leq 0\}$$

minimizes *F*:

$$A^* = \arg\min_{A \subset V} F(A)$$

Fujishige ‘91, Fujishige & Isotani ‘11

The minimum-norm-point algorithm

regularized problem

$$\min_x f(x) + \frac{1}{2}\|x\|^2$$

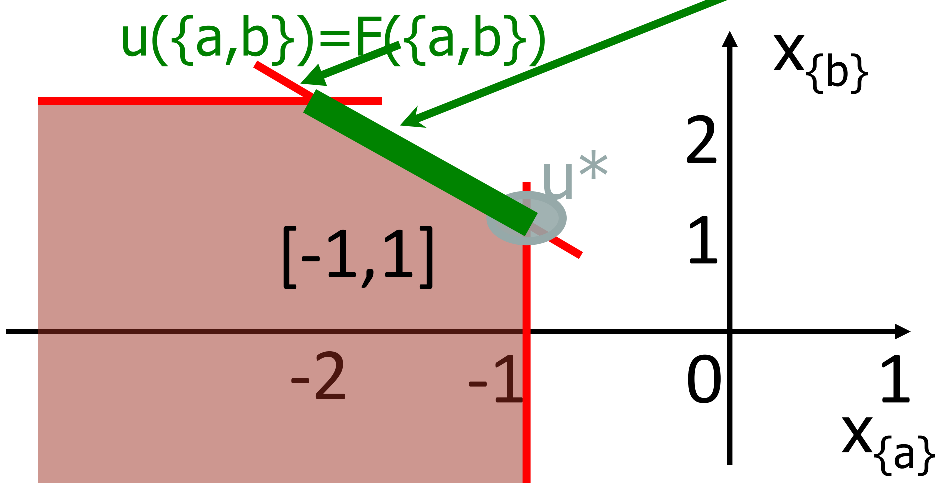
$\max_{y \in P_F} y^T x$

dual: minimum norm problem

$$u^* = \operatorname{argmin}_{u \in B_F} \frac{1}{2}\|u\|^2$$

Example: V = {a,b}

Base polytope B_F



$$A^* = \{i \mid u^*(i) \leq 0\}$$

minimizes *F*:

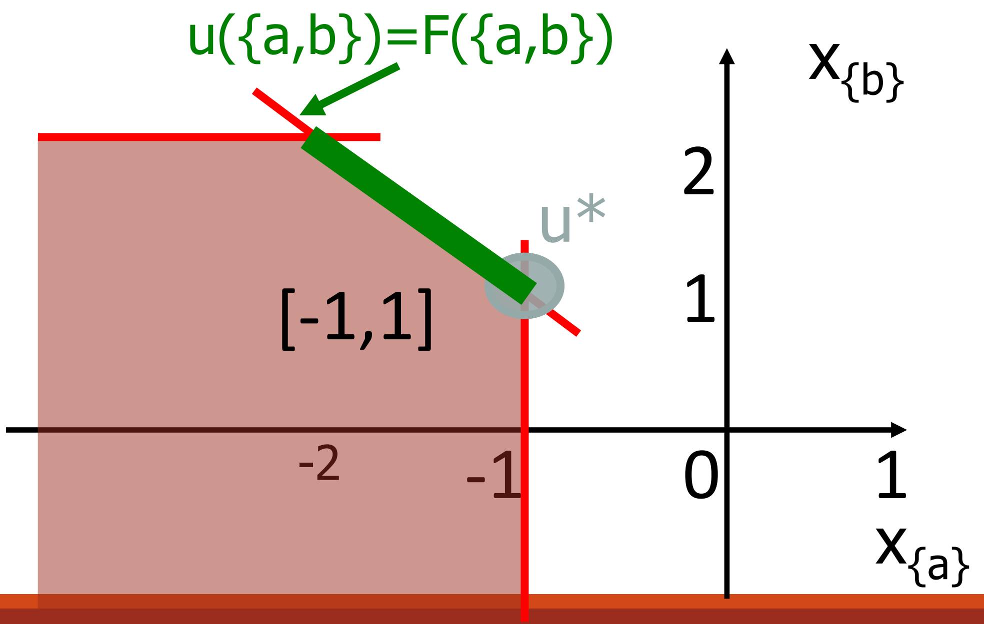
$$A^* = \arg\min_{A \subset V} F(A)$$

Fujishige '91, Fujishige & Isotani '11

The minimum-norm-point algorithm

1. find $$u^* = \arg\min_{u \in B_F} \frac{1}{2}\|u\|^2$$
1. $$A^* = \{i \mid u^*(i) \leq 0\}$$

can we solve this??



yes! 😊

recall: can solve

linear optimization over P_F

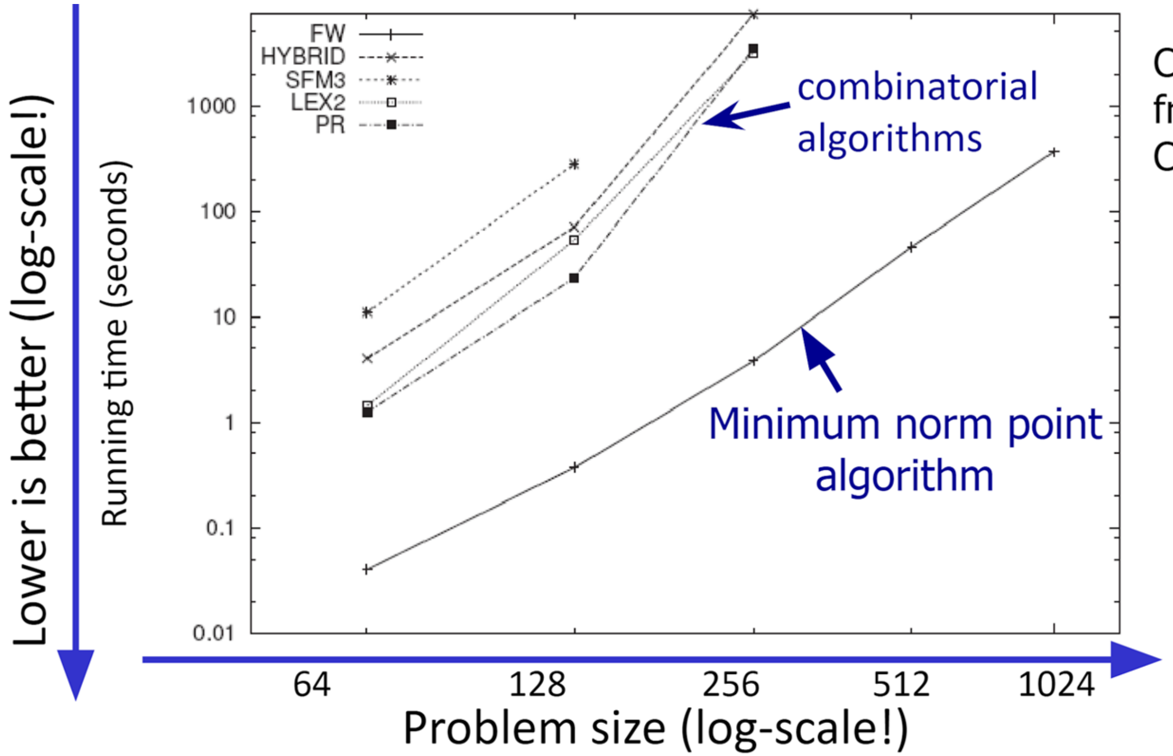
similar: optimization over B_F

can find u^*

(Frank-Wolfe algorithm)

Fujishige '91, Fujishige & Isotani '11

Empirical Performance



Cut functions from DIMACS Challenge

Minimum norm point algorithm: usually orders of magnitude faster

[Fujishige & Isotani '11]

Minimization of f using Lovasz Extension

① $\min_{x \in [0,1]^n} \tilde{f}(x) \leftarrow \hat{x} \leftarrow PGD\left(\frac{1}{\epsilon^2}\right)$

$$\left|f(\hat{x}) - f(x^*)\right| \le \epsilon$$

② Round $\hat{x}$

How to Round?

$$\nabla \tilde{f}(x) = h_{\sigma x}$$

Initial x^0

$$x^{t+1} \leftarrow x^t - \alpha h_{\sigma x^t}$$

$$\tilde{f}(\hat{x}) = \hat{x} \underbrace{h_{\sigma \hat{x}}}_{[f(S_i) - f(S_{i-1})]}$$

$$= \sum_{i=1}^{n} \alpha_i f(S_i^{\hat{x}}), \quad \sum_{i=1}^{n} \alpha_i = 1$$

Assume

$$f(S_i) > \tilde{f}(\hat{x}) \quad \forall i$$

$$\Rightarrow \tilde{f}(\hat{x}) > \sum_{i=1}^{\hat{n}} \alpha_i f(S_i) > \tilde{f}(\hat{x})$$

$\exists\, S_i$ s.t. $f(S_i) \le \tilde{f}(\hat{x})$

$\hat{x}$

$\downarrow$

$S_0 \subset S_1 \subset \cdots \subset S_K.$

$6\hat{x}$ = Permutation.

① $\hat{x} = [0, 0, 0, 1, 1, 0, 1]$

② $\hat{x} = [0.5, 1, 0.5, 0, 0, 1]$

$S_0 = \{2, 6\}$

$S_1 = \{1, 2, 3, 6\}$

$S_2 = \{1, 2, 3, 4, 5, 6\}$

Claim: $\forall\, i = 1:n$

③ $\hat{x} = [1, 0.8, 0.7, 0.4, 0.7, 0.9]$

$\{1\} \subseteq \{1, 6\} \subseteq \{1, 2, 6\}$

$f(S_i) = \tilde{f}(\hat{x})$

Assume $\hat{x}$ is exact minima

So pick any of the sets $S_0, S_1, \ldots, S_n$

What if PGD gives ε approximate sol^n

$|f(\hat{x}) - f(x^*)| \le \varepsilon$, then Claim: $\left| \min_{i=1:n} f(S_i) - f(x^*) \right| \le \varepsilon$

Submodular Minimization with Constraints

unconstrained:

- nontrivial algorithms, polynomial time

$$\min \; F(A) \quad \text{s.t. } A \subseteq V$$

constraints: e.g.

- limited cases doable: odd/even cardinality, inclusion/exclusion of a set

$$\min \; F(A) \quad \text{s.t. } |A| \geq k$$

$$\ldots$$

special case: balanced cut

General case: NP hard

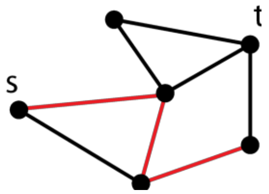
- hard to approximate within polynomial factors!
- But: special cases often still work well

[Lower bounds: Goel et al.`09, Iwata & Nagano `09, Jegelka & Bilmes `11]

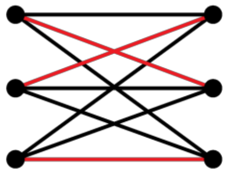
Submodular Minimization with Constraints

minimum…

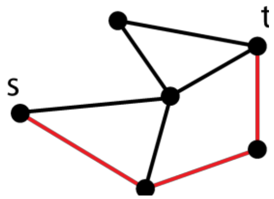
cut



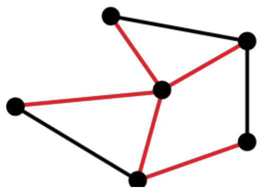
matching



path



spanning tree



ground set: edges in a graph

$$\min_{S \in \mathcal{C}} \sum_{e \in S} w(e) \quad \Longrightarrow \quad \min_{S \in \mathcal{C}} F(S)$$

Motivation: Image Segmentation

binary labeling: $x = e_A$

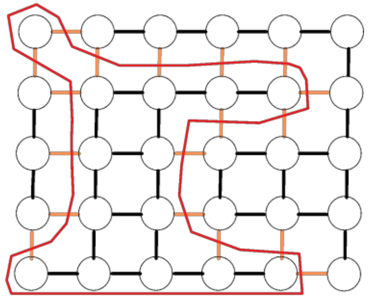
pairwise random field:

$$E(x) = \mathrm{Cut}(A)$$

What’s the problem?

aim

reality



minimum cut: prefer short cut = short object boundary

Co-operative Cuts

Minimum cut

implicit criterion:

short cut = short boundary

minimize

sum of edge weights

$$F(C) = \sum_{e \in C} w(e)$$

Minimum cooperative cut

new criterion:

boundary may be long if the boundary is homogeneous

minimize

submodular function of edges $F(C)$

not a sum of edge weights!

Constrained Submodular Min: Algorithms

$$\min_{S \in \mathcal{C}} \; F(S)$$

approximate optimization

convex relaxation

minimize surrogate function

approximation bounds dependent on *F*:

polynomial – constant – FPTAS

$O(n)$ $(1+\epsilon)$

[Goel et al.`09, Iwata & Nagano `09, Goemans et al. `09, Jegelka & Bilmes `11, Iyer et al. ICML `13, Kohli et al `13...]

Approximation Algorithms

- Given a min(max)-imization problem, obtaining the exact min(max)-imizer implies an approximation factor of 1.
- Since many problems are NP hard, it is not possible to achieve the exact optima in poly-time
- Define α as the approximation factor.
- A set X is the α-approximate minimizer if f(X) <= αf(X*) where X* is the global minima ($\alpha > 1$)
- A set X is the α-approximate maximizer if f(X) >= αf(X*) where X* is the global maxima ($\alpha < 1$)

Constrained Submodular Min Framework

minimize a series of surrogate functions

1. compute linear upper bound $\widehat{F}^i(S^i) = F(S^i)$

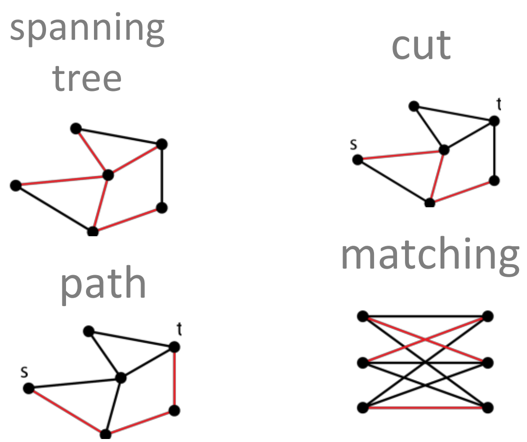
$$\hat{F}^i(S) = \sum_{e \in S} w^i(S)$$

2. Solve easy sum-of-weights problem:

$$S^i = \arg\min_{S \in \mathcal{C}} \hat{F}^i(S)$$

and repeat.

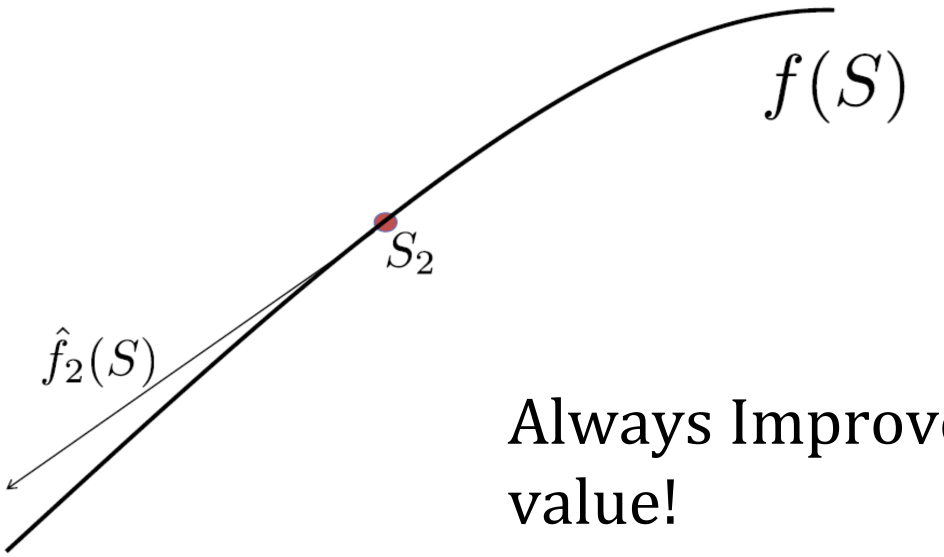
- efficient
- only need to solve sum-of-weights problems



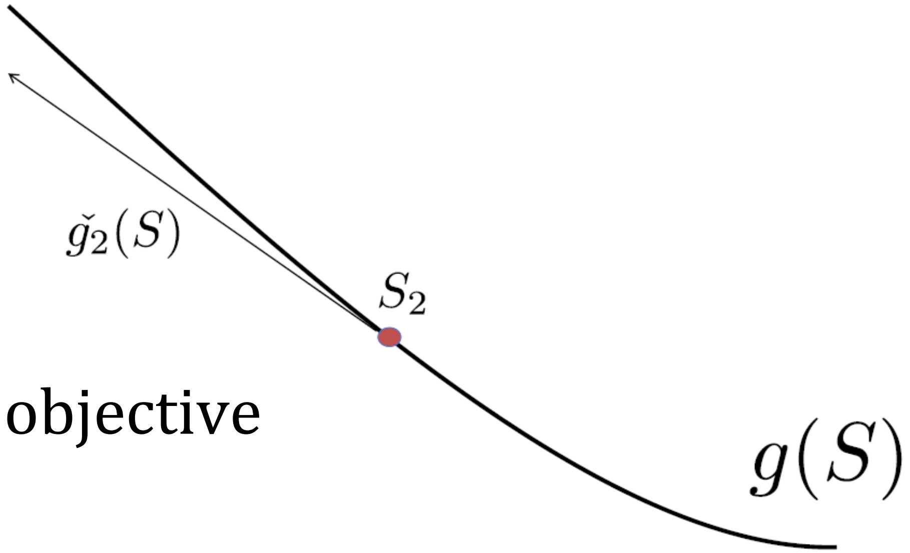
Majorization-Minimization Framework

For Min (Max) problems use the upper (lower) bounds as proxy functions!

Minimization



Maximization



Always Improves the objective value!

Constrained Submodular Minimization

minimize $F(S) : S \in \mathcal{C} =$ cut/path/matching/cardinality constraint...

Initialize $S_0 = \emptyset$;
for $i = 0, 1, \ldots$ **do**

compute modular upper bound $\hat{f}_i(S) = m^f_{S_i}(S) \geq f(S)$ based on S_i;

Set $S_{i+1} = \text{argmin}_{S \in \mathcal{C}} \hat{f}_i(S)$ - find best cut/path/matching...;
only need to solve linear-cost problem! ,
end

Iyer-Jegelka-Bilmes ICML 2013

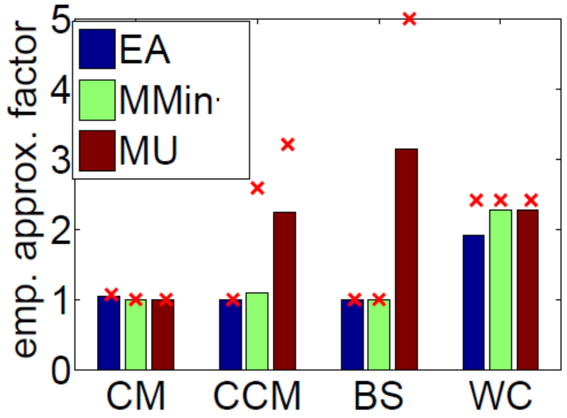
How does MMin perform?

minimize $F(S) : S \in \mathcal{C} =$ cut/path/matching/cardinality constraint...

In Theory...

- Mmin (IJB13) : $\frac{n}{1+(n-1)(1-\kappa)} \approx \frac{1}{1-\kappa}$
- EA* (Goemans et al) : $O(\frac{\sqrt{n}}{1+(\sqrt{n}-1)(1-\kappa)}) \approx \frac{1}{1-\kappa}$
- Mmin for subclasses (SCM): $1+\epsilon$

In Practice...



MMin 100x Faster compared to EA!

* Theoretically Tight

How does MMin perform?

minimize $F(S) : S \in \mathcal{C} =$ cut/path/matching/cardinality constraint...

For graph based problems, $m =$ number of edges, $n =$ number of vertices.

How good are these algorithms? $f(S) \leq \alpha f(S^*)$

Constraint:	MMin	EA	Lower bound
trees/matchings	$\frac{n}{1+(n-1)(1-\kappa_f)}$	$O(\frac{\sqrt{m}}{1+\sqrt{m}-1)(1-\kappa_f)})$	$\Omega(\frac{n}{1+(n-1)(1-\kappa_f)})$
cuts	$\frac{m}{1+(m-1)(1-\kappa_f)}$	$O(\frac{\sqrt{m}}{1+\sqrt{m}-1)(1-\kappa_f)})$	$\Omega(\frac{\sqrt{m}}{1+\sqrt{m}-1)(1-\kappa_f)})$
paths	$\frac{n}{1+(n-1)(1-\kappa_f)}$	$O(\frac{\sqrt{m}}{1+\sqrt{m}-1)(1-\kappa_f)})$	$\Omega(\frac{n^{2/3}}{1+(n^{2/3}-1)(1-\kappa_f)})$
cardinality(k)	$\frac{k}{1+(k-1)(1-\kappa_f)}$	$O(\frac{\sqrt{n}}{1+\sqrt{n}-1)(1-\kappa_f)})$	$\Omega(\frac{\sqrt{n}}{1+\sqrt{n}-1)(1-\kappa_f)})$

Worst case upper/lower bounds bounded by $O(\frac{1}{(1-\kappa_f)})$

Lecture 9

Submodular Maximization

Greedy & Beyond ...

Time Permit: ① Difference of Submod opt.

② Submod opt. s.t Submod Constraints

Master

1. Diff classes of fns
2. Diff kinds of constraints
3. Different settings

Submodular Maximization

Set Function

Selected set

$$\max_{\mathcal{A}\subseteq\mathcal{V}} F(\mathcal{A})$$

Selection cost

$$\text{subject to } C(\mathcal{A}) \le B$$

Budget

Constraints

F = Monotone Submodular,
Non Monotone Submodular,
Dispersion Functions,
....

F Models:

- Diversity
- Representation
- Coverage
- Information
- Importance
- ...

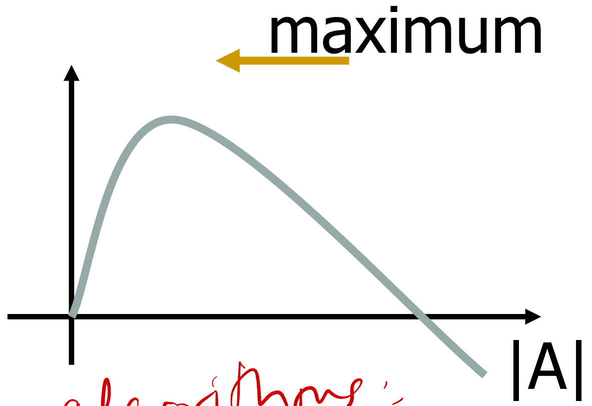
We shall study this and variants of this Master Optimization Problem!

Maximizing submodular functions

$$A^* = \arg\max_{A} F(A) \text{ s.t. } A \subseteq V$$

No constraint

Suppose we want for submodular F



Example:

- F(A) = U(A) – C(A) where U(A) is submodular utility, and C(A) is supermodular cost function

In general: NP hard. Moreover:

If F(A) can take negative values:
As hard to approximate as maximum independent set
(i.e., NP hard to get $O(n^{1-\varepsilon})$ approximation)

$\frac{1}{n}$

Approx algorithms:

① Alg with approx factor α
if Alg → $\hat{A}$ s.t $f(\hat{A}) \geq \alpha f(A^*)$
in Poly-time.

max: $\alpha \leq 1$

Exact maximization of SFs

Mixed integer programming (MIP)

- Series of mixed integer programs [Nemhauser et al ‘81]
- Constraint generation [Kawahara et al ‘09]

Branch-and-bound

- „Data-Correcting Algorithm“ [Goldengorin et al ’99]

Not common in ML!

Useful for small/moderate problems

All algorithms worst-case exponential!

NP Hard!!!

① Monotone f.

② Cardinality Constraints

Monotone Submodular Maximization

$$\max_{S} \ F(S) \ \text{ s.t. } |S| \le k$$

What is the Constraint?

C(S) = |S|

Max k-cover problem.

f: Set Cover.

NP hard

Uriel Fiege 1998

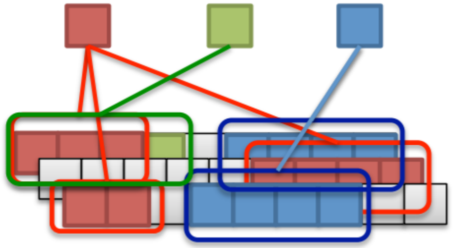
- greedy algorithm:

$S_0 = \emptyset$

for *i* = 0, ..., *k*-1

$$e^* = \arg\max_{e \in \mathcal{V} \setminus S_i} F(S_i \cup \{e\})$$

$$S_{i+1} = S_i \cup \{e^*\}$$



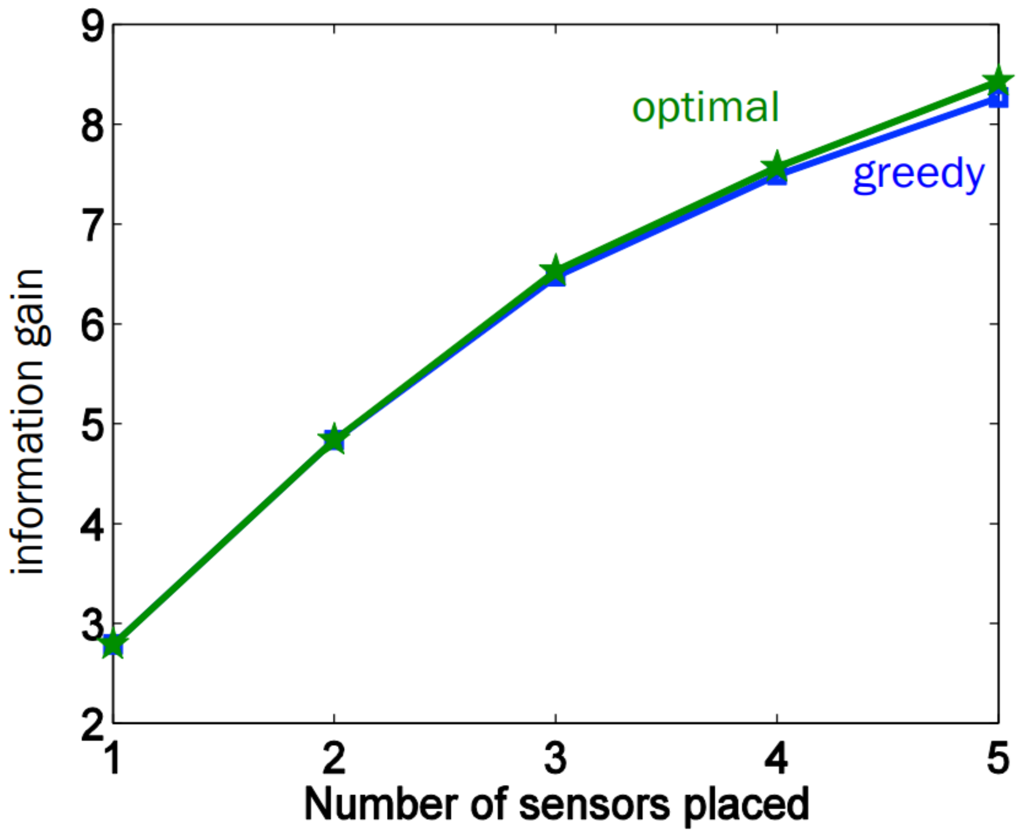
How "good" is S_k ?

(Nemhauser et al, 1978)

Approximation Guarantee!

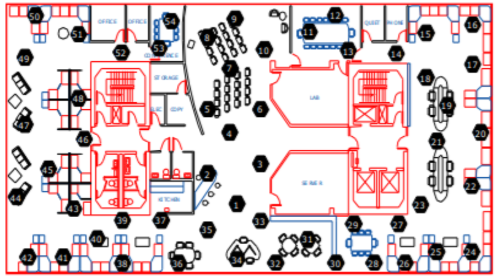
How good is Greedy in Practice?

empirically:



Emp: 0.98 - 0.99

sensor placement



How good is Greedy in Theory?

$$\max_S \; F(S) \;\text{ s.t. }\; |S| \le k$$

$1-\frac{1}{e}+\varepsilon$

NP hard!

Theorem *(Nemhauser, Fisher, Wolsey `78)*

F monotone submodular, S_k solution of greedy. Then

$$F(S_k) \;\ge\; \left(1-\frac{1}{e}\right) F(S^*)$$

Tight!

optimal solution

No Poly-time algorithm can do better than this in the worst case!

$$f(A^*) \le f(A_i \cup A^*) \quad \text{[monotone]}$$

$$= f(A_i) + \sum_{j=1}^{k} f(a_j^* \mid A_i \cup \{a_1^*, \dots, a_{j-1}^*\}) \quad \text{[Telescopic sum]}$$

$$\le f(A_i) + \sum_{j=1}^{k} f(a_j^* \mid A_i) \quad \text{[submod]}$$

$$\le f(A_i) + \sum_{j=1}^{k} f(a_i \mid A_i) \quad \text{[Greedy]}$$

$$= f(A_i) + k\, f(a_i \mid A_i)$$

$$= f(A_i) + k\,[f(A_{i+1}) - f(A_i)]$$

$$\Rightarrow \delta_i \le k[\delta_i - \delta_{i+1}]$$

$$\Rightarrow \delta_{i+1} \le \left(1 - \frac{1}{k}\right) \delta_i$$

A^* : opt soln

$A_1 \dots A_k$: k steps of greedy

k : card cons

$A^* = (a_1^*, \dots a_k^*)$

$$a_i = \underset{a \in V \setminus A_i}{\operatorname{argmax}} f(a \mid A_i)$$

$$\delta_i = f(A^*) - f(A_i)$$

$$\delta_k \le \left(1-\frac{1}{k}\right)^k \delta_0$$

$$\delta_k = f(A^*) - f(A_k)$$

$$\delta_0 = f(A^*) - f(\emptyset)$$

$$f(\emptyset) \ge 0$$

$$f(A^*) - f(A_k) \le \left(1-\frac{1}{k}\right)^k \left[f(A^*) - f(\emptyset)\right]$$

[non-neg of f] $\le \left(1-\frac{1}{k}\right)^k f(A^*)$

$$\boxed{\begin{array}{l} 1-x \le e^{-x} \\ \Rightarrow 1-\frac{1}{k} \le e^{-1/k} \\ \Rightarrow \left(1-\frac{1}{k}\right)^k \le \frac{1}{e} \end{array}}$$

$$\Rightarrow f(A_k) \ge \left[1 - \left(1-\frac{1}{k}\right)^k\right] f(A^*)$$

$$\ge \left[1 - \frac{1}{e}\right] f(A^*)$$

$1-\frac{1}{e} \simeq 0.63$

Enumerate all sets A s.t. $|A| \leq 3$ and pick the one with largest $f(A)$

Monotone Submodular – Budget Constraints

$$\max \quad F(S) \ \text{ s.t. } \sum_{e \in S} c(e) \leq B$$

$$A_3 = \operatorname*{argmax}_{A : |A| \leq 3} f(A)$$

1. run greedy: S_{gr} [cost agnostic greedy]
2. run a modified greedy: S_{mod} [cost sensitive greedy]

modified greedy

$$e^* = \arg\max \frac{F(S_i \cup \{e\}) - F(S_i)}{c(e)}$$

3. pick better of S_{gr}, S_{mod}

➔ approximation factor: $\frac{1}{2}\left(1 - \frac{1}{e}\right)$

"Leskovec 2007"

even better but less fast: partial enumeration *(Sviridenko, 2004)* or filtering *(Badanidiyuru & Vondrák 2014)*

Sviridenko 2004:

- Run the cost-sensitive greedy algorithm starting with all possible initial sets {I,j,k}
- $O(n^3)$ initial complexity
- 1 – 1/e approximation!

Sviridenko 2004.

Sviridenko 2004, Leskovec et al 2007

Summary: Greedy Algorithm Framework

[Used in Practice].

Monotone Submodular Function

$$\max_{S \subseteq V, c(S) \leq \mathcal{B}} f(S)$$

Cost of Summary Subset S (e.g. size)

Problem Formulation

Initialization $S \leftarrow \emptyset$.
repeat
Pick an element $v^* \in \text{argmax}_{v \in V \setminus S} \frac{f(v \cup S) - f(S)}{c(v)}$
Update $S \leftarrow S \cup v^*$
until Reaching the budget, i.e., $c(S) > \mathcal{B}$

Greedy Algorithm

Dual Problem: Monotone Submodular Cover [Gen Set Cover Problem]

$$\boxed{A^* = \operatorname{argmin} |A| \ \text{ s.t. } \ F(A) \geq Q}$$

F: set cover fn.

Dual Problem: Monotone Submodular Cover

$$\boxed{A^* = \operatorname{argmin} |A| \ \text{ s.t. } \ F(A) \geq Q}$$

$F \approx$ Acc

Greedy algorithm:

Start with $A := \emptyset$;

While $F(A) < Q$ and $|A| < n$

$s^* := \operatorname{argmax}_s F(A \cup \{s\})$

$A := A \cup \{s^*\}$

Stopping Conditions

instead of $|A| \leq k$.

if is $f(A) \geq Q$.

Dual Problem: Monotone Submodular Cover

[Submodular Set Cover]

$$A^* = \text{argmin } |A| \text{ s.t. } F(A) \geq Q$$

$\min\ c(A)$
s.t. $F(A) \geq Q$.

Greedy algorithm:

Start with $A := \emptyset$;
While $F(A) < Q$ and $|A| < n$
 $s^* := \text{argmax}_s F(A \cup \{s\})$
 $A := A \cup \{s^*\}$

ground set size

cost sensitive Greedy

$$\max_{s \in V \setminus A} \frac{F(s \mid A)}{c(s)}$$

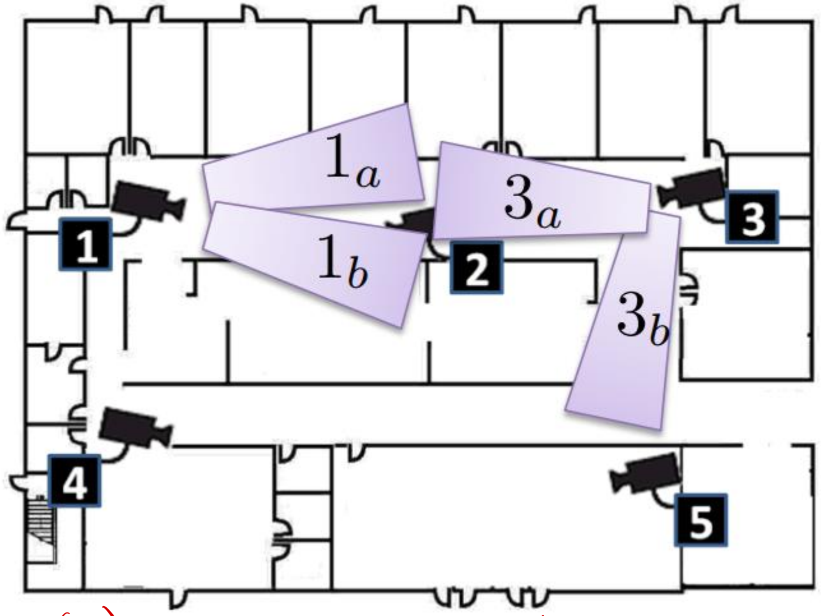
For bound, assume F is integral.
If not, just round it.

Theorem [Wolsey et al]: Greedy will return A_{greedy}
$|A_{greedy}| \leq (1+\log \max_s F(\{s\}))\ |A_{opt}|$

$\log n$

More Complex Constraints: Matroid Constraints

- Ground set: $V = \{1_a, 1_b, \ldots, 5_a, 5_b\}$
- Sensing quality model: $F : 2^V \to \mathbb{R}$



- Configuration (subset) is feasible if no camera is pointed in two directions at once

$|V| = 10$

- Constraints:

$P_1 = \{1_a, 1_b\}, \ldots, P_5 = \{5_a, 5_b\}$

require:

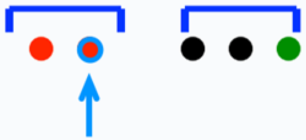
$|S \cap P_i| \leq 1$

$\max f(S)$

$S \in C$ (matroid constraints)

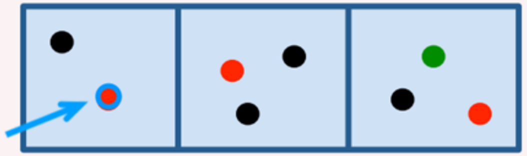
Independence Constraint in Matroid.

Matroid Constraints?



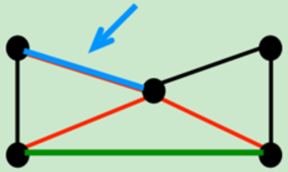
... $|S| \leq k$

Uniform matroid



... S contains at most one element from each group

Partition matroid

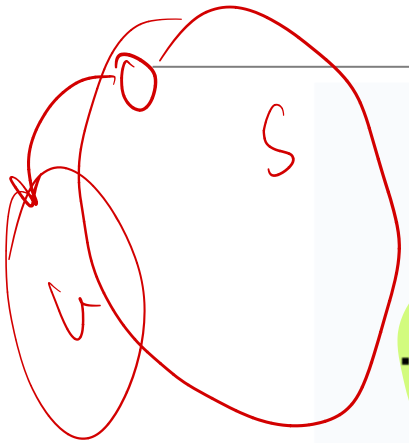


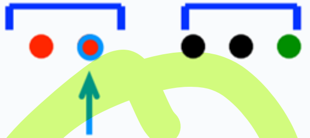
... S contains no cycles

Graphic matroid

- S independent ➔ $T \subseteq S$ also independent

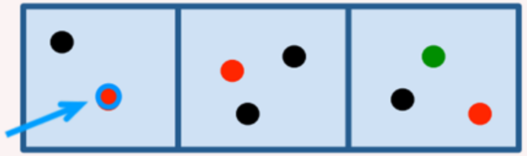
Matroid Constraints?





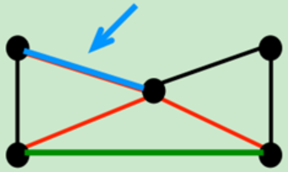
... $|S| \leq k$

Uniform matroid



... S contains at most one element from each group

Partition matroid



... S contains no cycles

Graphic matroid

- S independent ➔ $T \subseteq S$ also independent
- Exchange property: S, U independent, $|S| > |U|$
  ➔ some $e \in S$ can be added to U: $U \cup e$ independent
- All maximal independent sets have the same size

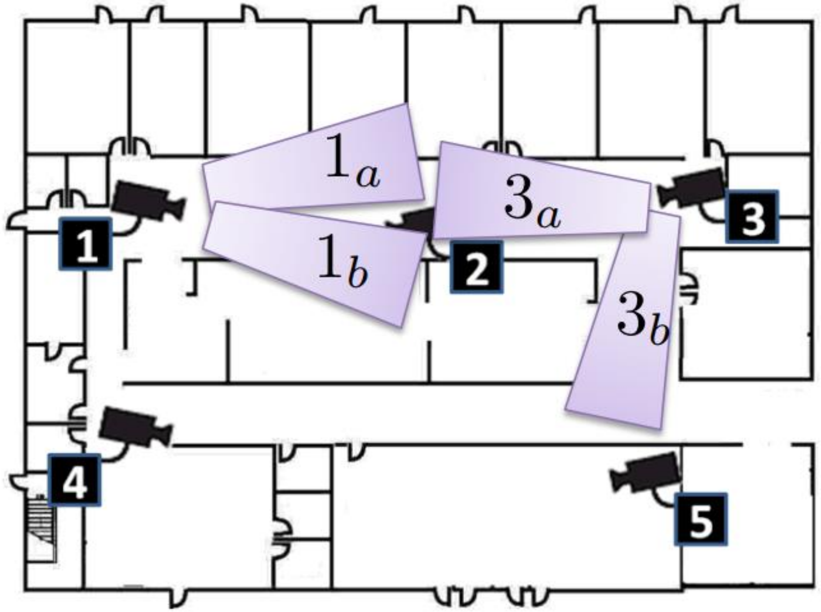
Monotone Submodular Maximization subject to Matroid Constraints

$S = \emptyset$

While $\exists e : S \cup e$ feasible

$\quad e^* \leftarrow \text{argmax}\{F(S \cup e) \mid S \cup e \text{ feasible}\}$

$\quad S \leftarrow S \cup e^*$



$|S \cap G_i| \leq 2.$

Monotone Submodular Maximization subject to Matroid Constraints

$S = \emptyset$

While $\exists e : S \cup e$ feasible

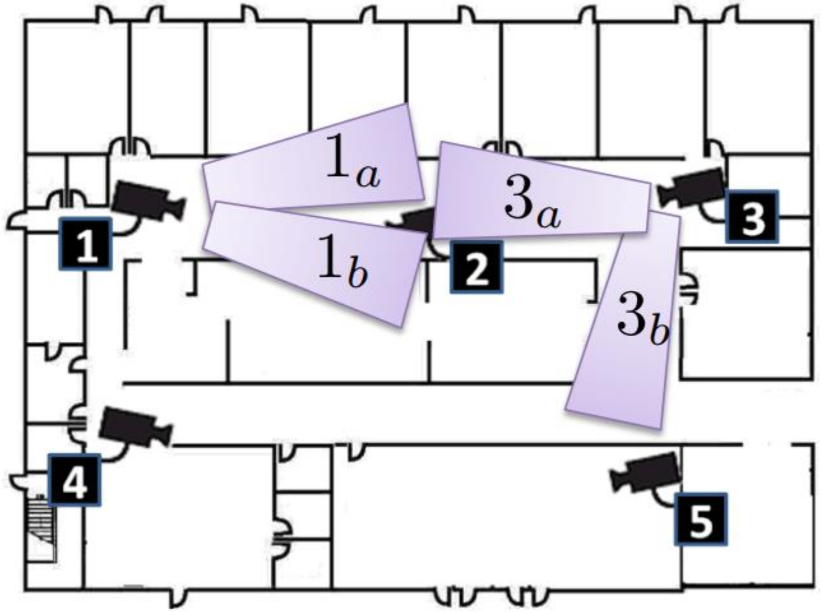
$e^* \leftarrow \text{argmax}\{F(S \cup e) \mid S \cup e \text{ feasible}\}$

$S \leftarrow S \cup e^*$

Theorem *(Nemhauser, Wolsey, Fisher 78)*

For monotone submodular functions:

$$F(S_{\text{greedy}}) \geq \frac{1}{2} F(S^*)$$



Non-Monotone Submodular Functions

Monotone Submodular Function

$$\max_{S \subseteq V, c(S) \leq \mathcal{B}} f(S)$$

Cost of Summary Subset S (e.g. size)

Start with $Y_0 = \emptyset$

for $i = 1$ *to* k **do**

Let $M_i = \text{argmax}_{X \subseteq V \setminus Y_{i-1}, |X| = k} \sum_{v \in X} f(v|Y_{i-1})$;

Choose y as a uniformly random element in M_i;

$Y_i = Y_{i-1} \cup y$;

return Y_k.

Theorem (Buchbinder et al 2014): The Randomized Greedy Algorithm achieves a 1/e approximation guarantee for Non-Monotone Submodular Maximization subject to cardinality constraints!

Budget into the Objective: Unconstrained Submodular Max

$A = \phi$ $B = V$

$A = \phi$

max $f(S)$

$c(S) \le B$

$\Downarrow$ modify

max $f(S) - \lambda c(S)$

Submodular Function

$$\max_{X \subseteq V} f(X) - \lambda c(X)$$

Cost of Summary Subset S (e.g. size)

Unconstrained Non-Monotone Submodular Maximization (USM)

Start: $A = \emptyset, \ B = \mathcal{V}$

for *i=1, ..., n* *//add or remove?*

ordering : $\sigma_1 \dots \sigma_n$

$A \subseteq B$.

add with probability

Linear time algo

$$\mathbb{P}(\text{add}) = \frac{\Delta_+}{\Delta_+ + \Delta_-}$$

if $\Delta_+ \ge 0$

$\boxed{\text{add to A}}$ or remove from B

$\Delta_+ = f(a|A)$

$\Delta_- = f(a|B \backslash a)$

$a \notin A, \ a \in B$

$f(a|B \backslash a) = f(B) - f(B \backslash a)$

Greedy can be arbitrarily bad!

Theorem (Buchbinder et al 2012): Bidirectional Greedy obtains a ½ approximation for USM!

$A = \emptyset$, $B = V$. $\quad 0 \qquad V = \{1,2,3,4\}$

$A = \{1\}$, $B = \{1,4\}$ $\quad 1$

$A = \{1\}$, $B = \{1,3,4\}$ $\quad 2$

$A = \{1,3\}$, $B = \{1,3,4\}$ $\quad 2$

$A = \{1,3,4\}$, $B = \{1,3,4\}$ $\quad 4$

$f(A) \geq 0$

Submodular + Dispersion Functions

Submodular + Dispersion

$$\max_{S \subseteq V, c(S) \leq \mathcal{B}} f(S)$$

Cost of Summary Subset S (e.g. size)

Problem Formulation

Initialization $S \leftarrow \emptyset$.
repeat
Pick an element $v^* \in \text{argmax}_{v \in V \setminus S} \frac{f(v \cup S) - f(S)}{c(v)}$
Update $S \leftarrow S \cup v^*$
until Reaching the budget, i.e., $c(S) > \mathcal{B}$

Greedy Algorithm

Theorem (Dasgupta et al 2013): The greedy algorithm achieves a ½ and ¼ approximation
For the Submodular + Disparity Sum and Submodular + Disparity Min Optimization Problems
Subject to cardinality constraints

Minorization-Maximization for Submodular Maximization

Submodular Maximization: $\max_{X \in \mathcal{C}} g(X)$

Unified Algorithm

Initialize $S_0 = \emptyset$;
for $i = 0, 1, \ldots$ **do**

subgradient

compute modular lower bound $\breve{g}_i = h^g_{S_i} \leq g$ based on S_i;

Set $S_{i+1} = \text{argmax}_{S \in \mathcal{C}} \breve{g}_i(S)$;

only need to solve linear-cost problem! ,

end

ICML - 2013

Theoretical Results?

[Subgradient]

Different Orderings => Known approximation Algorithms => Approximation bounds

Unconstrained Maximization

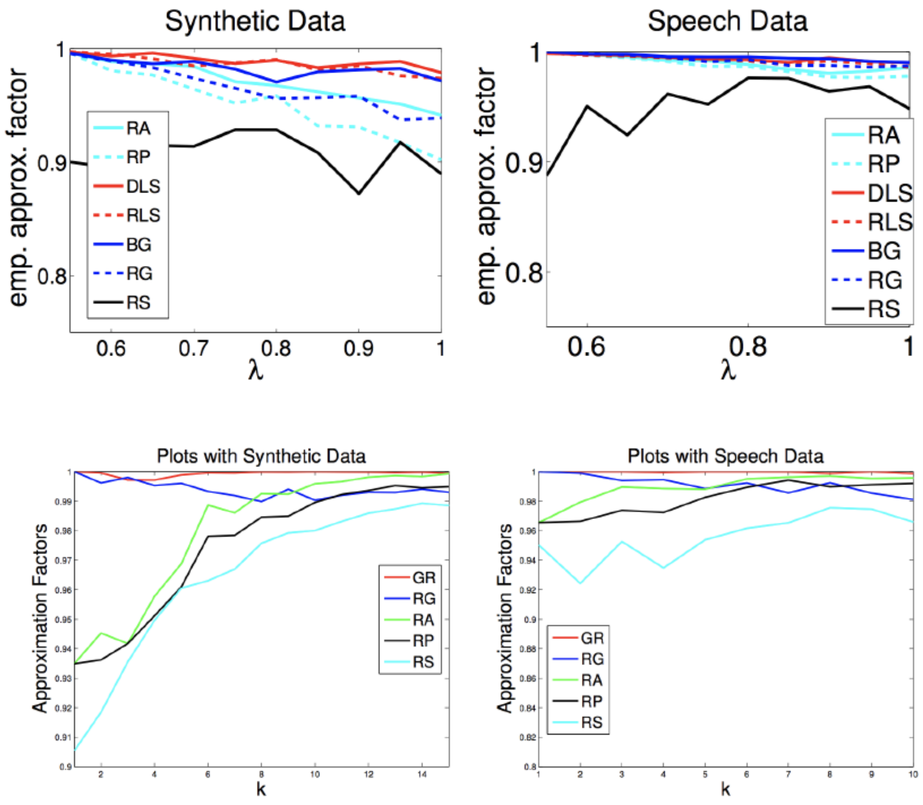
- Random Ordering (1/4)
- Local Search (1/3)
- Bidirectional Greedy (1/3)
- Randomized Greedy (1/2)
- ...

Constrained Maximization

- Random Ordering (k/n)
- Greedy (1 – 1/e)
- Randomized Greedy (1/e)
- Lazier than Lazy Greedy (1 – 1/e)
- ...

Iyer-Jegelka-Bilmes ICML 2013 **(Award Paper)**

Example 2: Empirically Performance



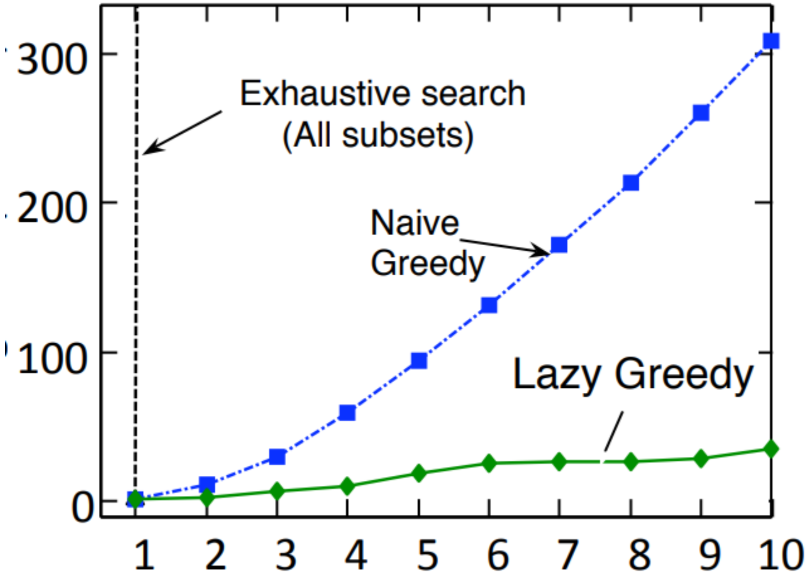
Observations:

- Greedy Algorithms often perform best in practice!
- Theoretically tightest algorithms do not often work the best!

Sol^n From 2k ≡ Sol^n from NG.

Implementational Tricks: Lazy greedy

- (Minoux 1982) that the greedy algorithm can be efficiently implemented via a lazy greedy algorithm.
- Idea is to accelerate the algorithm via priority queues and maintains upper bounds on the marginal gains
- In practice this brings the complexity to linear time



Complexity still $O(nkF)$ in worst case, where
F = Function Eval Complexity.

Can we do better?

Lazy Greedy Algorithm

for $i = 1:k$

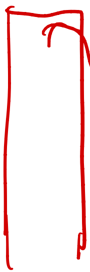
$$a_i = \underset{a \in V \setminus A_i}{\operatorname{argmax}} \frac{f(a|A_i)}{c(a)} \qquad O(n)$$

$$A_{i+1} = A_i \cup a_i$$

Total complexity = $O(nk)$

Initialize PQ: Compute all $f(i)$

sorted gains



Pop the top from PQ $\hat{a}_1$

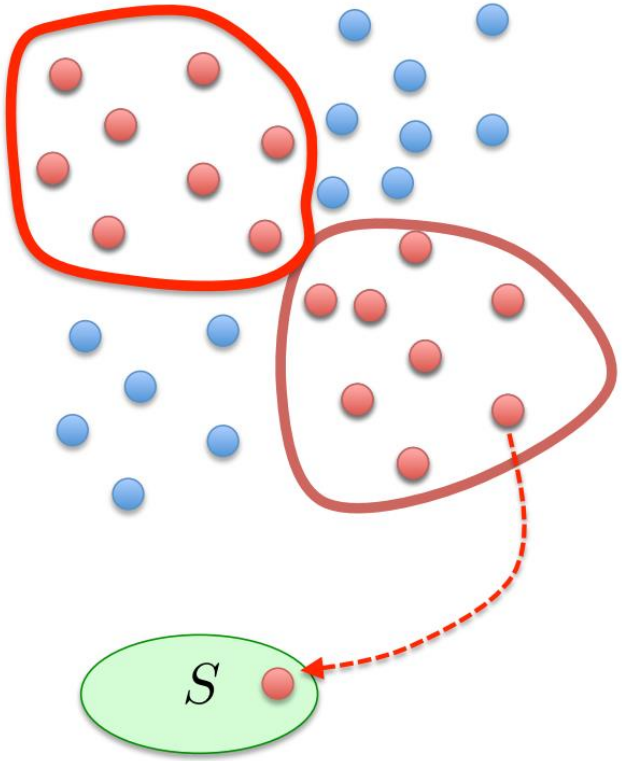
Step i of Greedy: A_i

Check: $f(\hat{a}_1 | A_i) \geq v_2 = f(\hat{a}_2 | A_{stale})$

$$\Rightarrow f(\hat{a}_1 | A_i) \geq f(j | A_i), \ \forall j \notin A_i$$

Making Lazy Greedy Faster: Stochastic Greedy

$K \cdot \frac{n}{k} \log \frac{1}{\varepsilon} = n \log \frac{1}{\varepsilon}$



$$\max_S \; F(S) \;\text{ s.t. }\; |S| \le k$$

for i=1…k:

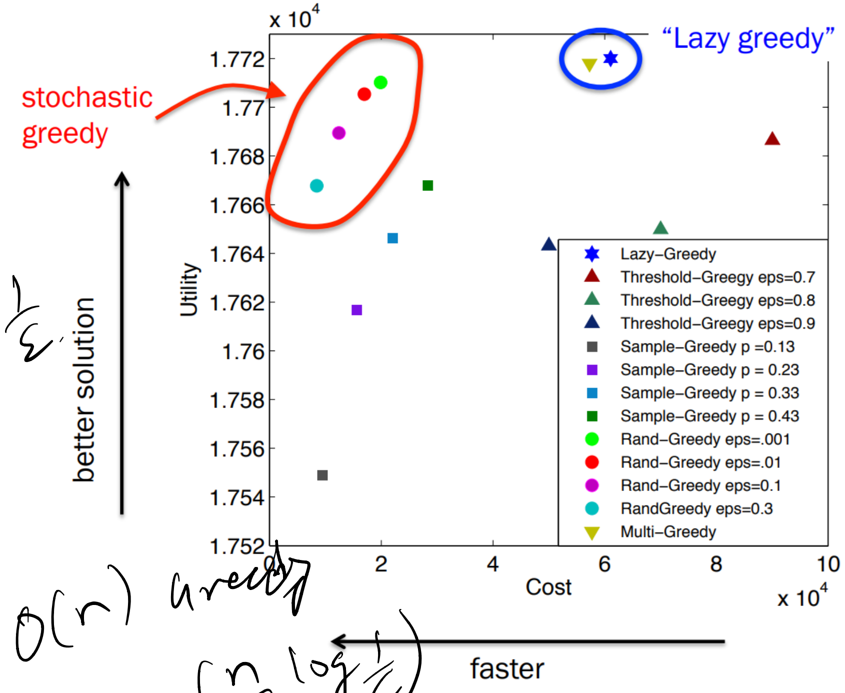
- randomly pick set T of size $\frac{n}{k} \log \frac{1}{\epsilon}$ $|T| = \frac{n}{k} \log \frac{1}{\varepsilon}$
- find best a element in T and add

$$a_i = \arg\max_{a \in T} F(a|S_{i-1}) \leftarrow O(n) \text{ greedy}$$

becomes $O\left(\frac{n}{k} \log \frac{1}{\varepsilon}\right)$

$$S_i \leftarrow S_{i-1} \cup \{a_i\}$$

$1 - \frac{1}{e} - \varepsilon$



AAAI 2015

Mirzasoleiman et al 2014, …

$f(e_i \mid S_v) \geq \text{tresh}_v$

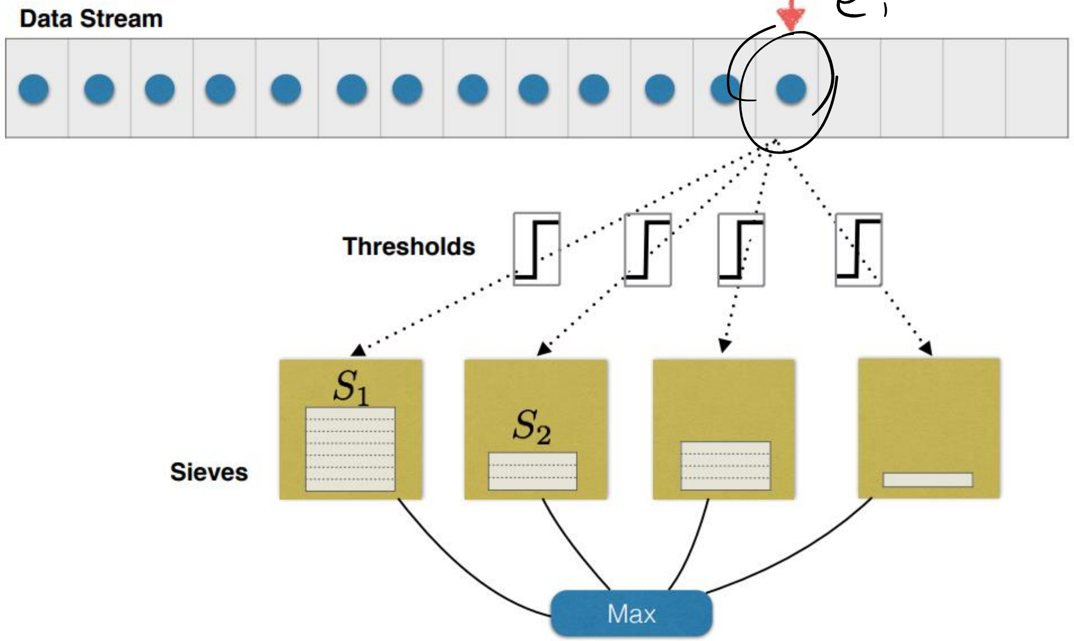
Streaming Submodular Maximization

Algorithm SIEVE-STREAMING

$O = \{(1+\epsilon)^i \mid i \in \mathbb{Z}\}$
For each $v \in O, S_v := \emptyset$ (maintain the sets only for the necessary v's lazily)
$m := 0$
for $i = 1$ to n **do**
  $m := \max(m, f(\{e_i\}))$
  $O_i = \{(1+\epsilon)^i \mid m \leq (1+\epsilon)^i \leq 2 \cdot k \cdot m\}$
  Delete all S_v such that $v \notin O_i$.
  for $v \in O_i$ **do**
    if $\triangle_f(e_i \mid S_v) \geq \frac{v/2 - f(S_v)}{k - |S_v|}$ and $|S_v| < k$ **then**
      $S_v := S_v \cup \{e_i\}$
return $\text{argmax}_{v \in O_n} f(S_v)$

100's Billions

e_i

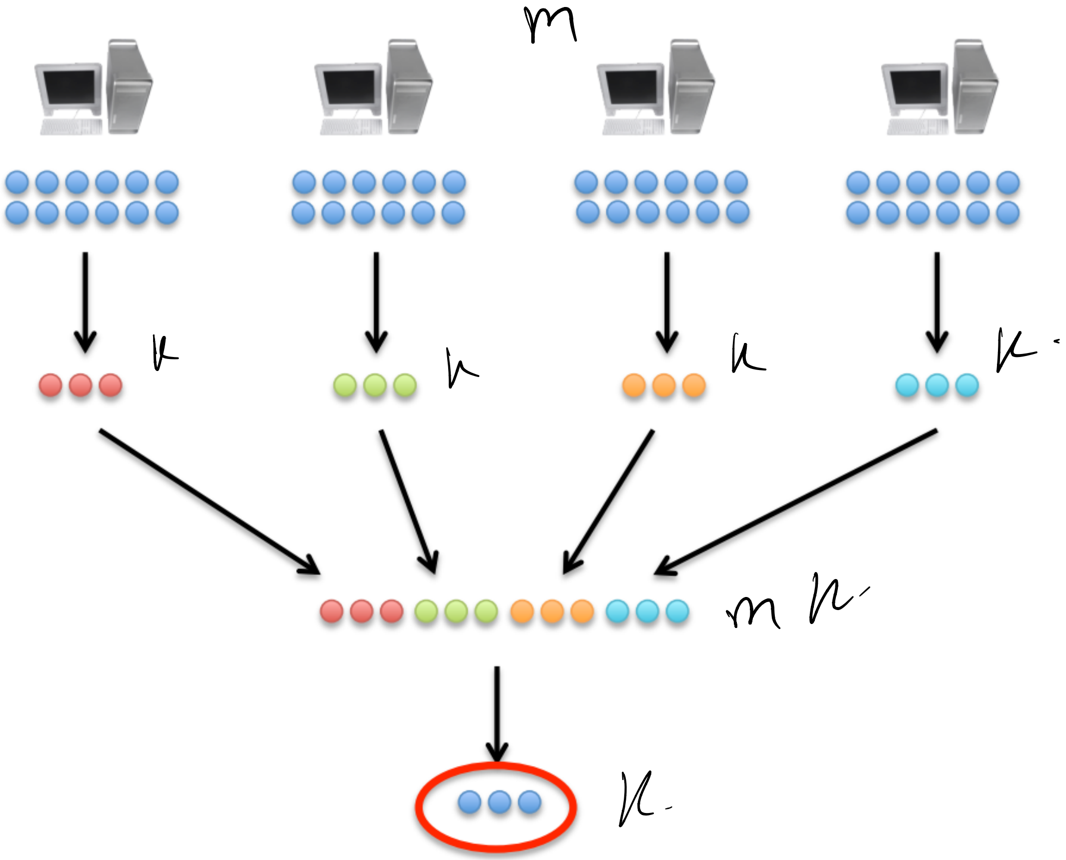


[Badanidiyuru et al 2014] **Theorem**: Sieve Streaming Is a single pass ½ approximation algorithm!

even more data ...

distributed greedy algorithm?

Distributed Greedy Algorithm I



Is this useful?

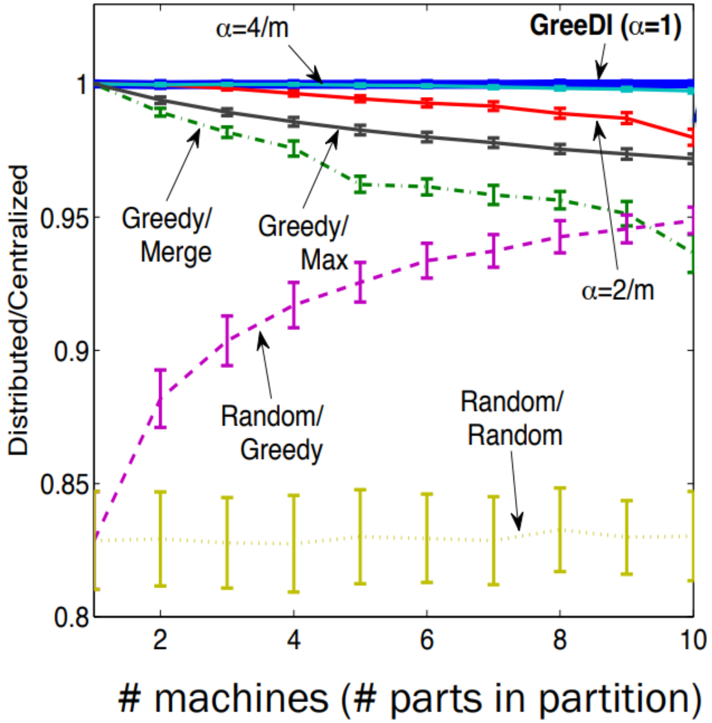
greedy is sequential.
pick in parallel??

pick k elements
on each machine.

combine and run
greedy again.

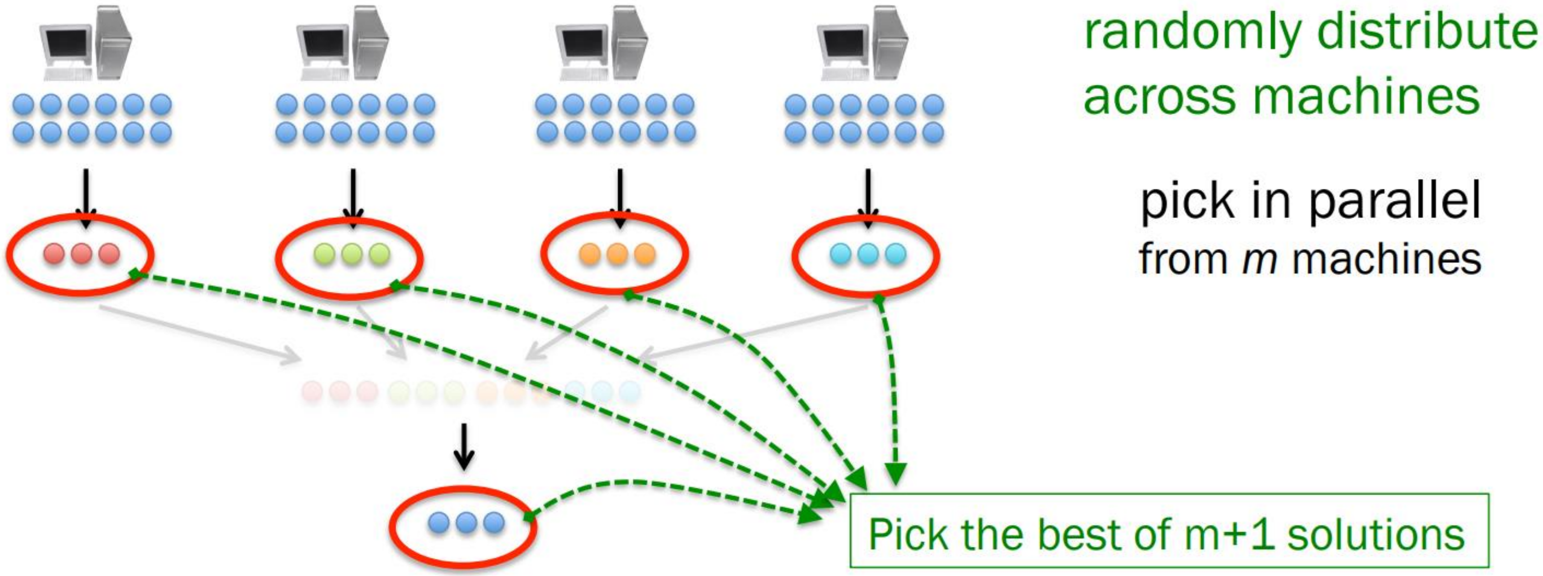
Approximation factor:

$$O\left(\frac{1}{\min\{\sqrt{k}, m\}}\right)$$



Mirzasoleiman et al 2013, …

Distributed Greedy Algorithm II



With greedy algorithm on both levels:
$\alpha = \beta = 1 - \frac{1}{e}$, overall factor:

$$\frac{1}{2}\left(1 - \frac{1}{e}\right)$$

$\frac{\alpha^2}{2\alpha} = \alpha/2$

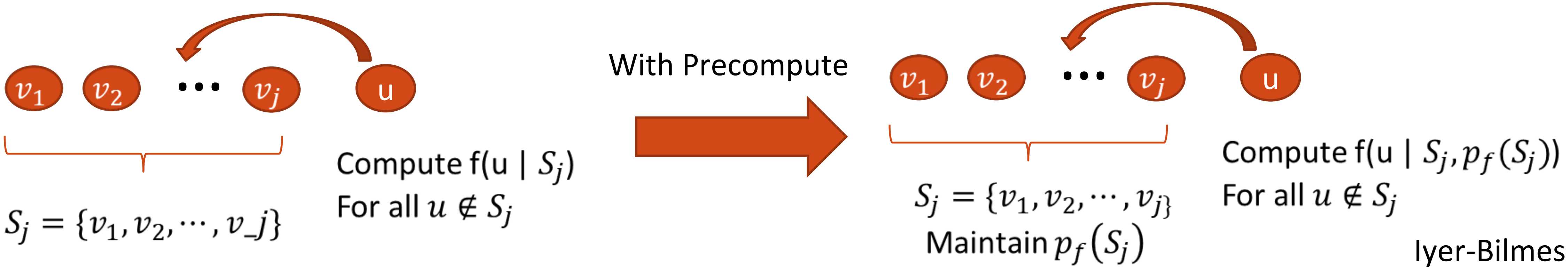
- each machine: $\alpha-$approximation algorithm
- level 2: $\beta-$ approximation algorithm

➔ overall approximation factor: $\mathbb{E}[F(\widehat{S})] \geq \dfrac{\alpha\beta}{\alpha+\beta} F(S^*)$

de Ponte Barbosa et al 2015, …

Memoization Framework to Implement Submodular Optimization Algorithms

- Idea of Memoization: Cache intermediate variables to be able to faster compute the gains of adding elements to sets
- Maintain precomputed statistics for each set X
- Given the precomputed statistics, the evaluation of the gains is much more efficient
- Significant speedups compared to simply using the value-oracle model



Iyer-Bilmes 2019

$$f(X) = \sum_{i=1}^{m} \psi(\phi_i(X))$$

features

Memoize $\underbrace{\phi_i(X)}$ for current set X

$$p_i^X = \phi_i(X)$$

$$\sum_{i=1}^{m} \left[\psi(p_i^X + \phi_i(a)) - \psi(p_i^X) \right]$$

Precomputed Statistics for Submodular Functions

Name	$f(X)$	$p_f(X)$	T_f^o	T_f^p
Facility Location	$\sum_{i \in V} \max_{k \in X} s_{ik}$	$[\max_{k \in X} s_{ik}, i \in V]$	$O(n^2)$	$O(n)$
Saturated Coverage	$\sum_{i \in V} \min\{\sum_{j \in X} s_{ij}, \alpha_i\}$	$[\sum_{j \in X} s_{ij}, i \in V]$	$O(n^2)$	$O(n)$
Graph Cut	$\lambda \sum_{i \in V} \sum_{j \in X} s_{ij} - \sum_{i,j \in X} s_{ij}$	$[\sum_{j \in X} s_{ij}, i \in V]$	$O(n^2)$	$O(n)$
Feature Based	$\sum_{i \in \mathcal{F}} \psi(w_i(X))$	$[w_i(X), i \in \mathcal{F}]$	$O(n\|\mathcal{F}\|)$	$O(\|\mathcal{F}\|)$
Set Cover	$w(\cup_{i \in X} U_i)$	$\cup_{i \in X} U_i$	$O(n\|U\|$	$\|U\|$
Prob. Set Cover	$\sum_{i \in \mathcal{U}} w_i[1 - \prod_{k \in X}(1 - p_{ik})]$	$[\prod_{k \in X}(1 - p_{ik}), i \in \mathcal{U}]$	$O(n\|\mathcal{U}\|)$	$O(\|\mathcal{U}\|)$
DPP	$\log\det(S_X))$	SVD(S_X)	$O(\|X\|^3)$	$O(\|X\|^2)$
Dispersion Min	$\min_{k,l \in X, k \neq l} d_{kl}$	$\min_{k,l \in X, k \neq l} d_{kl}$	$O(\|X\|^2)$	$O(\|X\|)$
Dispersion Sum	$\sum_{k,l \in X} d_{kl}$	$[\sum_{k \in X} d_{kl}, l \in X]$	$O(\|X\|^2)$	$O(\|X\|)$
Dispersion Min-Sum	$\sum_{k \in X} \min_{l \in X} d_{kl}$	$[\min_{k \in X} d_{kl}, l \in X]$	$O(\|X\|^2)$	$O(\|X\|)$

Table 1. List of Submodular Functions used, with the precompute statistics $p_f(X)$, gain evaluated using the precomputed statistics $p_f(X)$ and finally T_o^f as the cost of evaluation the function without memoization and T_p^f as the cost with memoization. It is easy to see that memoization saves an order of magnitude in computation.

(Iyer-Bilmes 2019)

Benefit of Memoization in Practice

Algorithm/Function	n = 4620	n = 192000	n = 10.2M
Lazy Greedy (Fac-Loc)	0.31	4.1	212.1
Lazy Greedy (Feat Based)	0.16	2.7	98.1
Min Norm Point (Complexity)	7.1	88.1	4321
SCSC (Feat-Based, Complexity)	1.2	12.3	567.1

Scaling across Dataset sizes

Algorithm/Function	Memoization	No Memoization
Lazy Greedy (Fac-Loc)	0.31	48
Lazy Greedy (Feat Based)	0.16	21
Min Norm Point (Complexity)	7.1	2023.1
SCSC (Feat-Based, Complexity)	1.2	101.2

Memoization vs No Memoization

Function	Ours	Gygli et al 2016	SFO (Krause et al 2008)
Facility Location	0.34	26.8	52
Graph Cut	0.39	35.7	43.2

SMTK vs Other ToolKits

Big Picture of Submodular Optimization

Submodular Functions

Minimization
- *Convex*
- Complexity/Cooperation/Attraction
- Ex: Bipartite Neighbor, Graph Cut/Ising, Concave over Mod

Maximization
- *Concave*
- Diversity/Representation/Coverage/Information
- Ex: Facility Location, DPPs, Set Cover, Dispersion etc.

Learning Submodular Functions

Minimization
- Log-Supermodular
- Ising Models
- Sampling, Mode, Approximate Partition Function

Maximization
- Log-Submodular
- DPPs
- Sampling, Partition Fn, Maximum Likelihood
- Max-Margin learning

Optimization Algorithms

Minimization
- SFM Algorithms
- Majorization-Minimization
- Continuous Relaxations (Lovasz Extension)

Maximization
- Greedy Algorithms
- Distributed Greedy/Streaming Greedy
- Randomized Greedy
- Bidirectional Greedy

Applications

Minimization
- Image Segmentation
- Sparse Reconstruction
- Limited Complexity Data Selection
- …

Maximization
- Data Subset Selection
- Image/Video Summarization
- Data Partitioning,
- …

Additional Reading

- Jeff's Class: http://j.ee.washington.edu/~bilmes/classes/ee596a_fall_2014/
- Stefanie Jegelka & Andreas Krause's 2013 ICML tutorial: http://techtalks.tv/talks/submodularity-in-machine-learning-new-directions-part-i/58125/
- Jeff's NIPS, 2013 tutorial on submodularity: http://melodi.ee.washington.edu/~bilmes/pgs/b2hd-bilmes2013-nips-tutorial.html and http://youtu.be/c4rBof38nKQ
- Andreas Krause's web page: http://submodularity.org
- Francis Bach's updated 2013 text: http://hal.archives-ouvertes.fr/docs/00/87/06/09/PDF/submodular_fot_revisd_hal.pdf
- My WACV 2019 Tutorial: https://sites.google.com/view/wacv2019summarization/home
- Tom McCormick's overview paper on submodular minimization: http://people.commerce.ubc.ca/faculty/mccormick/sfmchap8a.pdf

References

1. Mirzasoleiman, Baharan, Stefanie Jegelka, and Andreas Krause. **Streaming non-monotone submodular maximization: Personalized video summarization on the fly**. *arXiv preprint arXiv:1706.03583* (2017).

2. Rishabh Iyer and Jeff Bilmes, **A Memoization Framework for Scaling Submodular Optimization to Large Scale Problems,** Artificial Intelligence and Statistics (AISTATS) 2019

3. Rishabh Iyer, Stefanie Jegelka, Jeff Bilmes, **Fast semidifferential-based submodular function optimization**, International Conference on Machine Learning (ICML) 2013

4. Mirzasoleiman, Baharan, et al. **Distributed submodular maximization: Identifying representative elements in massive data.** Advances in Neural Information Processing Systems. 2013.

5. Barbosa, R., Ene, A., Nguyen, H., & Ward, J. (2015, June). **The power of randomization: Distributed submodular maximization on massive datasets**. In International Conference on Machine Learning (pp. 1236-1244).

6. Fisher, Marshall L., George L. Nemhauser, and Laurence A. Wolsey. **An analysis of approximations for maximizing submodular set functions**." Polyhedral combinatorics. Springer, Berlin, Heidelberg, 1978. 73-87.

7. Mirzasoleiman, Baharan, et al. **Lazier Than Lazy Greedy.** AAAI. 2015.

8. Buchbinder, Niv, et al. **Submodular maximization with cardinality constraints.** Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 2014.