Massachusetts Institute of Technology

CSAIL

IDSS MIT INSTITUTE FOR DATA, SYSTEMS, AND SOCIETY

Submodularity and Machine Learning

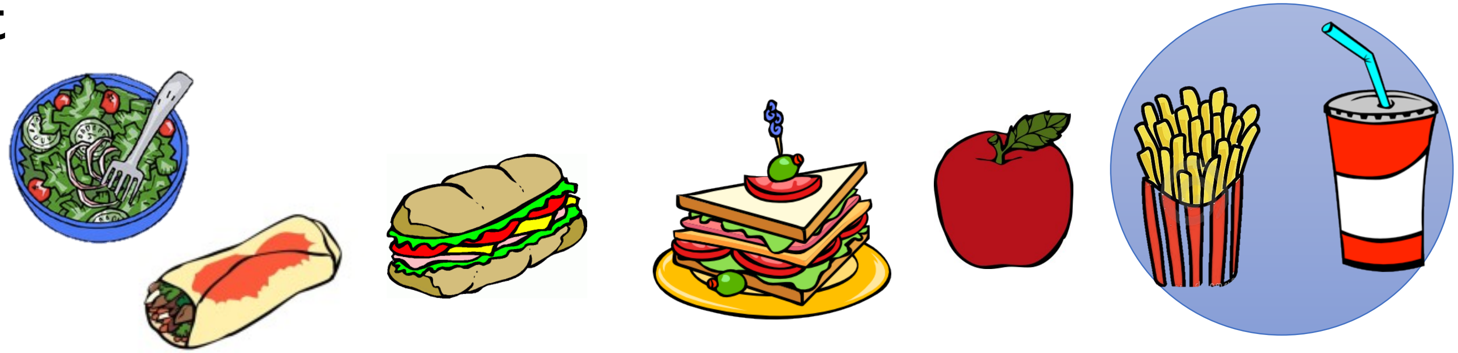
MLSS Tübingen, June 2017

Stefanie Jegelka
MIT

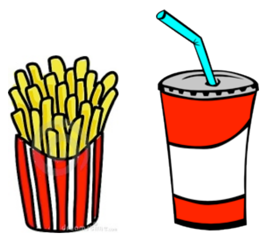
slides: people.csail.mit.edu/stefje/mlss/tuebingen2017.pdf
papers etc: people.csail.mit.edu/stefje/mlss/literature.pdf

Set functions

ground set

$\mathcal{V} = \{$  $\}$

$$F : 2^{\mathcal{V}} \rightarrow \mathbb{R}$$

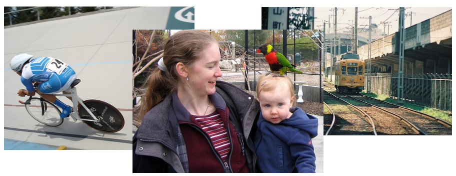
$F($  $) =$ cost of buying items together, or

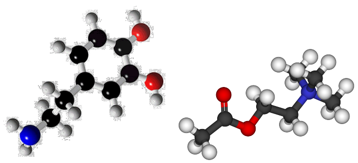
utility, or

probability, …

Machine Learning

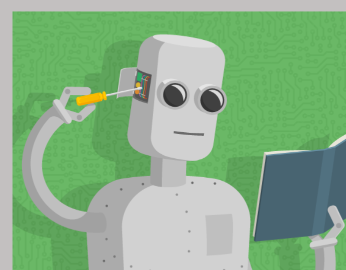
training examples





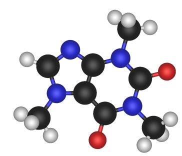
learn model

$$f(x, w)$$



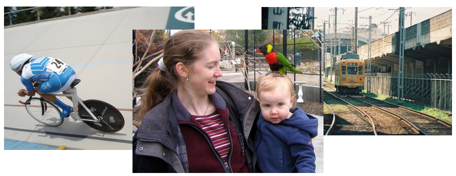
prediction

$f($  $, \hat{w}) =$ train

$f($  $, \hat{w}) =$ likely awakening effect

Machine Learning

training examples



learn model

$f(x, w)$

prediction

$f($ $, \hat{w}) =$ train

Informative Subsets

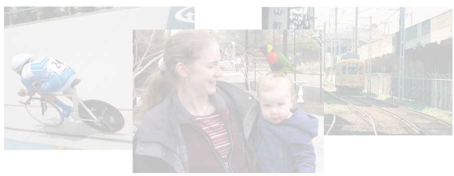
- Compression
- Summarization

- Placing sensors
- Designing experiments

$$F(S) = \text{``information''}$$

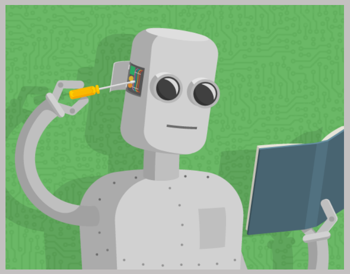
Machine Learning

training examples



learn model

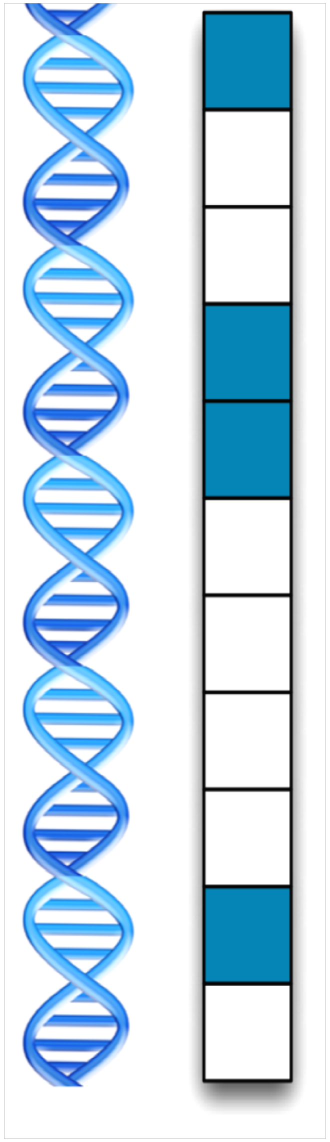
$$f(x, w)$$



prediction

$f($  $, \hat{w}) =$ train

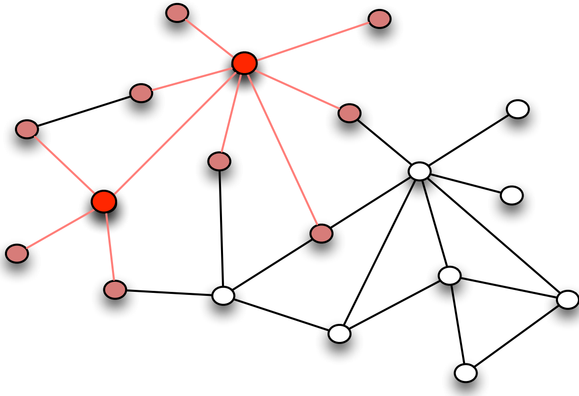
Variable (Coordinate) Selection



Only use few coordinates of x in $f(x, w)$

$$f(x, w) = \sum_{i=1}^{d} w_i x_i$$

$F(S) =$ "coherence"



Machine Learning

training examples

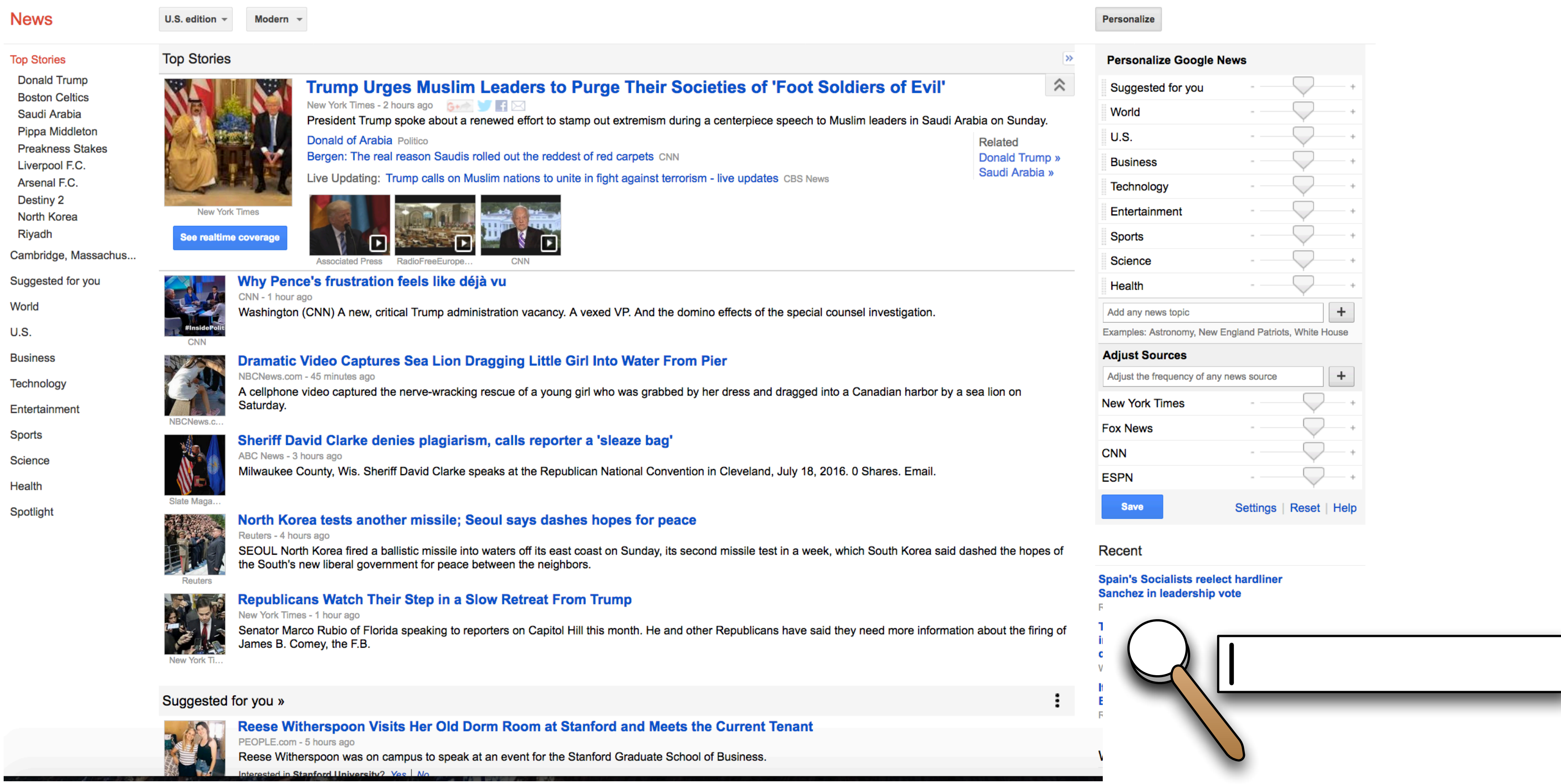
learn model

$$f(x, w)$$

prediction

$f($  $, \hat{w}) =$ train

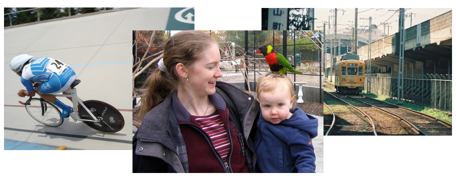
Summarization & Recommendation



$$F(S) = \text{relevance} + \text{diversity or coverage}$$

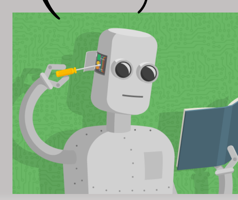
Machine Learning

training examples



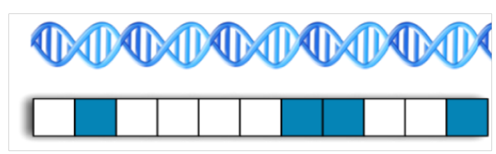
learn model

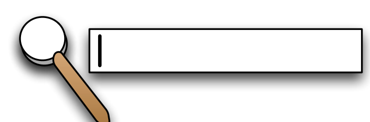
$$f(x, w)$$



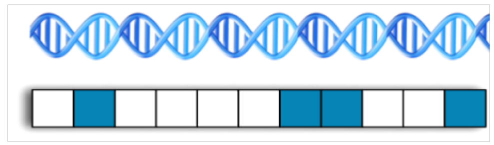
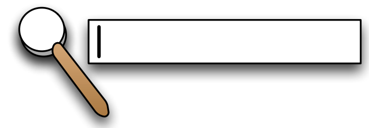
prediction

$f($  $, \hat{w}) =$ train





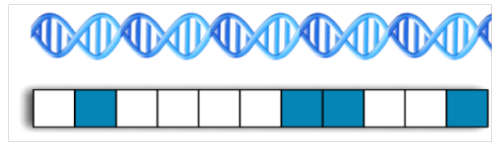
Machine Learning and Set functions

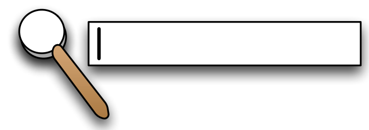




Common formalization: Find a set S that
maximizes / minimizes a set function F(S)

- difficult: 2^{100} possible subsets for just 100 items ☹
- This is large!
  fold a sheet of paper 100x. Height of the final pile:
  2^{100}x 0.1mm = **13.4 billion light years**!

Machine Learning and Set functions





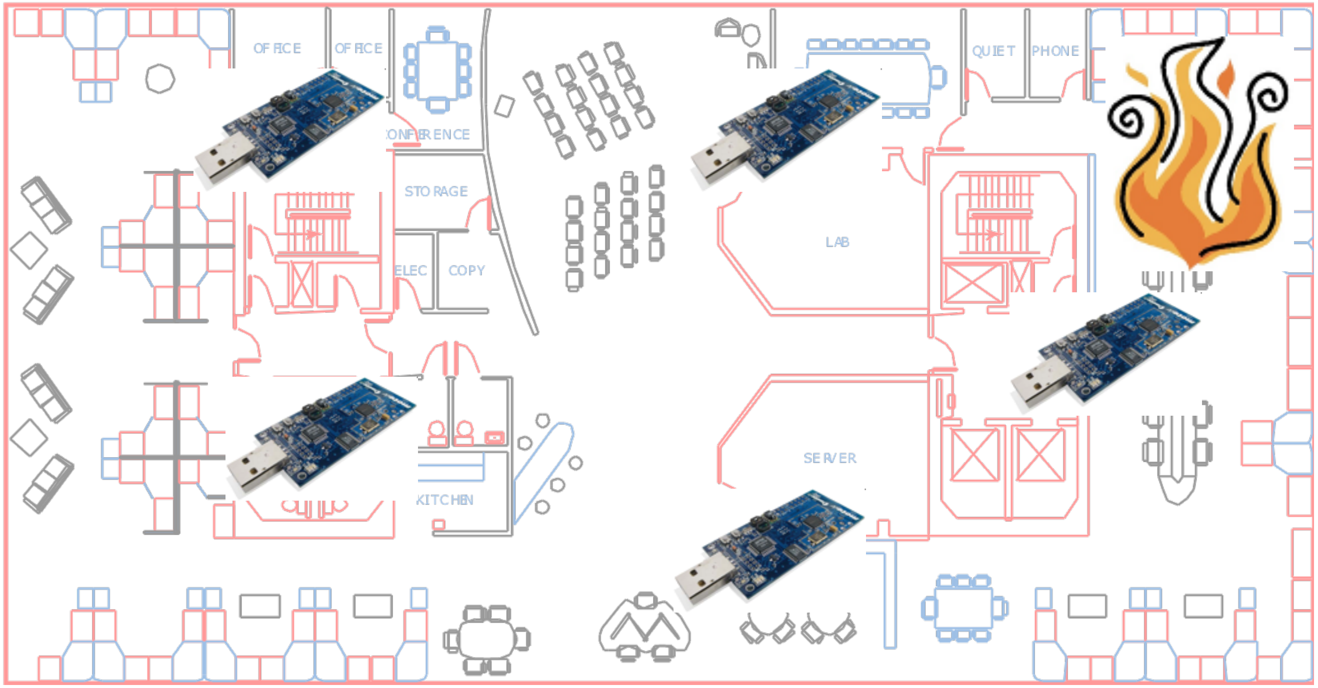
Common formalization: Find a set S that
maximizes / minimizes a set function F(S)

- difficult: 2^{100} possible subsets for just 100 items ☹
- Special properties help! ("10cm") ☺
  Submodularity

Roadmap

- What is submodularity and where does it come up?
- Optimization with submodular functions
- Further connections & directions

Sensing



$\mathcal{V}$ = all possible locations

F(S) = information gained from locations in *S*

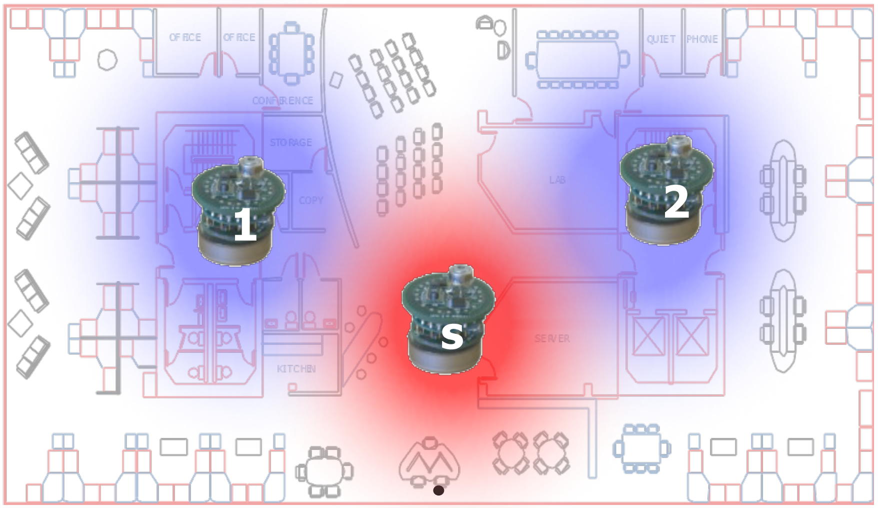
Marginal gain

- Given set function $F : 2^V \to \mathbb{R}$
- Marginal gain: $F(s|A) = F(A \cup \{s\}) - F(A)$

new sensor s

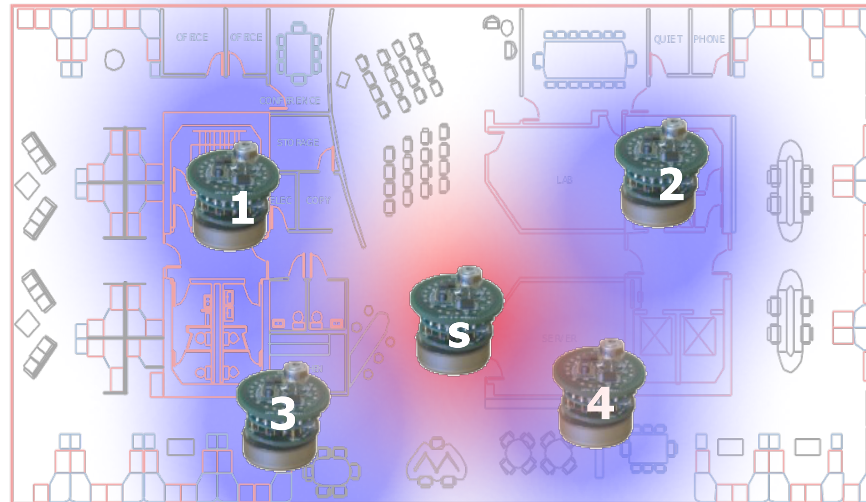
Diminishing gains

placement A = {1,2}



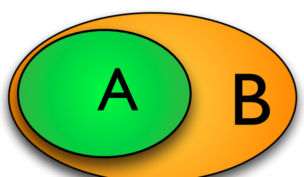
Big gain

placement B = {1,2,3,4}



small gain

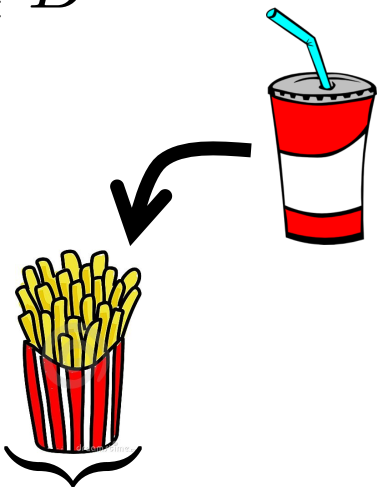
for all $A \subseteq B$
and s not in B



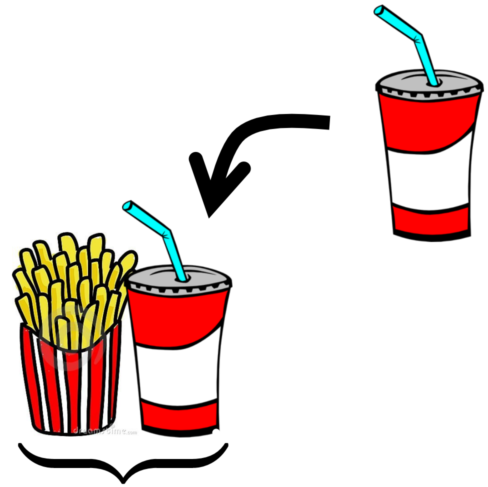
$$F(A \cup s) - F(A) \quad \geq \quad F(B \cup s) - F(B)$$

Diminishing marginal costs

$A \subseteq B$



A



B

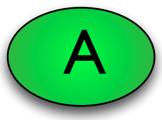
$$F(A \cup s) - F(A) \quad \geq \quad F(B \cup s) - F(B)$$

extra cost:
one drink

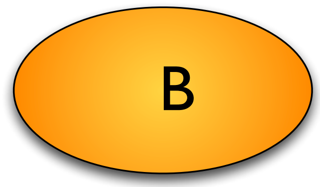
extra cost:
free refill ☺

Submodular set functions

- Diminishing gains: for all $A \subseteq B$



+ • e

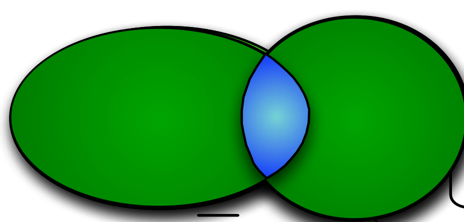


+ • e

$$F(A \cup e) - F(A) \geq F(B \cup e) - F(B)$$

- Union-Intersection: for all $S, T \subseteq \mathcal{V}$

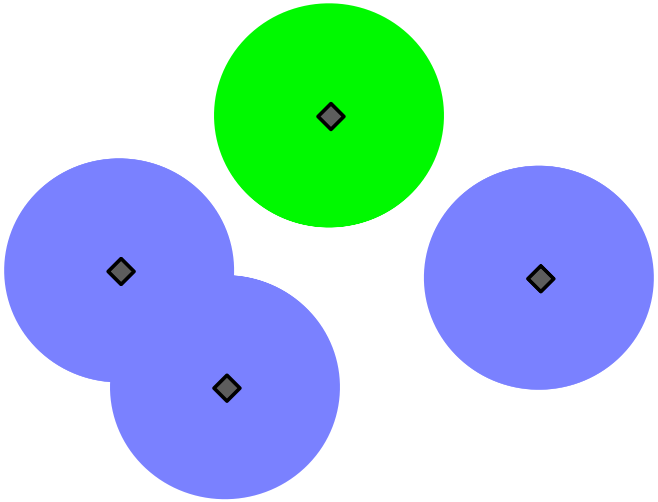
$F(S) + F(T)$

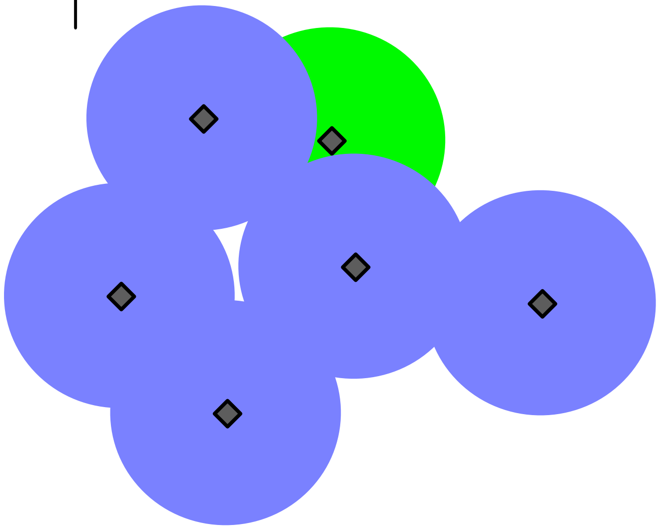


$\cup T) + F(S \cap T)$

Example: cover

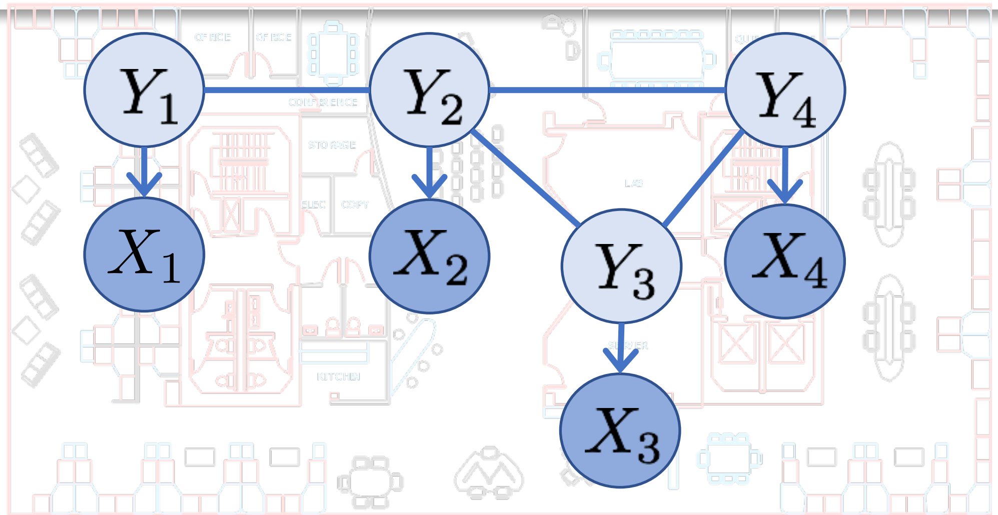
$$F(S) = \left| \bigcup_{v \in S} \text{area}(v) \right|$$





$$F(A \cup v) - F(A) \quad \geq \quad F(B \cup v) - F(B)$$

Example: sensing



- $\mathcal{V}$ = random variables we can possibly observe
- Utility to have sensors in locations *A*:

$$F(A) = H(\mathbf{Y}) - H(\mathbf{Y} \mid \mathbf{X}_A) = I(\mathbf{Y}; \mathbf{X}_A)$$

***uncertainty** about temperature* ***before** sensing*

***uncertainty** about temperature* ***after** sensing*

Mutual information

Example: entropy

$X_1, \ldots, X_n$ discrete random variables

$$F(S) = H(X_S) = \text{ joint entropy of variables indexed by } S$$

Exercise: meaning of diminishing returns here?

Example: entropy

$X_1, \ldots, X_n$ discrete random variables

$$F(S) = H(X_S) = \text{joint entropy of variables indexed by } S$$

$$A \subset B$$

$$\begin{aligned} H(X_{A \cup e}) - H(X_A) &= H(X_e | X_A) \\ &\leq H(X_e | X_B) \quad \text{"information never hurts"} \\ &= H(X_{B \cup e}) - H(X_B) \end{aligned}$$

discrete entropy is submodular!

Recommendation & Summarization

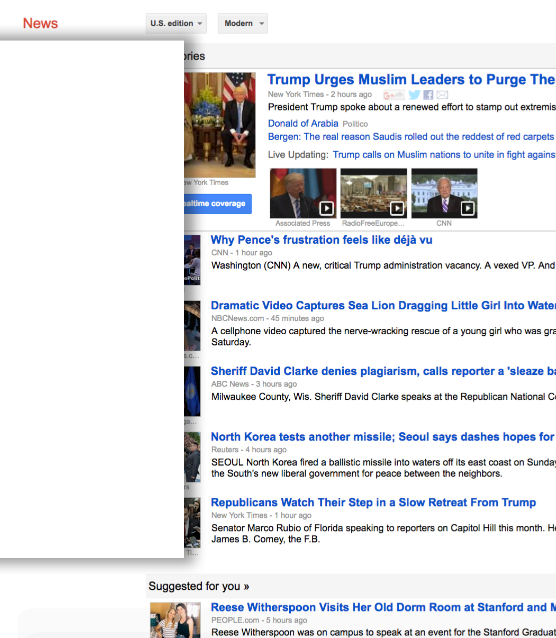
If you bought t
you may want
to add …

We want:

relevance & coverage

diversity

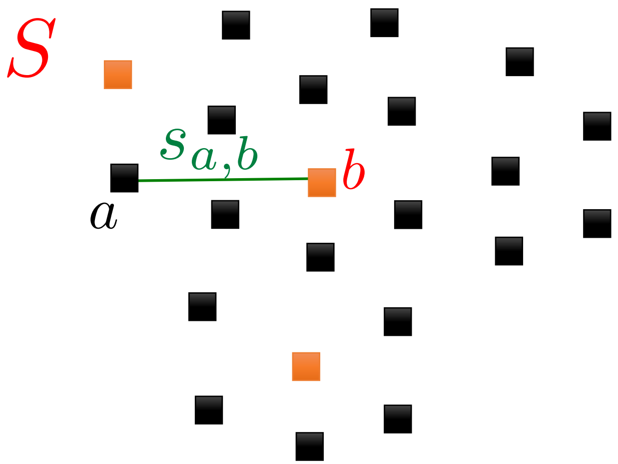
personalization

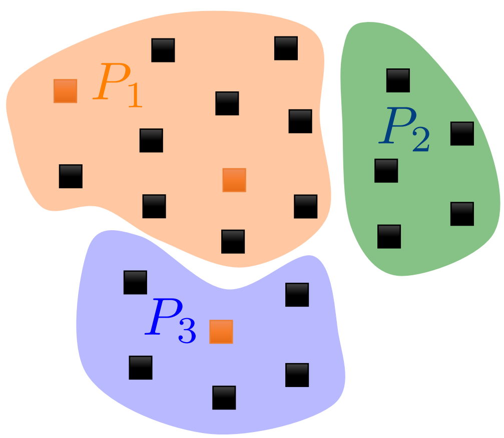


What could F(S) be?

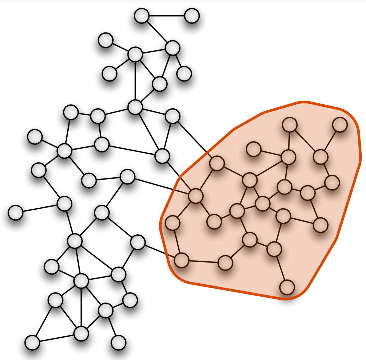
$$F(S) = \sum_{a \in \mathcal{V}} \max_{b \in S} s_{a,b}$$

$$F(S) = \sum_{j} \sqrt{|S \cap P_j|}$$





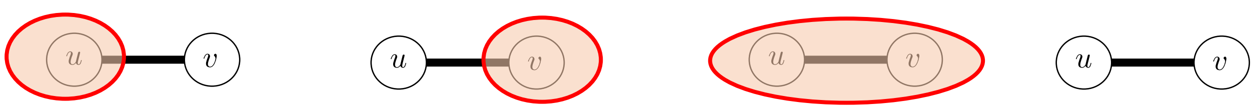
Example: graph cuts



cut for one edge:

$$F(S) = \sum_{u \in S, v \notin S} w_{uv}$$

$$\underbrace{F(\{u\})}_{F(S)} + \underbrace{F(\{v\})}_{F(T)} \geq \underbrace{F(\{u, v\}) + F(\emptyset)}_{F(S \cup T) + F(S \cap T)}$$



w_{uv} w_{uv} 0 0

- cut of one edge is submodular!
- large graph: sum of edges

sum of submodular functions is submodular

Examples of submodular functions

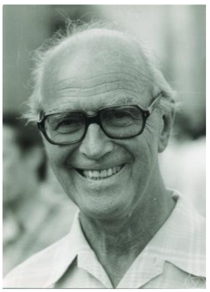
- Discrete entropy
- Mutual information

- Matrix rank (as a function of columns)

- Coverage
- Spread in social networks

- Graph cuts
- … many others!

Submodular functions (almost) everywhere!



THEORY OF CAPACITIES ([1])

by Gustave CHOQUET ([2])([3]).

INTRODUCTION

This work originated from the following]
significance had been emphasized by M. Brelot
Is the interior Newtonian capacity of an ar]
subset X of the space R^3 equal to the exte
capacity of X?

Submodular Functions, Matroids, and Certain Polyhedra⋆

Jack Edmonds

National Bureau of Standards, Washington, D.C.,

I

The viewpoint of the subject of matroids, and related areas of lattice theory, has always been, in one way or another, abstraction of algebraic dependence or, equivalently, abstraction of the incidence relations in geometric representations of algebra. Often one of the main derived facts is that all bases have the same cardinality. (See Van der Waerden, Section 33.)

Cores of Convex Games [1])

By LLOYD S. SHAPLEY [2])

ct: The core of an *n*-person game is the set of feasible outcomes tl
coalition of players. A convex game is defined as one that is ba
paper it is shown that the core of a convex game is not empty
structure. It is further shown that certain other cooperative solu
way to the core: The value of a convex game is the center of gravi

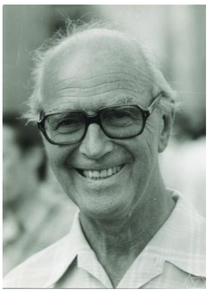
Submodular functions and convexity

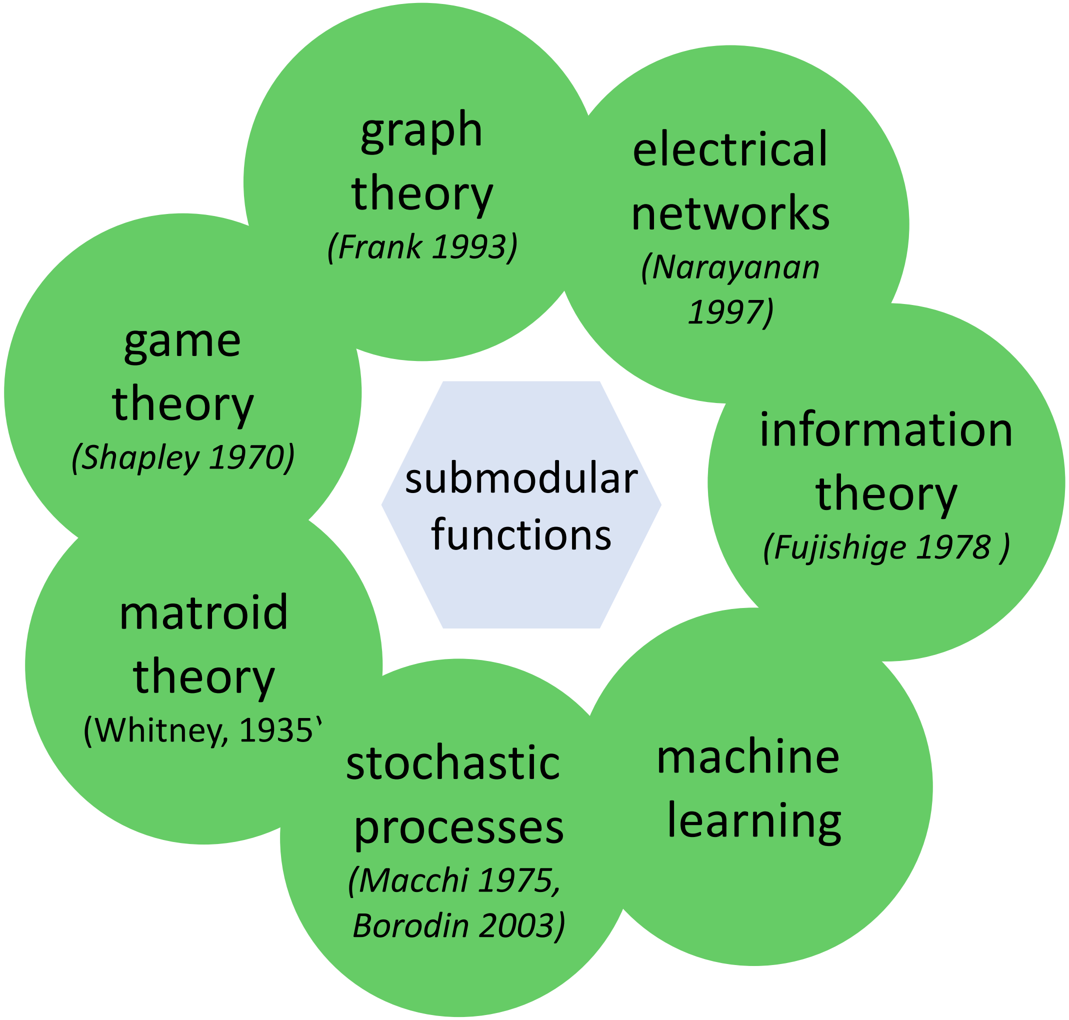
L. Lovász
Eötvös Loránd University, Department of Analysis I, Múzeu
Budapest, Hungary

0. Introduction

In "continuous" optimization convex functions play a central role. Bes
mentary tools like differentiation, various methods for finding the min
a convex function constitute the main body of nonlinear optimizat

Submodular functions (almost) everywhere!





Why are convex functions so important? *(Lovász, 1983)*

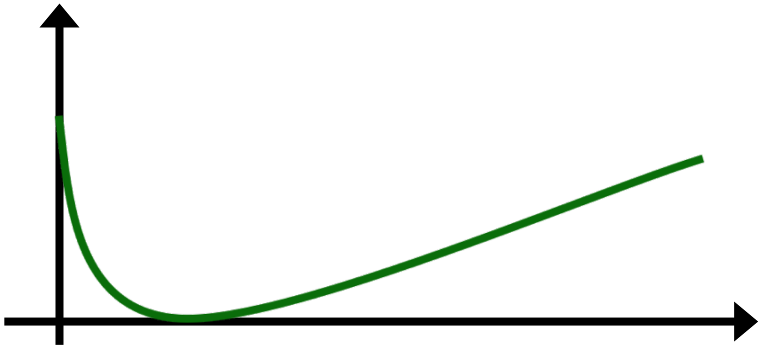
- "**occur in many models** in economy, engineering and other sciences", "often the only nontrivial property that can be stated in general"
- **preserved** under many operations and transformations: larger effective range of results
- sufficient structure for a "mathematically beautiful and practically useful **theory**"
- efficient **minimization**

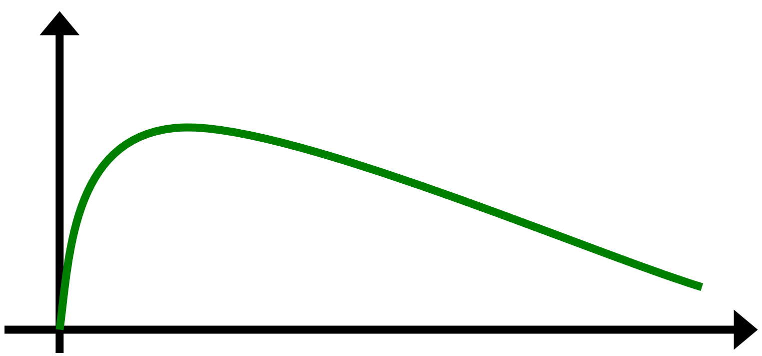
"It is less apparent, but we claim and hope to prove to a certain extent, that a similar role is played in discrete optimization by *submodular set-functions*" […]
they share the above four properties.

Submodularity …

discrete convexity ….

convex relaxation,
duality





… or concavity?

diminishing “derivative”

Roadmap

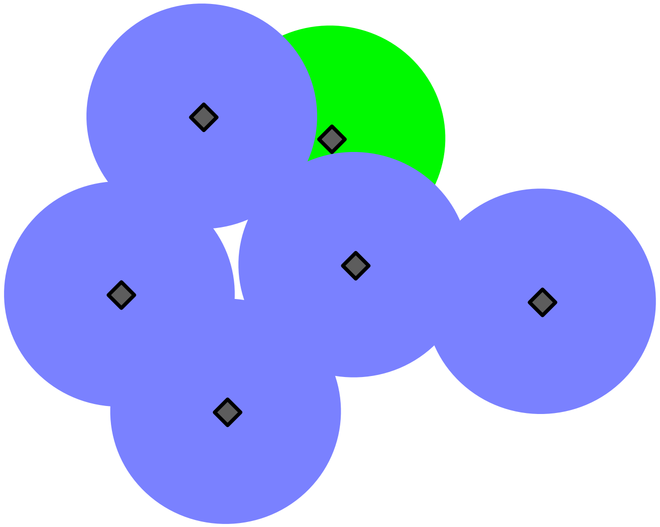
✓What is submodularity and where does it comes up?

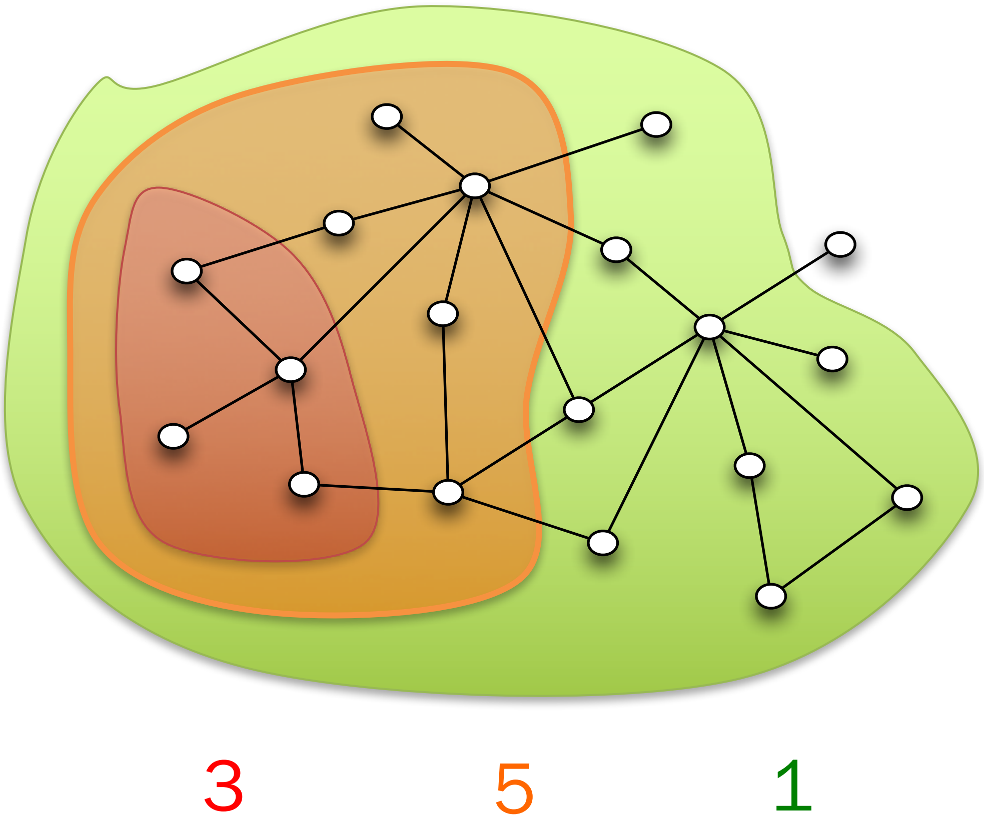
- Optimization with submodular functions

- Further connections & directions

Monotonicity

$$\text{if} \quad S \subseteq T \quad \text{then} \quad F(S) \leq F(T)$$





Maximizing a submodular function?

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

NP-hard ☹

Maximizing a submodular function?

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

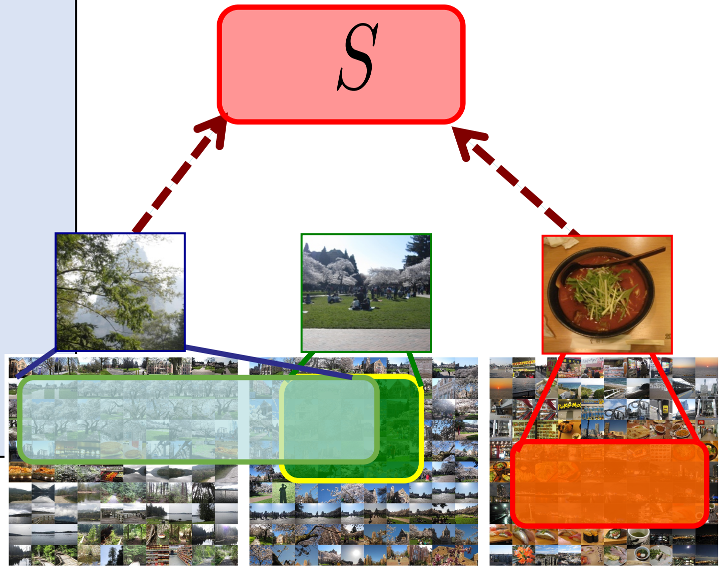
greedy algorithm:

$S_0 = \emptyset$

for *i = 0, …, k-1*

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

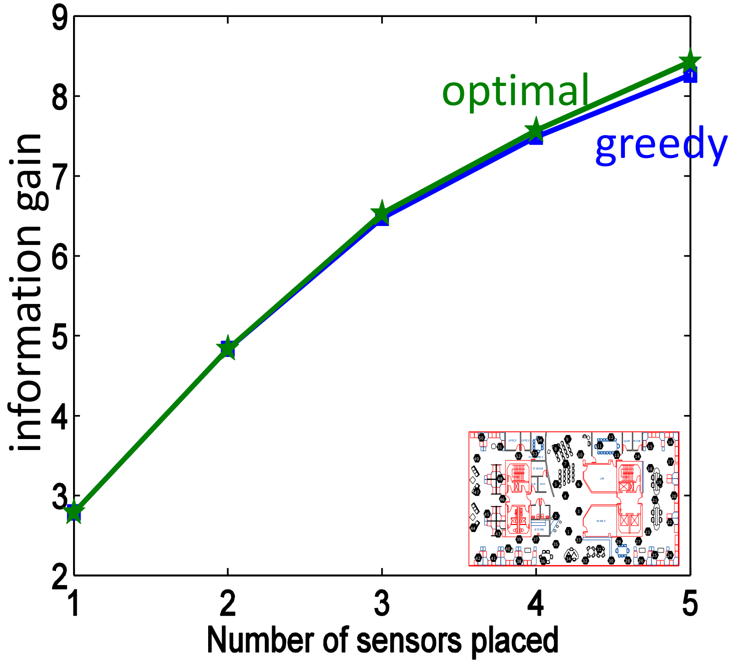
$$S_{i+1} = S_i \cup \{e^*\}$$



How "good" is S_k ?

How good is greedy?

empirically:

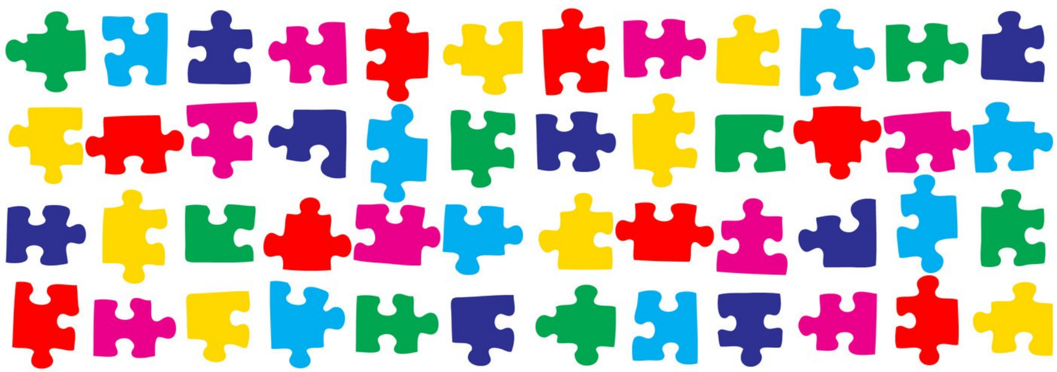


Theorem *(Nemhauser, Wolsey, Fisher 1978)*:
If F is monotone submodular, then Greedy is guaranteed to achieve at least 63% of optimum:

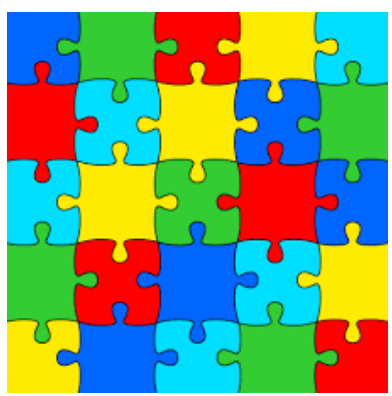
$$F(S_k) \geq \left(1 - \frac{1}{e}\right) F(S^*)$$

Why is this amazing?
Does it always work?

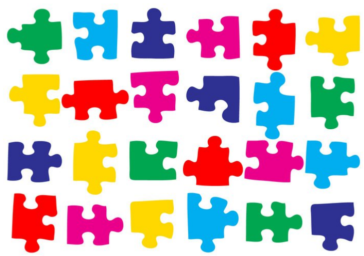
Greedy can fail … without submodularity

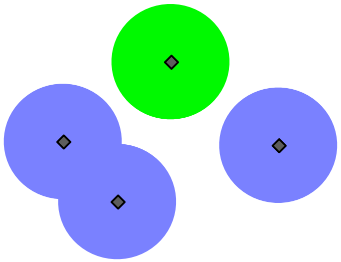


But: this never happens with diminishing returns! ☺

If $S =$  then $F(S) = 100$.
Otherwise, $F(S) = 0$

Recap: why does plain greedy work?





1. **Submodularity**: global information from local information
   Marginal gain of single item gives information about global value

2. **Monotonicity**: items can never harm (= reduce F)

Beyond greedy?

- Other constraints?
- Non-monotone functions?
- Large-scale greedy?

Greedy++

More complex constraints: budget

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

1. run greedy: S_{gr}
2. run a modified greedy: S_{mod}

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

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

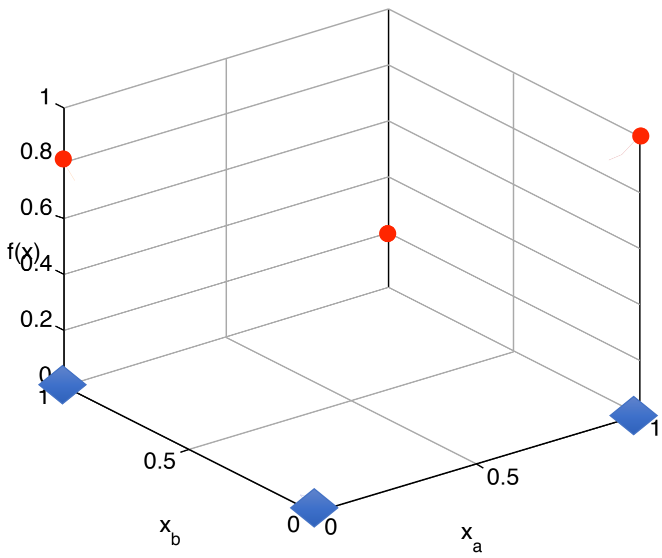
➔ approximation factor: $1 - \frac{1}{\sqrt{e}}$

even better but less fast: partial enumeration *(Sviridenko, '04)* or filtering *(Badanidiyuru & Vondrák '14)*

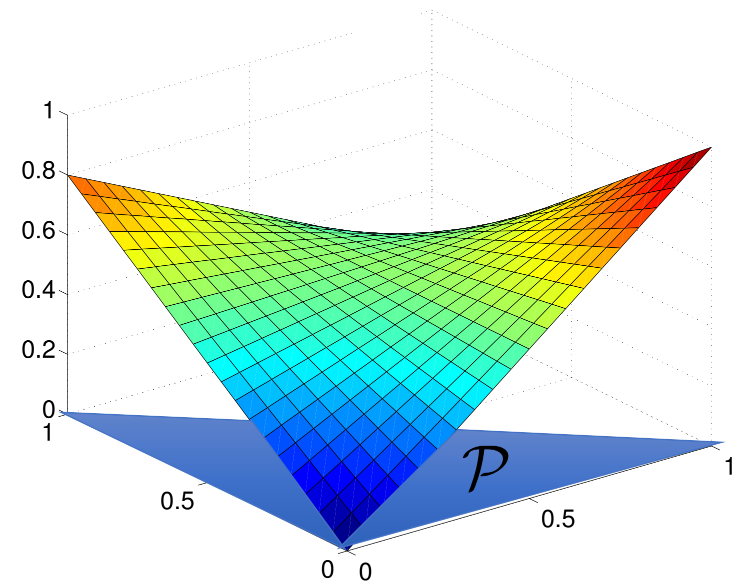
(Leskovec-Krause-Guestrin-Faloutsos-VanBriesen-Glance '07)

Relax: Discrete to continuous

$\max F(S)$



$\max \quad f_M(x)$



Algorithm: "continuous greedy"

1. approximately maximize f_M over $\mathcal{P} = \mathrm{conv}(\mathcal{I})$
2. round to discrete set

(Vondrák '08; Calinescu-Chekuri-Pal-Vondrák '11; Kulik-Shachnai-Tamir'11)

Beyond greedy? Greedy++

- Other constraints for monotone submodular functions? *Variants of greedy still work in many cases ("downward closed" constraints)*

- Non-monotone functions?

- Large-scale greedy?

Greedy can fail ...

$$F(A) = \left| \bigcup_{a \in A} \text{area}(a) \right| - \sum_{a \in A} c(a)$$

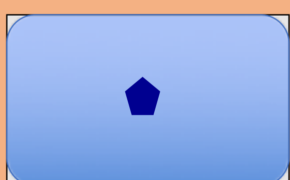
greedy solution:

$$F(A) = 40$$

optimal solution

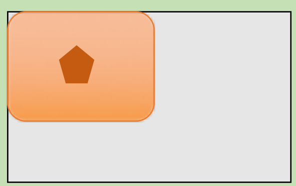
$$F(A) = 95$$

sensor 1



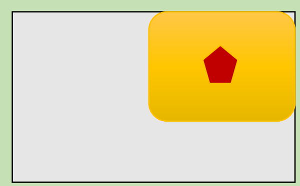
coverage: 100
cost: -60
gain 40

sensor 2



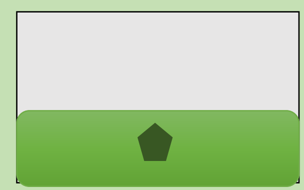
coverage: 30
cost: - 1
gain 29

sensor 3



coverage: 30
cost: - 1
gain 29

sensor 4



coverage: 40
cost: - 3
gain 37

Non-monotone maximization

- Generally inapproximable unless F is nonnegative
- Unconstrained maximization:
  - Local search *(Feige-Mirrokni-Vondrák'07)*
  - Double greedy: Optimal ½ approximation

    (Buchbinder-Feldman-Naor-Schwartz'12)
- Constrained maximization:
  - Cardinality constraints: randomized greedy *(Buchbinder-Feldman-Naor-Schwartz'14)*
  - Filtering based algorithms *(Mirzasoleiman-Badanidiyuru-Karbasi'16)*
  - More general constraints: Continuous local search via multilinear extension *(Chekuri—Vondrák-Zenklusen'11)*
- Distributed algorithms? yes!
  - divide-and-conquer *(de Ponte Barbosa-Ene-Nguyen-Ward '15)*
  - concurrency control / Hogwild *(Pan-Jegelka-Gonzalez-Bradley-Jordan '14)*

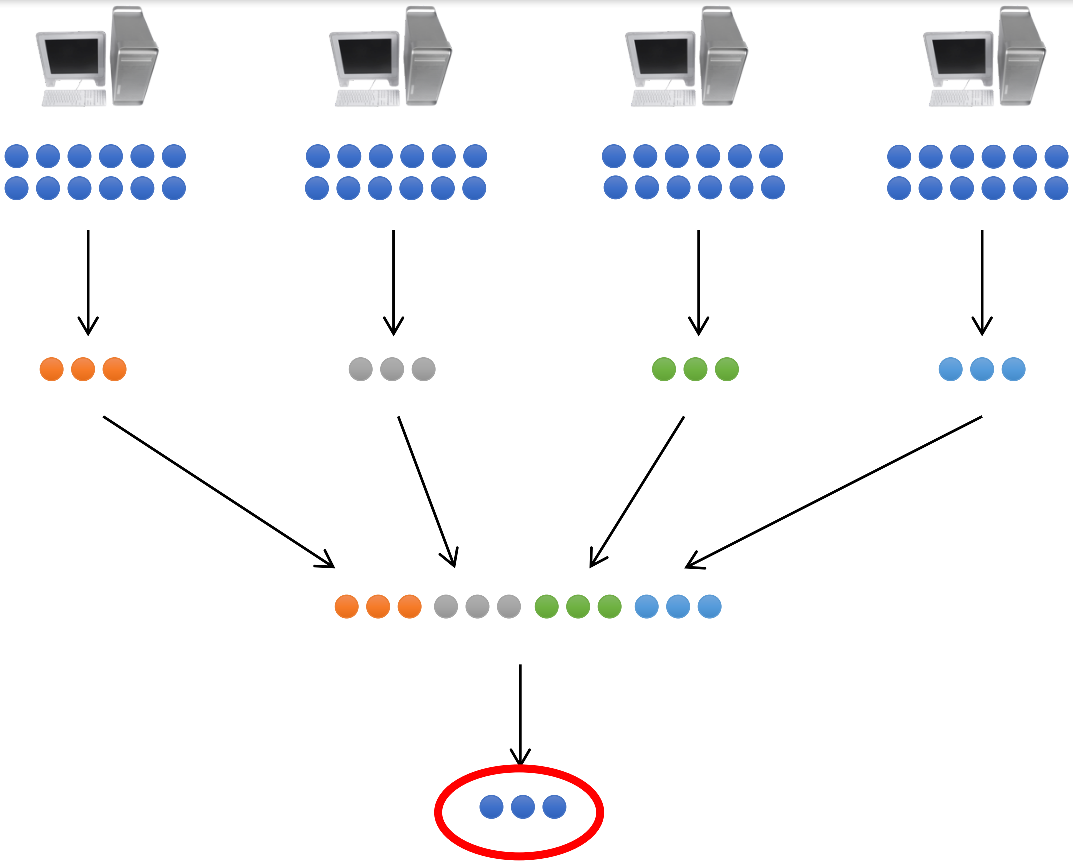
Beyond greedy? Greedy++

- Other constraints for monotone submodular functions?
  Variants of greedy still work in many cases ("downward closed" constraints)

- Non-monotone functions?
  Monotone greedy can fail, but other types of greedy ('double greedy') & local search work

- Large-scale greedy?

Distributed greedy algorithms



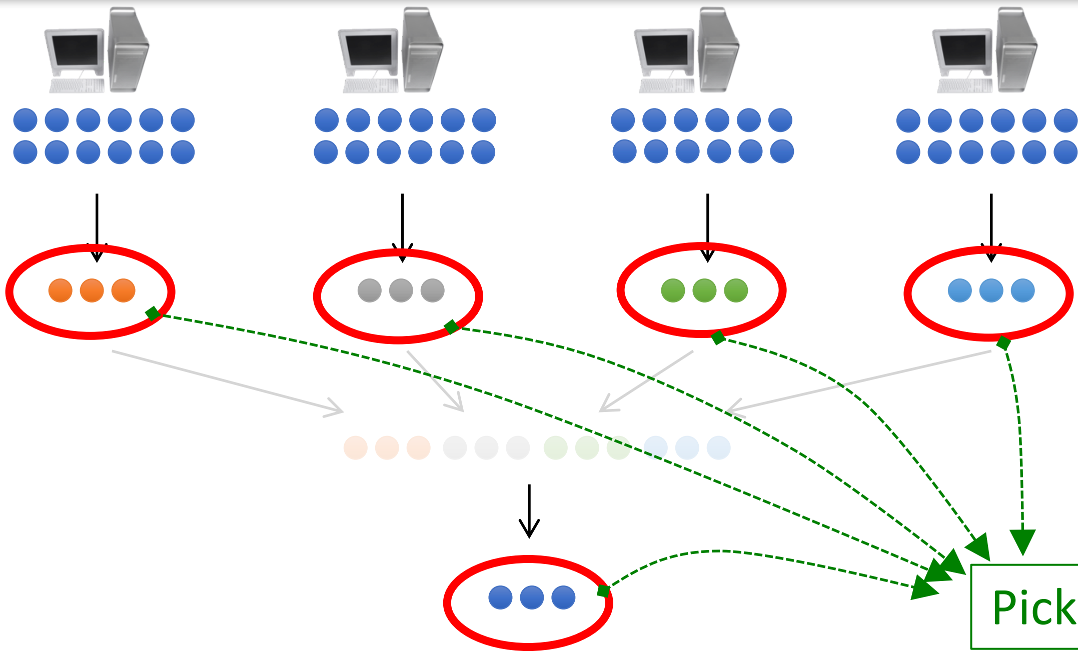
greedy is sequential.
pick in parallel??

pick *k* elements
on each machine.

combine and run
greedy again.

Is this useful?

Distributed greedy algorithms



pick in parallel
from *m* machines

Is this useful?

Pick the best of m+1 solutions

For any partition:

$$\frac{1}{\min\{\sqrt{k}, m\}}$$

Random partition:

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

Even better with geometric structure

(Mirzasoleiman-Karbasi-Sarkar-Krause’13, da Ponte Barbosa-Ene-Nguyen-Ward’15)

Beyond greedy? Greedy++

- Other constraints for monotone submodular functions?
  Variants of greedy still work in many cases ("downward closed" constraints)

- Non-monotone functions?
  Monotone greedy can fail, but other types of greedy ('double greedy') & local search work

- Large-scale greedy?
  Distributed, parallel, streaming versions for many cases

Roadmap

✓What is submodularity and where does it comes up?

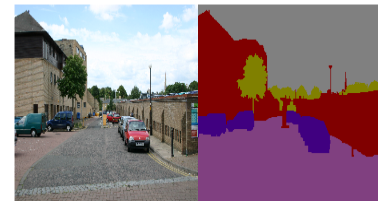
- Optimization with submodular functions
  ✓Maximization: greedy algorithms (diminishing returns)
  - Minimization?

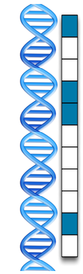
- Further connections & directions

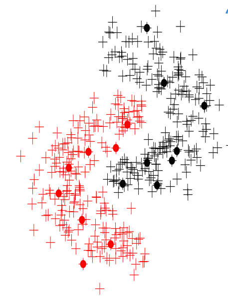
Submodular minimization

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

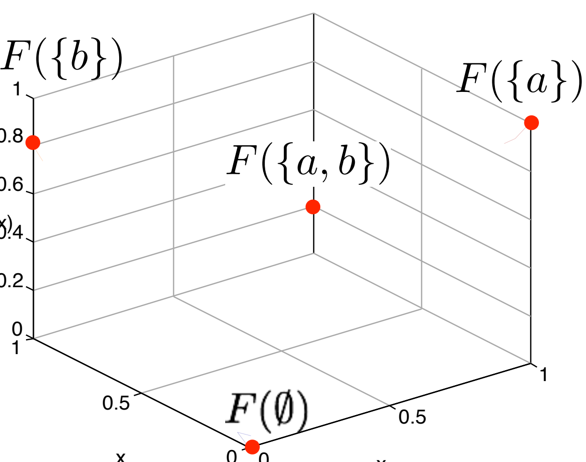
"maximize coherence"







Idea: relaxation

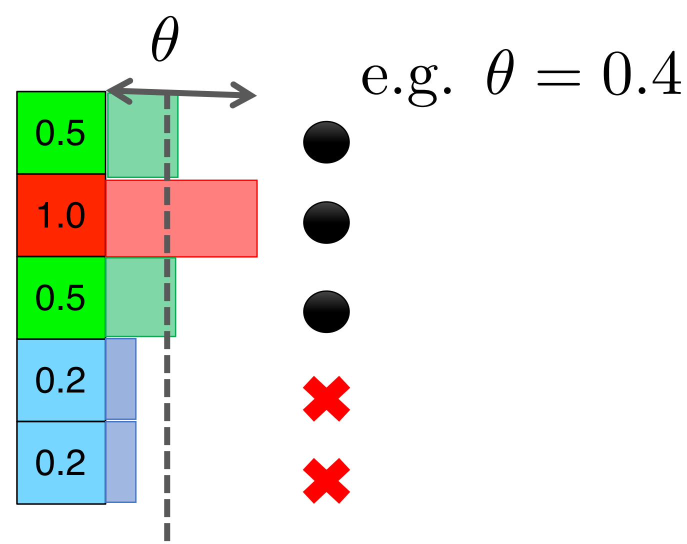


$$\min_{x \in \{0,1\}^n} \; F(x) \;\longrightarrow\; \min_{x \in [0,1]^n} \; f(x)$$

Lovasz extension

- expectation: $f(x) = \mathbb{E}_\theta[F(S_\theta)]$

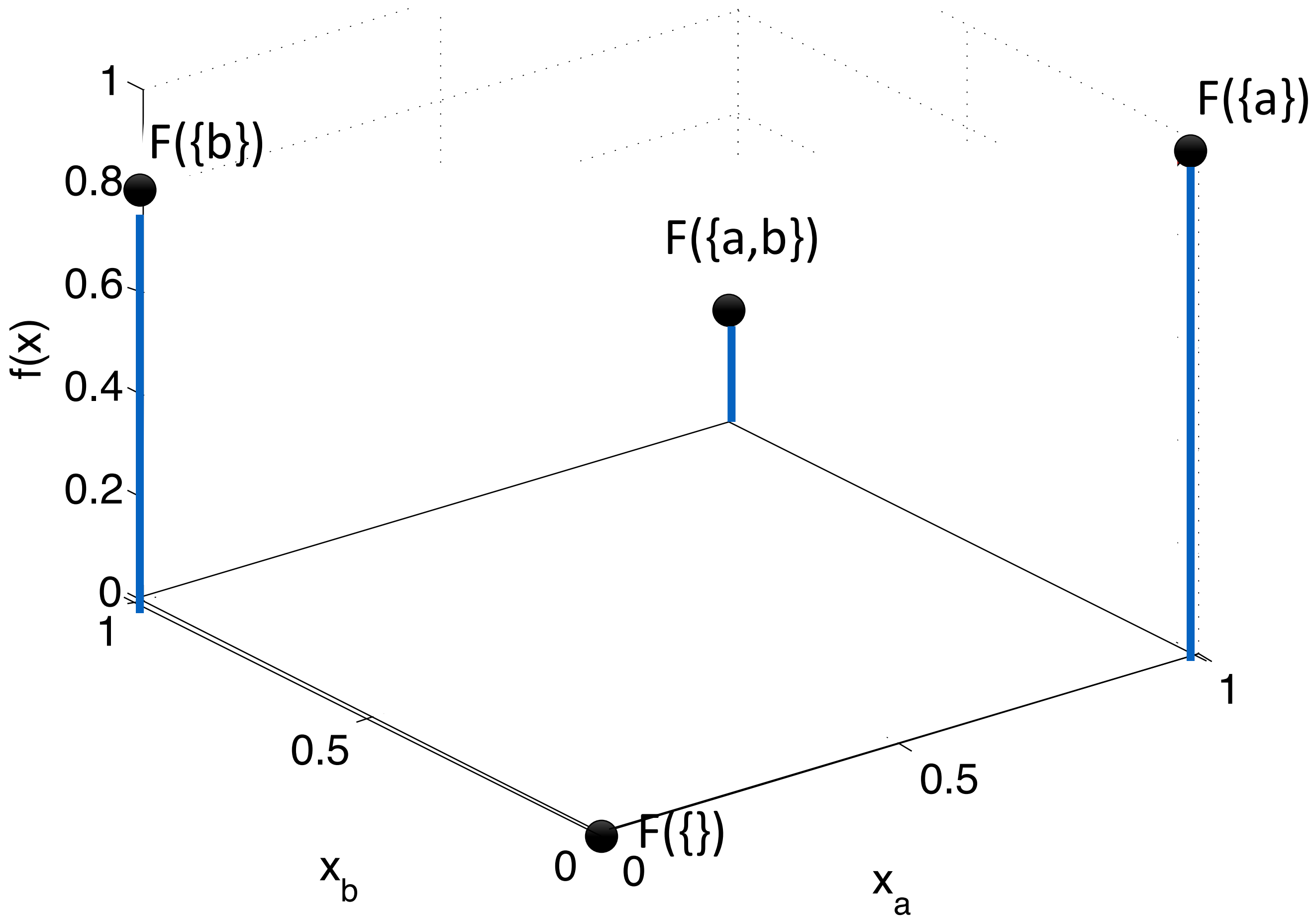
- sample threshold $\theta \in [0, 1]$ uniformly
- $S_\theta = \{e \mid x_e \geq \theta\}$



Each coordinate corresponds to an item

Lovász extension: example

$$f(x) = \mathbb{E}_\theta[F(S_\theta)]$$



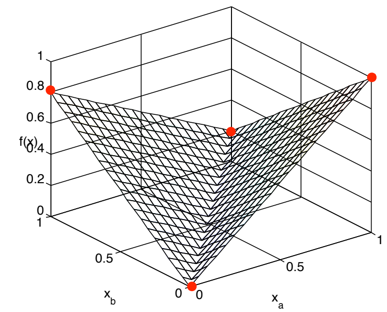
A	F(A)
{ }	0
{a}	1
{b}	.8
{a,b}	.2

Submodularity and convexity

$$f(x) = \mathbb{E}_{\theta \sim x}[F(S_\theta)]$$

if *F* is submodular, this is equivalent to:

$$f(x) = \max_{y \in \mathcal{B}_F} \; y^\top x$$

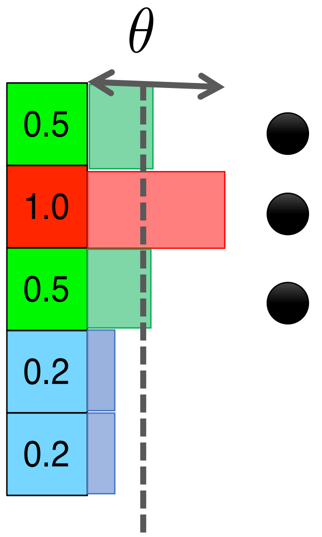


Theorem *(Edmonds 1971, Lovász 1983)*
Lovász extension is convex $\Leftrightarrow$ *F* is submodular.

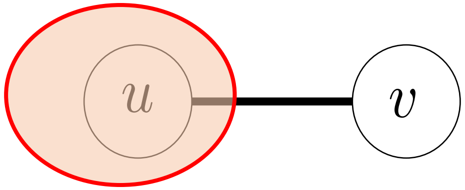
Examples of Lovasz extensions

1. $F(S) = \min\{|S|, 1\}$

   $f(x) = \max_i x_i$



2. Cut function: 2 items (nodes)



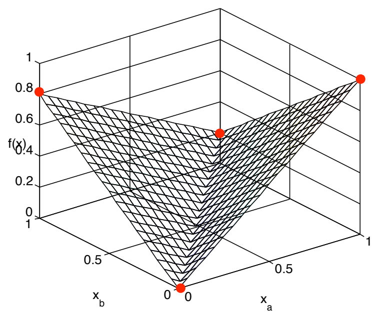
$$F(S) = \begin{cases} 1 & \text{if } |S| = 1 \\ 0 & \text{otherwise.} \end{cases}$$

$$f(x) = |x_u - x_v|$$

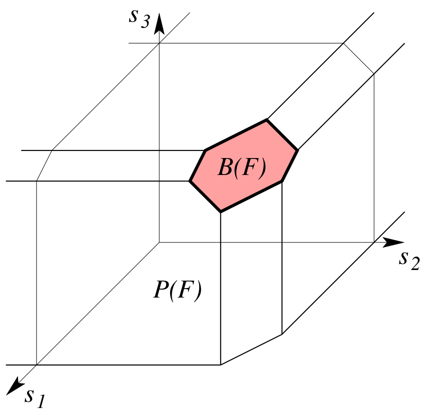
Base polytopes

$$f(x) = \mathbb{E}_{\theta \sim x}[F(S_\theta)]$$

if *F* is submodular, this is equivalent to:



$$f(x) = \max_{y \in \mathcal{B}_F} y^\top x$$



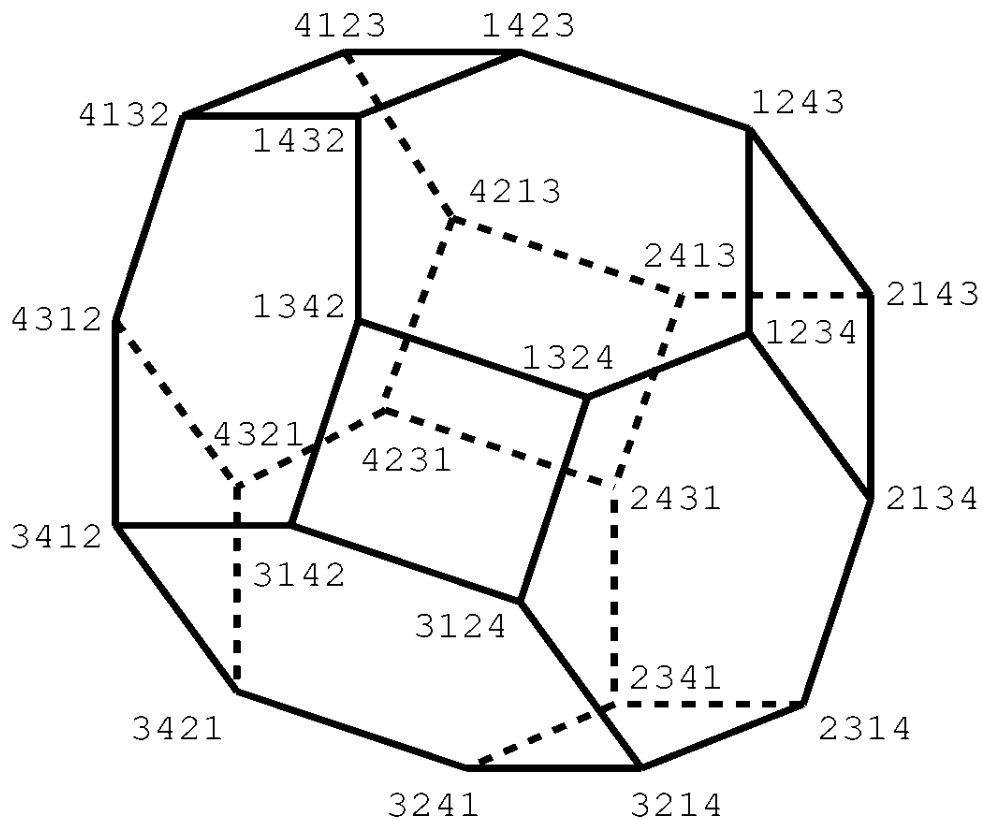
Base polytope: all vectors dominated by *F(S)*

$$\mathcal{B}_F = \{y \in \mathbb{R}^n \mid \forall S \subseteq \mathcal{V} \ \sum_{i \in S} y_i \leq F(S) \text{ and } \sum_{i=1}^{n} y_i = F(\mathcal{V})\}$$

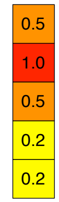
Examples of base polytopes

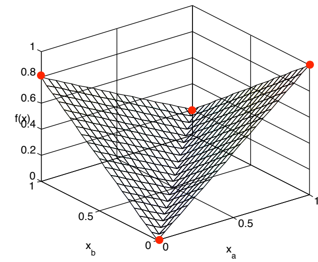
1. Probability simplex
   $$F(S) = \min\{|S|, 1\}$$

2. Permutahedron
   $$F(S) = \sum_{i=1}^{|S|}(n - i + 1)$$



Putting things together





$$\min_{S\subseteq\mathcal{V}} F(S) = \min_{x\in\{0,1\}^n} F(x) \quad \underset{=}{\longrightarrow} \quad \min_{x\in[0,1]^n} f(x)$$

1. relaxation: convex optimization ⟵ many ways to do Step 1
   computable subgradients

2. relaxation is exact!
   pick elements with positive coordinates $S^* = \{e \mid x_e^* > 0\}$

➔ submodular minimization in polynomial time!
(Grötschel, Lovász, Schrijver 1981)

Submodular minimization

convex optimization

- ellipsoid method
  (Grötschel-Lovasz-Schrijver 81)
- subgradient method …
  (…, Chakrabarty-Lee-Sidford-Wong 16)
- minimum-norm point / Fujishige-Wolfe algorithm (different relaxation)
  (Fujishige-Isotani 11)
- …

combinatorial methods

- first polynomial-time:
  (Schrijver 00, Iwata-Fleischer-Fujishige-01)
  …
- $O(n^4 T + n^5 \log M)$ *(Iwata 03)*
  $O(n^6 + n^5 T)$ *(Orlin 09)*

Latest:

$$O(n^2 T \log nM + n^3 \log^c nM)$$

$$O(n^3 T \log^2 n + n^4 \log^c n)$$

(Lee-Sidford-Wong 15)

Submodularity and convexity

- convex Lovasz extension
  - easy to compute: greedy algorithm (special polyhedra!)

- submodular minimization via convex optimization: exact
- duality results

- structured sparsity *(Bach 10)*
- decomposition & parallel algorithms
  (Komodakis-Paragios-Tziritas 11, Stobbe-Krause 10, Jegelka-Bach-Sra 13, Nishihara-Jegelka-Jordan 14, Ene-Nguyen 15)
- variational inference *(Djolonga-Krause 14)*
- …

Roadmap

✓What is submodularity and where does it comes up?

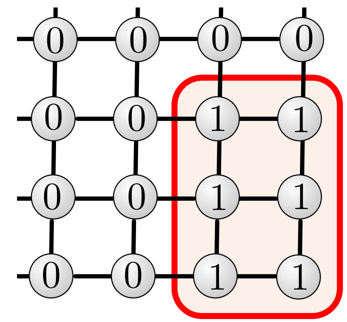
✓Optimization with submodular functions
- Maximization: greedy algorithms (discrete concavity) constraints manageable
- Minimization: convex relaxation (discrete convexity) constraints are hard

- Further connections & directions
  - Learning
  - Probability distributions & set functions
  - Integer & continuous functions

Log-supermodular distributions

$$P(S) \propto \exp(-F(S)) \qquad\qquad P(S)\,P(T) \;\le\; P(S \cup T)\,P(S \cap T)$$

Example: ferromagnetic Ising model / Conditional Random Field



"multivariate totally positive of order 2", "affiliated"

Benefits:

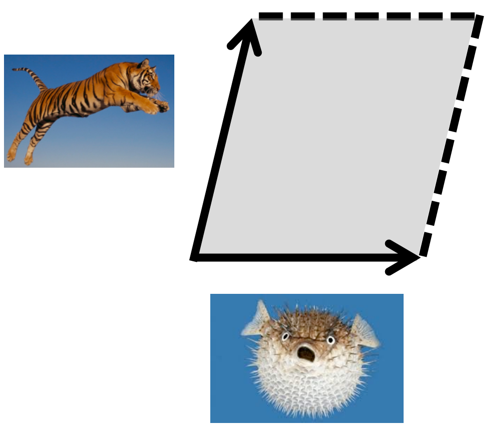
- finding the mode = minimizing a submodular function
- approximating partition function & marginals …

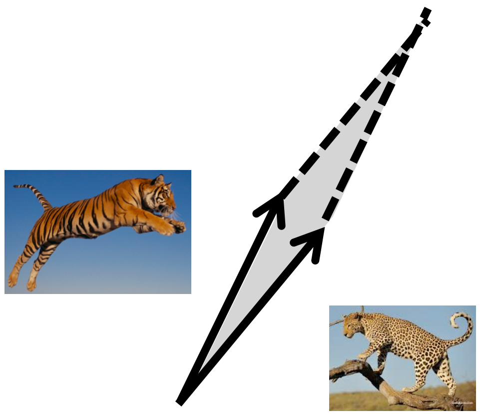
(Fortuin-Kasteleyn-Ginibre 71, Kolmogorov-Zabih 04, Djolonga-Krause 14, …)

Log-submodular distributions

$$P(S) \propto \exp(F(S)) \qquad P(S)\,P(T) \;\geq\; P(S \cup T)\,P(S \cap T)$$

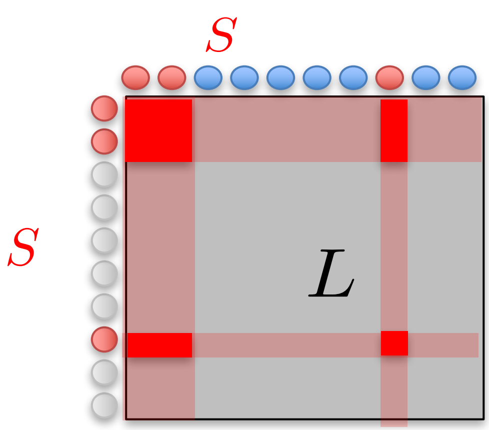
Example: Determinantal Point Processes / Volume sampling





$$P(S) \propto \mathrm{Vol}^2(\{v_i\}_{i \in S})$$

$$P(S) \propto \det(L_S)$$



Sub-family: "Strongly Rayleigh" distributions

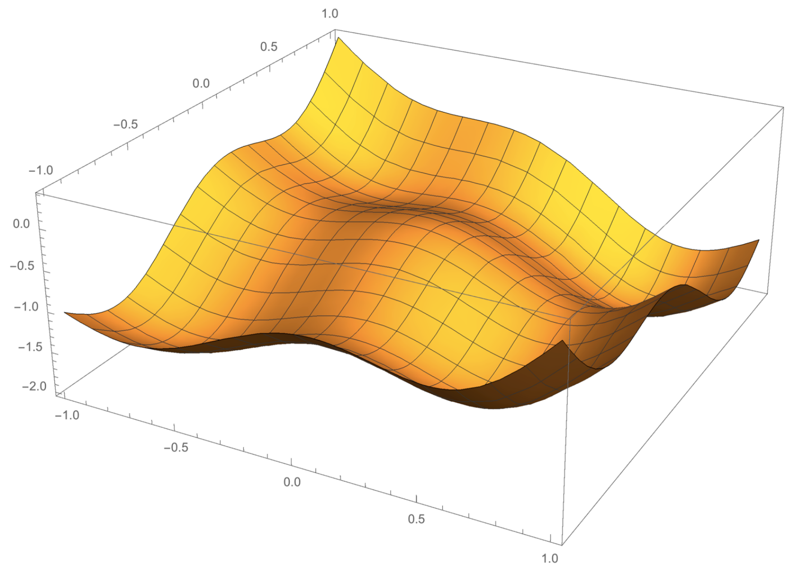
Benefits: sampling
(if negative association)

(Macchi 75, Feder-Mihail 82, Borodin 02, Deshpande-Rademacher-Vempala-Wang 06, Borcea-Bränden 09, Borcea-Bränden-Liggett 09, Kulesza-Taskar 12, Anari-Oveis Gharan-Rezaei 16, Li-Jegelka-Sra 16, …)

Submodularity more generally

- Integer and continuous functions

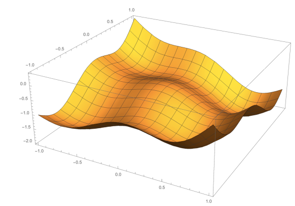
$$f(x) + f(y) \geq f(x \vee y) + f(x \wedge y)$$



- Many optimization results generalize ☺

(Milgrom-Shannon 94; Topkis 98; Murota 03; Kapralov-Post-Vondrak 10; Soma et al 2014-16; Bach 2015; Ene & Nguyen 2016; Bian-Mirzasoleiman-Buhmann-Krause 16)

Submodularity more generally

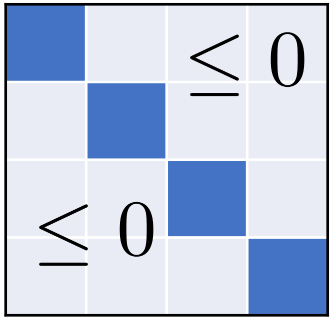


- Integer and continuous functions

$$f(x) + f(y) \;\geq\; f(x \vee y) + f(x \wedge y)$$

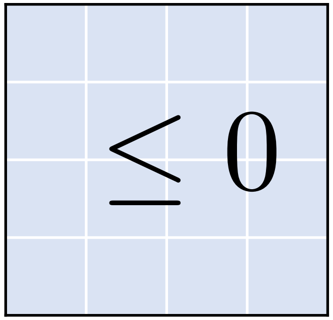
- Equivalent condition for differentiable functions:

$$\frac{\partial^2}{\partial x_i \partial x_j} f(x) \leq 0 \quad \forall i \neq j$$

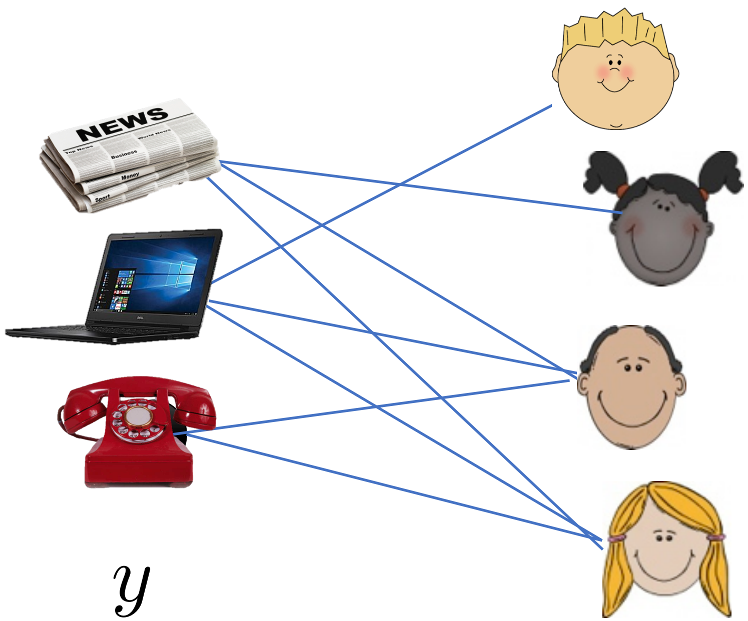


- *subclass:* diminishing returns

$$\frac{\partial^2}{\partial x_i \partial x_j} f(x) \leq 0 \quad \forall i, j$$



Application: robust optimization



$$\max_{y} \ \mathcal{I}(y;\theta) \quad \text{s.t.} \ \sum_s y_s \le B$$

infer θ from data.

robust optimization?

$$\max_{y} \ \min_{\theta \in R} \ \mathcal{I}(y;\theta)$$

nonconvex in θ ☹

But: submodular in θ ! ☺

(Staib & Jegelka, ICML 2017)

nonconvex optimization
lattice / continuous submodularity
many optimization results generalize

probability measures
log-supermodular (→positive assoc.)
log-submodular (←negative assoc.)
sampling, mode,
approx. partition function

submodular set functions

convexity:
minimization
maximize coherence

dim. returns (concavity):
maximization
maximize diversity

many examples:

- linear/modular functions
- entropy
- mutual information
- rank functions
- coverage
- diffusion in networks
- volume
- graph cut …