Submodular Functions, Optimization, and Applications to Machine Learning

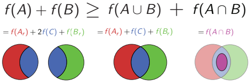
— Spring Quarter, Lecture 14 —

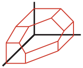
http://www.ee.washington.edu/people/faculty/bilmes/classes/ee563_spring_2018/

Prof. Jeff Bilmes

University of Washington, Seattle
Department of Electrical Engineering
http://melodi.ee.washington.edu/~bilmes

May 14th, 2018





Cumulative Outstanding Reading

- Read chapter 1 from Fujishige's book.
- Read chapter 2 from Fujishige's book.
- Read chapter 3 from Fujishige's book.
- Read chapter 4 from Fujishige's book.

Announcements, Assignments, and Reminders

- Next homework is posted on canvas. Due Thursday 5/10, 11:59pm.
- As always, if you have any questions about anything, please ask then via our discussion board (https://canvas.uw.edu/courses/1216339/discussion_topics). Can meet at odd hours via zoom (send message on canvas to schedule time to chat).

Class Road Map - EE563

- L1(3/26): Motivation, Applications, & Basic Definitions,
- L2(3/28): Machine Learning Apps (diversity, complexity, parameter, learning target, surrogate).
- L3(4/2): Info theory exs, more apps, definitions, graph/combinatorial examples
- L4(4/4): Graph and Combinatorial Examples, Matrix Rank, Examples and Properties, visualizations
- L5(4/9): More Examples/Properties/ Other Submodular Defs., Independence,
- L6(4/11): Matroids, Matroid Examples, Matroid Rank, Partition/Laminar Matroids
- L7(4/16): Laminar Matroids, System of Distinct Reps, Transversals, Transversal Matroid, Matroid Representation, Dual Matroids
- L8(4/18): Dual Matroids, Other Matroid Properties, Combinatorial Geometries, Matroids and Greedy.
- L9(4/23): Polyhedra, Matroid Polytopes, Matroids $\to$ Polymatroids
- L10(4/29): Matroids $\to$ Polymatroids, Polymatroids, Polymatroids and Greedy,
- L11(4/30): Polymatroids, Polymatroids and Greedy
- L12(5/2): Polymatroids and Greedy, Extreme Points, Cardinality Constrained Maximization
- L13(5/7): Constrained Submodular Maximization
- L14(5/9): Submodular Max w. Other Constraints, Cont. Extensions, Lovasz Extension
- L15(5/14):
- L16(5/16):
- L17(5/21):
- L18(5/23):
- L–(5/28): Memorial Day (holiday)
- L19(5/30):
- L21(6/4): Final Presentations maximization.

Last day of instruction, June 1st. Finals Week: June 2-8, 2018.

Priority Queue

- Use a priority queue Q as a data structure: operations include:
  - Insert an item (v, α) into queue, with $v \in V$ and $\alpha \in \mathbb{R}$.

$$\text{insert}(Q, (v, \alpha)) \tag{14.14}$$

  - Pop the item (v, α) with maximum value α off the queue.

$$(v, \alpha) \leftarrow \text{pop}(Q) \tag{14.15}$$

  - Query the value of the max item in the queue

$$\max(Q) \in \mathbb{R} \tag{14.16}$$

- On next slide, we call a popped item "fresh" if the value (v, α) popped has the correct value $\alpha = f(v|S_i)$. Use extra "bit" to store this info
- If a popped item is fresh, it must be the maximum — this can happen if, at given iteration, v was first popped and neither fresh nor maximum so placed back in the queue, and it then percolates back to the top at which point it is fresh — thereby avoid extra queue check.

Minoux's Accelerated Greedy Algorithm Submodular Max

Algorithm 1: Minoux's Accelerated Greedy Algorithm

1 Set $S_0 \leftarrow \emptyset$; $i \leftarrow 0$; Initialize priority queue Q ;
2 **for** $v \in E$ **do**
3   INSERT$(Q, f(v))$
4 **repeat**
5     $(v, \alpha) \leftarrow$ pop(Q) ;
6   **if** α *not "fresh"* **then**
7   recompute $\alpha \leftarrow f(v|S_i)$
8   **if** *(popped* α *in line 5 was "fresh") OR (*$\alpha \geq$ max(Q)*)* **then**
9   Set $S_{i+1} \leftarrow S_i \cup \{v\}$;
10         $i \leftarrow i + 1$;
11   **else**
12   insert$(Q, (v, \alpha))$
13 **until** $i = |E|$;

(Minimum) Submodular Set Cover

- Given polymatroid f, goal is to find a covering set of minimum cost:

$$S^* \in \underset{S \subseteq V}{\operatorname{argmin}} |S| \text{ such that } f(S) \geq \alpha \tag{14.14}$$

  where α is a "cover" requirement.
- Normally take $\alpha = f(V)$ but defining $f'(A) = \min \{f(A), \alpha\}$ we can take any α. Hence, we have equivalent formulation:

$$S^* \in \underset{S \subseteq V}{\operatorname{argmin}} |S| \text{ such that } f'(S) \geq f'(V) \tag{14.15}$$

- Note that this immediately generalizes standard set cover, in which case $f(A)$ is the cardinality of the union of sets indexed by A.
- Greedy Algorithm: Pick the first chain item S_i chosen by aforementioned greedy algorithm such that $f(S_i) \geq \alpha$ and output that as solution.

(Minimum) Submodular Set Cover: Approximation Analysis

- For integer valued f, this greedy algorithm an $O(\log(\max_{s \in V} f(\{s\})))$ approximation. Let S^* be optimal, and S^{G} be greedy solution, then

$$|S^{\mathsf{G}}| \leq |S^*| H(\max_{s \in V} f(\{s\})) = |S^*| O(\log_e(\max_{s \in V} f(\{s\}))) \tag{14.14}$$

  where H is the harmonic function, i.e., $H(d) = \sum_{i=1}^{d}(1/i)$.
- If f is not integral value, then bounds we get are of the form:

$$|S^{\mathsf{G}}| \leq |S^*| \left(1 + \log_e \frac{f(V)}{f(V) - f(S_{T-1})}\right) \tag{14.15}$$

  wehre S_T is the final greedy solution that occurs at step T.
- Set cover is hard to approximate with a factor better than $(1-\epsilon)\log \alpha$, where α is the desired cover constraint.

Curvature of a Submodular function

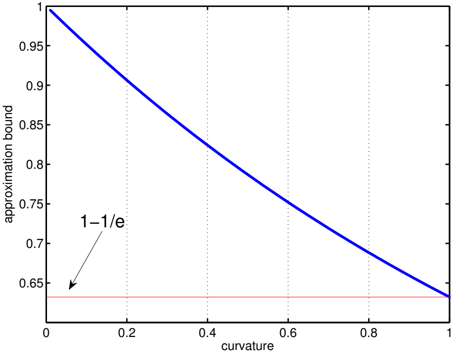
- By submodularity, total curvature can be computed in either form:

$$c \triangleq 1 - \min_{S, j \notin S : f(j|\emptyset) \neq 0} \frac{f(j|S)}{f(j|\emptyset)} = 1 - \min_{j : f(j|\emptyset) \neq 0} \frac{f(j|V \setminus \{j\})}{f(j|\emptyset)} \quad (14.17)$$

- Note: Matroid rank is either modular $c = 0$ or maximally curved $c = 1$ — hence, matroid rank can have only the extreme points of curvature, namely 0 or 1.
- Polymatroid functions are, in this sense, more nuanced, in that they allow non-extreme curvature, with $c \in [0, 1]$.
- It will be remembered the notion of "partial dependence" within polymatroid functions.

Curvature and approximation

- Curvature limitation can help the greedy algorithm in terms of approximation bounds.
- Conforti & Cornuéjols showed that greedy gives a $1/(1+c)$ approximation to $\max\{f(S) : S \in \mathcal{I}\}$ when f has total curvature c.
- Hence, greedy subject to matroid constraint is a $\max(1/(1+c), 1/2)$ approximation algorithm, and if $c < 1$ then it is better than 1/2 (e.g., with $c = 1/4$ then we have a 0.8 algorithm).
- For k-uniform matroid (i.e., k-cardinality constraints), then approximation factor becomes $\frac{1}{c}(1 - e^{-c})$



Generalizations

- Consider a k-uniform matroid $\mathcal{M} = (V, \mathcal{I})$ where $\mathcal{I} = \{S \subseteq V : |S| \leq k\}$, and consider problem $\max\{f(A) : A \in \mathcal{I}\}$

Generalizations

- Consider a k-uniform matroid $\mathcal{M} = (V, \mathcal{I})$ where $\mathcal{I} = \{S \subseteq V : |S| \leq k\}$, and consider problem $\max \{f(A) : A \in \mathcal{I}\}$
- Hence, the greedy algorithm is $1 - 1/e$ optimal for maximizing polymatroidal f subject to a k-uniform matroid constraint.

Generalizations

- Consider a k-uniform matroid $\mathcal{M} = (V, \mathcal{I})$ where $\mathcal{I} = \{S \subseteq V : |S| \leq k\}$, and consider problem $\max\{f(A) : A \in \mathcal{I}\}$
- Hence, the greedy algorithm is $1 - 1/e$ optimal for maximizing polymatroidal f subject to a k-uniform matroid constraint.
- Might be useful to allow an arbitrary matroid (e.g., partition matroid $\mathcal{I} = \{X \subseteq V : |X \cap V_i| \leq k_i$ for all $i = 1, \ldots, \ell\}$., or a transversal, etc).

Generalizations

- Consider a k-uniform matroid $\mathcal{M} = (V, \mathcal{I})$ where $\mathcal{I} = \{S \subseteq V : |S| \leq k\}$, and consider problem $\max\{f(A) : A \in \mathcal{I}\}$
- Hence, the greedy algorithm is $1 - 1/e$ optimal for maximizing polymatroidal f subject to a k-uniform matroid constraint.
- Might be useful to allow an arbitrary matroid (e.g., partition matroid $\mathcal{I} = \{X \subseteq V : |X \cap V_i| \leq k_i$ for all $i = 1, \ldots, \ell\}$., or a transversal, etc).
- Knapsack constraint: if each item $v \in V$ has a cost $c(v)$, we may ask for $c(S) \leq b$ where b is a budget, in units of costs.

Generalizations

- Consider a k-uniform matroid $\mathcal{M} = (V, \mathcal{I})$ where $\mathcal{I} = \{S \subseteq V : |S| \leq k\}$, and consider problem $\max\{f(A) : A \in \mathcal{I}\}$
- Hence, the greedy algorithm is $1 - 1/e$ optimal for maximizing polymatroidal f subject to a k-uniform matroid constraint.
- Might be useful to allow an arbitrary matroid (e.g., partition matroid $\mathcal{I} = \{X \subseteq V : |X \cap V_i| \leq k_i \text{ for all } i = 1, \dots, \ell\}$., or a transversal, etc).
- Knapsack constraint: if each item $v \in V$ has a cost $c(v)$, we may ask for $c(S) \leq b$ where b is a budget, in units of costs. Q: Is $\mathcal{I} = \{I : c(I) \leq b\}$ the independent sets of a matroid?

Generalizations

- Consider a k-uniform matroid $\mathcal{M} = (V, \mathcal{I})$ where $\mathcal{I} = \{S \subseteq V : |S| \leq k\}$, and consider problem $\max\{f(A) : A \in \mathcal{I}\}$
- Hence, the greedy algorithm is $1 - 1/e$ optimal for maximizing polymatroidal f subject to a k-uniform matroid constraint.
- Might be useful to allow an arbitrary matroid (e.g., partition matroid $\mathcal{I} = \{X \subseteq V : |X \cap V_i| \leq k_i \text{ for all } i = 1, \ldots, \ell\}$., or a transversal, etc).
- Knapsack constraint: if each item $v \in V$ has a cost $c(v)$, we may ask for $c(S) \leq b$ where b is a budget, in units of costs. Q: Is $\mathcal{I} = \{I : c(I) \leq b\}$ the independent sets of a matroid?
- We may wish to maximize f subject to multiple matroid constraints. I.e., $S \in \mathcal{I}_1, S \in \mathcal{I}_2, \ldots, S \in \mathcal{I}_p$ where $\mathcal{I}_i$ are independent sets of the i^{th} matroid.

Generalizations

- Consider a k-uniform matroid $\mathcal{M} = (V, \mathcal{I})$ where $\mathcal{I} = \{S \subseteq V : |S| \leq k\}$, and consider problem $\max \{f(A) : A \in \mathcal{I}\}$
- Hence, the greedy algorithm is $1 - 1/e$ optimal for maximizing polymatroidal f subject to a k-uniform matroid constraint.
- Might be useful to allow an arbitrary matroid (e.g., partition matroid $\mathcal{I} = \{X \subseteq V : |X \cap V_i| \leq k_i \text{ for all } i = 1, \ldots, \ell\}$., or a transversal, etc).
- Knapsack constraint: if each item $v \in V$ has a cost $c(v)$, we may ask for $c(S) \leq b$ where b is a budget, in units of costs. Q: Is $\mathcal{I} = \{I : c(I) \leq b\}$ the independent sets of a matroid?
- We may wish to maximize f subject to multiple matroid constraints. I.e., $S \in \mathcal{I}_1, S \in \mathcal{I}_2, \ldots, S \in \mathcal{I}_p$ where $\mathcal{I}_i$ are independent sets of the i^{th} matroid.
- Combinations of the above (e.g., knapsack & multiple matroid constraints).

Greedy over multiple matroids

- Obvious heuristic is to use the greedy step but always stay feasible.

Greedy over multiple matroids

- Obvious heuristic is to use the greedy step but always stay feasible.
- I.e., Starting with $S_0 = \emptyset$, we repeat the following greedy step

$$S_{i+1} = S_i \cup \left\{ \underset{v \in V \setminus S_i \,:\, S_i + v \in \bigcap_{i=1}^{p} \mathcal{I}_i}{\operatorname{argmax}} f(S_i \cup \{v\}) \right\} \tag{14.1}$$

Greedy over multiple matroids

- Obvious heuristic is to use the greedy step but always stay feasible.
- I.e., Starting with $S_0 = \emptyset$, we repeat the following greedy step

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i \,:\, S_i + v \in \bigcap_{i=1}^{p} \mathcal{I}_i} f(S_i \cup \{v\}) \right\} \tag{14.1}$$

- That is, we keep choosing next whatever feasible element looks best.

Greedy over multiple matroids

- Obvious heuristic is to use the greedy step but always stay feasible.
- I.e., Starting with $S_0 = \emptyset$, we repeat the following greedy step

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i \,:\, S_i + v \in \bigcap_{i=1}^{p} \mathcal{I}_i} f(S_i \cup \{v\}) \right\} \tag{14.1}$$

- That is, we keep choosing next whatever feasible element looks best.
- This algorithm is simple and also has a guarantee

Greedy over multiple matroids

- Obvious heuristic is to use the greedy step but always stay feasible.
- I.e., Starting with $S_0 = \emptyset$, we repeat the following greedy step

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i \,:\, S_i + v \in \bigcap_{i=1}^p \mathcal{I}_i} f(S_i \cup \{v\}) \right\} \tag{14.1}$$

- That is, we keep choosing next whatever feasible element looks best.
- This algorithm is simple and also has a guarantee

Theorem 14.3.1

Given a polymatroid function f, and set of matroids $\{M_j = (E, \mathcal{I}_j)\}_{j=1}^p$, the above greedy algorithm returns sets S_i such that for each i we have $f(S_i) \geq \frac{1}{p+1} \max_{|S| \leq i, S \in \bigcap_{i=1}^p \mathcal{I}_i} f(S)$, assuming such sets exists.

Greedy over multiple matroids

- Obvious heuristic is to use the greedy step but always stay feasible.
- I.e., Starting with $S_0 = \emptyset$, we repeat the following greedy step

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i \,:\, S_i + v \in \bigcap_{i=1}^p \mathcal{I}_i} f(S_i \cup \{v\}) \right\} \tag{14.1}$$

- That is, we keep choosing next whatever feasible element looks best.
- This algorithm is simple and also has a guarantee

Theorem 14.3.1

Given a polymatroid function f, and set of matroids $\{M_j = (E, \mathcal{I}_j)\}_{j=1}^p$, the above greedy algorithm returns sets S_i such that for each i we have $f(S_i) \geq \frac{1}{p+1} \max_{|S| \leq i, S \in \bigcap_{i=1}^p \mathcal{I}_i} f(S)$, assuming such sets exists.

- For one matroid, we have a 1/2 approximation.

Greedy over multiple matroids

- Obvious heuristic is to use the greedy step but always stay feasible.
- I.e., Starting with $S_0 = \emptyset$, we repeat the following greedy step

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i \,:\, S_i + v \in \bigcap_{i=1}^p \mathcal{I}_i} f(S_i \cup \{v\}) \right\} \tag{14.1}$$

- That is, we keep choosing next whatever feasible element looks best.
- This algorithm is simple and also has a guarantee

Theorem 14.3.1

Given a polymatroid function f, and set of matroids $\{M_j = (E, \mathcal{I}_j)\}_{j=1}^p$, the above greedy algorithm returns sets S_i such that for each i we have $f(S_i) \geq \frac{1}{p+1} \max_{|S| \leq i, S \in \bigcap_{i=1}^p \mathcal{I}_i} f(S)$, assuming such sets exists.

- For one matroid, we have a 1/2 approximation.
- Very easy algorithm, Minoux trick still possible, while addresses multiple matroid constraints

Greedy over multiple matroids

- Obvious heuristic is to use the greedy step but always stay feasible.
- I.e., Starting with $S_0 = \emptyset$, we repeat the following greedy step

$$S_{i+1} = S_i \cup \left\{ \underset{v \in V \setminus S_i \,:\, S_i + v \in \bigcap_{i=1}^p \mathcal{I}_i}{\operatorname{argmax}} f(S_i \cup \{v\}) \right\} \tag{14.1}$$

- That is, we keep choosing next whatever feasible element looks best.
- This algorithm is simple and also has a guarantee

Theorem 14.3.1

Given a polymatroid function f, and set of matroids $\{M_j = (E, \mathcal{I}_j)\}_{j=1}^p$, the above greedy algorithm returns sets S_i such that for each i we have $f(S_i) \geq \frac{1}{p+1} \max_{|S| \leq i, S \in \bigcap_{i=1}^p \mathcal{I}_i} f(S)$, assuming such sets exists.

- For one matroid, we have a 1/2 approximation.
- Very easy algorithm, Minoux trick still possible, while addresses multiple matroid constraints — but the bound is not that good when there are many matroids.

Matroid Intersection and Bipartite Matching

- Why might we want to do matroid intersection?

Matroid Intersection and Bipartite Matching

- Why might we want to do matroid intersection?
- Consider bipartite graph $G = (V, F, E)$. Define two partition matroids $M_V = (E, \mathcal{I}_V)$, and $M_F = (E, \mathcal{I}_F)$.

Matroid Intersection and Bipartite Matching

- Why might we want to do matroid intersection?
- Consider bipartite graph $G = (V, F, E)$. Define two partition matroids $M_V = (E, \mathcal{I}_V)$, and $M_F = (E, \mathcal{I}_F)$.
- Independence in each matroid corresponds to:

Matroid Intersection and Bipartite Matching

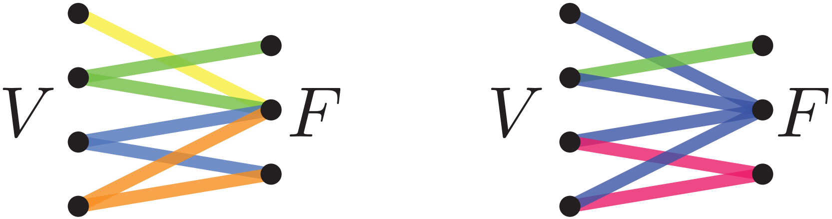
- Why might we want to do matroid intersection?
- Consider bipartite graph $G = (V, F, E)$. Define two partition matroids $M_V = (E, \mathcal{I}_V)$, and $M_F = (E, \mathcal{I}_F)$.
- Independence in each matroid corresponds to:
  1. $I \in \mathcal{I}_V$ if $|I \cap (V, f)| \leq 1$ for all $f \in F$,

Matroid Intersection and Bipartite Matching

- Why might we want to do matroid intersection?
- Consider bipartite graph $G = (V, F, E)$. Define two partition matroids $M_V = (E, \mathcal{I}_V)$, and $M_F = (E, \mathcal{I}_F)$.
- Independence in each matroid corresponds to:
  1. $I \in \mathcal{I}_V$ if $|I \cap (V, f)| \leq 1$ for all $f \in F$,
  2. and $I \in \mathcal{I}_F$ if $|I \cap (v, F)| \leq 1$ for all $v \in V$.

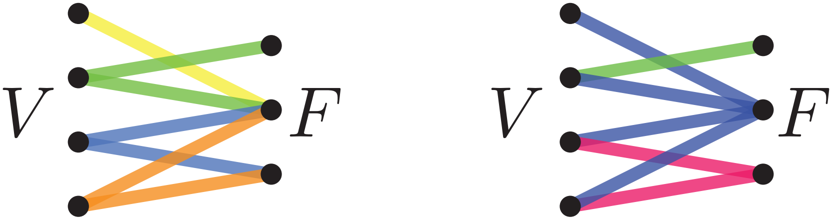
Matroid Intersection and Bipartite Matching

- Why might we want to do matroid intersection?
- Consider bipartite graph $G = (V, F, E)$. Define two partition matroids $M_V = (E, \mathcal{I}_V)$, and $M_F = (E, \mathcal{I}_F)$.
- Independence in each matroid corresponds to:
  1. $I \in \mathcal{I}_V$ if $|I \cap (V, f)| \leq 1$ for all $f \in F$,
  2. and $I \in \mathcal{I}_F$ if $|I \cap (v, F)| \leq 1$ for all $v \in V$.



Matroid Intersection and Bipartite Matching

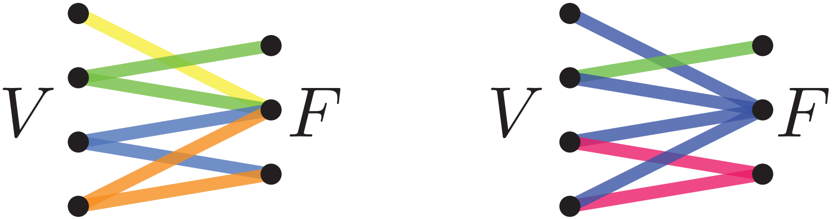
- Why might we want to do matroid intersection?
- Consider bipartite graph $G = (V, F, E)$. Define two partition matroids $M_V = (E, \mathcal{I}_V)$, and $M_F = (E, \mathcal{I}_F)$.
- Independence in each matroid corresponds to:
  1. $I \in \mathcal{I}_V$ if $|I \cap (V, f)| \leq 1$ for all $f \in F$,
  2. and $I \in \mathcal{I}_F$ if $|I \cap (v, F)| \leq 1$ for all $v \in V$.



- Therefore, a matching in G is simultaneously independent in both M_V and M_F and finding the maximum matching is finding the maximum cardinality set independent in both matroids.

Matroid Intersection and Bipartite Matching

- Why might we want to do matroid intersection?
- Consider bipartite graph $G = (V, F, E)$. Define two partition matroids $M_V = (E, \mathcal{I}_V)$, and $M_F = (E, \mathcal{I}_F)$.
- Independence in each matroid corresponds to:
  1. $I \in \mathcal{I}_V$ if $|I \cap (V, f)| \leq 1$ for all $f \in F$,
  2. and $I \in \mathcal{I}_F$ if $|I \cap (v, F)| \leq 1$ for all $v \in V$.



- Therefore, a matching in G is simultaneously independent in both M_V and M_F and finding the maximum matching is finding the maximum cardinality set independent in both matroids.
- In bipartite graph case, therefore, can be solved in polynomial time.

Matroid Intersection and Network Communication

- Let $G_1 = (V_1, E)$ and $G_2 = (V_2, E)$ be two graphs on an isomorphic set of edges (lets just give them same names E).

Matroid Intersection and Network Communication

- Let $G_1 = (V_1, E)$ and $G_2 = (V_2, E)$ be two graphs on an isomorphic set of edges (lets just give them same names E).
- Consider two cycle matroids associated with these graphs $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$. They might be very different (e.g., an edge might be between two distinct nodes in G_1 but the same edge is a loop in multi-graph G_2.)

Matroid Intersection and Network Communication

- Let $G_1 = (V_1, E)$ and $G_2 = (V_2, E)$ be two graphs on an isomorphic set of edges (lets just give them same names E).
- Consider two cycle matroids associated with these graphs $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$. They might be very different (e.g., an edge might be between two distinct nodes in G_1 but the same edge is a loop in multi-graph G_2.)
- We may wish to find the maximum size edge-induced subgraph that is still forest in both graphs (i.e., adding any edges will create a circuit in either M_1, M_2, or both).

Matroid Intersection and Network Communication

- Let $G_1 = (V_1, E)$ and $G_2 = (V_2, E)$ be two graphs on an isomorphic set of edges (lets just give them same names E).
- Consider two cycle matroids associated with these graphs $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$. They might be very different (e.g., an edge might be between two distinct nodes in G_1 but the same edge is a loop in multi-graph G_2.)
- We may wish to find the maximum size edge-induced subgraph that is still forest in both graphs (i.e., adding any edges will create a circuit in either M_1, M_2, or both).
- This is again a matroid intersection problem.

Matroid Intersection and TSP

- Definition: a Hamiltonian cycle is a cycle that passes through each node exactly once.

Matroid Intersection and TSP

- Definition: a Hamiltonian cycle is a cycle that passes through each node exactly once.
- Given directed graph G, goal is to find such a Hamiltonian cycle.

Matroid Intersection and TSP

- Definition: a Hamiltonian cycle is a cycle that passes through each node exactly once.
- Given directed graph G, goal is to find such a Hamiltonian cycle.
- From G with n nodes, create G' with $n+1$ nodes by duplicating (w.l.o.g.) a particular node $v_1 \in V(G)$ to v_1^+, v_1^-, and have all outgoing edges from v_1 come instead from v_1^- and all edges incoming to v_1 go instead to v_1^+.

Matroid Intersection and TSP

- Definition: a Hamiltonian cycle is a cycle that passes through each node exactly once.
- Given directed graph G, goal is to find such a Hamiltonian cycle.
- From G with n nodes, create G' with $n+1$ nodes by duplicating (w.l.o.g.) a particular node $v_1 \in V(G)$ to v_1^+, v_1^-, and have all outgoing edges from v_1 come instead from v_1^- and all edges incoming to v_1 go instead to v_1^+.
- Let M_1 be the cycle matroid on G'.

Matroid Intersection and TSP

- Definition: a Hamiltonian cycle is a cycle that passes through each node exactly once.
- Given directed graph G, goal is to find such a Hamiltonian cycle.
- From G with n nodes, create G' with $n+1$ nodes by duplicating (w.l.o.g.) a particular node $v_1 \in V(G)$ to v_1^+, v_1^-, and have all outgoing edges from v_1 come instead from v_1^- and all edges incoming to v_1 go instead to v_1^+.
- Let M_1 be the cycle matroid on G'.
- Let M_2 be the partition matroid having as independent sets those that have no more than one edge leaving any node — i.e., $I \in \mathcal{I}(M_2)$ if $|I \cap \delta^-(v)| \leq 1$ for all $v \in V(G')$.

Matroid Intersection and TSP

- Definition: a Hamiltonian cycle is a cycle that passes through each node exactly once.
- Given directed graph G, goal is to find such a Hamiltonian cycle.
- From G with n nodes, create G' with $n+1$ nodes by duplicating (w.l.o.g.) a particular node $v_1 \in V(G)$ to v_1^+, v_1^-, and have all outgoing edges from v_1 come instead from v_1^- and all edges incoming to v_1 go instead to v_1^+.
- Let M_1 be the cycle matroid on G'.
- Let M_2 be the partition matroid having as independent sets those that have no more than one edge leaving any node — i.e., $I \in \mathcal{I}(M_2)$ if $|I \cap \delta^-(v)| \leq 1$ for all $v \in V(G')$.
- Let M_3 be the partition matroid having as independent sets those that have no more than one edge entering any node — i.e., $I \in \mathcal{I}(M_3)$ if $|I \cap \delta^+(v)| \leq 1$ for all $v \in V(G')$.

Matroid Intersection and TSP

- Definition: a Hamiltonian cycle is a cycle that passes through each node exactly once.
- Given directed graph G, goal is to find such a Hamiltonian cycle.
- From G with n nodes, create G' with $n+1$ nodes by duplicating (w.l.o.g.) a particular node $v_1 \in V(G)$ to v_1^+, v_1^-, and have all outgoing edges from v_1 come instead from v_1^- and all edges incoming to v_1 go instead to v_1^+.
- Let M_1 be the cycle matroid on G'.
- Let M_2 be the partition matroid having as independent sets those that have no more than one edge leaving any node — i.e., $I \in \mathcal{I}(M_2)$ if $|I \cap \delta^-(v)| \leq 1$ for all $v \in V(G')$.
- Let M_3 be the partition matroid having as independent sets those that have no more than one edge entering any node — i.e., $I \in \mathcal{I}(M_3)$ if $|I \cap \delta^+(v)| \leq 1$ for all $v \in V(G')$.
- Then a Hamiltonian cycle exists iff there is an n-element intersection of M_1, M_2, and M_3.

Matroid Intersection and TSP

- Recall, the traveling salesperson problem (TSP) is the problem to, given a directed graph, start at a node, visit all cities, and return to the starting point. Optimization version does this tour at minimum cost.

Matroid Intersection and TSP

- Recall, the traveling salesperson problem (TSP) is the problem to, given a directed graph, start at a node, visit all cities, and return to the starting point. Optimization version does this tour at minimum cost.
- Since TSP is NP-complete, we obviously can't solve matroid intersections of 3 more matroids, unless P=NP.

Matroid Intersection and TSP

- Recall, the traveling salesperson problem (TSP) is the problem to, given a directed graph, start at a node, visit all cities, and return to the starting point. Optimization version does this tour at minimum cost.
- Since TSP is NP-complete, we obviously can't solve matroid intersections of 3 more matroids, unless P=NP.
- But bipartite graph example gives us hope for 2 matroids, as in that case we can easily solve $\max |X|$ s.t. $x \in \mathcal{I}_1 \cap \mathcal{I}_2$.

Greedy over multiple matroids: Generalized Bipartite Matching

- Generalized bipartite matching (i.e., max bipartite matching with submodular costs on the edges). Use two partition matroids (as mentioned earlier in class)

Greedy over multiple matroids: Generalized Bipartite Matching

- Generalized bipartite matching (i.e., max bipartite matching with submodular costs on the edges). Use two partition matroids (as mentioned earlier in class)
- Useful in natural language processing: Ex. Computer language translation, find an alignment between two language strings.

Greedy over multiple matroids: Generalized Bipartite Matching

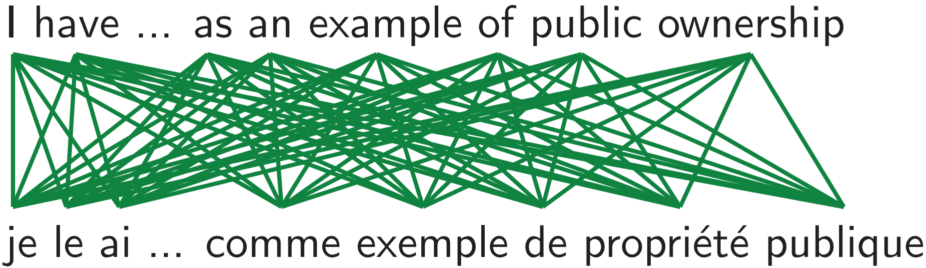
- Generalized bipartite matching (i.e., max bipartite matching with submodular costs on the edges). Use two partition matroids (as mentioned earlier in class)
- Useful in natural language processing: Ex. Computer language translation, find an alignment between two language strings.
- Consider bipartite graph $G = (E, F, V)$ where E and F are the left/right set of nodes, respectively, and V is the set of edges.

Greedy over multiple matroids: Generalized Bipartite Matching

- Generalized bipartite matching (i.e., max bipartite matching with submodular costs on the edges). Use two partition matroids (as mentioned earlier in class)
- Useful in natural language processing: Ex. Computer language translation, find an alignment between two language strings.
- Consider bipartite graph $G = (E, F, V)$ where E and F are the left/right set of nodes, respectively, and V is the set of edges.
- E corresponds to, say, an English language sentence and F corresponds to a French language sentence — goal is to form a matching (an alignment) between the two.

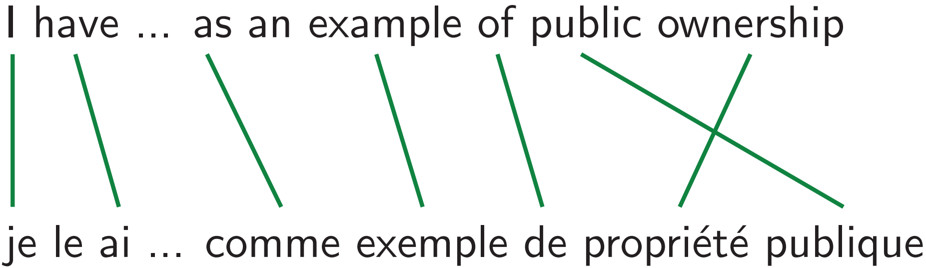
Greedy over > 1 matroids: Multiple Language Alignment

- Consider English string and French string, set up as a bipartite graph.



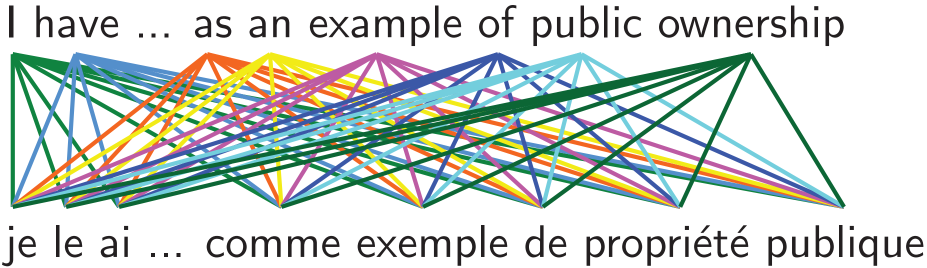
Greedy over > 1 matroids: Multiple Language Alignment

- One possible alignment, a matching, with score as sum of edge weights.



Greedy over > 1 matroids: Multiple Language Alignment

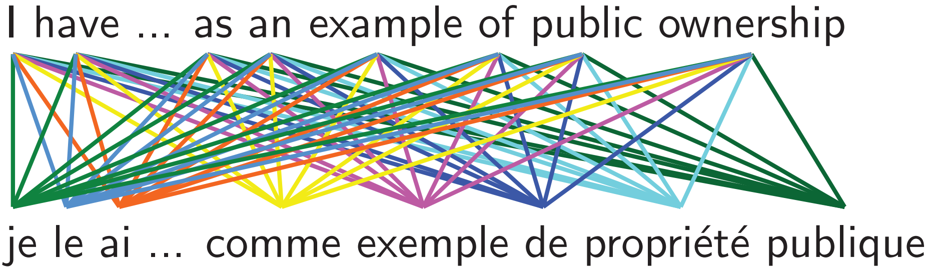
- Edges incident to English words constitute an edge partition



- The two edge partitions can be used to set up two 1-partition matroids on the edges.
- For each matroid, a set of edges is independent only if the set intersects each partition block no more than one time.

Greedy over > 1 matroids: Multiple Language Alignment

- Edges incident to French words constitute an edge partition



- The two edge partitions can be used to set up two 1-partition matroids on the edges.
- For each matroid, a set of edges is independent only if the set intersects each partition block no more than one time.

Greedy over > 1 matroids: Multiple Language Alignment

- Typical to use bipartite matching to find an alignment between the two language strings.

Greedy over > 1 matroids: Multiple Language Alignment

- Typical to use bipartite matching to find an alignment between the two language strings.
- As we saw, this is equivalent to two 1-partition matroids and a non-negative modular cost function on the edges.

Greedy over > 1 matroids: Multiple Language Alignment

- Typical to use bipartite matching to find an alignment between the two language strings.
- As we saw, this is equivalent to two 1-partition matroids and a non-negative modular cost function on the edges.
- We can generalize this using a polymatroid cost function on the edges, and two k-partition matroids, allowing for "fertility" in the models:

Greedy over > 1 matroids: Multiple Language Alignment

- Typical to use bipartite matching to find an alignment between the two language strings.
- As we saw, this is equivalent to two 1-partition matroids and a non-negative modular cost function on the edges.
- We can generalize this using a polymatroid cost function on the edges, and two k-partition matroids, allowing for "fertility" in the models:

Fertility at most 1

Greedy over > 1 matroids: Multiple Language Alignment

- Typical to use bipartite matching to find an alignment between the two language strings.
- As we saw, this is equivalent to two 1-partition matroids and a non-negative modular cost function on the edges.
- We can generalize this using a polymatroid cost function on the edges, and two k-partition matroids, allowing for "fertility" in the models:

Fertility at most 2

Greedy over > 1 matroids: Multiple Language Alignment

- Generalizing further, each block of edges in each partition matroid can have its own "fertility" limit:
  $\mathcal{I} = \{X \subseteq V : |X \cap V_i| \leq k_i$ for all $i = 1, \ldots, \ell\}$.

Greedy over > 1 matroids: Multiple Language Alignment

- Generalizing further, each block of edges in each partition matroid can have its own "fertility" limit: $\mathcal{I} = \{X \subseteq V : |X \cap V_i| \leq k_i \text{ for all } i = 1, \ldots, \ell\}$.
- Maximizing submodular function subject to multiple matroid constraints addresses this problem.

Greedy over multiple matroids: Submodular Welfare

- Submodular Welfare Maximization: Consider E a set of m goods to be distributed/partitioned among n people ("players").

Greedy over multiple matroids: Submodular Welfare

- Submodular Welfare Maximization: Consider E a set of m goods to be distributed/partitioned among n people ("players").
- Each players has a submodular "valuation" function, $g_i : 2^E \to \mathbb{R}_+$ that measures how "desirable" or "valuable" a given subset $A \subseteq E$ of goods are to that player.

Greedy over multiple matroids: Submodular Welfare

- Submodular Welfare Maximization: Consider E a set of m goods to be distributed/partitioned among n people ("players").
- Each players has a submodular "valuation" function, $g_i : 2^E \to \mathbb{R}_+$ that measures how "desirable" or "valuable" a given subset $A \subseteq E$ of goods are to that player.
- Assumption: No good can be shared between multiple players, each good must be allocated to a single player.

Greedy over multiple matroids: Submodular Welfare

- Submodular Welfare Maximization: Consider E a set of m goods to be distributed/partitioned among n people ("players").
- Each players has a submodular "valuation" function, $g_i : 2^E \to \mathbb{R}_+$ that measures how "desirable" or "valuable" a given subset $A \subseteq E$ of goods are to that player.
- Assumption: No good can be shared between multiple players, each good must be allocated to a single player.
- Goal of submodular welfare: Partition the goods $E = E_1 \cup E_2 \cup \dots \cup E_n$ into n blocks in order to maximize the submodular social welfare, measured as:

$$\text{submodular-social-welfare}(E_1, E_2, \dots, E_n) = \sum_{i=1}^{n} g_i(E_i). \tag{14.2}$$

Greedy over multiple matroids: Submodular Welfare

- Submodular Welfare Maximization: Consider E a set of m goods to be distributed/partitioned among n people ("players").
- Each players has a submodular "valuation" function, $g_i : 2^E \to \mathbb{R}_+$ that measures how "desirable" or "valuable" a given subset $A \subseteq E$ of goods are to that player.
- Assumption: No good can be shared between multiple players, each good must be allocated to a single player.
- Goal of submodular welfare: Partition the goods $E = E_1 \cup E_2 \cup \dots \cup E_n$ into n blocks in order to maximize the submodular social welfare, measured as:

$$\text{submodular-social-welfare}(E_1, E_2, \dots, E_n) = \sum_{i=1}^{n} g_i(E_i). \tag{14.2}$$

- We can solve this via submodular maximization subject to multiple matroid independence constraints as we next describe ...

Submodular Welfare: Submodular Max over matroid partition

- Create new ground set E' as disjoint union of n copies of the ground set. I.e.,

$$E' = \underbrace{E \uplus E \uplus \cdots \uplus E}_{n\times} \tag{14.3}$$

Submodular Welfare: Submodular Max over matroid partition

- Create new ground set E' as disjoint union of n copies of the ground set. I.e.,

$$E' = \underbrace{E \uplus E \uplus \cdots \uplus E}_{n\times} \tag{14.3}$$

- Let $E^{(i)} \subset E'$ be the i^{th} block of E'.

Submodular Welfare: Submodular Max over matroid partition

- Create new ground set E' as disjoint union of n copies of the ground set. I.e.,

$$E' = \underbrace{E \uplus E \uplus \cdots \uplus E}_{n\times} \tag{14.3}$$

- Let $E^{(i)} \subset E'$ be the i^{th} block of E'.
- For any $e \in E$, the corresponding element in $E^{(i)}$ is called $(e, i) \in E^{(i)}$ (each original element is tagged by integer).

Submodular Welfare: Submodular Max over matroid partition

- Create new ground set E' as disjoint union of n copies of the ground set. I.e.,

$$E' = \underbrace{E \uplus E \uplus \cdots \uplus E}_{n \times} \tag{14.3}$$

- Let $E^{(i)} \subset E'$ be the i^{th} block of E'.
- For any $e \in E$, the corresponding element in $E^{(i)}$ is called $(e, i) \in E^{(i)}$ (each original element is tagged by integer).
- For $e \in E$, define $E_e = \{(e', i) \in E' : e' = e\}$.

Submodular Welfare: Submodular Max over matroid partition

- Create new ground set E' as disjoint union of n copies of the ground set. I.e.,

$$E' = \underbrace{E \uplus E \uplus \cdots \uplus E}_{n\times} \tag{14.3}$$

- Let $E^{(i)} \subset E'$ be the i^{th} block of E'.
- For any $e \in E$, the corresponding element in $E^{(i)}$ is called $(e, i) \in E^{(i)}$ (each original element is tagged by integer).
- For $e \in E$, define $E_e = \{(e', i) \in E' : e' = e\}$.
- Hence, $\{E_e\}_{e \in E}$ is a partition of E', each block of the partition for one of the original elements in E.

Submodular Welfare: Submodular Max over matroid partition

- Create new ground set E' as disjoint union of n copies of the ground set. I.e.,

$$E' = \underbrace{E \uplus E \uplus \cdots \uplus E}_{n\times} \tag{14.3}$$

- Let $E^{(i)} \subset E'$ be the i^{th} block of E'.
- For any $e \in E$, the corresponding element in $E^{(i)}$ is called $(e, i) \in E^{(i)}$ (each original element is tagged by integer).
- For $e \in E$, define $E_e = \{(e', i) \in E' : e' = e\}$.
- Hence, $\{E_e\}_{e \in E}$ is a partition of E', each block of the partition for one of the original elements in E.
- Create a 1-partition matroid $\mathcal{M} = (E', \mathcal{I})$ where

$$\mathcal{I} = \left\{S \subseteq E' : \forall e \in E, |S \cap E_e| \leq 1\right\} \tag{14.4}$$

Submodular Welfare: Submodular Max over matroid partition

- Hence, S is independent in matroid $\mathcal{M} = (E', I)$ if S uses each original element no more than once.

Submodular Welfare: Submodular Max over matroid partition

- Hence, S is independent in matroid $\mathcal{M} = (E', I)$ if S uses each original element no more than once.
- Create submodular function $f' : 2^{E'} \to \mathbb{R}_+$ with $f'(S) = \sum_{i=1}^n g_i(S \cap E^{(i)})$.

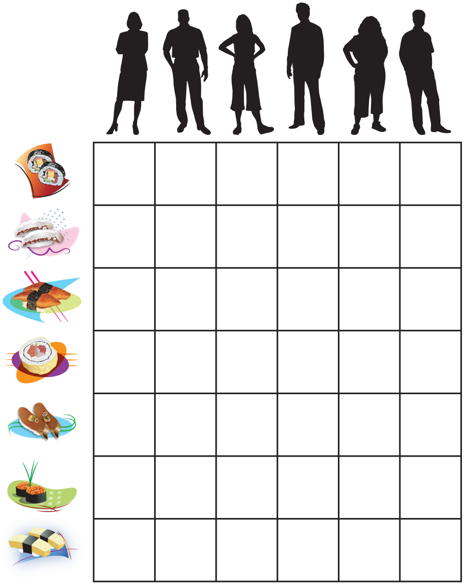
Submodular Welfare: Submodular Max over matroid partition

- Hence, S is independent in matroid $\mathcal{M} = (E', I)$ if S uses each original element no more than once.
- Create submodular function $f' : 2^{E'} \to \mathbb{R}_+$ with $f'(S) = \sum_{i=1}^{n} g_i(S \cap E^{(i)})$.
- Submodular welfare maximization becomes matroid constrained submodular max $\max \{f'(S) : S \in \mathcal{I}\}$, so greedy algorithm gives a $1/2$ approximation.

Submodular Social Welfare

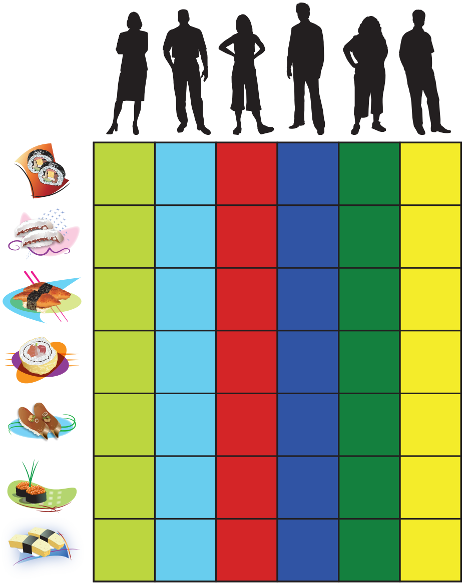
- Have $n = 6$ people (who don't like to share) and $|E| = m = 7$ pieces of sushi. E.g., $e \in E$ might be $e =$ "salmon roll".

Submodular Social Welfare



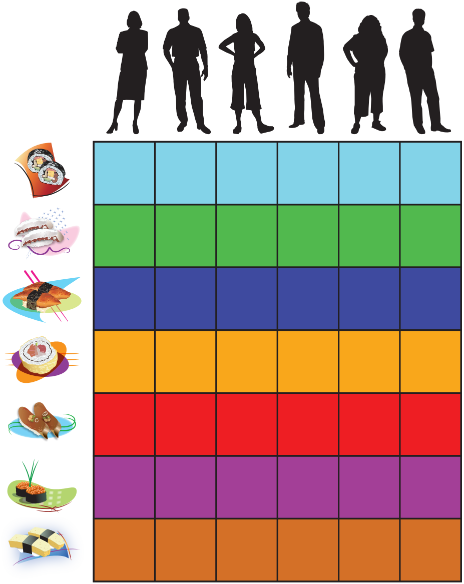
- Have $n = 6$ people (who don't like to share) and $|E| = m = 7$ pieces of sushi. E.g., $e \in E$ might be $e =$ "salmon roll".
- Goal: distribute sushi to people to maximize social welfare.

Submodular Social Welfare



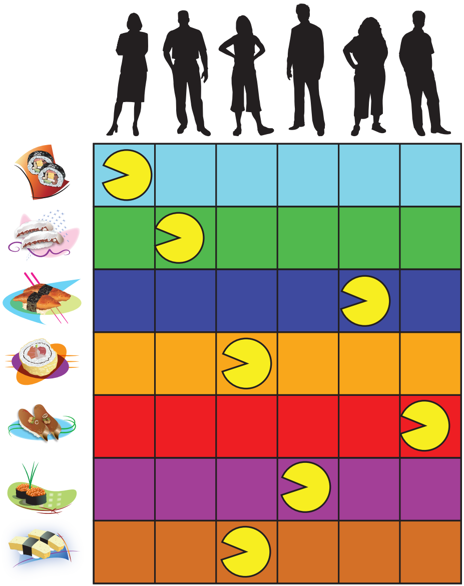
- Have $n = 6$ people (who don't like to share) and $|E| = m = 7$ pieces of sushi. E.g., $e \in E$ might be $e =$ "salmon roll".
- Goal: distribute sushi to people to maximize social welfare.
- Ground set disjoint union $E \uplus E \uplus E \uplus E \uplus E \uplus E$.

Submodular Social Welfare



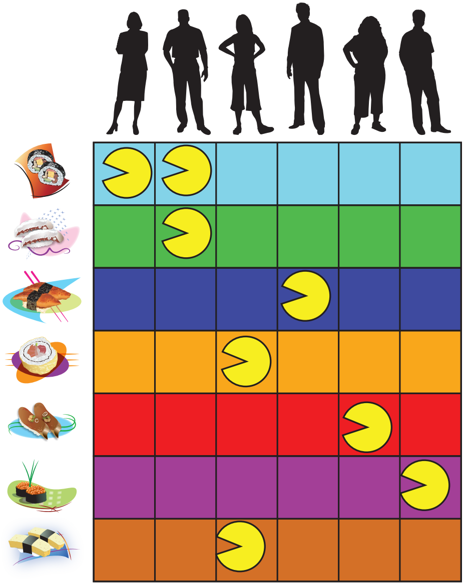
- Have $n = 6$ people (who don't like to share) and $|E| = m = 7$ pieces of sushi. E.g., $e \in E$ might be $e =$ "salmon roll".
- Goal: distribute sushi to people to maximize social welfare.
- Ground set disjoint union $E \uplus E \uplus E \uplus E \uplus E \uplus E$.
- Partition matroid partitions: $E_{e_1} \cup E_{e_2} \cup E_{e_3} \cup E_{e_4} \cup E_{e_5} \cup E_{e_6} \cup E_{e_7}$.

Submodular Social Welfare



- Have $n = 6$ people (who don't like to share) and $|E| = m = 7$ pieces of sushi. E.g., $e \in E$ might be $e =$ "salmon roll".
- Goal: distribute sushi to people to maximize social welfare.
- Ground set disjoint union $E \uplus E \uplus E \uplus E \uplus E \uplus E$.
- Partition matroid partitions: $E_{e_1} \cup E_{e_2} \cup E_{e_3} \cup E_{e_4} \cup E_{e_5} \cup E_{e_6} \cup E_{e_7}$.
- independent allocation

Submodular Social Welfare



- Have $n = 6$ people (who don't like to share) and $|E| = m = 7$ pieces of sushi. E.g., $e \in E$ might be $e =$ "salmon roll".
- Goal: distribute sushi to people to maximize social welfare.
- Ground set disjoint union $E \uplus E \uplus E \uplus E \uplus E \uplus E$.
- Partition matroid partitions: $E_{e_1} \cup E_{e_2} \cup E_{e_3} \cup E_{e_4} \cup E_{e_5} \cup E_{e_6} \cup E_{e_7}$.
- independent allocation
- non-independent allocation

Monotone Submodular over Knapsack Constraint

- The constraint $|A| \leq k$ is a simple cardinality constraint.

Monotone Submodular over Knapsack Constraint

- The constraint $|A| \leq k$ is a simple cardinality constraint.
- Consider a non-negative integral modular function $c : E \to \mathbb{Z}_+$.

Monotone Submodular over Knapsack Constraint

- The constraint $|A| \leq k$ is a simple cardinality constraint.
- Consider a non-negative integral modular function $c : E \to \mathbb{Z}_+$.
- A knapsack constraint would be of the form $c(A) \leq b$ where B is some integer budget that must not be exceeded. That is $\max\{f(A) : A \subseteq V, c(A) \leq b\}$.

Monotone Submodular over Knapsack Constraint

- The constraint $|A| \leq k$ is a simple cardinality constraint.
- Consider a non-negative integral modular function $c : E \rightarrow \mathbb{Z}_+$.
- A knapsack constraint would be of the form $c(A) \leq b$ where B is some integer budget that must not be exceeded. That is $\max\{f(A) : A \subseteq V, c(A) \leq b\}$.
- Important: A knapsack constraint yields an independence system (down closed) but it is not a matroid!

Monotone Submodular over Knapsack Constraint

- The constraint $|A| \leq k$ is a simple cardinality constraint.
- Consider a non-negative integral modular function $c : E \to \mathbb{Z}_+$.
- A knapsack constraint would be of the form $c(A) \leq b$ where B is some integer budget that must not be exceeded. That is $\max \{f(A) : A \subseteq V, c(A) \leq b\}$.
- Important: A knapsack constraint yields an independence system (down closed) but it is not a matroid!
- $c(e)$ may be seen as the cost of item e and if $c(e) = 1$ for all e, then we recover the cardinality constraint we saw earlier.

Monotone Submodular over Knapsack Constraint

- Greedy can be seen as choosing the best gain: Starting with $S_0 = \emptyset$, we repeat the following greedy step

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i} \Big(f(S_i \cup \{v\}) - f(S_i) \Big) \right\} \tag{14.5}$$

the gain is $f(\{v\}|S_i) = f(S_i + v) - f(S_i)$, so greedy just chooses next the currently unselected element with greatest gain.

Monotone Submodular over Knapsack Constraint

- Greedy can be seen as choosing the best gain: Starting with $S_0 = \emptyset$, we repeat the following greedy step

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i} \Big(f(S_i \cup \{v\}) - f(S_i) \Big) \right\} \tag{14.5}$$

  the gain is $f(\{v\}|S_i) = f(S_i + v) - f(S_i)$, so greedy just chooses next the currently unselected element with greatest gain.

- Core idea in knapsack case: Greedy can be extended to choose next whatever looks cost-normalized best, i.e., Starting some initial set S_0, we repeat the following cost-normalized greedy step

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i} \frac{f(S_i \cup \{v\}) - f(S_i)}{c(v)} \right\} \tag{14.6}$$

  which we repeat until $c(S_{i+1}) > b$ and then take S_i as the solution.

A Knapsack Constraint

- There are a number of ways of getting approximation bounds using this strategy.
- If we run the normalized greedy procedure starting with $S_0 = \emptyset$, and compare the solution found with the max of the singletons $\max_{v \in V} f(\{v\})$, choosing the max, then we get a $(1 - e^{-1/2}) \approx 0.39$ approximation, in $O(n^2)$ time (Minoux trick also possible for further speed)
- Partial enumeration: On the other hand, we can get a $(1 - e^{-1}) \approx 0.63$ approximation in $O(n^5)$ time if we run the above procedure starting from all sets of cardinality three (so restart for all S_0 such that $|S_0| = 3$), and compare that with the best singleton and pairwise solution.
- Extending something similar to this to d simultaneous knapsack constraints is possible as well.

Local Search Algorithms

From J. Vondrak

- Local search involves switching up to t elements, as long as it provides a (non-trivial) improvement; can iterate in several phases. Some examples follow:
- 1/3 approximation to unconstrained non-monotone maximization [Feige, Mirrokni, Vondrak, 2007]
- $1/(k + 2 + \frac{1}{k} + \delta_t)$ approximation for non-monotone maximization subject to k matroids [Lee, Mirrokni, Nagarajan, Sviridenko, 2009]
- $1/(k + \delta_t)$ approximation for monotone submodular maximization subject to $k \geq 2$ matroids [Lee, Sviridenko, Vondrak, 2010].

What About Non-monotone

- Alternatively, we may wish to maximize non-monotone submodular functions. This includes of course graph cuts, and this problem is APX-hard, so maximizing non-monotone functions, even unconstrainedly, is hard.

What About Non-monotone

- Alternatively, we may wish to maximize non-monotone submodular functions. This includes of course graph cuts, and this problem is APX-hard, so maximizing non-monotone functions, even unconstrainedly, is hard.
- If f is an arbitrary submodular function (so neither polymatroidal, nor necessarily positive or negative), then verifying if the maximum of f is positive or negative is already NP-hard.

What About Non-monotone

- Alternatively, we may wish to maximize non-monotone submodular functions. This includes of course graph cuts, and this problem is APX-hard, so maximizing non-monotone functions, even unconstrainedly, is hard.
- If f is an arbitrary submodular function (so neither polymatroidal, nor necessarily positive or negative), then verifying if the maximum of f is positive or negative is already NP-hard.
- Therefore, submodular function max in such case is inapproximable unless P=NP (since any such procedure would give us the sign of the max).

What About Non-monotone

- Alternatively, we may wish to maximize non-monotone submodular functions. This includes of course graph cuts, and this problem is APX-hard, so maximizing non-monotone functions, even unconstrainedly, is hard.
- If f is an arbitrary submodular function (so neither polymatroidal, nor necessarily positive or negative), then verifying if the maximum of f is positive or negative is already NP-hard.
- Therefore, submodular function max in such case is inapproximable unless P=NP (since any such procedure would give us the sign of the max).
- Thus, any approximation algorithm must be for unipolar submodular functions. E.g., non-negative but otherwise arbitrary submodular functions.

What About Non-monotone

- Alternatively, we may wish to maximize non-monotone submodular functions. This includes of course graph cuts, and this problem is APX-hard, so maximizing non-monotone functions, even unconstrainedly, is hard.
- If f is an arbitrary submodular function (so neither polymatroidal, nor necessarily positive or negative), then verifying if the maximum of f is positive or negative is already NP-hard.
- Therefore, submodular function max in such case is inapproximable unless P=NP (since any such procedure would give us the sign of the max).
- Thus, any approximation algorithm must be for unipolar submodular functions. E.g., non-negative but otherwise arbitrary submodular functions.
- We may get a $(\frac{1}{3} - \frac{\epsilon}{n})$ approximation for maximizing non-monotone non-negative submodular functions, with most $O(\frac{1}{\epsilon} n^3 \log n)$ function calls using approximate local maxima.

Submodularity and local optima

- Given any submodular function f, a set $S \subseteq V$ is a local maximum of f if $f(S - v) \leq f(S)$ for all $v \in S$ and $f(S + v) \leq f(S)$ for all $v \in V \setminus S$ (i.e., local in a Hamming ball of radius 1).

Submodularity and local optima

- Given any submodular function f, a set $S \subseteq V$ is a local maximum of f if $f(S - v) \leq f(S)$ for all $v \in S$ and $f(S + v) \leq f(S)$ for all $v \in V \setminus S$ (i.e., local in a Hamming ball of radius 1).
- The following interesting result is true for any submodular function:

Submodularity and local optima

- Given any submodular function f, a set $S \subseteq V$ is a local maximum of f if $f(S - v) \leq f(S)$ for all $v \in S$ and $f(S + v) \leq f(S)$ for all $v \in V \setminus S$ (i.e., local in a Hamming ball of radius 1).
- The following interesting result is true for any submodular function:

Lemma 14.3.2

Given a submodular function f, if S is a local maximum of f, and $I \subseteq S$ or $I \supseteq S$, then $f(I) \leq f(S)$.

Submodularity and local optima

- Given any submodular function f, a set $S \subseteq V$ is a local maximum of f if $f(S - v) \leq f(S)$ for all $v \in S$ and $f(S + v) \leq f(S)$ for all $v \in V \setminus S$ (i.e., local in a Hamming ball of radius 1).
- The following interesting result is true for any submodular function:

Lemma 14.3.2

Given a submodular function f, if S is a local maximum of f, and $I \subseteq S$ or $I \supseteq S$, then $f(I) \leq f(S)$.

- Idea of proof: Given $v_1, v_2 \in S$, suppose $f(S - v_1) \leq f(S)$ and $f(S - v_2) \leq f(S)$. Submodularity requires $f(S - v_1) + f(S - v_2) \geq f(S) + f(S - v_1 - v_2)$ which would be impossible unless $f(S - v_1 - v_2) \leq f(S)$.

Submodularity and local optima

- Given any submodular function f, a set $S \subseteq V$ is a local maximum of f if $f(S - v) \leq f(S)$ for all $v \in S$ and $f(S + v) \leq f(S)$ for all $v \in V \setminus S$ (i.e., local in a Hamming ball of radius 1).
- The following interesting result is true for any submodular function:

Lemma 14.3.2

Given a submodular function f, if S is a local maximum of f, and $I \subseteq S$ or $I \supseteq S$, then $f(I) \leq f(S)$.

- Idea of proof: Given $v_1, v_2 \in S$, suppose $f(S - v_1) \leq f(S)$ and $f(S - v_2) \leq f(S)$. Submodularity requires $f(S - v_1) + f(S - v_2) \geq f(S) + f(S - v_1 - v_2)$ which would be impossible unless $f(S - v_1 - v_2) \leq f(S)$.
- Similarly, given $v_1, v_2 \notin S$, and $f(S + v_1) \leq f(S)$ and $f(S + v_2) \leq f(S)$. Submodularity requires $f(S + v_1) + f(S + v_2) \geq f(S) + f(S + v_1 + v_2)$ which requires $f(S + v_1 + v_2) \leq f(S)$.

Submodularity and local optima

- Given any submodular function f, a set $S \subseteq V$ is a local maximum of f if $f(S - v) \leq f(S)$ for all $v \in S$ and $f(S + v) \leq f(S)$ for all $v \in V \setminus S$ (i.e., local in a Hamming ball of radius 1).
- The following interesting result is true for any submodular function:

Lemma 14.3.2

Given a submodular function f, if S is a local maximum of f, and $I \subseteq S$ or $I \supseteq S$, then $f(I) \leq f(S)$.

- In other words, once we have identified a local maximum, the two intervals in the Boolean lattice $[\emptyset, S]$ and $[S, V]$ can be ruled out as a possible improvement over S.

Submodularity and local optima

- Given any submodular function f, a set $S \subseteq V$ is a local maximum of f if $f(S - v) \leq f(S)$ for all $v \in S$ and $f(S + v) \leq f(S)$ for all $v \in V \setminus S$ (i.e., local in a Hamming ball of radius 1).
- The following interesting result is true for any submodular function:

Lemma 14.3.2

Given a submodular function f, if S is a local maximum of f, and $I \subseteq S$ or $I \supseteq S$, then $f(I) \leq f(S)$.

- In other words, once we have identified a local maximum, the two intervals in the Boolean lattice $[\emptyset, S]$ and $[S, V]$ can be ruled out as a possible improvement over S.
- Finding a local maximum is already hard (PLS-complete), but it is possible to find an approximate local maximum relatively efficiently.

Submodularity and local optima

- Given any submodular function f, a set $S \subseteq V$ is a local maximum of f if $f(S - v) \leq f(S)$ for all $v \in S$ and $f(S + v) \leq f(S)$ for all $v \in V \setminus S$ (i.e., local in a Hamming ball of radius 1).
- The following interesting result is true for any submodular function:

Lemma 14.3.2

Given a submodular function f, if S is a local maximum of f, and $I \subseteq S$ or $I \supseteq S$, then $f(I) \leq f(S)$.

- In other words, once we have identified a local maximum, the two intervals in the Boolean lattice $[\emptyset, S]$ and $[S, V]$ can be ruled out as a possible improvement over S.
- Finding a local maximum is already hard (PLS-complete), but it is possible to find an approximate local maximum relatively efficiently.
- This is the approach that yields the $(\frac{1}{3} - \frac{\epsilon}{n})$ approximation algorithm.

Linear time algorithm unconstrained non-monotone max

- Tight randomized tight $1/2$ approximation algorithm for unconstrained non-monotone non-negative submodular maximization.

Linear time algorithm unconstrained non-monotone max

- Tight randomized tight $1/2$ approximation algorithm for unconstrained non-monotone non-negative submodular maximization.
- Buchbinder, Feldman, Naor, Schwartz 2012.

Linear time algorithm unconstrained non-monotone max

- Tight randomized tight $1/2$ approximation algorithm for unconstrained non-monotone non-negative submodular maximization.
- Buchbinder, Feldman, Naor, Schwartz 2012. Recall $[a]_+ = \max(a, 0)$.

Linear time algorithm unconstrained non-monotone max

- Tight randomized tight $1/2$ approximation algorithm for unconstrained non-monotone non-negative submodular maximization.
- Buchbinder, Feldman, Naor, Schwartz 2012. Recall $[a]_+ = \max(a, 0)$.

Algorithm 5: Randomized Linear-time non-monotone submodular max

Set $L \leftarrow \emptyset$; $U \leftarrow V$ /* Lower L, upper U. Invariant: $L \subseteq U$ */ ;
Order elements of $V = (v_1, v_2, \ldots, v_n)$ arbitrarily ;
for $i \leftarrow 0 \ldots |V|$ **do**
  $a \leftarrow [f(v_i|L)]_+$; $b \leftarrow [-f(U|U \setminus \{v_i\})]_+$;
  if $a = b = 0$ **then** $p \leftarrow 1/2$;
  ;
  else $p \leftarrow a/(a+b)$;
  ;
  if *Flip of coin with* $\Pr($*heads*$) = p$ *draws heads* **then**
    $L \leftarrow L \cup \{v_i\}$;
  Otherwise /* if the coin drew tails, an event with prob. $1 - p$ */
    $U \leftarrow U \setminus \{v\}$
return L *(which is the same as* U *at this point)*

Linear time algorithm unconstrained non-monotone max

- Each "sweep" of the algorithm is $O(n)$.

Linear time algorithm unconstrained non-monotone max

- Each "sweep" of the algorithm is $O(n)$.
- Running the algorithm $1\times$ (with an arbitrary variable order) results in a $1/3$ approximation.

Linear time algorithm unconstrained non-monotone max

- Each "sweep" of the algorithm is $O(n)$.
- Running the algorithm $1\times$ (with an arbitrary variable order) results in a $1/3$ approximation.
- The $1/2$ guarantee is in expected value (the expected solution has the $1/2$ guarantee).

Linear time algorithm unconstrained non-monotone max

- Each "sweep" of the algorithm is $O(n)$.
- Running the algorithm $1\times$ (with an arbitrary variable order) results in a $1/3$ approximation.
- The $1/2$ guarantee is in expected value (the expected solution has the $1/2$ guarantee).
- In practice, run it multiple times, each with a different random permutation of the elements, and then take the cumulative best.

Linear time algorithm unconstrained non-monotone max

- Each "sweep" of the algorithm is $O(n)$.
- Running the algorithm $1\times$ (with an arbitrary variable order) results in a $1/3$ approximation.
- The $1/2$ guarantee is in expected value (the expected solution has the $1/2$ guarantee).
- In practice, run it multiple times, each with a different random permutation of the elements, and then take the cumulative best.
- It may be possible to choose the random order smartly to get better results in practice.

More general still: multiple constraints different types

- In the past several years, there has been a plethora of papers on maximizing both monotone and non-monotone submodular functions under various combinations of one or more knapsack and/or matroid constraints.

More general still: multiple constraints different types

- In the past several years, there has been a plethora of papers on maximizing both monotone and non-monotone submodular functions under various combinations of one or more knapsack and/or matroid constraints.
- The approximation quality is usually some function of the number of matroids, and is often not a function of the number of knapsacks.

More general still: multiple constraints different types

- In the past several years, there has been a plethora of papers on maximizing both monotone and non-monotone submodular functions under various combinations of one or more knapsack and/or matroid constraints.
- The approximation quality is usually some function of the number of matroids, and is often not a function of the number of knapsacks.
- Often the computational costs of the algorithms are prohibitive (e.g., exponential in k) with large constants, so these algorithms might not scale.

More general still: multiple constraints different types

- In the past several years, there has been a plethora of papers on maximizing both monotone and non-monotone submodular functions under various combinations of one or more knapsack and/or matroid constraints.
- The approximation quality is usually some function of the number of matroids, and is often not a function of the number of knapsacks.
- Often the computational costs of the algorithms are prohibitive (e.g., exponential in k) with large constants, so these algorithms might not scale.
- On the other hand, these algorithms offer deep and interesting intuition into submodular functions, beyond what we have covered here.

Some results on submodular maximization

- As we've seen, we can get $1 - 1/e$ for non-negative monotone submodular (polymatroid) functions with greedy algorithm under cardinality constraints, and this is tight.

Some results on submodular maximization

- As we've seen, we can get $1 - 1/e$ for non-negative monotone submodular (polymatroid) functions with greedy algorithm under cardinality constraints, and this is tight.
- For general matroid, greedy reduces to $1/2$ approximation (as we've seen).

Some results on submodular maximization

- As we've seen, we can get $1 - 1/e$ for non-negative monotone submodular (polymatroid) functions with greedy algorithm under cardinality constraints, and this is tight.
- For general matroid, greedy reduces to $1/2$ approximation (as we've seen).
- We can recover $1 - 1/e$ approximation using the continuous greedy algorithm on the multilinear extension and then using pipage rounding to re-integerize the solution (see J. Vondrak's publications).

Some results on submodular maximization

- As we've seen, we can get $1 - 1/e$ for non-negative monotone submodular (polymatroid) functions with greedy algorithm under cardinality constraints, and this is tight.
- For general matroid, greedy reduces to $1/2$ approximation (as we've seen).
- We can recover $1 - 1/e$ approximation using the continuous greedy algorithm on the multilinear extension and then using pipage rounding to re-integerize the solution (see J. Vondrak's publications).
- More general constraints are possible too, as we see on the next table (for references, see Jan Vondrak's publications http://theory.stanford.edu/~jvondrak/).

Submodular Max Summary - From J. Vondrak

Monotone Maximization

Constraint	Approximation	Hardness	Technique
$\lvert S\rvert \leq k$	$1-1/e$	$1-1/e$	greedy
matroid	$1-1/e$	$1-1/e$	multilinear ext.
$O(1)$ knapsacks	$1-1/e$	$1-1/e$	multilinear ext.
k matroids	$k+\epsilon$	$k/\log k$	local search
k matroids and $O(1)$ knapsacks	$O(k)$	$k/\log k$	multilinear ext.

Nonmonotone Maximization

Constraint	Approximation	Hardness	Technique
Unconstrained	$1/2$	$1/2$	combinatorial
matroid	$1/e$	0.48	multilinear ext.
$O(1)$ knapsacks	$1/e$	0.49	multilinear ext.
k matroids	$k+O(1)$	$k/\log k$	local search
k matroids and $O(1)$ knapsacks	$O(k)$	$k/\log k$	multilinear ext.

Continuous Extensions of Discrete Set Functions

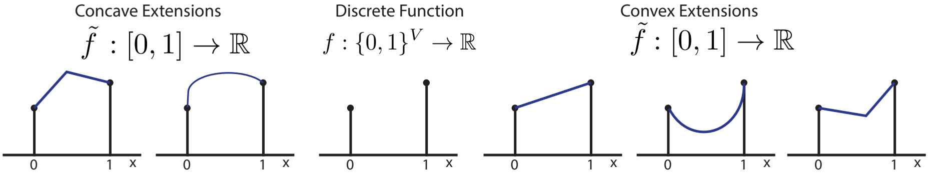
- Any function $f : 2^V \rightarrow \mathbb{R}$ (equivalently $f : \{0,1\}^V \rightarrow \mathbb{R}$) can be extended to a continuous function in the sense $\tilde{f} : [0,1]^V \rightarrow \mathbb{R}$.

Continuous Extensions of Discrete Set Functions

- Any function $f : 2^V \to \mathbb{R}$ (equivalently $f : \{0,1\}^V \to \mathbb{R}$) can be extended to a continuous function in the sense $\tilde{f} : [0,1]^V \to \mathbb{R}$.
- This may be tight (i.e., $\tilde{f}(\mathbf{1}_A) = f(A)$ for all A). I.e., the extension $\tilde{f}$ coincides with f at the hypercube vertices.

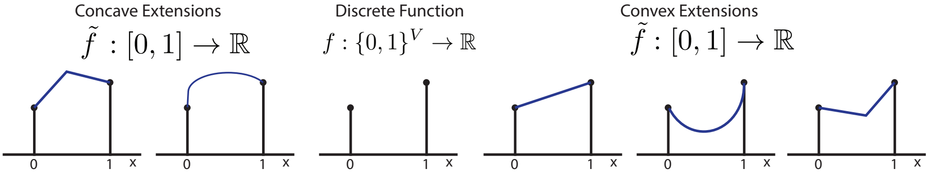
Continuous Extensions of Discrete Set Functions

- Any function $f : 2^V \to \mathbb{R}$ (equivalently $f : \{0,1\}^V \to \mathbb{R}$) can be extended to a continuous function in the sense $\tilde{f} : [0,1]^V \to \mathbb{R}$.
- This may be tight (i.e., $\tilde{f}(\mathbf{1}_A) = f(A)$ for all A). I.e., the extension $\tilde{f}$ coincides with f at the hypercube vertices.
- In fact, any such discrete function defined on the vertices of the n-D hypercube $\{0,1\}^n$ has a variety of both convex and concave extensions tight at the vertices (Crama & Hammer'11). Example $n = 1$,



Continuous Extensions of Discrete Set Functions

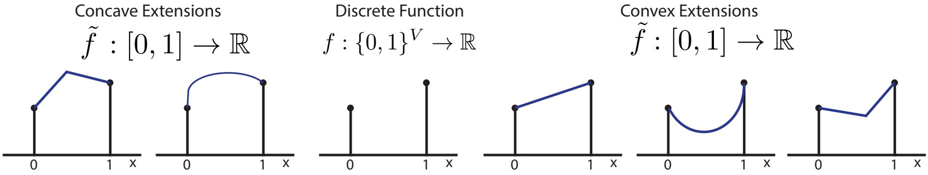
- Any function $f : 2^V \to \mathbb{R}$ (equivalently $f : \{0,1\}^V \to \mathbb{R}$) can be extended to a continuous function in the sense $\tilde{f} : [0,1]^V \to \mathbb{R}$.
- This may be tight (i.e., $\tilde{f}(\mathbf{1}_A) = f(A)$ for all A). I.e., the extension $\tilde{f}$ coincides with f at the hypercube vertices.
- In fact, any such discrete function defined on the vertices of the n-D hypercube $\{0,1\}^n$ has a variety of both convex and concave extensions tight at the vertices (Crama & Hammer'11). Example $n = 1$,



- Since there are an exponential number of vertices $\{0,1\}^n$, important questions regarding such extensions is:

Continuous Extensions of Discrete Set Functions

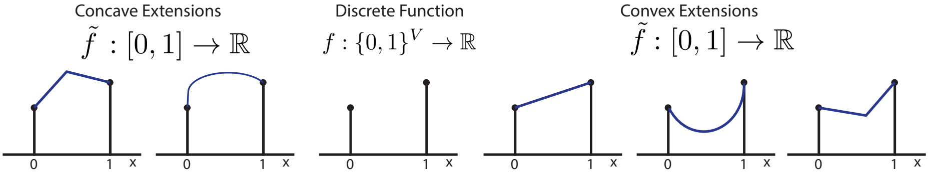
- Any function $f : 2^V \to \mathbb{R}$ (equivalently $f : \{0,1\}^V \to \mathbb{R}$) can be extended to a continuous function in the sense $\tilde{f} : [0,1]^V \to \mathbb{R}$.
- This may be tight (i.e., $\tilde{f}(\mathbf{1}_A) = f(A)$ for all A). I.e., the extension $\tilde{f}$ coincides with f at the hypercube vertices.
- In fact, any such discrete function defined on the vertices of the n-D hypercube $\{0,1\}^n$ has a variety of both convex and concave extensions tight at the vertices (Crama & Hammer'11). Example $n = 1$,



- Since there are an exponential number of vertices $\{0,1\}^n$, important questions regarding such extensions is:
  1. When are they computationally feasible to obtain or estimate?

Continuous Extensions of Discrete Set Functions

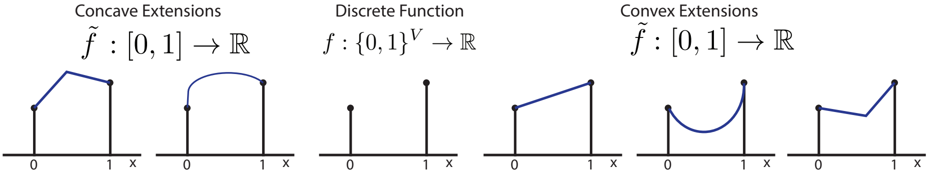
- Any function $f : 2^V \to \mathbb{R}$ (equivalently $f : \{0,1\}^V \to \mathbb{R}$) can be extended to a continuous function in the sense $\tilde{f} : [0,1]^V \to \mathbb{R}$.
- This may be tight (i.e., $\tilde{f}(\mathbf{1}_A) = f(A)$ for all A). I.e., the extension $\tilde{f}$ coincides with f at the hypercube vertices.
- In fact, any such discrete function defined on the vertices of the n-D hypercube $\{0,1\}^n$ has a variety of both convex and concave extensions tight at the vertices (Crama & Hammer'11). Example $n = 1$,



- Since there are an exponential number of vertices $\{0,1\}^n$, important questions regarding such extensions is:
  1. When are they computationally feasible to obtain or estimate?
  2. When do they have nice mathematical properties?

Continuous Extensions of Discrete Set Functions

- Any function $f : 2^V \to \mathbb{R}$ (equivalently $f : \{0,1\}^V \to \mathbb{R}$) can be extended to a continuous function in the sense $\tilde{f} : [0,1]^V \to \mathbb{R}$.
- This may be tight (i.e., $\tilde{f}(\mathbf{1}_A) = f(A)$ for all A). I.e., the extension $\tilde{f}$ coincides with f at the hypercube vertices.
- In fact, any such discrete function defined on the vertices of the n-D hypercube $\{0,1\}^n$ has a variety of both convex and concave extensions tight at the vertices (Crama & Hammer'11). Example $n = 1$,



- Since there are an exponential number of vertices $\{0,1\}^n$, important questions regarding such extensions is:
  1. When are they computationally feasible to obtain or estimate?
  2. When do they have nice mathematical properties?
  3. When are they useful for something practical?

Def: Convex Envelope of a function

- Given any function $h : \mathbb{R}^n \to \mathbb{R}$, define new function $\check{h} : \mathbf{R}^n \to \mathbb{R}$ via:

$$\check{h}(x) = \sup \{g(x) : g \text{ is convex } \& \ g(y) \leq h(y), \forall y \in \mathbb{R}^n\} \tag{14.7}$$

Def: Convex Envelope of a function

- Given any function $h : \mathbb{R}^n \to \mathbb{R}$, define new function $\check{h} : \mathbf{R}^n \to \mathbb{R}$ via:

$$\check{h}(x) = \sup \{g(x) : g \text{ is convex } \& \ g(y) \leq h(y), \forall y \in \mathbb{R}^n\} \tag{14.7}$$

- I.e., (1) $\check{h}(x)$ is convex, (2) $\check{h}(x) \leq h(x), \forall x$, and (3) if $g(x)$ is any convex function having the property that $g(x) \leq h(x), \forall x$, then $g(x) \leq \check{h}(x)$.

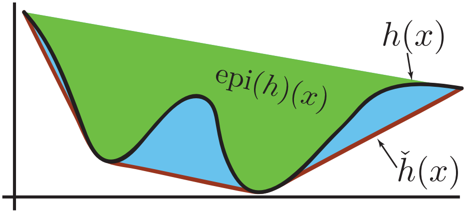
Def: Convex Envelope of a function

- Given any function $h : \mathbb{R}^n \to \mathbb{R}$, define new function $\check{h} : \mathbf{R}^n \to \mathbb{R}$ via:

$$\check{h}(x) = \sup \{g(x) : g \text{ is convex } \& \ g(y) \leq h(y), \forall y \in \mathbb{R}^n\} \quad (14.7)$$

- I.e., (1) $\check{h}(x)$ is convex, (2) $\check{h}(x) \leq h(x), \forall x$, and (3) if $g(x)$ is any convex function having the property that $g(x) \leq h(x), \forall x$, then $g(x) \leq \check{h}(x)$.
- Alternatively,

$$\check{h}(x) = \inf \{t : (x,t) \in \text{convexhull}(\text{epigraph}(h))\} \quad (14.8)$$



Convex Closure of Discrete Set Functions

- Given set function $f : 2^V \to \mathbb{R}$, an arbitrary (i.e., not necessarily submodular nor supermodular) set function, define a function $\check{f} : [0,1]^V \to \mathbb{R}$, as

$$\check{f}(x) = \min_{p \in \triangle^n(x)} \sum_{S \subseteq V} p_S f(S) \tag{14.9}$$

where $\triangle^n(x) =$
$\left\{ p \in \mathbb{R}^{2^n} : \sum_{S \subseteq V} p_S = 1,\ p_S \geq 0 \forall S \subseteq V,\ \&\ \sum_{S \subseteq V} p_S \mathbf{1}_S = x \right\}$

Convex Closure of Discrete Set Functions

- Given set function $f : 2^V \to \mathbb{R}$, an arbitrary (i.e., not necessarily submodular nor supermodular) set function, define a function $\check{f} : [0,1]^V \to \mathbb{R}$, as

$$\check{f}(x) = \min_{p \in \triangle^n(x)} \sum_{S \subseteq V} p_S f(S) \tag{14.9}$$

where $\triangle^n(x) =$
$\left\{ p \in \mathbb{R}^{2^n} : \sum_{S \subseteq V} p_S = 1,\ p_S \geq 0 \forall S \subseteq V,\ \&\ \sum_{S \subseteq V} p_S \mathbf{1}_S = x \right\}$

- Hence, $\triangle^n(x)$ is the set of all probability distributions over the 2^n vertices of the hypercube, and where the expected value of the characteristic vectors of those points is equal to x, i.e., for any $p \in \triangle^n(x)$, $E_{S \sim p}(\mathbf{1}_S) = \sum_{S \subseteq V} p_S \mathbf{1}_S = x$.

Convex Closure of Discrete Set Functions

- Given set function $f : 2^V \to \mathbb{R}$, an arbitrary (i.e., not necessarily submodular nor supermodular) set function, define a function $\check{f} : [0,1]^V \to \mathbb{R}$, as

$$\check{f}(x) = \min_{p \in \triangle^n(x)} \sum_{S \subseteq V} p_S f(S) \tag{14.9}$$

where $\triangle^n(x) =$
$\left\{p \in \mathbb{R}^{2^n} : \sum_{S \subseteq V} p_S = 1,\ p_S \geq 0 \forall S \subseteq V,\ \&\ \sum_{S \subseteq V} p_S \mathbf{1}_S = x\right\}$

- Hence, $\triangle^n(x)$ is the set of all probability distributions over the 2^n vertices of the hypercube, and where the expected value of the characteristic vectors of those points is equal to x, i.e., for any $p \in \triangle^n(x)$, $E_{S \sim p}(\mathbf{1}_S) = \sum_{S \subseteq V} p_S \mathbf{1}_S = x$.
- Hence, $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$

Convex Closure of Discrete Set Functions

- Given set function $f : 2^V \to \mathbb{R}$, an arbitrary (i.e., not necessarily submodular nor supermodular) set function, define a function $\check{f} : [0,1]^V \to \mathbb{R}$, as

$$\check{f}(x) = \min_{p \in \triangle^n(x)} \sum_{S \subseteq V} p_S f(S) \tag{14.9}$$

where $\triangle^n(x) =$
$\left\{ p \in \mathbb{R}^{2^n} : \sum_{S \subseteq V} p_S = 1,\ p_S \geq 0 \forall S \subseteq V, \ \&\ \sum_{S \subseteq V} p_S \mathbf{1}_S = x \right\}$

- Hence, $\triangle^n(x)$ is the set of all probability distributions over the 2^n vertices of the hypercube, and where the expected value of the characteristic vectors of those points is equal to x, i.e., for any $p \in \triangle^n(x)$, $E_{S \sim p}(\mathbf{1}_S) = \sum_{S \subseteq V} p_S \mathbf{1}_S = x$.
- Hence, $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$
- Note, this is not (necessarily) the Lovász extension, rather this is a convex extension.

Convex Closure of Discrete Set Functions

- Given, $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$, there are several things we'd like to show:

Convex Closure of Discrete Set Functions

- Given, $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$, there are several things we'd like to show:
  1. That $\check{f}$ is tight (i.e., $\forall S \subseteq V$, we have $\check{f}(\mathbf{1}_S) = f(S)$).

Convex Closure of Discrete Set Functions

- Given, $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$, there are several things we'd like to show:
  1. That $\check{f}$ is tight (i.e., $\forall S \subseteq V$, we have $\check{f}(\mathbf{1}_S) = f(S)$).
  2. That $\check{f}$ is convex (and consequently, that any arbitrary set function has a tight convex extension).

Convex Closure of Discrete Set Functions

- Given, $\breve{f}(x) = \min_{p \in \triangle^n(x)} E_{S\sim p}[f(S)]$, there are several things we'd like to show:
  1. That $\breve{f}$ is tight (i.e., $\forall S \subseteq V$, we have $\breve{f}(\mathbf{1}_S) = f(S)$).
  2. That $\breve{f}$ is convex (and consequently, that any arbitrary set function has a tight convex extension).
  3. That the convex closure $\breve{f}$ is the convex envelope of the function defined only on the hypercube vertices, and that takes value $f(S)$ at $\mathbf{1}_S$.

Convex Closure of Discrete Set Functions

- Given, $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$, there are several things we'd like to show:
  1. That $\check{f}$ is tight (i.e., $\forall S \subseteq V$, we have $\check{f}(\mathbf{1}_S) = f(S)$).
  2. That $\check{f}$ is convex (and consequently, that any arbitrary set function has a tight convex extension).
  3. That the convex closure $\check{f}$ is the convex envelope of the function defined only on the hypercube vertices, and that takes value $f(S)$ at $\mathbf{1}_S$.
  4. The definition of the Lovász extension of a set function, and that $\check{f}$ is the Lovász extension iff f is submodular.

Tightness of Convex Closure

Lemma 14.4.1

$\forall A \subseteq V$, *we have* $\check{f}(\mathbf{1}_A) = f(A)$.

Proof.

- Define p^x to be an achiving argmin in $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$.

Tightness of Convex Closure

Lemma 14.4.1

$\forall A \subseteq V$, *we have* $\check{f}(\mathbf{1}_A) = f(A)$.

Proof.

- Define p^x to be an achiving argmin in $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$.
- Take an arbitrary A, so that $\mathbf{1}_A = \sum_{S \subseteq V} p_S^{\mathbf{1}_A} \mathbf{1}_S = \mathbf{1}_A$.

Tightness of Convex Closure

Lemma 14.4.1

$\forall A \subseteq V$, *we have* $\breve{f}(\mathbf{1}_A) = f(A)$.

Proof.

- Define p^x to be an achiving argmin in $\breve{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$.
- Take an arbitrary A, so that $\mathbf{1}_A = \sum_{S \subseteq V} p_S^{\mathbf{1}_A} \mathbf{1}_S = \mathbf{1}_A$.
- Suppose $\exists S'$ with $S' \setminus A \neq 0$ having $p_{S'}^{\mathbf{1}_A} > 0$. This would mean, for any $v \in S' \setminus A$, that $\left(\sum_S p_S^{\mathbf{1}_A} \mathbf{1}_S\right)(v) > 0$, a contradiction.

Tightness of Convex Closure

Lemma 14.4.1

$\forall A \subseteq V$, *we have* $\check{f}(\mathbf{1}_A) = f(A)$.

Proof.

- Define p^x to be an achiving argmin in $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$.
- Take an arbitrary A, so that $\mathbf{1}_A = \sum_{S \subseteq V} p_S^{\mathbf{1}_A} \mathbf{1}_S = \mathbf{1}_A$.
- Suppose $\exists S'$ with $S' \setminus A \neq 0$ having $p_{S'}^{\mathbf{1}_A} > 0$. This would mean, for any $v \in S' \setminus A$, that $\left(\sum_S p_S^{\mathbf{1}_A} \mathbf{1}_S\right)(v) > 0$, a contradiction.
- Suppose $\exists S'$ s.t. $A \setminus S' \neq \emptyset$ with $p_{S'}^{\mathbf{1}_A} > 0$.

Tightness of Convex Closure

Lemma 14.4.1

$\forall A \subseteq V$, *we have* $\breve{f}(\mathbf{1}_A) = f(A)$.

Proof.

- Define p^x to be an achiving argmin in $\breve{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$.
- Take an arbitrary A, so that $\mathbf{1}_A = \sum_{S \subseteq V} p_S^{\mathbf{1}_A} \mathbf{1}_S = \mathbf{1}_A$.
- Suppose $\exists S'$ with $S' \setminus A \neq 0$ having $p_{S'}^{\mathbf{1}_A} > 0$. This would mean, for any $v \in S' \setminus A$, that $\left(\sum_S p_S^{\mathbf{1}_A} \mathbf{1}_S\right)(v) > 0$, a contradiction.
- Suppose $\exists S'$ s.t. $A \setminus S' \neq \emptyset$ with $p_{S'}^{\mathbf{1}_A} > 0$.
- Then, for any $v \in A \setminus S'$, consider below leading to a contradiction

$$\underbrace{p_{S'} \mathbf{1}_{S'}}_{>0} + \underbrace{\sum_{\substack{S \subseteq A \\ S \neq S'}} p_S \mathbf{1}_S}_{\text{can't sum to 1}} \Rightarrow \Bigg(\sum_{\substack{S \subseteq A \\ S \neq S'}} p_s \mathbf{1}_S \Bigg)(v) < 1 \tag{14.10}$$

I.e., $v \in A$ so it must get value 1, but since $v \notin S'$, v is deficient.

Convexity of the Convex Closure

Lemma 14.4.2

$\breve{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$ *is convex in* $[0,1]^V$.

Proof.

- Let $x, y \in [0,1]^V$, $0 \leq \lambda \leq 1$, and $z = \lambda x + (1-\lambda) y$, then

$$\lambda \breve{f}(x) + (1-\lambda)\breve{f}(y) = \lambda \sum_S p^x_S f(S) + (1-\lambda) \sum_S p^y_S f(S) \tag{14.11}$$

$$= \sum_S (\lambda p^x_S + (1-\lambda) p^y_S) f(S) \tag{14.12}$$

$$= \sum_S p^{z'}_S f(S) \geq \min_{p \in \triangle^n(z)} E_{S \sim p}[f(S)] \tag{14.13}$$

$$= \breve{f}(z) = \breve{f}(\lambda x + (1-\lambda) y) \tag{14.14}$$

Convexity of the Convex Closure

Lemma 14.4.2

$\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S\sim p}[f(S)]$ *is convex in* $[0,1]^V$.

Proof.

- Let $x, y \in [0,1]^V$, $0 \leq \lambda \leq 1$, and $z = \lambda x + (1-\lambda)y$, then

$$\lambda \check{f}(x) + (1-\lambda)\check{f}(y) = \lambda \sum_S p^x_S f(S) + (1-\lambda)\sum_S p^y_S f(S) \quad (14.11)$$

$$= \sum_S (\lambda p^x_S + (1-\lambda) p^y_S) f(S) \quad (14.12)$$

$$= \sum_S p^{z'}_S f(S) \geq \min_{p \in \triangle^n(z)} E_{S\sim p}[f(S)] \quad (14.13)$$

$$= \check{f}(z) = \check{f}(\lambda x + (1-\lambda)y) \quad (14.14)$$

- Note that $p^{z'}_S = \lambda p^x_S + (1-\lambda)p^y_S$ and is feasible in the min since $\sum_S p^{z'}_S = 1$, $p^{z'}_S \geq 0$ and $\sum_S p^{z'}_S \mathbf{1}_S = z$.

Def: Convex Envelope of a function

- Given any function $h : \mathbb{R}^n \to \mathbb{R}$, define new function $\check{h} : \mathbf{R}^n \to \mathbb{R}$ via:

$$\check{h}(x) = \sup \{g(x) : g \text{ is convex } \& \; g(y) \leq h(y), \forall y \in \mathbb{R}^n\} \tag{14.7}$$

Convex Closure is the Convex Envelope

Lemma 14.4.3

$\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$ *is the convex envelope.*

Proof.

- Suppose $\exists$ a convex $\bar{f}$ with $\bar{f}(\mathbf{1}_A) = f(A) = \check{f}(\mathbf{1}_A), \forall A \subseteq V$ and $\exists x \in [0,1]^V$ s.t. $\bar{f}(x) > \check{f}(x)$.
- Define p^x to be an achiving argmin in $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$. Hence, we have $x = \sum_S p^x_S \mathbf{1}_S$. Thus

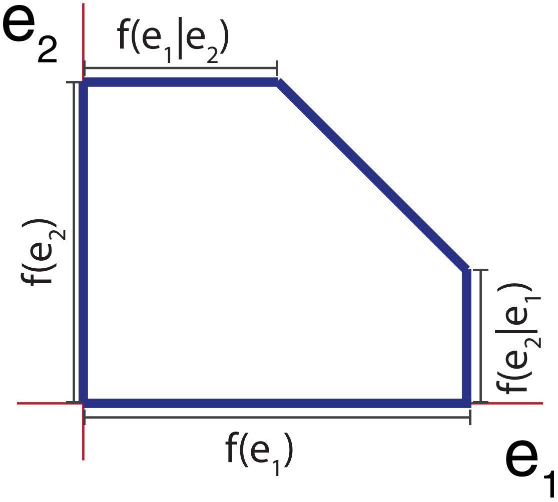
$$\check{f}(x) = \sum_S p^x_S f(S) = \sum_S p^x_S \bar{f}(\mathbf{1}_S) \tag{14.15}$$

$$< \bar{f}(x) = \bar{f}\Big(\sum_S p^x_S \mathbf{1}_S\Big) \tag{14.16}$$

but this contradicts the convexity of $\bar{f}$.

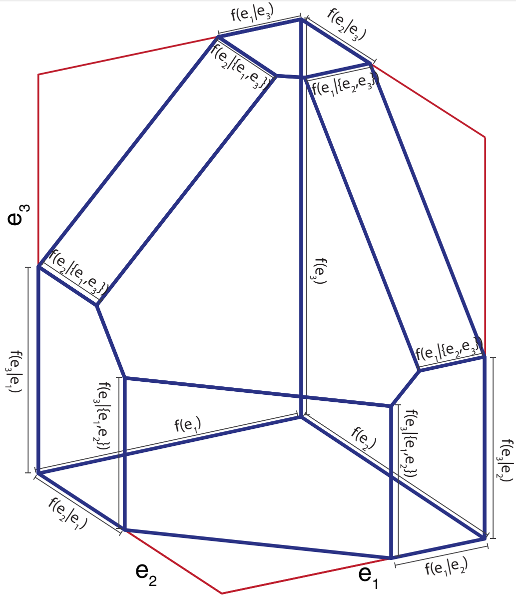
Polymatroid with labeled edge lengths

- Recall $f(e|A) = f(A+e) - f(A)$
- Notice how submodularity, $f(e|B) \leq f(e|A)$ for $A \subseteq B$, defines the shape of the polytope.
- In fact, we have strictness here $f(e|B) < f(e|A)$ for $A \subset B$.
- Also, consider how the greedy algorithm proceeds along the edges of the polytope.



Polymatroid with labeled edge lengths

- Recall $f(e|A) = f(A+e) - f(A)$
- Notice how submodularity, $f(e|B) \leq f(e|A)$ for $A \subseteq B$, defines the shape of the polytope.
- In fact, we have strictness here $f(e|B) < f(e|A)$ for $A \subset B$.
- Also, consider how the greedy algorithm proceeds along the edges of the polytope.



Optimization over P_f

- Consider the following optimization. Given $w \in \mathbb{R}^E$,

$$\begin{aligned} &\text{maximize} && w^\intercal x && (14.17a) \\ &\text{subject to} && x \in P_f && (14.17b) \end{aligned}$$

Optimization over P_f

- Consider the following optimization. Given $w \in \mathbb{R}^E$,

$$\begin{aligned} \text{maximize} \quad & w^\intercal x && (14.17a) \\ \text{subject to} \quad & x \in P_f && (14.17b) \end{aligned}$$

- Since P_f is down closed, if $\exists e \in E$ with $w(e) < 0$ then the solution above is unboundedly large.

Optimization over P_f

- Consider the following optimization. Given $w \in \mathbb{R}^E$,

$$\begin{aligned} \text{maximize} \quad & w^\intercal x & \text{(14.17a)} \\ \text{subject to} \quad & x \in P_f & \text{(14.17b)} \end{aligned}$$

- Since P_f is down closed, if $\exists e \in E$ with $w(e) < 0$ then the solution above is unboundedly large. Hence, assume $w \in \mathbb{R}_+^E$.

Optimization over P_f

- Consider the following optimization. Given $w \in \mathbb{R}^E$,

$$\begin{aligned} \text{maximize} \quad & w^\intercal x && (14.17a) \\ \text{subject to} \quad & x \in P_f && (14.17b) \end{aligned}$$

- Since P_f is down closed, if $\exists e \in E$ with $w(e) < 0$ then the solution above is unboundedly large. Hence, assume $w \in \mathbb{R}_+^E$.
- Due to Theorem ??, any $x \in P_f$ with $x \notin B_f$ is dominated by $x \leq y \in B_f$ which can only increase $w^\intercal x \leq w^\intercal y$ when $w \in \mathbb{R}_+^E$.

Optimization over P_f

- Consider the following optimization. Given $w \in \mathbb{R}^E$,

$$\begin{aligned} &\text{maximize} && w^\intercal x && (14.17a) \\ &\text{subject to} && x \in P_f && (14.17b) \end{aligned}$$

- Since P_f is down closed, if $\exists e \in E$ with $w(e) < 0$ then the solution above is unboundedly large. Hence, assume $w \in \mathbb{R}_+^E$.
- Due to Theorem ??, any $x \in P_f$ with $x \notin B_f$ is dominated by $x \leq y \in B_f$ which can only increase $w^\intercal x \leq w^\intercal y$ when $w \in \mathbb{R}_+^E$.
- Hence, the problem is equivalent to: given $w \in \mathbb{R}_+^E$,

$$\begin{aligned} &\text{maximize} && w^\intercal x && (14.18a) \\ &\text{subject to} && x \in B_f && (14.18b) \end{aligned}$$

Optimization over P_f

- Consider the following optimization. Given $w \in \mathbb{R}^E$,

$$\begin{aligned} &\text{maximize} && w^\intercal x && (14.17a) \\ &\text{subject to} && x \in P_f && (14.17b) \end{aligned}$$

- Since P_f is down closed, if $\exists e \in E$ with $w(e) < 0$ then the solution above is unboundedly large. Hence, assume $w \in \mathbb{R}^E_+$.
- Due to Theorem **??**, any $x \in P_f$ with $x \notin B_f$ is dominated by $x \leq y \in B_f$ which can only increase $w^\intercal x \leq w^\intercal y$ when $w \in \mathbb{R}^E_+$.
- Hence, the problem is equivalent to: given $w \in \mathbb{R}^E_+$,

$$\begin{aligned} &\text{maximize} && w^\intercal x && (14.18a) \\ &\text{subject to} && x \in B_f && (14.18b) \end{aligned}$$

- Moreover, we can have $w \in \mathbb{R}^E$ if we insist on $x \in B_f$.

A continuous extension of f

- Consider again optimization problem. Given $w \in \mathbb{R}^E$,

$$\begin{aligned} &\text{maximize} && w^\intercal x && (14.19a) \\ &\text{subject to} && x \in B_f && (14.19b) \end{aligned}$$

A continuous extension of f

- Consider again optimization problem. Given $w \in \mathbb{R}^E$,

$$\begin{aligned} &\text{maximize} && w^\intercal x && (14.19a) \\ &\text{subject to} && x \in B_f && (14.19b) \end{aligned}$$

- We may consider this optimization problem a function $\breve{f} : \mathbb{R}^E \to \mathbb{R}$ of $w \in \mathbb{R}^E$, defined as:

$$\breve{f}(w) = \max(wx : x \in B_f) \quad (14.20)$$

A continuous extension of f

- Consider again optimization problem. Given $w \in \mathbb{R}^E$,

$$\begin{aligned} \text{maximize} \quad & w^\intercal x && (14.19a) \\ \text{subject to} \quad & x \in B_f && (14.19b) \end{aligned}$$

- We may consider this optimization problem a function $\breve{f} : \mathbb{R}^E \to \mathbb{R}$ of $w \in \mathbb{R}^E$, defined as:

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.20}$$

- Hence, for any w, from the solution to the above theorem (as we have seen), we can compute the value of this function using Edmond's greedy algorithm.

Edmond's Theorem: The Greedy Algorithm

- Edmonds proved that the solution to $\breve{f}(w) = \max(wx : x \in B_f)$ is solved by the greedy algorithm iff f is submodular.
- In particular, sort choose element order $(e_1, e_2, \ldots, e_m)$ based on decreasing w,so that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
- Define the chain with i^{th} element $E_i = \{e_1, e_2, \ldots, e_i\}$.
- Define a vector $x^* \in \mathbb{R}^V$ where element e_i has value $x(e_i) = f(e_i | E_{i-1})$ for all $i \in V$.
- Then $\langle w, x^* \rangle = \max(wx : x \in B_f)$

Theorem 14.5.1 (Edmonds)

If $f : 2^E \to \mathbb{R}_+$ is given, and B is a polytope in $\mathbb{R}_+^E$ of the form $B = \left\{x \in \mathbb{R}_+^E : x(A) \leq f(A), \forall A \subseteq E, x(E) = f(E)\right\}$, then the greedy solution to the problem $\max(w^\intercal x : x \in P)$ is $\forall w$ optimum iff f is monotone non-decreasing submodular (i.e., iff P is a polymatroid).

A continuous extension of submodular f

- That is, given a submodular function f, a $w \in \mathbb{R}^E$, choose element order $(e_1, e_2, \ldots, e_m)$ based on decreasing w,so that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.

A continuous extension of submodular f

- That is, given a submodular function f, a $w \in \mathbb{R}^E$, choose element order $(e_1, e_2, \ldots, e_m)$ based on decreasing w,so that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
- Define the chain with i^{th} element $E_i = \{e_1, e_2, \ldots, e_i\}$, we have

$$\breve{f}(w)$$

A continuous extension of submodular f

- That is, given a submodular function f, a $w \in \mathbb{R}^E$, choose element order $(e_1, e_2, \ldots, e_m)$ based on decreasing w,so that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
- Define the chain with i^{th} element $E_i = \{e_1, e_2, \ldots, e_i\}$, we have

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.21}$$

A continuous extension of submodular f

- That is, given a submodular function f, a $w \in \mathbb{R}^E$, choose element order $(e_1, e_2, \ldots, e_m)$ based on decreasing w,so that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
- Define the chain with i^{th} element $E_i = \{e_1, e_2, \ldots, e_i\}$, we have

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.21}$$

$$= \sum_{i=1}^{m} w(e_i) f(e_i | E_{i-1}) = \sum_{i=1}^{m} w(e_i) x(e_i) \tag{14.22}$$

A continuous extension of submodular f

- That is, given a submodular function f, a $w \in \mathbb{R}^E$, choose element order $(e_1, e_2, \ldots, e_m)$ based on decreasing w,so that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
- Define the chain with i^{th} element $E_i = \{e_1, e_2, \ldots, e_i\}$, we have

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.21}$$

$$= \sum_{i=1}^{m} w(e_i) f(e_i | E_{i-1}) = \sum_{i=1}^{m} w(e_i) x(e_i) \tag{14.22}$$

$$= \sum_{i=1}^{m} w(e_i)(f(E_i) - f(E_{i-1})) \tag{14.23}$$

A continuous extension of submodular f

- That is, given a submodular function f, a $w \in \mathbb{R}^E$, choose element order $(e_1, e_2, \dots, e_m)$ based on decreasing w,so that $w(e_1) \geq w(e_2) \geq \dots \geq w(e_m)$.
- Define the chain with i^{th} element $E_i = \{e_1, e_2, \dots, e_i\}$, we have

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.21}$$

$$= \sum_{i=1}^{m} w(e_i) f(e_i | E_{i-1}) = \sum_{i=1}^{m} w(e_i) x(e_i) \tag{14.22}$$

$$= \sum_{i=1}^{m} w(e_i)(f(E_i) - f(E_{i-1})) \tag{14.23}$$

$$= w(e_m) f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1})) f(E_i) \tag{14.24}$$

A continuous extension of submodular f

- That is, given a submodular function f, a $w \in \mathbb{R}^E$, choose element order $(e_1, e_2, \ldots, e_m)$ based on decreasing w,so that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
- Define the chain with i^{th} element $E_i = \{e_1, e_2, \ldots, e_i\}$, we have

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.21}$$

$$= \sum_{i=1}^{m} w(e_i) f(e_i | E_{i-1}) = \sum_{i=1}^{m} w(e_i) x(e_i) \tag{14.22}$$

$$= \sum_{i=1}^{m} w(e_i)(f(E_i) - f(E_{i-1})) \tag{14.23}$$

$$= w(e_m) f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1})) f(E_i) \tag{14.24}$$

- We say that $\emptyset \triangleq E_0 \subset E_1 \subset E_2 \subset \cdots \subset E_m = E$ forms a chain based on w.

A continuous extension of submodular f

- Definition of the continuous extension, once again, for reference:

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.25}$$

A continuous extension of submodular f

- Definition of the continuous extension, once again, for reference:

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.25}$$

- Therefore, if f is a submodular function, we can write

$$\breve{f}(w)$$

A continuous extension of submodular f

- Definition of the continuous extension, once again, for reference:

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.25}$$

- Therefore, if f is a submodular function, we can write

$$\breve{f}(w) = w(e_m)f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1}))f(E_i) \tag{14.26}$$

A continuous extension of submodular f

- Definition of the continuous extension, once again, for reference:

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.25}$$

- Therefore, if f is a submodular function, we can write

$$\breve{f}(w) = w(e_m)f(E_m) + \sum_{i=1}^{m-1}(w(e_i) - w(e_{i+1}))f(E_i) \tag{14.26}$$

$$= \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.27}$$

A continuous extension of submodular f

- Definition of the continuous extension, once again, for reference:

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.25}$$

- Therefore, if f is a submodular function, we can write

$$\breve{f}(w) = w(e_m)f(E_m) + \sum_{i=1}^{m-1}(w(e_i) - w(e_{i+1}))f(E_i) \tag{14.26}$$

$$= \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.27}$$

where $\lambda_m = w(e_m)$ and otherwise $\lambda_i = w(e_i) - w(e_{i+1})$, where the elements are sorted descending according to w as before.

A continuous extension of submodular f

- Definition of the continuous extension, once again, for reference:

$$\breve{f}(w) = \max(wx : x \in B_f) \tag{14.25}$$

- Therefore, if f is a submodular function, we can write

$$\breve{f}(w) = w(e_m)f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1}))f(E_i) \tag{14.26}$$

$$= \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.27}$$

  where $\lambda_m = w(e_m)$ and otherwise $\lambda_i = w(e_i) - w(e_{i+1})$, where the elements are sorted descending according to w as before.

- Convex analysis $\Rightarrow$ $\breve{f}(w) = \max(wx : x \in P)$ is always convex in w for any set $P \subseteq R^E$, since a maximum of a set of linear functions (true even when f is not submodular or P is not itself a convex set).

An extension of f

- Recall, for any such $w \in \mathbb{R}^E$, we have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \underbrace{(w_1 - w_2)}_{\lambda_1} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \underbrace{(w_2 - w_3)}_{\lambda_2} \begin{pmatrix} 1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \cdots + \underbrace{(w_{n-1} - w_n)}_{\lambda_{m-1}} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 0 \end{pmatrix} + \underbrace{(w_m)}_{\lambda_m} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix} \tag{14.28}$$

An extension of f

- Recall, for any such $w \in \mathbb{R}^E$, we have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \underbrace{(w_1 - w_2)}_{\lambda_1} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \underbrace{(w_2 - w_3)}_{\lambda_2} \begin{pmatrix} 1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \cdots + \underbrace{(w_{n-1} - w_n)}_{\lambda_{m-1}} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 0 \end{pmatrix} + \underbrace{(w_m)}_{\lambda_m} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix} \tag{14.28}$$

- If we take w in decreasing order, then each coefficient of the vectors is non-negative (except possibly the last one, $\lambda_m = w_m$).

An extension of f

- Recall, for any such $w \in \mathbb{R}^E$, we have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \underbrace{(w_1 - w_2)}_{\lambda_1} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \underbrace{(w_2 - w_3)}_{\lambda_2} \begin{pmatrix} 1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \cdots + \underbrace{(w_{n-1} - w_n)}_{\lambda_{m-1}} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 0 \end{pmatrix} + \underbrace{(w_m)}_{\lambda_m} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix} \tag{14.28}$$

- If we take w in decreasing order, then each coefficient of the vectors is non-negative (except possibly the last one, $\lambda_m = w_m$).
- Often, we take $w \in \mathbb{R}_+^V$ or even $w \in [0,1]^V$, where $\lambda_m \geq 0$.

An extension of f

- Define sets E_i based on this decreasing order of w as follows, for $i = 0, \dots, n$

$$E_i \stackrel{\text{def}}{=} \{e_1, e_2, \dots, e_i\} \tag{14.29}$$

An extension of f

- Define sets E_i based on this decreasing order of w as follows, for $i = 0, \dots, n$

$$E_i \stackrel{\text{def}}{=} \{e_1, e_2, \dots, e_i\} \tag{14.29}$$

- Note that

$$\mathbf{1}_{E_0} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \mathbf{1}_{E_1} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \dots, \mathbf{1}_{E_\ell} = \begin{pmatrix} \left.\begin{matrix} 1 \\ 1 \\ \vdots \\ 1 \end{matrix}\right\} \ell\times \\ \left.\begin{matrix} 0 \\ 0 \\ \vdots \\ 0 \end{matrix}\right\} (n-\ell)\times \end{pmatrix}, \text{ etc.}$$

An extension of f

- Define sets E_i based on this decreasing order of w as follows, for $i = 0, \dots, n$

$$E_i \stackrel{\text{def}}{=} \{e_1, e_2, \dots, e_i\} \tag{14.29}$$

- Note that

$$\mathbf{1}_{E_0} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \mathbf{1}_{E_1} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \dots, \mathbf{1}_{E_\ell} = \begin{pmatrix} \left.\begin{matrix} 1 \\ 1 \\ \vdots \\ 1 \end{matrix}\right\} \ell\times \\ \left.\begin{matrix} 0 \\ 0 \\ \vdots \\ 0 \end{matrix}\right\} (n-\ell)\times \end{pmatrix}, \text{ etc.}$$

- Hence, from the previous and current slide, we have $w = \sum_{i=1}^{m} \lambda_i \mathbf{1}_{E_i}$

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.
- Take $w = \mathbf{1}_A$ for some $A \subseteq E$, so w is vertex of the hypercube.

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.
- Take $w = \mathbf{1}_A$ for some $A \subseteq E$, so w is vertex of the hypercube.
- Order the elements of E in decreasing order of w so that $w(e_1) \geq w(e_2) \geq w(e_3) \geq \cdots \geq w(e_m)$.

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.
- Take $w = \mathbf{1}_A$ for some $A \subseteq E$, so w is vertex of the hypercube.
- Order the elements of E in decreasing order of w so that $w(e_1) \geq w(e_2) \geq w(e_3) \geq \cdots \geq w(e_m)$.
- This means

$$w = (w(e_1), w(e_2), \ldots, w(e_m)) = (\underbrace{1, 1, 1, \ldots, 1}_{|A| \text{ times}}, \underbrace{0, 0, \ldots, 0}_{m-|A| \text{ times}}) \tag{14.30}$$

so that $1_A(i) = 1$ if $i \leq |A|$, and $1_A(i) = 0$ otherwise.

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.
- Take $w = \mathbf{1}_A$ for some $A \subseteq E$, so w is vertex of the hypercube.
- Order the elements of E in decreasing order of w so that $w(e_1) \geq w(e_2) \geq w(e_3) \geq \cdots \geq w(e_m)$.
- This means

$$w = (w(e_1), w(e_2), \ldots, w(e_m)) = (\underbrace{1, 1, 1, \ldots, 1}_{|A| \text{ times}}, \underbrace{0, 0, \ldots, 0}_{m-|A| \text{ times}}) \tag{14.30}$$

so that $1_A(i) = 1$ if $i \leq |A|$, and $1_A(i) = 0$ otherwise.

- For any $f : 2^E \to \mathbb{R}$, $w = \mathbf{1}_A$, since $E_{|A|} = \{e_1, e_2, \ldots, e_{|A|}\} = A$:

$$\breve{f}(w)$$

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.
- Take $w = \mathbf{1}_A$ for some $A \subseteq E$, so w is vertex of the hypercube.
- Order the elements of E in decreasing order of w so that $w(e_1) \geq w(e_2) \geq w(e_3) \geq \cdots \geq w(e_m)$.
- This means

$$w = (w(e_1), w(e_2), \ldots, w(e_m)) = (\underbrace{1, 1, 1, \ldots, 1}_{|A| \text{ times}}, \underbrace{0, 0, \ldots, 0}_{m-|A| \text{ times}}) \tag{14.30}$$

so that $1_A(i) = 1$ if $i \leq |A|$, and $1_A(i) = 0$ otherwise.

- For any $f : 2^E \to \mathbb{R}$, $w = \mathbf{1}_A$, since $E_{|A|} = \{e_1, e_2, \ldots, e_{|A|}\} = A$:

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i)$$

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.
- Take $w = \mathbf{1}_A$ for some $A \subseteq E$, so w is vertex of the hypercube.
- Order the elements of E in decreasing order of w so that $w(e_1) \geq w(e_2) \geq w(e_3) \geq \cdots \geq w(e_m)$.
- This means

$$w = (w(e_1), w(e_2), \ldots, w(e_m)) = (\underbrace{1, 1, 1, \ldots, 1}_{|A| \text{ times}}, \underbrace{0, 0, \ldots, 0}_{m-|A| \text{ times}}) \tag{14.30}$$

so that $1_A(i) = 1$ if $i \leq |A|$, and $1_A(i) = 0$ otherwise.

- For any $f : 2^E \to \mathbb{R}$, $w = \mathbf{1}_A$, since $E_{|A|} = \{e_1, e_2, \ldots, e_{|A|}\} = A$:

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) = w(e_m) f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1}) f(E_i)$$

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.
- Take $w = \mathbf{1}_A$ for some $A \subseteq E$, so w is vertex of the hypercube.
- Order the elements of E in decreasing order of w so that $w(e_1) \geq w(e_2) \geq w(e_3) \geq \dots \geq w(e_m)$.
- This means

$$w = (w(e_1), w(e_2), \dots, w(e_m)) = (\underbrace{1, 1, 1, \dots, 1}_{|A| \text{ times}}, \underbrace{0, 0, \dots, 0}_{m-|A| \text{ times}}) \quad (14.30)$$

so that $1_A(i) = 1$ if $i \leq |A|$, and $1_A(i) = 0$ otherwise.
- For any $f : 2^E \to \mathbb{R}$, $w = \mathbf{1}_A$, since $E_{|A|} = \{e_1, e_2, \dots, e_{|A|}\} = A$:

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) = w(e_m) f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1}) f(E_i)$$

$$= \mathbf{1}_A(m) f(E_m) + \sum_{i=1}^{m-1} (\mathbf{1}_A(i) - \mathbf{1}_A(i+1)) f(E_i) \quad (14.31)$$

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.
- Take $w = \mathbf{1}_A$ for some $A \subseteq E$, so w is vertex of the hypercube.
- Order the elements of E in decreasing order of w so that $w(e_1) \geq w(e_2) \geq w(e_3) \geq \dots \geq w(e_m)$.
- This means

$$w = (w(e_1), w(e_2), \dots, w(e_m)) = (\underbrace{1, 1, 1, \dots, 1}_{|A| \text{ times}}, \underbrace{0, 0, \dots, 0}_{m-|A| \text{ times}}) \tag{14.30}$$

so that $1_A(i) = 1$ if $i \leq |A|$, and $1_A(i) = 0$ otherwise.

- For any $f : 2^E \to \mathbb{R}$, $w = \mathbf{1}_A$, since $E_{|A|} = \{e_1, e_2, \dots, e_{|A|}\} = A$:

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) = w(e_m) f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1}) f(E_i)$$

$$= \mathbf{1}_A(m) f(E_m) + \sum_{i=1}^{m-1} (\mathbf{1}_A(i) - \mathbf{1}_A(i+1)) f(E_i) \tag{14.31}$$

$$= (\mathbf{1}_A(|A|) - \mathbf{1}_A(|A|+1)) f(E_{|A|}) = f(E_{|A|})$$

From $\breve{f}$ back to f, even when f is not submodular

- From the continuous $\breve{f}$, we can recover $f(A)$ for any $A \subseteq V$.
- Take $w = \mathbf{1}_A$ for some $A \subseteq E$, so w is vertex of the hypercube.
- Order the elements of E in decreasing order of w so that $w(e_1) \geq w(e_2) \geq w(e_3) \geq \cdots \geq w(e_m)$.
- This means

$$w = (w(e_1), w(e_2), \ldots, w(e_m)) = (\underbrace{1, 1, 1, \ldots, 1}_{|A| \text{ times}}, \underbrace{0, 0, \ldots, 0}_{m-|A| \text{ times}}) \tag{14.30}$$

so that $1_A(i) = 1$ if $i \leq |A|$, and $1_A(i) = 0$ otherwise.

- For any $f : 2^E \to \mathbb{R}$, $w = \mathbf{1}_A$, since $E_{|A|} = \{e_1, e_2, \ldots, e_{|A|}\} = A$:

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) = w(e_m) f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1}) f(E_i)$$

$$= \mathbf{1}_A(m) f(E_m) + \sum_{i=1}^{m-1} (\mathbf{1}_A(i) - \mathbf{1}_A(i+1)) f(E_i) \tag{14.31}$$

$$= (\mathbf{1}_A(|A|) - \mathbf{1}_A(|A|+1)) f(E_{|A|}) = f(E_{|A|}) = f(A) \tag{14.32}$$

From $\breve{f}$ back to f

- We can view $\breve{f} : [0,1]^E \rightarrow \mathbb{R}$ defined on the hypercube, with f defined as $\breve{f}$ evaluated on the hypercube extreme points (vertices).

From $\breve{f}$ back to f

- We can view $\breve{f} : [0,1]^E \to \mathbb{R}$ defined on the hypercube, with f defined as $\breve{f}$ evaluated on the hypercube extreme points (vertices).
- To summarize, with $\breve{f}(\mathbf{1}_A) = \sum_{i=1}^m \lambda_i f(E_i)$, we have

$$\breve{f}(\mathbf{1}_A) = f(A), \tag{14.33}$$

From $\breve{f}$ back to f

- We can view $\breve{f} : [0,1]^E \to \mathbb{R}$ defined on the hypercube, with f defined as $\breve{f}$ evaluated on the hypercube extreme points (vertices).
- To summarize, with $\breve{f}(\mathbf{1}_A) = \sum_{i=1}^m \lambda_i f(E_i)$, we have

$$\breve{f}(\mathbf{1}_A) = f(A), \tag{14.33}$$

- . . . and when f is submodular, we also have have

$$\breve{f}(\mathbf{1}_A) = \max\{\mathbf{1}_A{}^\intercal x : x \in B_f\} \tag{14.34}$$

$$= \max\{\mathbf{1}_A{}^\intercal x : x(B) \leq f(B), \forall B \subseteq E\} \tag{14.35}$$

From $\breve{f}$ back to f

- We can view $\breve{f} : [0,1]^E \to \mathbb{R}$ defined on the hypercube, with f defined as $\breve{f}$ evaluated on the hypercube extreme points (vertices).
- To summarize, with $\breve{f}(\mathbf{1}_A) = \sum_{i=1}^m \lambda_i f(E_i)$, we have

$$\breve{f}(\mathbf{1}_A) = f(A), \tag{14.33}$$

- ... and when f is submodular, we also have have

$$\breve{f}(\mathbf{1}_A) = \max \{ \mathbf{1}_A{}^\intercal x : x \in B_f \} \tag{14.34}$$

$$= \max \{ \mathbf{1}_A{}^\intercal x : x(B) \leq f(B), \forall B \subseteq E \} \tag{14.35}$$

- Note when considering only $\breve{f} : [0,1]^E \to \mathbb{R}$, then any $w \in [0,1]^E$ is in positive orthant, and we have

$$\breve{f}(w) = \max \{ w^\intercal x : x \in P_f \} \tag{14.36}$$

An extension of an arbitrary $f : 2^V \to \mathbb{R}$

- Thus, for any $f : 2^E \to \mathbb{R}$, even non-submodular f, we can define an extension, having $\breve{f}(\mathbf{1}_A) = f(A)$, $\forall A$, in this way where

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.37}$$

with the $E_i = \{e_1, \dots, e_i\}$'s defined based on sorted descending order of w as in $w(e_1) \geq w(e_2) \geq \dots \geq w(e_m)$, and where

$$\text{for } i \in \{1, \dots, m\}, \quad \lambda_i = \begin{cases} w(e_i) - w(e_{i+1}) & \text{if } i < m \\ w(e_m) & \text{if } i = m \end{cases} \tag{14.38}$$

so that $w = \sum_{i=1}^{m} \lambda_i \mathbf{1}_{E_i}$.

An extension of an arbitrary $f : 2^V \to \mathbb{R}$

- Thus, for any $f : 2^E \to \mathbb{R}$, even non-submodular f, we can define an extension, having $\breve{f}(\mathbf{1}_A) = f(A)$, $\forall A$, in this way where

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.37}$$

with the $E_i = \{e_1, \dots, e_i\}$'s defined based on sorted descending order of w as in $w(e_1) \geq w(e_2) \geq \dots \geq w(e_m)$, and where

$$\text{for } i \in \{1, \dots, m\}, \quad \lambda_i = \begin{cases} w(e_i) - w(e_{i+1}) & \text{if } i < m \\ w(e_m) & \text{if } i = m \end{cases} \tag{14.38}$$

so that $w = \sum_{i=1}^m \lambda_i \mathbf{1}_{E_i}$.
- $w = \sum_{i=1}^m \lambda_i \mathbf{1}_{E_i}$ is an interpolation of certain hypercube vertices.

An extension of an arbitrary $f : 2^V \to \mathbb{R}$

- Thus, for any $f : 2^E \to \mathbb{R}$, even non-submodular f, we can define an extension, having $\breve{f}(\mathbf{1}_A) = f(A)$, $\forall A$, in this way where

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.37}$$

with the $E_i = \{e_1, \ldots, e_i\}$'s defined based on sorted descending order of w as in $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$, and where

$$\text{for } i \in \{1, \ldots, m\}, \quad \lambda_i = \begin{cases} w(e_i) - w(e_{i+1}) & \text{if } i < m \\ w(e_m) & \text{if } i = m \end{cases} \tag{14.38}$$

so that $w = \sum_{i=1}^m \lambda_i \mathbf{1}_{E_i}$.

- $w = \sum_{i=1}^m \lambda_i \mathbf{1}_{E_i}$ is an interpolation of certain hypercube vertices.
- $\breve{f}(w) = \sum_{i=1}^m \lambda_i f(E_i)$ is the associated interpolation of the values of f at sets corresponding to each hypercube vertex.

An extension of an arbitrary $f : 2^V \to \mathbb{R}$

- Thus, for any $f : 2^E \to \mathbb{R}$, even non-submodular f, we can define an extension, having $\breve{f}(\mathbf{1}_A) = f(A)$, $\forall A$, in this way where

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.37}$$

with the $E_i = \{e_1, \dots, e_i\}$'s defined based on sorted descending order of w as in $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$, and where

$$\text{for } i \in \{1, \dots, m\}, \quad \lambda_i = \begin{cases} w(e_i) - w(e_{i+1}) & \text{if } i < m \\ w(e_m) & \text{if } i = m \end{cases} \tag{14.38}$$

so that $w = \sum_{i=1}^m \lambda_i \mathbf{1}_{E_i}$.

- $w = \sum_{i=1}^m \lambda_i \mathbf{1}_{E_i}$ is an interpolation of certain hypercube vertices.
- $\breve{f}(w) = \sum_{i=1}^m \lambda_i f(E_i)$ is the associated interpolation of the values of f at sets corresponding to each hypercube vertex.
- This extension is called the Lovász extension!

Weighted gains vs. weighted functions

- Again sorting E descending in w, the extension summarized:

$$\breve{f}(w) = \sum_{i=1}^{m} w(e_i) f(e_i | E_{i-1}) \tag{14.39}$$

$$= \sum_{i=1}^{m} w(e_i)(f(E_i) - f(E_{i-1})) \tag{14.40}$$

$$= w(e_m) f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1})) f(E_i) \tag{14.41}$$

$$= \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.42}$$

Weighted gains vs. weighted functions

- Again sorting E descending in w, the extension summarized:

$$\breve{f}(w) = \sum_{i=1}^{m} w(e_i) f(e_i | E_{i-1}) \tag{14.39}$$

$$= \sum_{i=1}^{m} w(e_i)(f(E_i) - f(E_{i-1})) \tag{14.40}$$

$$= w(e_m) f(E_m) + \sum_{i=1}^{m-1} (w(e_i) - w(e_{i+1})) f(E_i) \tag{14.41}$$

$$= \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.42}$$

- So $\breve{f}(w)$ seen either as sum of weighted gain evaluations (Eqn. (14.39)), or as sum of weighted function evaluations (Eqn. (14.42)).

Summary: comparison of the two extension forms

- So if f is submodular, then we can write $\breve{f}(w) = \max(wx : x \in B_f)$ (which is clearly convex) in the form:

$$\breve{f}(w) = \max(wx : x \in B_f) = \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.43}$$

where $w = \sum_{i=1}^{m} \lambda_i \mathbf{1}_{E_i}$ and $E_i = \{e_1, \ldots, e_i\}$ defined based on sorted descending order $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.

Summary: comparison of the two extension forms

- So if f is submodular, then we can write $\breve{f}(w) = \max(wx : x \in B_f)$ (which is clearly convex) in the form:

$$\breve{f}(w) = \max(wx : x \in B_f) = \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.43}$$

where $w = \sum_{i=1}^{m} \lambda_i \mathbf{1}_{E_i}$ and $E_i = \{e_1, \ldots, e_i\}$ defined based on sorted descending order $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.

- On the other hand, for any f (even non-submodular), we can produce an extension $\breve{f}$ having the form

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.44}$$

where $w = \sum_{i=1}^{m} \lambda_i \mathbf{1}_{E_i}$ and $E_i = \{e_1, \ldots, e_i\}$ defined based on sorted descending order $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.

Summary: comparison of the two extension forms

- So if f is submodular, then we can write $\breve{f}(w) = \max(wx : x \in B_f)$ (which is clearly convex) in the form:

$$\breve{f}(w) = \max(wx : x \in B_f) = \sum_{i=1}^{m} \lambda_i f(E_i) \quad (14.43)$$

where $w = \sum_{i=1}^{m} \lambda_i \mathbf{1}_{E_i}$ and $E_i = \{e_1, \dots, e_i\}$ defined based on sorted descending order $w(e_1) \geq w(e_2) \geq \dots \geq w(e_m)$.

- On the other hand, for any f (even non-submodular), we can produce an extension $\breve{f}$ having the form

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) \quad (14.44)$$

where $w = \sum_{i=1}^{m} \lambda_i \mathbf{1}_{E_i}$ and $E_i = \{e_1, \dots, e_i\}$ defined based on sorted descending order $w(e_1) \geq w(e_2) \geq \dots \geq w(e_m)$.

- In both Eq. (14.43) and Eq. (14.44), we have $\breve{f}(\mathbf{1}_A) = f(A)$, $\forall A$, but Eq. (14.44), might not be convex.

Summary: comparison of the two extension forms

- So if f is submodular, then we can write $\breve{f}(w) = \max(wx : x \in B_f)$ (which is clearly convex) in the form:

$$\breve{f}(w) = \max(wx : x \in B_f) = \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.43}$$

  where $w = \sum_{i=1}^{m} \lambda_i \mathbf{1}_{E_i}$ and $E_i = \{e_1, \ldots, e_i\}$ defined based on sorted descending order $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
- On the other hand, for any f (even non-submodular), we can produce an extension $\breve{f}$ having the form

$$\breve{f}(w) = \sum_{i=1}^{m} \lambda_i f(E_i) \tag{14.44}$$

  where $w = \sum_{i=1}^{m} \lambda_i \mathbf{1}_{E_i}$ and $E_i = \{e_1, \ldots, e_i\}$ defined based on sorted descending order $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
- In both Eq. (14.43) and Eq. (14.44), we have $\breve{f}(\mathbf{1}_A) = f(A)$, $\forall A$, but Eq. (14.44), might not be convex.
- Submodularity is sufficient for convexity, but is it necessary?

The Lovász extension of $f : 2^E \to \mathbb{R}$

- Lovász showed that if a function $\breve{f}(w)$ defined as in Eqn. (14.37) is convex, then f must be submodular.

The Lovász extension of $f : 2^E \to \mathbb{R}$

- Lovász showed that if a function $\breve{f}(w)$ defined as in Eqn. (14.37) is convex, then f must be submodular.
- This continuous extension $\breve{f}$ of f, in any case (f being submodular or not), is typically called the Lovász extension of f (but also sometimes called the Choquet integral, or the Lovász-Edmonds extension).

Lovász Extension, Submodularity and Convexity

Theorem 14.5.2

A function $f : 2^E \to \mathbb{R}$ is submodular iff its Lovász extension $\breve{f}$ of f is convex.

Proof.

- We've already seen that if f is submodular, its extension can be written via Eqn.(14.37) due to the greedy algorithm, and therefore is also equivalent to $\breve{f}(w) = \max \{wx : x \in P_f\}$, and thus is convex.

. . .

Lovász Extension, Submodularity and Convexity

Theorem 14.5.2

A function $f : 2^E \to \mathbb{R}$ is submodular iff its Lovász extension $\breve{f}$ of f is convex.

Proof.

- We've already seen that if f is submodular, its extension can be written via Eqn.(14.37) due to the greedy algorithm, and therefore is also equivalent to $\breve{f}(w) = \max\{wx : x \in P_f\}$, and thus is convex.
- Conversely, suppose the Lovász extension $\breve{f}(w) = \sum_i \lambda_i f(E_i)$ of some function $f : 2^E \to \mathbb{R}$ is a convex function.

...

Lovász Extension, Submodularity and Convexity

Theorem 14.5.2

A function $f : 2^E \to \mathbb{R}$ is submodular iff its Lovász extension $\breve{f}$ of f is convex.

Proof.

- We've already seen that if f is submodular, its extension can be written via Eqn.(14.37) due to the greedy algorithm, and therefore is also equivalent to $\breve{f}(w) = \max\{wx : x \in P_f\}$, and thus is convex.
- Conversely, suppose the Lovász extension $\breve{f}(w) = \sum_i \lambda_i f(E_i)$ of some function $f : 2^E \to \mathbb{R}$ is a convex function.
- We note that, based on the extension definition, in particular the definition of the $\{\lambda_i\}_i$, we have that $\breve{f}(\alpha w) = \alpha \breve{f}(w)$ for any $\alpha \in \mathbb{R}_+$. I.e., f is a positively homogeneous convex function.

...

Lovász Extension, Submodularity and Convexity

. . . proof of Thm. 14.5.2 cont.

- Earlier, we saw that $\breve{f}(\mathbf{1}_A) = f(A)$ for all $A \subseteq E$.

. . .

Lovász Extension, Submodularity and Convexity

. . . proof of Thm. 14.5.2 cont.

- Earlier, we saw that $\breve{f}(\mathbf{1}_A) = f(A)$ for all $A \subseteq E$.
- Now, given $A, B \subseteq E$, we will show that

$$\breve{f}(\mathbf{1}_A + \mathbf{1}_B) = \breve{f}(\mathbf{1}_{A \cup B} + \mathbf{1}_{A \cap B}) \tag{14.45}$$

$$= f(A \cup B) + f(A \cap B). \tag{14.46}$$

. . .

Lovász Extension, Submodularity and Convexity

. . . proof of Thm. 14.5.2 cont.

- Earlier, we saw that $\breve{f}(\mathbf{1}_A) = f(A)$ for all $A \subseteq E$.
- Now, given $A, B \subseteq E$, we will show that

$$\breve{f}(\mathbf{1}_A + \mathbf{1}_B) = \breve{f}(\mathbf{1}_{A\cup B} + \mathbf{1}_{A\cap B}) \quad (14.45)$$

$$= f(A \cup B) + f(A \cap B). \quad (14.46)$$

- Let $C = A \cap B$, order E based on decreasing $w = \mathbf{1}_A + \mathbf{1}_B$ so that

$$w = (w(e_1), w(e_2), \dots, w(e_m)) \quad (14.47)$$

$$= (\underbrace{2, 2, \dots, 2}_{i\in C}, \underbrace{1, 1, \dots, 1}_{i\in A\triangle B}, \underbrace{0, 0, \dots, 0}_{i\in E\setminus(A\cup B)}) \quad (14.48)$$

. . .

Lovász Extension, Submodularity and Convexity

. . . proof of Thm. 14.5.2 cont.

- Earlier, we saw that $\breve{f}(\mathbf{1}_A) = f(A)$ for all $A \subseteq E$.
- Now, given $A, B \subseteq E$, we will show that

$$\breve{f}(\mathbf{1}_A + \mathbf{1}_B) = \breve{f}(\mathbf{1}_{A\cup B} + \mathbf{1}_{A\cap B}) \tag{14.45}$$

$$= f(A \cup B) + f(A \cap B). \tag{14.46}$$

- Let $C = A \cap B$, order E based on decreasing $w = \mathbf{1}_A + \mathbf{1}_B$ so that

$$w = (w(e_1), w(e_2), \dots, w(e_m)) \tag{14.47}$$

$$= (\underbrace{2, 2, \dots, 2}_{i\in C}, \underbrace{1, 1, \dots, 1}_{i\in A\triangle B}, \underbrace{0, 0, \dots, 0}_{i\in E\setminus(A\cup B)}) \tag{14.48}$$

- Then, considering $\breve{f}(w) = \sum_i \lambda_i f(E_i)$, we have $\lambda_{|C|} = 1$, $\lambda_{|A\cup B|} = 1$, and $\lambda_i = 0$ for $i \notin \{|C|, |A \cup B|\}$.

. . .

Lovász Extension, Submodularity and Convexity

... proof of Thm. 14.5.2 cont.

- Earlier, we saw that $\breve{f}(\mathbf{1}_A) = f(A)$ for all $A \subseteq E$.
- Now, given $A, B \subseteq E$, we will show that

$$\breve{f}(\mathbf{1}_A + \mathbf{1}_B) = \breve{f}(\mathbf{1}_{A\cup B} + \mathbf{1}_{A\cap B}) \tag{14.45}$$

$$= f(A \cup B) + f(A \cap B). \tag{14.46}$$

- Let $C = A \cap B$, order E based on decreasing $w = \mathbf{1}_A + \mathbf{1}_B$ so that

$$w = (w(e_1), w(e_2), \dots, w(e_m)) \tag{14.47}$$

$$= (\underbrace{2, 2, \dots, 2}_{i\in C}, \underbrace{1, 1, \dots, 1}_{i\in A\triangle B}, \underbrace{0, 0, \dots, 0}_{i\in E\setminus(A\cup B)}) \tag{14.48}$$

- Then, considering $\breve{f}(w) = \sum_i \lambda_i f(E_i)$, we have $\lambda_{|C|} = 1$, $\lambda_{|A\cup B|} = 1$, and $\lambda_i = 0$ for $i \notin \{|C|, |A \cup B|\}$.
- But then $E_{|C|} = A \cap B$ and $E_{|A\cup B|} = A \cup B$. Therefore, $\breve{f}(w) = \breve{f}(\mathbf{1}_A + \mathbf{1}_B) = f(A \cap B) + f(A \cup B)$.

...

Lovász Extension, Submodularity and Convexity

. . . proof of Thm. 14.5.2 cont.

- Also, since $\breve{f}$ is convex (by assumption) and positively homogeneous, we have for any $A, B \subseteq E$,

$$0.5[f(A \cap B) + f(A \cup B)] \tag{14.52}$$

□

Lovász Extension, Submodularity and Convexity

. . . proof of Thm. 14.5.2 cont.

- Also, since $\breve{f}$ is convex (by assumption) and positively homogeneous, we have for any $A, B \subseteq E$,

$$0.5[f(A \cap B) + f(A \cup B)] = 0.5[\breve{f}(\mathbf{1}_A + \mathbf{1}_B)] \tag{14.49}$$

(14.52)

□

Lovász Extension, Submodularity and Convexity

... proof of Thm. 14.5.2 cont.

- Also, since $\breve{f}$ is convex (by assumption) and positively homogeneous, we have for any $A, B \subseteq E$,

$$0.5[f(A \cap B) + f(A \cup B)] = 0.5[\breve{f}(\mathbf{1}_A + \mathbf{1}_B)] \tag{14.49}$$

$$= \breve{f}(0.5\mathbf{1}_A + 0.5\mathbf{1}_B) \tag{14.50}$$

(14.52)

□

Lovász Extension, Submodularity and Convexity

. . . proof of Thm. 14.5.2 cont.

- Also, since $\breve{f}$ is convex (by assumption) and positively homogeneous, we have for any $A, B \subseteq E$,

$$0.5[f(A \cap B) + f(A \cup B)] = 0.5[\breve{f}(\mathbf{1}_A + \mathbf{1}_B)] \quad (14.49)$$

$$= \breve{f}(0.5\mathbf{1}_A + 0.5\mathbf{1}_B) \quad (14.50)$$

$$\leq 0.5\breve{f}(\mathbf{1}_A) + 0.5\breve{f}(\mathbf{1}_B) \quad (14.51)$$

$$(14.52)$$

□

Lovász Extension, Submodularity and Convexity

... proof of Thm. 14.5.2 cont.

- Also, since $\breve{f}$ is convex (by assumption) and positively homogeneous, we have for any $A, B \subseteq E$,

$$0.5[f(A \cap B) + f(A \cup B)] = 0.5[\breve{f}(\mathbf{1}_A + \mathbf{1}_B)] \tag{14.49}$$

$$= \breve{f}(0.5\mathbf{1}_A + 0.5\mathbf{1}_B) \tag{14.50}$$

$$\leq 0.5\breve{f}(\mathbf{1}_A) + 0.5\breve{f}(\mathbf{1}_B) \tag{14.51}$$

$$= 0.5(f(A) + f(B)) \tag{14.52}$$

□

Lovász Extension, Submodularity and Convexity

. . . proof of Thm. 14.5.2 cont.

- Also, since $\breve{f}$ is convex (by assumption) and positively homogeneous, we have for any $A, B \subseteq E$,

$$0.5[f(A \cap B) + f(A \cup B)] = 0.5[\breve{f}(\mathbf{1}_A + \mathbf{1}_B)] \tag{14.49}$$

$$= \breve{f}(0.5\mathbf{1}_A + 0.5\mathbf{1}_B) \tag{14.50}$$

$$\leq 0.5\breve{f}(\mathbf{1}_A) + 0.5\breve{f}(\mathbf{1}_B) \tag{14.51}$$

$$= 0.5(f(A) + f(B)) \tag{14.52}$$

- Thus, we have shown that for any $A, B \subseteq E$,

$$f(A \cup B) + f(A \cap B) \leq f(A) + f(B) \tag{14.53}$$

so f must be submodular.

□

Lovász ext. vs. <u>the</u> concave closure of submodular function

- The above theorem showed that the Lovász extension is convex iff f is submodular.

Lovász ext. vs. the concave closure of submodular function

- The above theorem showed that the Lovász extension is convex iff f is submodular.
- Our next theorem shows that the Lovász extension coincides precisely with the convex closure iff f is submodular.

Lovász ext. vs. the concave closure of submodular function

- The above theorem showed that the Lovász extension is convex iff f is submodular.
- Our next theorem shows that the Lovász extension coincides precisely with the convex closure iff f is submodular.
- I.e., not only is the Lovász extension convex for f submodular, it is the convex closure when f is convex.

Lovász ext. vs. the concave closure of submodular function

- The above theorem showed that the Lovász extension is convex iff f is submodular.
- Our next theorem shows that the Lovász extension coincides precisely with the convex closure iff f is submodular.
- I.e., not only is the Lovász extension convex for f submodular, it is the convex closure when f is convex.
- Hence, convex closure is easy to evaluate when f is submodular and is this particular form iff f is submodular.

Lovász ext. vs. the concave closure of submodular function

Theorem 14.5.3

Let $\breve{f}(w) = \max(wx : x \in B_f) = \sum_{i=1}^{m} \lambda_i f(E_i)$ *be the Lovász extension and* $\check{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$ *be the convex closure. Then* $\breve{f}$ *and* $\check{f}$ *coincide iff* f *is submodular.*

Proof.

- Assume f is submodular.

...

Lovász ext. vs. the concave closure of submodular function

Theorem 14.5.3

Let $\breve{f}(w) = \max(wx : x \in B_f) = \sum_{i=1}^{m} \lambda_i f(E_i)$ *be the Lovász extension and* $\breve{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$ *be the convex closure. Then* $\breve{f}$ *and* $\breve{f}$ *coincide iff* f *is submodular.*

Proof.

- Assume f is submodular.
- Given x, let p^x be an achieving argmin in $\breve{f}(x)$ that also maximizes $\sum_S p_S^x |S|^2$.

...

Lovász ext. vs. the concave closure of submodular function

Theorem 14.5.3

Let $\breve{f}(w) = \max(wx : x \in B_f) = \sum_{i=1}^m \lambda_i f(E_i)$ *be the Lovász extension and* $\breve{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$ *be the convex closure. Then* $\breve{f}$ *and* $\breve{f}$ *coincide iff* f *is submodular.*

Proof.

- Assume f is submodular.
- Given x, let p^x be an achieving argmin in $\breve{f}(x)$ that also maximizes $\sum_S p_S^x |S|^2$.
- Suppose $\exists A, B \subseteq V$ that are crossing (i.e., $A \not\subseteq B$, $B \not\subseteq A$) and positive and w.l.o.g., $p_A^x \geq p_B^x > 0$.

. . .

Lovász ext. vs. the concave closure of submodular function

Theorem 14.5.3

Let $\breve{f}(w) = \max(wx : x \in B_f) = \sum_{i=1}^m \lambda_i f(E_i)$ be the Lovász extension and $\breve{f}(x) = \min_{p \in \triangle^n(x)} E_{S \sim p}[f(S)]$ be the convex closure. Then $\breve{f}$ and $\breve{f}$ coincide iff f is submodular.

Proof.

- Assume f is submodular.
- Given x, let p^x be an achieving argmin in $\breve{f}(x)$ that also maximizes $\sum_S p^x_S |S|^2$.
- Suppose $\exists A, B \subseteq V$ that are crossing (i.e., $A \not\subseteq B$, $B \not\subseteq A$) and positive and w.l.o.g., $p^x_A \geq p^x_B > 0$.
- Then we may update p^x as follows:

$$\bar{p}^x_A \leftarrow p^x_A - p^x_B \qquad \bar{p}^x_B \leftarrow p^x_B - p^x_B \tag{14.54}$$

$$\bar{p}^x_{A \cup B} \leftarrow p^x_{A \cup B} + p^x_B \qquad \bar{p}^x_{A \cap B} \leftarrow p^x_{A \cap B} + p^x_B \tag{14.55}$$

and by submodularity, this does not increase $\sum_S p^x_S f(S)$.

...

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- This does increase $\sum_S p^x_S |S|^2$ however since

$$|A \cup B|^2 + |A \cap B|^2 = (|A| + |B \setminus A|)^2 + (|B| - |B \setminus A|)^2 \quad (14.56)$$

$$= |A|^2 + |B|^2 + 2|B \setminus A|(|A| - |B| + |B \setminus A|) \quad (14.57)$$

$$\geq |A|^2 + |B|^2 \quad (14.58)$$

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- This does increase $\sum_S p^x_S |S|^2$ however since

$$|A \cup B|^2 + |A \cap B|^2 = (|A| + |B \setminus A|)^2 + (|B| - |B \setminus A|)^2 \quad (14.56)$$

$$= |A|^2 + |B|^2 + 2|B \setminus A|(|A| - |B| + |B \setminus A|) \quad (14.57)$$

$$\geq |A|^2 + |B|^2 \quad (14.58)$$

- Contradiction! Hence, there can be no crossing sets A, B and we must have, for any A, B with $p^x_A > 0$ and $p^x_B > 0$ either $A \subset B$ or $B \subset A$.

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- This does increase $\sum_S p^x_S |S|^2$ however since

$$|A \cup B|^2 + |A \cap B|^2 = (|A| + |B \setminus A|)^2 + (|B| - |B \setminus A|)^2 \quad (14.56)$$

$$= |A|^2 + |B|^2 + 2|B \setminus A|(|A| - |B| + |B \setminus A|) \quad (14.57)$$

$$\geq |A|^2 + |B|^2 \quad (14.58)$$

- Contradiction! Hence, there can be no crossing sets A, B and we must have, for any A, B with $p^x_A > 0$ and $p^x_B > 0$ either $A \subset B$ or $B \subset A$.
- Hence, the sets $\{A \subseteq V : p^x_A > 0\}$ form a chain and can be as large only as size $n = |V|$.

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- This does increase $\sum_S p^x_S |S|^2$ however since

$$|A \cup B|^2 + |A \cap B|^2 = (|A| + |B \setminus A|)^2 + (|B| - |B \setminus A|)^2 \quad (14.56)$$

$$= |A|^2 + |B|^2 + 2|B \setminus A|(|A| - |B| + |B \setminus A|) \quad (14.57)$$

$$\geq |A|^2 + |B|^2 \quad (14.58)$$

- Contradiction! Hence, there can be no crossing sets A, B and we must have, for any A, B with $p^x_A > 0$ and $p^x_B > 0$ either $A \subset B$ or $B \subset A$.
- Hence, the sets $\{A \subseteq V : p^x_A > 0\}$ form a chain and can be as large only as size $n = |V|$.
- This is the same chain that defines the Lovász extension $\breve{f}(x)$, namely $\emptyset = E_0 \subseteq E_1 \subseteq E_2 \subset \ldots$ where $E_i = \{e_1, e_2, \ldots, e_i\}$ and e_i is orderd so that $x(e_1) \geq x(e_2) \geq \cdots \geq x(e_n)$.

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- Next, assume f is not submodular. We must show that the Lovász extension $\breve{f}(x)$ and the concave closure $\breve{f}(x)$ need not coincide.

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- Next, assume f is not submodular. We must show that the Lovász extension $\breve{f}(x)$ and the concave closure $\check{f}(x)$ need not coincide.
- Since f is not submodular, $\exists S$ and $i, j \notin S$ such that $f(S) + f(S + i + j) > f(S + i) + f(S + j)$, a strict violation of submodularity.

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- Next, assume f is not submodular. We must show that the Lovász extension $\breve{f}(x)$ and the concave closure $\check{f}(x)$ need not coincide.
- Since f is not submodular, $\exists S$ and $i, j \notin S$ such that $f(S) + f(S+i+j) > f(S+i) + f(S+j)$, a strict violation of submodularity.
- Consider $x = \mathbf{1}_S + \frac{1}{2}\mathbf{1}_{\{i,j\}}$.

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- Next, assume f is not submodular. We must show that the Lovász extension $\breve{f}(x)$ and the concave closure $\breve{f}(x)$ need not coincide.
- Since f is not submodular, $\exists S$ and $i, j \notin S$ such that $f(S) + f(S+i+j) > f(S+i) + f(S+j)$, a strict violation of submodularity.
- Consider $x = \mathbf{1}_S + \frac{1}{2}\mathbf{1}_{\{i,j\}}$.
- Then $\breve{f}(x) = \frac{1}{2}f(S) + \frac{1}{2}f(S+i+j)$ and p^x is feasible for $\breve{f}$ with $p^x_S = 1/2$ and $p^x_{S+i+j} = 1/2$.

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- Next, assume f is not submodular. We must show that the Lovász extension $\breve{f}(x)$ and the concave closure $\breve{f}(x)$ need not coincide.
- Since f is not submodular, $\exists S$ and $i, j \notin S$ such that $f(S) + f(S+i+j) > f(S+i) + f(S+j)$, a strict violation of submodularity.
- Consider $x = \mathbf{1}_S + \frac{1}{2}\mathbf{1}_{\{i,j\}}$.
- Then $\breve{f}(x) = \frac{1}{2}f(S) + \frac{1}{2}f(S+i+j)$ and p^x is feasible for $\breve{f}$ with $p^x_S = 1/2$ and $p^x_{S+i+j} = 1/2$.
- An alternate feasible distribution for x in the convex closure is $\bar{p}^x_{S+i} = \bar{p}^x_{S+j} = 1/2$.

Lovász ext. vs. the concave closure of submodular function

. . . proof cont.

- Next, assume f is not submodular. We must show that the Lovász extension $\breve{f}(x)$ and the concave closure $\breve{f}(x)$ need not coincide.
- Since f is not submodular, $\exists S$ and $i, j \notin S$ such that $f(S) + f(S+i+j) > f(S+i) + f(S+j)$, a strict violation of submodularity.
- Consider $x = \mathbf{1}_S + \frac{1}{2}\mathbf{1}_{\{i,j\}}$.
- Then $\breve{f}(x) = \frac{1}{2}f(S) + \frac{1}{2}f(S+i+j)$ and p^x is feasible for $\breve{f}$ with $p^x_S = 1/2$ and $p^x_{S+i+j} = 1/2$.
- An alternate feasible distribution for x in the convex closure is $\bar{p}^x_{S+i} = \bar{p}^x_{S+j} = 1/2$.
- This gives

$$\breve{f}(x) \leq \frac{1}{2}[f(S+i) + f(S+j)] < \breve{f}(x) \tag{14.59}$$

meaning $\breve{f}(x) \neq \breve{f}(x)$.