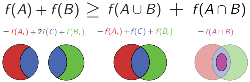
Submodular Functions, Optimization, and Applications to Machine Learning
— Spring Quarter, Lecture 12 —

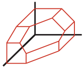
http://www.ee.washington.edu/people/faculty/bilmes/classes/ee563_spring_2018/

Prof. Jeff Bilmes

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

May 7th, 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 → Polymatroids
- L10(4/29): Matroids → Polymatroids, Polymatroids, Polymatroids and Greedy,
- L11(4/30): Polymatroids, Polymatroids and Greedy
- L12(5/2):
- L13(5/7):
- L14(5/9):
- 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.

Vector rank, rank(x), is submodular

- Recall that the matroid rank function is submodular.
- The vector rank function rank(x) also satisfies a form of submodularity, namely one defined on the real lattice.

Theorem 12.2.1 (vector rank and submodularity)

Let P be a polymatroid polytope. The vector rank function rank : $\mathbb{R}_+^E \to \mathbb{R}$ with rank$(x) = \max(y(E) : y \leq x, y \in P)$ satisfies, for all $u, v \in \mathbb{R}_+^E$

$$\text{rank}(u) + \text{rank}(v) \geq \text{rank}(u \vee v) + \text{rank}(u \wedge v) \tag{12.1}$$

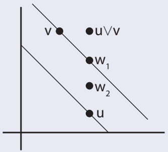
More on polymatroids

Theorem 12.2.1

A polymatroid can equivalently be defined as a pair (E, P) where E is a finite ground set and $P \subseteq R_+^E$ is a compact non-empty set of independent vectors such that

1. *every subvector of an independent vector is independent (if $x \in P$ and $y \leq x$ then $y \in P$, i.e., down closed)*
2. *If $u, v \in P$ (i.e., are independent) and $u(E) < v(E)$, then there exists a vector $w \in P$ such that*

$$u < w \leq u \vee v \qquad (12.20)$$



Corollary 12.2.2

The independent vectors of a polymatroid form a convex polyhedron in $\mathbb{R}_+^E$.

More on polymatroids

For any compact set P, b is a base of P if it is a maximal subvector within P. Recall the bases of matroids. In fact, we can define a polymatroid via vector bases (analogous to how a matroid can be defined via matroid bases).

Theorem 12.2.1

A polymatroid can equivalently be defined as a pair (E, P) where E is a finite ground set and $P \subseteq R_+^E$ is a compact non-empty set of independent vectors such that

1. *every subvector of an independent vector is independent (if $x \in P$ and $y \leq x$ then $y \in P$, i.e., down closed)*
2. *if b, c are bases of P and d is such that $b \wedge c < d < b$, then there exists an f, with $d \wedge c < f \leq c$ such that $d \vee f$ is a base of P*
3. *All of the bases of P have the same rank.*

Note, all three of the above are required for a polymatroid (a matroid analogy would require the equivalent of only the first two).

Matroid instance of Theorem ??

- Considering Theorem **??**, the matroid case is now a special case, where we have that:

Corollary 12.2.2

We have that:

$$\max\{y(E) : y \in P_{\textit{ind. set}}(M), y \leq x\} = \min\{r_M(A) + x(E \setminus A) : A \subseteq E\} \tag{12.21}$$

where r_M is the matroid rank function of some matroid.

Polymatroidal polyhedron and greedy

- Let $(E, \mathcal{I})$ be a set system and $w \in \mathbb{R}_+^E$ be a weight vector.
- Recall greedy algorithm: Set $A = \emptyset$, and repeatedly choose $y \in E \setminus A$ such that $A \cup \{y\} \in \mathcal{I}$ with $w(y)$ as large as possible, stopping when no such y exists.
- For a matroid, we saw that independence system $(E, \mathcal{I})$ is a matroid iff for each weight function $w \in \mathbb{R}_+^E$, the greedy algorithm leads to a set $I \in \mathcal{I}$ of maximum weight $w(I)$.
- Stated succinctly, considering $\max \{w(I) : I \in \mathcal{I}\}$, then $(E, \mathcal{I})$ is a matroid iff greedy works for this maximization.
- Can we also characterize a polymatroid in this way?
- That is, if we consider $\max \left\{ wx : x \in P_f^+ \right\}$, where P_f^+ represents the "independent vectors", is it the case that P_f^+ is a polymatroid iff greedy works for this maximization?
- Can we, ultimately, even relax things so that $w \in \mathbb{R}^E$?

Polymatroidal polyhedron and greedy

- What is the greedy solution in this setting, when $w \in \mathbb{R}^E$?
- Sort elements of E w.r.t. w so that, w.l.o.g. $E = (e_1, e_2, \ldots, e_m)$ with $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
- Let $k+1$ be the first point (if any) at which we are non-positive, i.e., $w(e_k) > 0$ and $0 \geq w(e_{k+1)}$.
- Next define partial accumulated sets E_i, for $i = 0 \ldots m$, we have w.r.t. the above sorted order:

$$E_i \stackrel{\text{def}}{=} \{e_1, e_2, \ldots e_i\} \tag{12.22}$$

(note $E_0 = \emptyset$, $f(E_0) = 0$, and E and E_i is always sorted w.r.t w).

- The greedy solution is the vector $x \in \mathbb{R}_+^E$ with elements defined as:

$$x(e_1) \stackrel{\text{def}}{=} f(E_1) = f(e_1) = f(e_1|E_0) = f(e_1|\emptyset) \tag{12.23}$$

$$x(e_i) \stackrel{\text{def}}{=} f(E_i) - f(E_{i-1}) = f(e_i|E_{i-1}) \text{ for } i = 2 \ldots k \tag{12.24}$$

$$x(e_i) \stackrel{\text{def}}{=} 0 \text{ for } i = k+1 \ldots m = |E| \tag{12.25}$$

Polymatroidal polyhedron and greedy

Theorem 12.2.2

The vector $x \in \mathbb{R}_+^E$ as previously defined using the greedy algorithm maximizes wx over P_f^+, with $w \in \mathbb{R}_+^E$, if f is submodular.

Proof.

- Consider the LP strong duality equation:

$$\max(wx : x \in P_f^+) = \min\Big(\sum_{A \subseteq E} y_A f(A) : y \in \mathbb{R}_+^{2^E}, \sum_{A \subseteq E} y_A \mathbf{1}_A \geq w\Big) \tag{12.21}$$

- Sort E by w descending, and define the following vector $y \in \mathbb{R}_+^{2^E}$ as

$$y_{E_i} \leftarrow w(e_i) - w(e_{i+1}) \text{ for } i = 1 \ldots (m-1), \tag{12.22}$$

$$y_E \leftarrow w(e_m), \text{ and} \tag{12.23}$$

$$y_A \leftarrow 0 \text{ otherwise} \tag{12.24}$$

Polymatroidal polyhedron and greedy

- Thus, restating the above results into a single complete theorem, we have a result very similar to what we saw for matroids (i.e., Theorem 9.4.1)

Theorem 12.2.2

If $f : 2^E \to \mathbb{R}_+$ is given, and P is a polytope in $\mathbb{R}_+^E$ of the form $P = \{x \in \mathbb{R}_+^E : x(A) \leq f(A), \forall A \subseteq E\}$, 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).

Multiple Polytopes associated with arbitrary f

- Given an arbitrary submodular function $f : 2^V \to R$ (not necessarily a polymatroid function, so it need not be positive, monotone, etc.).

Multiple Polytopes associated with arbitrary f

- Given an arbitrary submodular function $f : 2^V \to R$ (not necessarily a polymatroid function, so it need not be positive, monotone, etc.).
- If $f(\emptyset) \neq 0$, can set $f'(A) = f(A) - f(\emptyset)$ without destroying submodularity. This does not change any minima, (i.e., $\text{argmin}_A f(A) = \text{argmin}_{A'} f'(A)$) so assume all functions are normalized $f(\emptyset) = 0$.
  Note that due to constraint $x(\emptyset) \leq f(\emptyset)$, we must have $f(\emptyset) \geq 0$ since if not (i.e., if $f(\emptyset) < 0$), then P_f^+ doesn't exist.
  Another form of normalization can do is:

$$f'(A) = \begin{cases} f(A) & \text{if } A \neq \emptyset \\ 0 & \text{if } A = \emptyset \end{cases} \tag{12.1}$$

  This preserves submodularity due to $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$, and if $A \cap B = \emptyset$ then r.h.s. only gets smaller when $f(\emptyset) \geq 0$.

Multiple Polytopes associated with arbitrary f

- Given an arbitrary submodular function $f : 2^V \to R$ (not necessarily a polymatroid function, so it need not be positive, monotone, etc.).
- If $f(\emptyset) \neq 0$, can set $f'(A) = f(A) - f(\emptyset)$ without destroying submodularity. This does not change any minima, (i.e., $\text{argmin}_A f(A) = \text{argmin}_{A'} f'(A)$) so assume all functions are normalized $f(\emptyset) = 0$.
- We can define several polytopes:

$$P_f = \left\{x \in \mathbb{R}^E : x(S) \leq f(S), \forall S \subseteq E\right\} \tag{12.1}$$

$$P_f^+ = P_f \cap \left\{x \in \mathbb{R}^E : x \geq 0\right\} \tag{12.2}$$

$$B_f = P_f \cap \left\{x \in \mathbb{R}^E : x(E) = f(E)\right\} \tag{12.3}$$

Multiple Polytopes associated with arbitrary f

- Given an arbitrary submodular function $f : 2^V \to R$ (not necessarily a polymatroid function, so it need not be positive, monotone, etc.).
- If $f(\emptyset) \neq 0$, can set $f'(A) = f(A) - f(\emptyset)$ without destroying submodularity. This does not change any minima, (i.e., $\text{argmin}_A f(A) = \text{argmin}_{A'} f'(A)$) so assume all functions are normalized $f(\emptyset) = 0$.
- We can define several polytopes:

$$P_f = \left\{x \in \mathbb{R}^E : x(S) \leq f(S), \forall S \subseteq E\right\} \tag{12.1}$$

$$P_f^+ = P_f \cap \left\{x \in \mathbb{R}^E : x \geq 0\right\} \tag{12.2}$$

$$B_f = P_f \cap \left\{x \in \mathbb{R}^E : x(E) = f(E)\right\} \tag{12.3}$$

- P_f is what is sometimes called the extended polytope (sometimes notated as EP_f.

Multiple Polytopes associated with arbitrary f

- Given an arbitrary submodular function $f : 2^V \to R$ (not necessarily a polymatroid function, so it need not be positive, monotone, etc.).
- If $f(\emptyset) \neq 0$, can set $f'(A) = f(A) - f(\emptyset)$ without destroying submodularity. This does not change any minima, (i.e., $\operatorname{argmin}_A f(A) = \operatorname{argmin}_{A'} f'(A)$) so assume all functions are normalized $f(\emptyset) = 0$.
- We can define several polytopes:

$$P_f = \left\{x \in \mathbb{R}^E : x(S) \leq f(S), \forall S \subseteq E\right\} \tag{12.1}$$

$$P_f^+ = P_f \cap \left\{x \in \mathbb{R}^E : x \geq 0\right\} \tag{12.2}$$

$$B_f = P_f \cap \left\{x \in \mathbb{R}^E : x(E) = f(E)\right\} \tag{12.3}$$

- P_f is what is sometimes called the extended polytope (sometimes notated as EP_f.
- P_f^+ is P_f restricted to the positive orthant.

Multiple Polytopes associated with arbitrary f

- Given an arbitrary submodular function $f : 2^V \to R$ (not necessarily a polymatroid function, so it need not be positive, monotone, etc.).
- If $f(\emptyset) \neq 0$, can set $f'(A) = f(A) - f(\emptyset)$ without destroying submodularity. This does not change any minima, (i.e., $\text{argmin}_A f(A) = \text{argmin}_{A'} f'(A)$) so assume all functions are normalized $f(\emptyset) = 0$.
- We can define several polytopes:

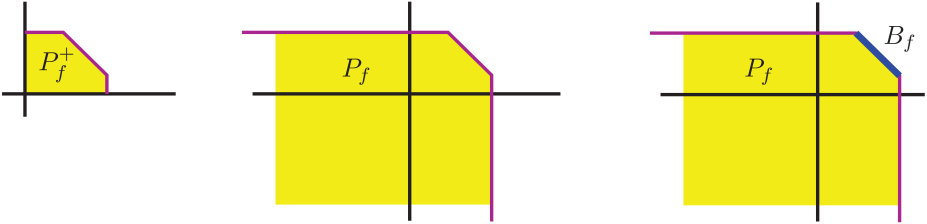
$$P_f = \left\{x \in \mathbb{R}^E : x(S) \leq f(S), \forall S \subseteq E\right\} \tag{12.1}$$

$$P_f^+ = P_f \cap \left\{x \in \mathbb{R}^E : x \geq 0\right\} \tag{12.2}$$

$$B_f = P_f \cap \left\{x \in \mathbb{R}^E : x(E) = f(E)\right\} \tag{12.3}$$

- P_f is what is sometimes called the extended polytope (sometimes notated as EP_f.
- P_f^+ is P_f restricted to the positive orthant.
- B_f is called the base polytope, analogous to the base in matroid.

Multiple Polytopes in 2D associated with f

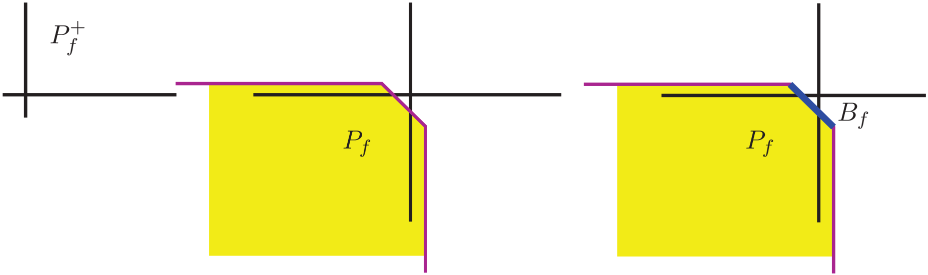


$$P_f^+ = P_f \cap \{x \in \mathbb{R}^E : x \geq 0\} \tag{12.4}$$

$$P_f = \{x \in \mathbb{R}^E : x(S) \leq f(S), \forall S \subseteq E\} \tag{12.5}$$

$$B_f = P_f \cap \{x \in \mathbb{R}^E : x(E) = f(E)\} \tag{12.6}$$

Multiple Polytopes in 2D associated with f

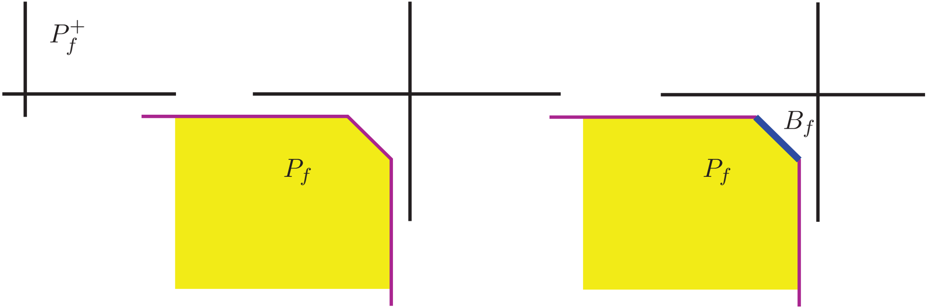


$$P_f^+ = P_f \cap \left\{x \in \mathbb{R}^E : x \geq 0\right\} \tag{12.4}$$

$$P_f = \left\{x \in \mathbb{R}^E : x(S) \leq f(S), \forall S \subseteq E\right\} \tag{12.5}$$

$$B_f = P_f \cap \left\{x \in \mathbb{R}^E : x(E) = f(E)\right\} \tag{12.6}$$

Multiple Polytopes in 2D associated with f

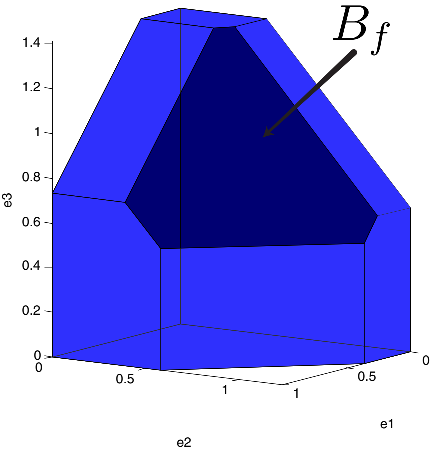


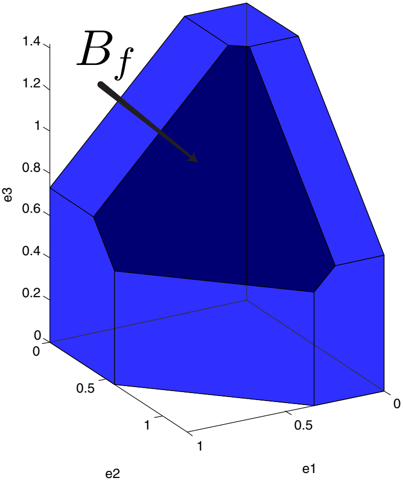
$$P_f^+ = P_f \cap \left\{x \in \mathbb{R}^E : x \geq 0\right\} \tag{12.4}$$

$$P_f = \left\{x \in \mathbb{R}^E : x(S) \leq f(S), \forall S \subseteq E\right\} \tag{12.5}$$

$$B_f = P_f \cap \left\{x \in \mathbb{R}^E : x(E) = f(E)\right\} \tag{12.6}$$

Base Polytope in 3D





$$P_f = \left\{x \in \mathbb{R}^E : x(S) \le f(S), \forall S \subseteq E\right\} \tag{12.7}$$

$$B_f = P_f \cap \left\{x \in \mathbb{R}^E : x(E) = f(E)\right\} \tag{12.8}$$

A polymatroid function's polyhedron is a polymatroid.

Theorem 12.3.1

Let f be a submodular function defined on subsets of E. For any $x \in \mathbb{R}^E$, we have:

$$\textit{rank}(x) = \max\left(y(E) : y \leq x, y \in P_f\right) = \min\left(x(A) + f(E \setminus A) : A \subseteq E\right) \quad (12.9)$$

Essentially the same theorem as Theorem 10.4.1, but note P_f rather than P_f^+. Taking $x = 0$ we get:

Corollary 12.3.2

Let f be a submodular function defined on subsets of E. We have:

$$\textit{rank}(0) = \max\left(y(E) : y \leq 0, y \in P_f\right) = \min\left(f(A) : A \subseteq E\right) \quad (12.10)$$

Proof of Theorem 12.3.1

Proof Thm 12.3.1: $\max\left(y(E) : y \leq x, y \in P_f\right) = \min\left(x(A) + f(E \setminus A) : A \subseteq E\right)$.

- Let y^* be optimal solution of the l.h.s. and let $A \subseteq E$ be any subset.

□

Proof of Theorem 12.3.1

Proof Thm 12.3.1: $\max\left(y(E) : y \leq x, y \in P_f\right) = \min\left(x(A) + f(E \setminus A) : A \subseteq E\right)$.

- Let y^* be optimal solution of the l.h.s. and let $A \subseteq E$ be any subset.
- Then $y^*(E) = y^*(A) + y^*(E \setminus A) \leq f(A) + x(E \setminus A)$ since if $y^* \in P_f$, $y^*(A) \leq f(A)$ and since $y^* \leq x$, $y^*(E \setminus A) \leq x(E \setminus A)$. This is a form of weak duality.

□

Proof of Theorem 12.3.1

Proof Thm 12.3.1: $\max\left(y(E) : y \leq x, y \in P_f\right) = \min\left(x(A) + f(E \setminus A) : A \subseteq E\right)$.

- Let y^* be optimal solution of the l.h.s. and let $A \subseteq E$ be any subset.
- Then $y^*(E) = y^*(A) + y^*(E \setminus A) \leq f(A) + x(E \setminus A)$ since if $y^* \in P_f$, $y^*(A) \leq f(A)$ and since $y^* \leq x$, $y^*(E \setminus A) \leq x(E \setminus A)$. This is a form of weak duality.
- For any $e \in E$, if $y^*(e) < x(e)$, must be some reason other than the constraint $y^* \leq x$, namely it must be that $\exists T \in \mathcal{D}(y^*)$ with $e \in T$ (i.e., e is a member of at least one of the tight sets).

□

Proof of Theorem 12.3.1

Proof Thm 12.3.1: $\max\left(y(E) : y \leq x, y \in P_f\right) = \min\left(x(A) + f(E \setminus A) : A \subseteq E\right)$.

- Let y^* be optimal solution of the l.h.s. and let $A \subseteq E$ be any subset.
- Then $y^*(E) = y^*(A) + y^*(E \setminus A) \leq f(A) + x(E \setminus A)$ since if $y^* \in P_f$, $y^*(A) \leq f(A)$ and since $y^* \leq x$, $y^*(E \setminus A) \leq x(E \setminus A)$. This is a form of weak duality.
- For any $e \in E$, if $y^*(e) < x(e)$, must be some reason other than the constraint $y^* \leq x$, namely it must be that $\exists T \in \mathcal{D}(y^*)$ with $e \in T$ (i.e., e is a member of at least one of the tight sets). I.e., given $e \notin \text{sat}(y^*)$, then $y^*(A) < f(A) \forall A \ni e$ including $\{e\}$, hence $x(e) < f(e)$. Conversely, $e \in \text{sat}(y^*)$ means $y^*(T) = f(T)$ for some $T \in \mathcal{D}(y^*)$.

□

Proof of Theorem 12.3.1

Proof Thm 12.3.1: $\max(y(E) : y \leq x, y \in P_f) = \min(x(A) + f(E \setminus A) : A \subseteq E)$.

- Let y^* be optimal solution of the l.h.s. and let $A \subseteq E$ be any subset.
- Then $y^*(E) = y^*(A) + y^*(E \setminus A) \leq f(A) + x(E \setminus A)$ since if $y^* \in P_f$, $y^*(A) \leq f(A)$ and since $y^* \leq x$, $y^*(E \setminus A) \leq x(E \setminus A)$. This is a form of weak duality.
- For any $e \in E$, if $y^*(e) < x(e)$, must be some reason other than the constraint $y^* \leq x$, namely it must be that $\exists T \in \mathcal{D}(y^*)$ with $e \in T$ (i.e., e is a member of at least one of the tight sets). I.e., given $e \notin \text{sat}(y^*)$, then $y^*(A) < f(A) \forall A \ni e$ including $\{e\}$, hence $x(e) < f(e)$. Conversely, $e \in \text{sat}(y^*)$ means $y^*(T) = f(T)$ for some $T \in \mathcal{D}(y^*)$.
- Hence, for all $e \notin \text{sat}(y^*)$ we have $y^*(e) = x(e)$, and moreover $y^*(\text{sat}(y^*)) = f(\text{sat}(y^*))$ by definition.

□

Proof of Theorem 12.3.1

Proof Thm 12.3.1: $\max\left(y(E) : y \leq x, y \in P_f\right) = \min\left(x(A) + f(E \setminus A) : A \subseteq E\right)$.

- Let y^* be optimal solution of the l.h.s. and let $A \subseteq E$ be any subset.
- Then $y^*(E) = y^*(A) + y^*(E \setminus A) \leq f(A) + x(E \setminus A)$ since if $y^* \in P_f$, $y^*(A) \leq f(A)$ and since $y^* \leq x$, $y^*(E \setminus A) \leq x(E \setminus A)$. This is a form of weak duality.
- For any $e \in E$, if $y^*(e) < x(e)$, must be some reason other than the constraint $y^* \leq x$, namely it must be that $\exists T \in \mathcal{D}(y^*)$ with $e \in T$ (i.e., e is a member of at least one of the tight sets). I.e., given $e \notin \text{sat}(y^*)$, then $y^*(A) < f(A) \forall A \ni e$ including $\{e\}$, hence $x(e) < f(e)$. Conversely, $e \in \text{sat}(y^*)$ means $y^*(T) = f(T)$ for some $T \in \mathcal{D}(y^*)$.
- Hence, for all $e \notin \text{sat}(y^*)$ we have $y^*(e) = x(e)$, and moreover $y^*(\text{sat}(y^*)) = f(\text{sat}(y^*))$ by definition.
- Thus $y^*(\text{sat}(y^*)) + y^*(E \setminus \text{sat}(y^*)) = f(\text{sat}(y^*)) + x(E \setminus \text{sat}(y^*))$, strong duality, showing that the two sides are equal for y^*.

∎

Greedy and P_f

- In Theorem 11.4.1 (i.e., greedy solution in P_f^+), we can relax P_f^+ to P_f (prime and dual feasibiity still hold as does strong duality).

Greedy and P_f

- In Theorem 11.4.1 (i.e., greedy solution in P_f^+), we can relax P_f^+ to P_f (prime and dual feasibiity still hold as does strong duality).
- The proof, that is, shows that $x \in P_f$, not just P_f^+.

Greedy and P_f

- In Theorem 11.4.1 (i.e., greedy solution in P_f^+), we can relax P_f^+ to P_f (prime and dual feasibiity still hold as does strong duality).
- The proof, that is, shows that $x \in P_f$, not just P_f^+.
- If $\exists e$ such that $w(e) < 0$ then $\max(wx : x \in P_f) = \infty$ since we can let $x_e \to \infty$, unless we ignore the negative elements or assume $w \geq 0$.

Greedy and P_f

- In Theorem 11.4.1 (i.e., greedy solution in P_f^+), we can relax P_f^+ to P_f (prime and dual feasibiity still hold as does strong duality).
- The proof, that is, shows that $x \in P_f$, not just P_f^+.
- If $\exists e$ such that $w(e) < 0$ then $\max(wx : x \in P_f) = \infty$ since we can let $x_e \to \infty$, unless we ignore the negative elements or assume $w \geq 0$.
- Moreover, in either P_f, or P_f^+ case, since the greedy constructed an x has $x(E) = f(E)$, we have that the greedy $x \in B_f$.

Greedy and P_f

- In Theorem 11.4.1 (i.e., greedy solution in P_f^+), we can relax P_f^+ to P_f (prime and dual feasibiity still hold as does strong duality).
- The proof, that is, shows that $x \in P_f$, not just P_f^+.
- If $\exists e$ such that $w(e) < 0$ then $\max(wx : x \in P_f) = \infty$ since we can let $x_e \to \infty$, unless we ignore the negative elements or assume $w \geq 0$.
- Moreover, in either P_f, or P_f^+ case, since the greedy constructed an x has $x(E) = f(E)$, we have that the greedy $x \in B_f$.
- In fact, we will see, in the next section, that the greedy x is a vertex of B_f.

Greedy and P_f

- Recall that Theorem 10.4.1 states that $\max\left(y(E) : y \leq x, y \in P_f^+\right) = \min\left(x(A) + f(E \setminus A) : A \subseteq E\right)$

Greedy and P_f

- Recall that Theorem 10.4.1 states that $\max\left(y(E) : y \leq x, y \in P_f^+\right) = \min\left(x(A) + f(E \setminus A) : A \subseteq E\right)$
- Theorem 11.4.1 states that greedy algorithm maximizes wx over P_f^+ for $w \in \mathbb{R}_+^E$ with f being submodular.

Greedy and P_f

- Recall that Theorem 10.4.1 states that
$\max\left(y(E) : y \leq x, y \in P_f^+\right) = \min\left(x(A) + f(E \setminus A) : A \subseteq E\right)$
- Theorem 11.4.1 states that greedy algorithm maximizes wx over P_f^+ for $w \in \mathbb{R}_+^E$ with f being submodular.
- Above implies that Theorem 11.4.1 can be generalized to over P_f and that greedy solution gives a point in B_f, even for arbitrary finite w.

Polymatroid extreme points

- The greedy algorithm does more than solve $\max(wx : x \in P_f^+)$. We can use it to generate vertices of polymatroidal polytopes.

Polymatroid extreme points

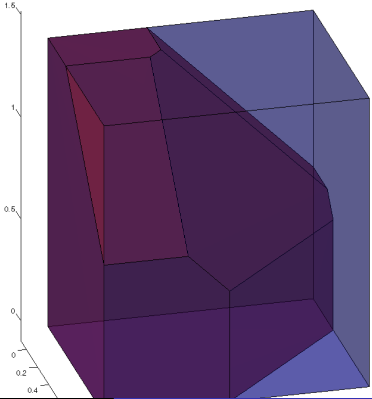
- The greedy algorithm does more than solve $\max(wx : x \in P_f^+)$. We can use it to generate vertices of polymatroidal polytopes.
- Consider P_f^+ and also $C_f^+ \stackrel{\text{def}}{=} \{x : x \in \mathbb{R}_+^E, x(e) \leq f(e), \forall e \in E\}$

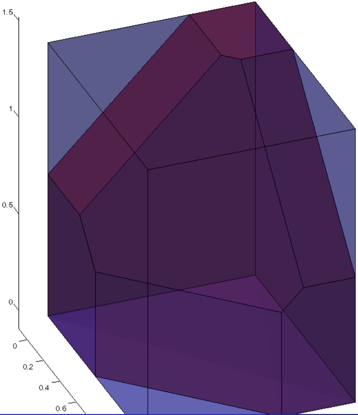
Polymatroid extreme points

- The greedy algorithm does more than solve $\max(wx : x \in P_f^+)$. We can use it to generate vertices of polymatroidal polytopes.
- Consider P_f^+ and also $C_f^+ \stackrel{\text{def}}{=} \{x : x \in \mathbb{R}_+^E, x(e) \leq f(e), \forall e \in E\}$
- Then ordering $A = (a_1, \ldots, a_{|A|})$ arbitrarily with $A_i = \{a_1, \ldots, a_i\}$, $f(A) = \sum_i f(a_i | A_{i-1}) \leq \sum_i f(a_i)$, and hence $P_f^+ \subseteq C_f^+$.

Polymatroid extreme points

- The greedy algorithm does more than solve $\max(wx : x \in P_f^+)$. We can use it to generate vertices of polymatroidal polytopes.
- Consider P_f^+ and also $C_f^+ \stackrel{\text{def}}{=} \{x : x \in \mathbb{R}_+^E, x(e) \leq f(e), \forall e \in E\}$
- Then ordering $A = (a_1, \dots, a_{|A|})$ arbitrarily with $A_i = \{a_1, \dots, a_i\}$, $f(A) = \sum_i f(a_i|A_{i-1}) \leq \sum_i f(a_i)$, and hence $P_f^+ \subseteq C_f^+$.





Polymatroid extreme points

- Since $w \in \mathbb{R}_+^E$ is arbitrary, it may be that any $e \in E$ is max (i.e., is such that $w(e) > w(e')$ for $e' \in E \setminus \{e\}$).

Polymatroid extreme points

- Since $w \in \mathbb{R}_+^E$ is arbitrary, it may be that any $e \in E$ is max (i.e., is such that $w(e) > w(e')$ for $e' \in E \setminus \{e\}$).
- Thus, intuitively, any first vertex of the polytope away from the origin might be obtained by advancing along the corresponding axis.

Polymatroid extreme points

- Since $w \in \mathbb{R}_+^E$ is arbitrary, it may be that any $e \in E$ is max (i.e., is such that $w(e) > w(e')$ for $e' \in E \setminus \{e\}$).
- Thus, intuitively, any first vertex of the polytope away from the origin might be obtained by advancing along the corresponding axis.
- Recall, base polytope defined as the extreme face of P_f. I.e.,

$$B_f = P_f \cap \left\{x \in \mathbb{R}_+^E : x(E) = f(E)\right\} \tag{12.11}$$

Polymatroid extreme points

- Since $w \in \mathbb{R}_+^E$ is arbitrary, it may be that any $e \in E$ is max (i.e., is such that $w(e) > w(e')$ for $e' \in E \setminus \{e\}$).
- Thus, intuitively, any first vertex of the polytope away from the origin might be obtained by advancing along the corresponding axis.
- Recall, base polytope defined as the extreme face of P_f. I.e.,

$$B_f = P_f \cap \left\{x \in \mathbb{R}_+^E : x(E) = f(E)\right\} \tag{12.11}$$

Polymatroid extreme points

- Since $w \in \mathbb{R}_+^E$ is arbitrary, it may be that any $e \in E$ is max (i.e., is such that $w(e) > w(e')$ for $e' \in E \setminus \{e\}$).
- Thus, intuitively, any first vertex of the polytope away from the origin might be obtained by advancing along the corresponding axis.
- Recall, base polytope defined as the extreme face of P_f. I.e.,

$$B_f = P_f \cap \left\{x \in \mathbb{R}_+^E : x(E) = f(E)\right\} \tag{12.11}$$

- Also, intuitively, we can continue advancing along the skeletal edges of the polytope to reach any other vertex, given the appropriate ordering. If we advance in all dimensions, we'll reach a vertex in B_f, and if we advance only in some dimensions, we'll reach a vertex in $P_f \cap \left\{x \in \mathbb{R}_+^E : x(A) = 0 \text{ for some } A\right\}$.

Polymatroid extreme points

- Since $w \in \mathbb{R}_+^E$ is arbitrary, it may be that any $e \in E$ is max (i.e., is such that $w(e) > w(e')$ for $e' \in E \setminus \{e\}$).
- Thus, intuitively, any first vertex of the polytope away from the origin might be obtained by advancing along the corresponding axis.
- Recall, base polytope defined as the extreme face of P_f. I.e.,

$$B_f = P_f \cap \left\{x \in \mathbb{R}_+^E : x(E) = f(E)\right\} \tag{12.11}$$

- Also, intuitively, we can continue advancing along the skeletal edges of the polytope to reach any other vertex, given the appropriate ordering. If we advance in all dimensions, we'll reach a vertex in B_f, and if we advance only in some dimensions, we'll reach a vertex in $P_f \cap \left\{x \in \mathbb{R}_+^E : x(A) = 0 \text{ for some } A\right\}$.
- We formalize this next:

Polymatroid extreme points

- Given any arbitrary order of $E = (e_1, e_2, \ldots, e_m)$, define $E_i = (e_1, e_2, \ldots, e_i)$.

Polymatroid extreme points

- Given any arbitrary order of $E = (e_1, e_2, \ldots, e_m)$, define $E_i = (e_1, e_2, \ldots, e_i)$.
- As before, a vector x is generated by E_i using the greedy procedure as follows

$$x(e_1) = f(E_1) = f(e_1) \tag{12.12}$$

$$x(e_j) = f(E_j) - f(E_{j-1}) = f(e_j|E_{j-1}) \text{ for } 2 \leq j \leq i \tag{12.13}$$

Polymatroid extreme points

- Given any arbitrary order of $E = (e_1, e_2, \dots, e_m)$, define $E_i = (e_1, e_2, \dots, e_i)$.
- As before, a vector x is generated by E_i using the greedy procedure as follows

$$x(e_1) = f(E_1) = f(e_1) \tag{12.12}$$

$$x(e_j) = f(E_j) - f(E_{j-1}) = f(e_j|E_{j-1}) \text{ for } 2 \leq j \leq i \tag{12.13}$$

- An extreme point of P_f is a point that is not a convex combination of two other distinct points in P_f. Equivalently, an extreme point corresponds to setting certain inequalities in the specification of P_f to be equalities, so that there is a unique single point solution.

Polymatroid extreme points

Theorem 12.4.1

For a given ordering $E = (e_1, \dots, e_m)$ *of* E *and a given* $E_i = (e_1, \dots, e_i)$ *and* x *generated by* E_i *using the greedy procedure* ($x(e_i) = f(e_i | E_{i-1})$)*, then* x *is an extreme point of* P_f *when* f *is submodular.*

Polymatroid extreme points

Theorem 12.4.1

For a given ordering $E = (e_1, \ldots, e_m)$ of E and a given $E_i = (e_1, \ldots, e_i)$ and x generated by E_i using the greedy procedure ($x(e_i) = f(e_i|E_{i-1})$), then x is an extreme point of P_f when f is submodular.

Proof.

- We already saw that $x \in P_f$ (Theorem 11.4.1).

Polymatroid extreme points

Theorem 12.4.1

For a given ordering $E = (e_1, \ldots, e_m)$ of E and a given $E_i = (e_1, \ldots, e_i)$ and x generated by E_i using the greedy procedure ($x(e_i) = f(e_i|E_{i-1})$), then x is an extreme point of P_f when f is submodular.

Proof.

- We already saw that $x \in P_f$ (Theorem 11.4.1).
- To show that x is an extreme point of P_f, note that it is the unique solution of the following system of equations

$$x(E_j) = f(E_j) \text{ for } 1 \leq j \leq i \leq m \tag{12.14}$$

$$x(e) = 0 \text{ for } e \in E \setminus E_i \tag{12.15}$$

There are $i \leq m$ equations and $i \leq m$ unknowns, and simple Gaussian elimination gives us back the x constructed via the Greedy algorithm!!

Polymatroid extreme points

- As an example, we have $x(E_1) = x(e_1) = f(e_1)$

Polymatroid extreme points

- As an example, we have $x(E_1) = x(e_1) = f(e_1)$
- $x(E_2) = x(e_1) + x(e_2) = f(e_1, e_2)$ so $x(e_2) = f(e_1, e_2) - x(e_1) = f(e_1, e_2) - f(e_1) = f(e_2|e_1)$.

Polymatroid extreme points

- As an example, we have $x(E_1) = x(e_1) = f(e_1)$
- $x(E_2) = x(e_1) + x(e_2) = f(e_1, e_2)$ so $x(e_2) = f(e_1, e_2) - x(e_1) = f(e_1, e_2) - f(e_1) = f(e_2|e_1)$.
- $x(E_3) = x(e_1) + x(e_2) + x(e_3) = f(e_1, e_2, e_3)$ so $x(e_3) = f(e_1, e_2, e_3) - x(e_2) - x(e_1) = f(e_1, e_2, e_3) - f(e_1, e_2) = f(e_3|e_1, e_2)$

Polymatroid extreme points

- As an example, we have $x(E_1) = x(e_1) = f(e_1)$
- $x(E_2) = x(e_1) + x(e_2) = f(e_1, e_2)$ so $x(e_2) = f(e_1, e_2) - x(e_1) = f(e_1, e_2) - f(e_1) = f(e_2|e_1)$.
- $x(E_3) = x(e_1) + x(e_2) + x(e_3) = f(e_1, e_2, e_3)$ so $x(e_3) = f(e_1, e_2, e_3) - x(e_2) - x(e_1) = f(e_1, e_2, e_3) - f(e_1, e_2) = f(e_3|e_1, e_2)$
- And so on . . . , but we see that this is just Gaussian elimination.

Polymatroid extreme points

- As an example, we have $x(E_1) = x(e_1) = f(e_1)$
- $x(E_2) = x(e_1) + x(e_2) = f(e_1, e_2)$ so $x(e_2) = f(e_1, e_2) - x(e_1) = f(e_1, e_2) - f(e_1) = f(e_2|e_1)$.
- $x(E_3) = x(e_1) + x(e_2) + x(e_3) = f(e_1, e_2, e_3)$ so $x(e_3) = f(e_1, e_2, e_3) - x(e_2) - x(e_1) = f(e_1, e_2, e_3) - f(e_1, e_2) = f(e_3|e_1, e_2)$
- And so on . . . , but we see that this is just Gaussian elimination.
- Also, since $x \in P_f$, for each i, we see that,

$$x(E_j) = f(E_j) \quad \text{for } 1 \leq j \leq i \tag{12.16}$$

$$x(A) \leq f(A), \forall A \subseteq E \tag{12.17}$$

Polymatroid extreme points

- As an example, we have $x(E_1) = x(e_1) = f(e_1)$
- $x(E_2) = x(e_1) + x(e_2) = f(e_1, e_2)$ so $x(e_2) = f(e_1, e_2) - x(e_1) = f(e_1, e_2) - f(e_1) = f(e_2|e_1)$.
- $x(E_3) = x(e_1) + x(e_2) + x(e_3) = f(e_1, e_2, e_3)$ so $x(e_3) = f(e_1, e_2, e_3) - x(e_2) - x(e_1) = f(e_1, e_2, e_3) - f(e_1, e_2) = f(e_3|e_1, e_2)$
- And so on . . . , but we see that this is just Gaussian elimination.
- Also, since $x \in P_f$, for each i, we see that,

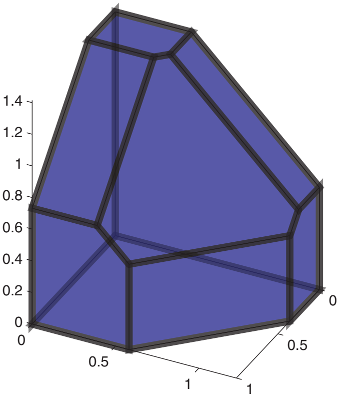
$$x(E_j) = f(E_j) \quad \text{for } 1 \leq j \leq i \tag{12.16}$$

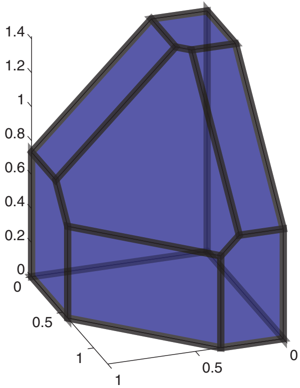
$$x(A) \leq f(A), \forall A \subseteq E \tag{12.17}$$

- Thus, the greedy procedure provides a modular function lower bound on f that is tight on all points E_i in the order. This can be useful in its own right, as it provides subgradients and subdifferential structure.

Polymatroid extreme points

some examples





Polymatroid extreme points

- Moreover, we have (and will ultimately prove)

Corollary 12.4.2

If x is an extreme point of P_f and $B \subseteq E$ is given such that $\text{supp}(x) = \{e \in E : x(e) \neq 0\} \subseteq B \subseteq \cup(A : x(A) = f(A)) = \text{sat}(x)$, then x is generated using greedy by some ordering of B.

Polymatroid extreme points

- Moreover, we have (and will ultimately prove)

Corollary 12.4.2

If x is an extreme point of P_f and $B \subseteq E$ is given such that $\text{supp}(x) = \{e \in E : x(e) \neq 0\} \subseteq B \subseteq \cup(A : x(A) = f(A)) = \text{sat}(x)$, then x is generated using greedy by some ordering of B.

- Note, $\text{sat}(x) = \text{cl}(x) = \cup(A : x(A) = f(A))$ is also called the closure of x (recall that sets A such that $x(A) = f(A)$ are called tight, and such sets are closed under union and intersection, as seen in Lecture 10, Theorem 10.4.3)

Polymatroid extreme points

- Moreover, we have (and will ultimately prove)

Corollary 12.4.2

If x is an extreme point of P_f and $B \subseteq E$ is given such that $\text{supp}(x) = \{e \in E : x(e) \neq 0\} \subseteq B \subseteq \cup(A : x(A) = f(A)) = \text{sat}(x)$, then x is generated using greedy by some ordering of B.

- Note, $\text{sat}(x) = \text{cl}(x) = \cup(A : x(A) = f(A))$ is also called the closure of x (recall that sets A such that $x(A) = f(A)$ are called tight, and such sets are closed under union and intersection, as seen in Lecture 10, Theorem 10.4.3)
- Thus, $\text{cl}(x)$ is a tight set.

Polymatroid extreme points

- Moreover, we have (and will ultimately prove)

Corollary 12.4.2

If x is an extreme point of P_f and $B \subseteq E$ is given such that $\text{supp}(x) = \{e \in E : x(e) \neq 0\} \subseteq B \subseteq \cup(A : x(A) = f(A)) = \text{sat}(x)$, then x is generated using greedy by some ordering of B.

- Note, $\text{sat}(x) = \text{cl}(x) = \cup(A : x(A) = f(A))$ is also called the closure of x (recall that sets A such that $x(A) = f(A)$ are called tight, and such sets are closed under union and intersection, as seen in Lecture 10, Theorem 10.4.3)
- Thus, $\text{cl}(x)$ is a tight set.
- Also, $\text{supp}(x) = \{e \in E : x(e) \neq 0\}$ is called the support of x.

Polymatroid extreme points

- Moreover, we have (and will ultimately prove)

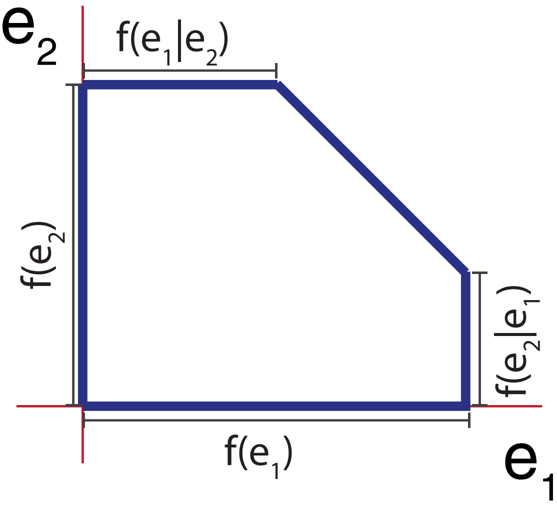
Corollary 12.4.2

If x is an extreme point of P_f and $B \subseteq E$ is given such that $\mathrm{supp}(x) = \{e \in E : x(e) \neq 0\} \subseteq B \subseteq \cup(A : x(A) = f(A)) = \mathrm{sat}(x)$, then x is generated using greedy by some ordering of B.

- Note, $\mathrm{sat}(x) = \mathrm{cl}(x) = \cup(A : x(A) = f(A))$ is also called the closure of x (recall that sets A such that $x(A) = f(A)$ are called tight, and such sets are closed under union and intersection, as seen in Lecture 10, Theorem 10.4.3)
- Thus, $\mathrm{cl}(x)$ is a tight set.
- Also, $\mathrm{supp}(x) = \{e \in E : x(e) \neq 0\}$ is called the support of x.
- For arbitrary x, $\mathrm{supp}(x)$ is not necessarily tight, but for an extreme point, $\mathrm{supp}(x)$ is.

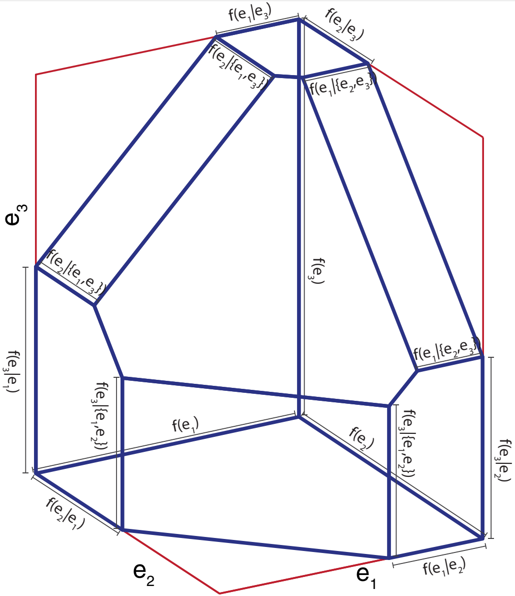
Polymatroid with labeled edge lengths

- 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.



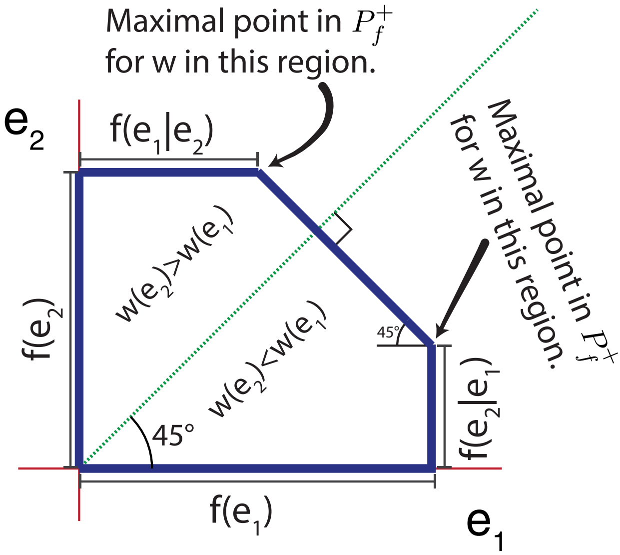
Polymatroid with labeled edge lengths

- 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.



Intuition: why greedy works with polymatroids

- Given w, the goal is to find $x = (x(e_1), x(e_2))$ that maximizes $x^\intercal w = x(e_1)w(e_1) + x(e_2)w(e_2)$.
- If $w(e_2) > w(e_1)$ the upper extreme point indicated maximizes $x^\intercal w$ over $x \in P_f^+$.
- If $w(e_2) < w(e_1)$ the lower extreme point indicated maximizes $x^\intercal w$ over $x \in P_f^+$.



Maximization of Submodular Functions

- Submodular maximization is quite useful.

Maximization of Submodular Functions

- Submodular maximization is quite useful.
- Applications: sensor placement, facility location, document summarization, or any kind of covering problem (choose a small set of elements that cover some domain as much as possible).

Maximization of Submodular Functions

- Submodular maximization is quite useful.
- Applications: sensor placement, facility location, document summarization, or any kind of covering problem (choose a small set of elements that cover some domain as much as possible).
- For polymatroid function (or any monotone non-decreasing function), unconstrained maximization is trivial (take ground set).

Maximization of Submodular Functions

- Submodular maximization is quite useful.
- Applications: sensor placement, facility location, document summarization, or any kind of covering problem (choose a small set of elements that cover some domain as much as possible).
- For polymatroid function (or any monotone non-decreasing function), unconstrained maximization is trivial (take ground set).
- Thus, when we do monotone submodular maximization we find the maximum under some constraint.

Maximization of Submodular Functions

- Submodular maximization is quite useful.
- Applications: sensor placement, facility location, document summarization, or any kind of covering problem (choose a small set of elements that cover some domain as much as possible).
- For polymatroid function (or any monotone non-decreasing function), unconstrained maximization is trivial (take ground set).
- Thus, when we do monotone submodular maximization we find the maximum under some constraint.
- There is also a sort of dual problem that is often considered together with max, and those are minimum cover problems (to be defined).

The Set Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.

The Set Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \ldots, m\}$ be the set of integers.

The Set Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \ldots, m\}$ be the set of integers.
- Define $f : 2^V \to \mathbb{Z}_+$ as $f(X) = |\bigcup_{v \in X} E_v|$

The Set Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \ldots, m\}$ be the set of integers.
- Define $f : 2^V \to \mathbb{Z}_+$ as $f(X) = |\bigcup_{v \in X} E_v|$
- Then f is the set cover function. As we say, f is monotone submodular (a polymatroid).

The Set Cover Problem

- Let E be a ground set and let $E_1, E_2, \dots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \dots, m\}$ be the set of integers.
- Define $f : 2^V \to \mathbb{Z}_+$ as $f(X) = |\bigcup_{v \in X} E_v|$
- Then f is the set cover function. As we say, f is monotone submodular (a polymatroid).
- The set cover problem asks for the smallest subset X of V such that $f(X) = |E|$ (smallest subset of the subsets of E) where E is still covered. I.e.,

$$\text{minimize} |X| \text{ subject to } f(X) \geq |E| \tag{12.18}$$

The Set Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \ldots, m\}$ be the set of integers.
- Define $f : 2^V \to \mathbb{Z}_+$ as $f(X) = |\bigcup_{v \in X} E_v|$
- Then f is the set cover function. As we say, f is monotone submodular (a polymatroid).
- The set cover problem asks for the smallest subset X of V such that $f(X) = |E|$ (smallest subset of the subsets of E) where E is still covered. I.e.,

$$\text{minimize}|X| \text{ subject to } f(X) \geq |E| \tag{12.18}$$

- We might wish to use a more general modular function $m(X)$ rather than cardinality $|X|$.

The Set Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \ldots, m\}$ be the set of integers.
- Define $f : 2^V \to \mathbb{Z}_+$ as $f(X) = |\bigcup_{v \in X} E_v|$
- Then f is the set cover function. As we say, f is monotone submodular (a polymatroid).
- The set cover problem asks for the smallest subset X of V such that $f(X) = |E|$ (smallest subset of the subsets of E) where E is still covered. I.e.,

$$\text{minimize} |X| \text{ subject to } f(X) \geq |E| \tag{12.18}$$

- We might wish to use a more general modular function $m(X)$ rather than cardinality $|X|$.
- This problem is NP-hard, and Feige in 1998 showed that it cannot be approximated with a ratio better than $(1 - \epsilon) \log n$ unless NP is slightly superpolynomial ($n^{O(\log \log n)}$).

What About Non-monotone

- So even simple case of cardinality constrained submodular function maximization is NP-hard.
- This will be true of most submodular max (and related) problems.
- Hence, the only hope is approximation algorithms. Question is, what is the tradeoff between running time and approximation quality, and is it possible to get tight bounds (i.e., an algorithm that achieves an approximation ratio, and a proof that one can't do better than that unless some extremely unlike event were to be true, such as P=NP).

The Max k-Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.

The Max k-Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \ldots, m\}$ be the set of integers.

The Max k-Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \ldots, m\}$ be the set of integers.
- Define $f : 2^V \to \mathbb{Z}_+$ as $f(X) = |\bigcup_{v \in V} E_v|$

The Max k-Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \ldots, m\}$ be the set of integers.
- Define $f : 2^V \to \mathbb{Z}_+$ as $f(X) = |\bigcup_{v \in V} E_v|$
- Then f is the set cover function. As we saw, f is monotone submodular (a polymatroid).

The Max k-Cover Problem

- Let E be a ground set and let $E_1, E_2, \ldots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \ldots, m\}$ be the set of integers.
- Define $f : 2^V \to \mathbb{Z}_+$ as $f(X) = |\bigcup_{v \in V} E_v|$
- Then f is the set cover function. As we saw, f is monotone submodular (a polymatroid).
- The max k cover problem asks, given a k, what sized k set of sets X can we choose that covers the most? I.e., that maximizes $f(X)$ as in:

$$\max f(X) \text{ subject to } |X| \leq k \tag{12.19}$$

The Max k-Cover Problem

- Let E be a ground set and let $E_1, E_2, \dots, E_m$ be a set of subsets.
- Let $V = \{1, 2, \dots, m\}$ be the set of integers.
- Define $f : 2^V \to \mathbb{Z}_+$ as $f(X) = |\bigcup_{v \in V} E_v|$
- Then f is the set cover function. As we saw, f is monotone submodular (a polymatroid).
- The max k cover problem asks, given a k, what sized k set of sets X can we choose that covers the most? I.e., that maximizes $f(X)$ as in:

$$\max f(X) \text{ subject to } |X| \leq k \tag{12.19}$$

- This problem is NP-hard, and Feige in 1998 showed that it cannot be approximated with a ratio better than $(1 - 1/e)$.

Cardinality Constrained Max. of Polymatroid Functions

- Now we are given an arbitrary polymatroid function f.

Cardinality Constrained Max. of Polymatroid Functions

- Now we are given an arbitrary polymatroid function f.
- Given k, goal is: find $A^* \in \operatorname{argmax}\{f(A) : |A| \leq k\}$

Cardinality Constrained Max. of Polymatroid Functions

- Now we are given an arbitrary polymatroid function f.
- Given k, goal is: find $A^* \in \operatorname{argmax} \{f(A) : |A| \leq k\}$
- w.l.o.g., we can find $A^* \in \operatorname{argmax} \{f(A) : |A| = k\}$

Cardinality Constrained Max. of Polymatroid Functions

- Now we are given an arbitrary polymatroid function f.
- Given k, goal is: find $A^* \in \operatorname{argmax} \{f(A) : |A| \leq k\}$
- w.l.o.g., we can find $A^* \in \operatorname{argmax} \{f(A) : |A| = k\}$
- An important result by Nemhauser et. al. (1978) states that for normalized ($f(\emptyset) = 0$) monotone submodular functions (i.e., polymatroids) can be approximately maximized using a simple greedy algorithm.

Cardinality Constrained Max. of Polymatroid Functions

- Now we are given an arbitrary polymatroid function f.
- Given k, goal is: find $A^* \in \operatorname{argmax} \{f(A) : |A| \leq k\}$
- w.l.o.g., we can find $A^* \in \operatorname{argmax} \{f(A) : |A| = k\}$
- An important result by Nemhauser et. al. (1978) states that for normalized ($f(\emptyset) = 0$) monotone submodular functions (i.e., polymatroids) can be approximately maximized using a simple greedy algorithm.
- Starting with $S_0 = \emptyset$, we repeat the following greedy step for $i = 0 \ldots (k-1)$:

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i} f(S_i \cup \{v\}) \right\} \tag{12.20}$$

The Greedy Algorithm for Submodular Max

A bit more precisely:

Algorithm 1: The Greedy Algorithm

```
Set S_0 ← ∅ ;
for i ← 0 ... |E| − 1 do
    Choose v_i as follows:
      v_i ∈ argmax_{v∈V\S_i} f({v}|S_i) = argmax_{v∈V\S_i} f(S_i ∪ {v}) ;
    Set S_{i+1} ← S_i ∪ {v_i} ;
```

Set $S_0 \leftarrow \emptyset$;

for $i \leftarrow 0 \ldots |E| - 1$ **do**

Choose v_i as follows:

$$v_i \in \operatorname{argmax}_{v \in V \setminus S_i} f(\{v\}|S_i) = \operatorname{argmax}_{v \in V \setminus S_i} f(S_i \cup \{v\}) \;;$$

Set $S_{i+1} \leftarrow S_i \cup \{v_i\}$;

Greedy Algorithm for Card. Constrained Submodular Max

- This algorithm has a guarantee

Greedy Algorithm for Card. Constrained Submodular Max

- This algorithm has a guarantee

Theorem 12.5.1

Given a polymatroid function f , the above greedy algorithm returns sets S_i such that for each i we have $f(S_i) \geq (1 - 1/e) \max_{|S| \leq i} f(S)$.

Greedy Algorithm for Card. Constrained Submodular Max

- This algorithm has a guarantee

Theorem 12.5.1

Given a polymatroid function f , the above greedy algorithm returns sets S_i such that for each i we have $f(S_i) \geq (1 - 1/e) \max_{|S| \leq i} f(S)$.

- To approximately find $A^* \in \operatorname{argmax} \{ f(A) : |A| \leq k \}$, we repeat the greedy step until $k = i + 1$:

Greedy Algorithm for Card. Constrained Submodular Max

- This algorithm has a guarantee

Theorem 12.5.1

Given a polymatroid function f , the above greedy algorithm returns sets S_i such that for each i we have $f(S_i) \geq (1 - 1/e) \max_{|S| \leq i} f(S)$.

- To approximately find $A^* \in \operatorname{argmax} \{f(A) : |A| \leq k\}$, we repeat the greedy step until $k = i + 1$:
- Again, since this generalizes max k-cover, Feige (1998) showed that this can't be improved. Unless $P = NP$, no polynomial time algorithm can do better than $(1 - 1/e + \epsilon)$ for any $\epsilon > 0$.

The Greedy Algorithm: $1 - 1/e$ intuition.

- At step $i < k$, greedy chooses v_i to maximize $f(v|S_i)$.

The Greedy Algorithm: $1 - 1/e$ intuition.

- At step $i < k$, greedy chooses v_i to maximize $f(v|S_i)$.
- Let S^* be optimal solution (of size k) and $\text{OPT} = f(S^*)$.

The Greedy Algorithm: $1 - 1/e$ intuition.

- At step $i < k$, greedy chooses v_i to maximize $f(v|S_i)$.
- Let S^* be optimal solution (of size k) and OPT $= f(S^*)$. By submodularity, we will show:

$$\exists v \in V \setminus S_i : f(v|S_i) = f(S_i + v|S_i) \geq \frac{1}{k}(\text{OPT} - f(S_i)) \tag{12.21}$$

The Greedy Algorithm: $1 - 1/e$ intuition.

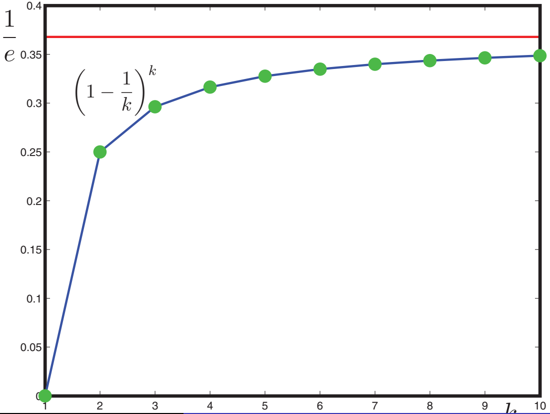
- At step $i < k$, greedy chooses v_i to maximize $f(v|S_i)$.
- Let S^* be optimal solution (of size k) and OPT $= f(S^*)$. By submodularity, we will show:

$$\exists v \in V \setminus S_i : f(v|S_i) = f(S_i + v|S_i) \geq \frac{1}{k}(\text{OPT} - f(S_i)) \tag{12.21}$$

The Greedy Algorithm: $1 - 1/e$ intuition.

- At step $i < k$, greedy chooses v_i to maximize $f(v|S_i)$.
- Let S^* be optimal solution (of size k) and OPT $= f(S^*)$. By submodularity, we will show:

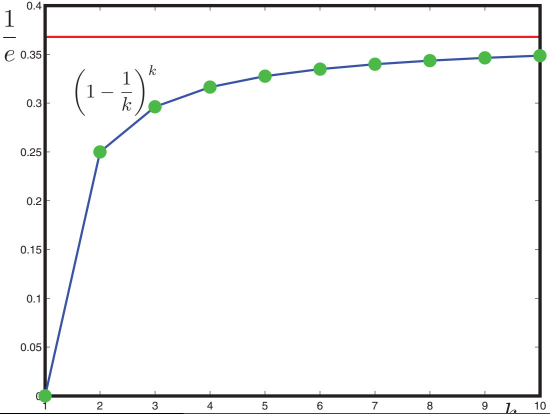
$$\exists v \in V \setminus S_i : f(v|S_i) = f(S_i + v|S_i) \geq \frac{1}{k}(\text{OPT} - f(S_i)) \tag{12.21}$$



The Greedy Algorithm: $1 - 1/e$ intuition.

- At step $i < k$, greedy chooses v_i to maximize $f(v|S_i)$.
- Let S^* be optimal solution (of size k) and OPT $= f(S^*)$. By submodularity, we will show:

$$\exists v \in V \setminus S_i : f(v|S_i) = f(S_i + v|S_i) \geq \frac{1}{k}(\text{OPT} - f(S_i)) \quad (12.21)$$



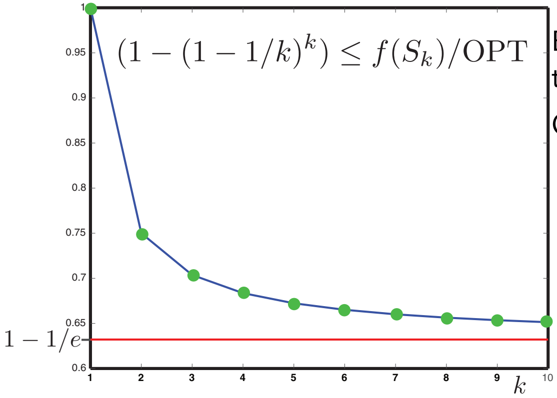
Equation (12.30) will show that Equation (12.21) $\Rightarrow$:

$$\begin{aligned} &\text{OPT} - f(S_{i+1}) \\ &\qquad \leq (1 - 1/k)(\text{OPT} - f(S_i)) \\ &\Rightarrow \text{OPT} - f(S_k) \\ &\qquad \leq (1 - 1/k)^k \text{OPT} \\ &\qquad \leq 1/e\text{OPT} \\ &\Rightarrow \text{OPT}(1 - 1/e) \leq f(S_k) \end{aligned}$$

The Greedy Algorithm: $1 - 1/e$ intuition.

- At step $i < k$, greedy chooses v_i to maximize $f(v|S_i)$.
- Let S^* be optimal solution (of size k) and OPT $= f(S^*)$. By submodularity, we will show:

$$\exists v \in V \setminus S_i : f(v|S_i) = f(S_i + v|S_i) \geq \frac{1}{k}(\text{OPT} - f(S_i)) \qquad (12.21)$$



Equation (12.30) will show that Equation (12.21) $\Rightarrow$:

$$\begin{aligned}
&\text{OPT} - f(S_{i+1}) \\
&\qquad \leq (1-1/k)(\text{OPT} - f(S_i)) \\
&\Rightarrow \text{OPT} - f(S_k) \\
&\qquad \leq (1-1/k)^k \text{OPT} \\
&\qquad \leq 1/e\text{OPT} \\
&\Rightarrow \text{OPT}(1-1/e) \leq f(S_k)
\end{aligned}$$

Cardinality Constrained Polymatroid Max Theorem

Theorem 12.5.2 (Nemhauser et al. 1978)

Given non-negative monotone submodular function $f : 2^V \to \mathbb{R}_+$, *define* $\{S_i\}_{i \geq 0}$ *to be the chain formed by the greedy algorithm (Eqn. (12.20)). Then for all* $k, \ell \in \mathbb{Z}_{++}$, *we have:*

$$f(S_\ell) \geq (1 - e^{-\ell/k}) \max_{S:|S| \leq k} f(S) \tag{12.22}$$

and in particular, for $\ell = k$, *we have* $f(S_k) \geq (1 - 1/e) \max_{S:|S| \leq k} f(S)$.

Cardinality Constrained Polymatroid Max Theorem

Theorem 12.5.2 (Nemhauser et al. 1978)

Given non-negative monotone submodular function $f : 2^V \to \mathbb{R}_+$, *define* $\{S_i\}_{i \geq 0}$ *to be the chain formed by the greedy algorithm (Eqn. (12.20)). Then for all* $k, \ell \in \mathbb{Z}_{++}$, *we have:*

$$f(S_\ell) \geq (1 - e^{-\ell/k}) \max_{S:|S| \leq k} f(S) \tag{12.22}$$

and in particular, for $\ell = k$, *we have* $f(S_k) \geq (1 - 1/e) \max_{S:|S| \leq k} f(S)$.

- k is size of optimal set, i.e., OPT $= f(S^*)$ with $|S^*| = k$

Cardinality Constrained Polymatroid Max Theorem

Theorem 12.5.2 (Nemhauser et al. 1978)

Given non-negative monotone submodular function $f : 2^V \to \mathbb{R}_+$, define $\{S_i\}_{i \geq 0}$ to be the chain formed by the greedy algorithm (Eqn. (12.20)). Then for all $k, \ell \in \mathbb{Z}_{++}$, we have:

$$f(S_\ell) \geq (1 - e^{-\ell/k}) \max_{S:|S| \leq k} f(S) \tag{12.22}$$

and in particular, for $\ell = k$, we have $f(S_k) \geq (1 - 1/e) \max_{S:|S|\leq k} f(S)$.

- k is size of optimal set, i.e., OPT $= f(S^*)$ with $|S^*| = k$
- ℓ is size of set we are choosing (i.e., we choose S_ℓ from greedy chain).

Cardinality Constrained Polymatroid Max Theorem

Theorem 12.5.2 (Nemhauser et al. 1978)

Given non-negative monotone submodular function $f : 2^V \to \mathbb{R}_+$, define $\{S_i\}_{i \geq 0}$ to be the chain formed by the greedy algorithm (Eqn. (12.20)). Then for all $k, \ell \in \mathbb{Z}_{++}$, we have:

$$f(S_\ell) \geq (1 - e^{-\ell/k}) \max_{S:|S| \leq k} f(S) \tag{12.22}$$

and in particular, for $\ell = k$, we have $f(S_k) \geq (1 - 1/e) \max_{S:|S| \leq k} f(S)$.

- k is size of optimal set, i.e., OPT $= f(S^*)$ with $|S^*| = k$
- ℓ is size of set we are choosing (i.e., we choose S_ℓ from greedy chain).
- Bound is how well does S_ℓ (of size ℓ) do relative to S^*, the optimal set of size k.

Cardinality Constrained Polymatroid Max Theorem

Theorem 12.5.2 (Nemhauser et al. 1978)

Given non-negative monotone submodular function $f : 2^V \to \mathbb{R}_+$*, define* $\{S_i\}_{i \geq 0}$ *to be the chain formed by the greedy algorithm (Eqn. (12.20)). Then for all* $k, \ell \in \mathbb{Z}_{++}$*, we have:*

$$f(S_\ell) \geq (1 - e^{-\ell/k}) \max_{S:|S| \leq k} f(S) \qquad (12.22)$$

and in particular, for $\ell = k$*, we have* $f(S_k) \geq (1 - 1/e) \max_{S:|S| \leq k} f(S)$*.*

- k is size of optimal set, i.e., OPT $= f(S^*)$ with $|S^*| = k$
- ℓ is size of set we are choosing (i.e., we choose S_ℓ from greedy chain).
- Bound is how well does S_ℓ (of size ℓ) do relative to S^*, the optimal set of size k.
- Intuitively, bound should get worse when $\ell < k$ and get better when $\ell > k$.

Cardinality Constrained Polymatroid Max Theorem

Proof of Theorem 12.5.2.

...

Cardinality Constrained Polymatroid Max Theorem

Proof of Theorem 12.5.2.

- Fix ℓ (number of items greedy will chose) and k (size of optimal set to compare against).

...

Cardinality Constrained Polymatroid Max Theorem

Proof of Theorem 12.5.2.

- Fix ℓ (number of items greedy will chose) and k (size of optimal set to compare against).
- Set $S^* \in \operatorname{argmax}\{f(S) : |S| \leq k\}$

...

Cardinality Constrained Polymatroid Max Theorem

Proof of Theorem 12.5.2.

- Fix ℓ (number of items greedy will chose) and k (size of optimal set to compare against).
- Set $S^* \in \operatorname{argmax} \{f(S) : |S| \leq k\}$
- w.l.o.g. assume $|S^*| = k$.

...

Cardinality Constrained Polymatroid Max Theorem

Proof of Theorem 12.5.2.

- Fix ℓ (number of items greedy will chose) and k (size of optimal set to compare against).
- Set $S^* \in \operatorname{argmax}\{f(S) : |S| \leq k\}$
- w.l.o.g. assume $|S^*| = k$.
- Order $S^* = (v_1^*, v_2^*, \ldots, v_k^*)$ arbitrarily.

...

Cardinality Constrained Polymatroid Max Theorem

Proof of Theorem 12.5.2.

- Fix ℓ (number of items greedy will chose) and k (size of optimal set to compare against).
- Set $S^* \in \operatorname{argmax}\{f(S) : |S| \leq k\}$
- w.l.o.g. assume $|S^*| = k$.
- Order $S^* = (v_1^*, v_2^*, \dots, v_k^*)$ arbitrarily.
- Let $S_i = (v_1, v_2, \dots, v_i)$ be the greedy order chain chosen by the algorithm, for $i \in \{1, 2, \dots, \ell\}$.

...

Cardinality Constrained Polymatroid Max Theorem

Proof of Theorem 12.5.2.

- Fix ℓ (number of items greedy will chose) and k (size of optimal set to compare against).
- Set $S^* \in \operatorname{argmax}\{f(S) : |S| \leq k\}$
- w.l.o.g. assume $|S^*| = k$.
- Order $S^* = (v_1^*, v_2^*, \dots, v_k^*)$ arbitrarily.
- Let $S_i = (v_1, v_2, \dots, v_i)$ be the greedy order chain chosen by the algorithm, for $i \in \{1, 2, \dots, \ell\}$.
- Then the following inequalities (on the next slide) follow:

...

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

. . .

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- For all $i < \ell$, we have

$$f(S^*)$$

. . .

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- For all $i < \ell$, we have

$$f(S^*) \leq f(S^* \cup S_i)$$

. . .

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- For all $i < \ell$, we have

$$f(S^*) \leq f(S^* \cup S_i) = f(S_i) + f(S^*|S_i) \tag{12.23}$$

. . .

Cardinality Constrained Polymatroid Max Theorem

... proof of Theorem 12.5.2 cont.

- For all $i < \ell$, we have

$$f(S^*) \leq f(S^* \cup S_i) = f(S_i) + f(S^*|S_i) \tag{12.23}$$

$$= f(S_i) + \sum_{j=1}^{k} f(v_j^*|S_i \cup \{v_1^*, v_2^*, \ldots, v_{j-1}^*\}) \tag{12.24}$$

...

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- For all $i < \ell$, we have

$$f(S^*) \leq f(S^* \cup S_i) = f(S_i) + f(S^*|S_i) \tag{12.23}$$

$$= f(S_i) + \sum_{j=1}^{k} f(v_j^*|S_i \cup \{v_1^*, v_2^*, \dots, v_{j-1}^*\}) \tag{12.24}$$

$$\leq f(S_i) + \sum_{v \in S^*} f(v|S_i) \tag{12.25}$$

. . .

Cardinality Constrained Polymatroid Max Theorem

…proof of Theorem 12.5.2 cont.

- For all $i < \ell$, we have

$$f(S^*) \leq f(S^* \cup S_i) = f(S_i) + f(S^*|S_i) \tag{12.23}$$

$$= f(S_i) + \sum_{j=1}^{k} f(v_j^*|S_i \cup \{v_1^*, v_2^*, \ldots, v_{j-1}^*\}) \tag{12.24}$$

$$\leq f(S_i) + \sum_{v \in S^*} f(v|S_i) \tag{12.25}$$

$$\leq f(S_i) + \sum_{v \in S^*} f(v_{i+1}|S_i)$$

…

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- For all $i < \ell$, we have

$$f(S^*) \leq f(S^* \cup S_i) = f(S_i) + f(S^*|S_i) \tag{12.23}$$

$$= f(S_i) + \sum_{j=1}^{k} f(v_j^*|S_i \cup \{v_1^*, v_2^*, \dots, v_{j-1}^*\}) \tag{12.24}$$

$$\leq f(S_i) + \sum_{v \in S^*} f(v|S_i) \tag{12.25}$$

$$\leq f(S_i) + \sum_{v \in S^*} f(v_{i+1}|S_i) = f(S_i) + \sum_{v \in S^*} f(S_{i+1}|S_i) \tag{12.26}$$

. . .

Cardinality Constrained Polymatroid Max Theorem

... proof of Theorem 12.5.2 cont.

- For all $i < \ell$, we have

$$f(S^*) \leq f(S^* \cup S_i) = f(S_i) + f(S^*|S_i) \tag{12.23}$$

$$= f(S_i) + \sum_{j=1}^{k} f(v_j^*|S_i \cup \{v_1^*, v_2^*, \dots, v_{j-1}^*\}) \tag{12.24}$$

$$\leq f(S_i) + \sum_{v \in S^*} f(v|S_i) \tag{12.25}$$

$$\leq f(S_i) + \sum_{v \in S^*} f(v_{i+1}|S_i) = f(S_i) + \sum_{v \in S^*} f(S_{i+1}|S_i) \tag{12.26}$$

$$= f(S_i) + k f(S_{i+1}|S_i) \tag{12.27}$$

...

Cardinality Constrained Polymatroid Max Theorem

... proof of Theorem 12.5.2 cont.

- For all $i < \ell$, we have

$$f(S^*) \leq f(S^* \cup S_i) = f(S_i) + f(S^*|S_i) \tag{12.23}$$

$$= f(S_i) + \sum_{j=1}^{k} f(v_j^*|S_i \cup \{v_1^*, v_2^*, \ldots, v_{j-1}^*\}) \tag{12.24}$$

$$\leq f(S_i) + \sum_{v \in S^*} f(v|S_i) \tag{12.25}$$

$$\leq f(S_i) + \sum_{v \in S^*} f(v_{i+1}|S_i) = f(S_i) + \sum_{v \in S^*} f(S_{i+1}|S_i) \tag{12.26}$$

$$= f(S_i) + kf(S_{i+1}|S_i) \tag{12.27}$$

- Therefore, we have Equation 12.21, i.e.,:

$$f(S^*) - f(S_i) \leq kf(S_{i+1}|S_i) = k(f(S_{i+1}) - f(S_i)) \tag{12.28}$$

...

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- Define gap $\delta_i \triangleq f(S^*) - f(S_i)$, so $\delta_i - \delta_{i+1} = f(S_{i+1}) - f(S_i)$,

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- Define gap $\delta_i \triangleq f(S^*) - f(S_i)$, so $\delta_i - \delta_{i+1} = f(S_{i+1}) - f(S_i)$, giving

$$\delta_i \leq k(\delta_i - \delta_{i+1}) \tag{12.29}$$

or

Cardinality Constrained Polymatroid Max Theorem

... proof of Theorem 12.5.2 cont.

- Define gap $\delta_i \triangleq f(S^*) - f(S_i)$, so $\delta_i - \delta_{i+1} = f(S_{i+1}) - f(S_i)$, giving

$$\delta_i \leq k(\delta_i - \delta_{i+1}) \tag{12.29}$$

or

$$\delta_{i+1} \leq (1 - \frac{1}{k})\delta_i \tag{12.30}$$

Cardinality Constrained Polymatroid Max Theorem

... proof of Theorem 12.5.2 cont.

- Define gap $\delta_i \triangleq f(S^*) - f(S_i)$, so $\delta_i - \delta_{i+1} = f(S_{i+1}) - f(S_i)$, giving

$$\delta_i \leq k(\delta_i - \delta_{i+1}) \tag{12.29}$$

  or

$$\delta_{i+1} \leq (1 - \frac{1}{k})\delta_i \tag{12.30}$$

- The relationship between δ_0 and δ_ℓ is then

$$\delta_l \leq (1 - \frac{1}{k})^\ell \delta_0 \tag{12.31}$$

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- Define gap $\delta_i \triangleq f(S^*) - f(S_i)$, so $\delta_i - \delta_{i+1} = f(S_{i+1}) - f(S_i)$, giving

$$\delta_i \leq k(\delta_i - \delta_{i+1}) \tag{12.29}$$

  or

$$\delta_{i+1} \leq (1 - \frac{1}{k})\delta_i \tag{12.30}$$

- The relationship between δ_0 and δ_ℓ is then

$$\delta_l \leq (1 - \frac{1}{k})^\ell \delta_0 \tag{12.31}$$

- Now, $\delta_0 = f(S^*) - f(\emptyset) \leq f(S^*)$ since $f \geq 0$.

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- Define gap $\delta_i \triangleq f(S^*) - f(S_i)$, so $\delta_i - \delta_{i+1} = f(S_{i+1}) - f(S_i)$, giving

$$\delta_i \leq k(\delta_i - \delta_{i+1}) \tag{12.29}$$

or

$$\delta_{i+1} \leq (1 - \frac{1}{k})\delta_i \tag{12.30}$$

- The relationship between δ_0 and δ_ℓ is then

$$\delta_l \leq (1 - \frac{1}{k})^\ell \delta_0 \tag{12.31}$$

- Now, $\delta_0 = f(S^*) - f(\emptyset) \leq f(S^*)$ since $f \geq 0$.
- Also, by variational bound $1 - x \leq e^{-x}$ for $x \in \mathbb{R}$, we have

$$\delta_\ell \leq (1 - \frac{1}{k})^\ell \delta_0 \leq e^{-\ell/k} f(S^*) \tag{12.32}$$

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

□

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- When we identify $\delta_l = f(S^*) - f(S_\ell)$, a bit of rearranging then gives:

$$f(S_\ell) \geq (1 - e^{-\ell/k}) f(S^*) \tag{12.33}$$

□

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- When we identify $\delta_l = f(S^*) - f(S_\ell)$, a bit of rearranging then gives:

$$f(S_\ell) \geq (1 - e^{-\ell/k}) f(S^*) \tag{12.33}$$

□

- With $\ell = k$, when picking k items, greedy gets $(1 - 1/e) \approx 0.6321$ bound. This means that if S_k is greedy solution of size k, and S^* is an optimal solution of size k, $f(S_k) \geq (1 - 1/e) f(S^*) \approx 0.6321 f(S^*)$.

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- When we identify $\delta_l = f(S^*) - f(S_\ell)$, a bit of rearranging then gives:

$$f(S_\ell) \geq (1 - e^{-\ell/k}) f(S^*) \tag{12.33}$$

□

- With $\ell = k$, when picking k items, greedy gets $(1 - 1/e) \approx 0.6321$ bound. This means that if S_k is greedy solution of size k, and S^* is an optimal solution of size k, $f(S_k) \geq (1 - 1/e) f(S^*) \approx 0.6321 f(S^*)$.
- What if we want to guarantee a solution no worse than $.95 f(S^*)$ where $|S^*| = k$?

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- When we identify $\delta_l = f(S^*) - f(S_\ell)$, a bit of rearranging then gives:

$$f(S_\ell) \geq (1 - e^{-\ell/k}) f(S^*) \tag{12.33}$$

□

- With $\ell = k$, when picking k items, greedy gets $(1 - 1/e) \approx 0.6321$ bound. This means that if S_k is greedy solution of size k, and S^* is an optimal solution of size k, $f(S_k) \geq (1 - 1/e) f(S^*) \approx 0.6321 f(S^*)$.
- What if we want to guarantee a solution no worse than $.95 f(S^*)$ where $|S^*| = k$? Set $0.95 = (1 - e^{-\ell/k})$, which gives $\ell = \lceil -k \ln(1 - 0.95) \rceil = 4k$.

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- When we identify $\delta_l = f(S^*) - f(S_\ell)$, a bit of rearranging then gives:

$$f(S_\ell) \geq (1 - e^{-\ell/k}) f(S^*) \tag{12.33}$$

□

- With $\ell = k$, when picking k items, greedy gets $(1 - 1/e) \approx 0.6321$ bound. This means that if S_k is greedy solution of size k, and S^* is an optimal solution of size k, $f(S_k) \geq (1 - 1/e) f(S^*) \approx 0.6321 f(S^*)$.
- What if we want to guarantee a solution no worse than $.95 f(S^*)$ where $|S^*| = k$? Set $0.95 = (1 - e^{-\ell/k})$, which gives $\ell = \lceil -k \ln(1 - 0.95) \rceil = 4k$. And $\lceil -\ln(1 - 0.999) \rceil = 7$.

Cardinality Constrained Polymatroid Max Theorem

. . . proof of Theorem 12.5.2 cont.

- When we identify $\delta_l = f(S^*) - f(S_\ell)$, a bit of rearranging then gives:

$$f(S_\ell) \geq (1 - e^{-\ell/k}) f(S^*) \tag{12.33}$$

□

- With $\ell = k$, when picking k items, greedy gets $(1 - 1/e) \approx 0.6321$ bound. This means that if S_k is greedy solution of size k, and S^* is an optimal solution of size k, $f(S_k) \geq (1 - 1/e) f(S^*) \approx 0.6321 f(S^*)$.
- What if we want to guarantee a solution no worse than $.95 f(S^*)$ where $|S^*| = k$? Set $0.95 = (1 - e^{-\ell/k})$, which gives $\ell = \lceil -k \ln(1 - 0.95) \rceil = 4k$. And $\lceil -\ln(1 - 0.999) \rceil = 7$.
- So solution, in the worst case, quickly gets very good. Typical/practical case is much better.

Greedy running time

- Greedy computes a new maximum $n = |V|$ times, and each maximum computation requires $O(n)$ comparisons, leading to $O(n^2)$ computation for greedy.

Greedy running time

- Greedy computes a new maximum $n = |V|$ times, and each maximum computation requires $O(n)$ comparisons, leading to $O(n^2)$ computation for greedy.
- This is the best we can do for arbitrary functions, but $O(n^2)$ is not practical to some.

Greedy running time

- Greedy computes a new maximum $n = |V|$ times, and each maximum computation requires $O(n)$ comparisons, leading to $O(n^2)$ computation for greedy.
- This is the best we can do for arbitrary functions, but $O(n^2)$ is not practical to some.
- Greedy can be made much faster in practice by a simple strategy made possible, once again, via the use of submodularity.

Greedy running time

- Greedy computes a new maximum $n = |V|$ times, and each maximum computation requires $O(n)$ comparisons, leading to $O(n^2)$ computation for greedy.
- This is the best we can do for arbitrary functions, but $O(n^2)$ is not practical to some.
- Greedy can be made much faster in practice by a simple strategy made possible, once again, via the use of submodularity.
- This is called Minoux's 1977 Accelerated Greedy strategy (and has been rediscovered a few times, e.g., "Lazy greedy"), and runs much faster while still producing same answer.

Greedy running time

- Greedy computes a new maximum $n = |V|$ times, and each maximum computation requires $O(n)$ comparisons, leading to $O(n^2)$ computation for greedy.
- This is the best we can do for arbitrary functions, but $O(n^2)$ is not practical to some.
- Greedy can be made much faster in practice by a simple strategy made possible, once again, via the use of submodularity.
- This is called Minoux's 1977 Accelerated Greedy strategy (and has been rediscovered a few times, e.g., "Lazy greedy"), and runs much faster while still producing same answer.
- We describe it next:

Minoux's Accelerated Greedy for Submodular Functions

- At stage i in the algorithm, we have a set of gains $f(v|S_i)$ for all $v \notin S_i$. Store these values $\alpha_v \leftarrow f(v|S_i)$ in sorted priority queue.

Minoux's Accelerated Greedy for Submodular Functions

- At stage i in the algorithm, we have a set of gains $f(v|S_i)$ for all $v \notin S_i$. Store these values $\alpha_v \leftarrow f(v|S_i)$ in sorted priority queue.
- Priority queue, $O(1)$ to find max, $O(\log n)$ to insert in right place.

Minoux's Accelerated Greedy for Submodular Functions

- At stage i in the algorithm, we have a set of gains $f(v|S_i)$ for all $v \notin S_i$. Store these values $\alpha_v \leftarrow f(v|S_i)$ in sorted priority queue.
- Priority queue, $O(1)$ to find max, $O(\log n)$ to insert in right place.
- Once we choose a max v, then set $S_{i+1} \leftarrow S_i + v$.

Minoux's Accelerated Greedy for Submodular Functions

- At stage i in the algorithm, we have a set of gains $f(v|S_i)$ for all $v \notin S_i$. Store these values $\alpha_v \leftarrow f(v|S_i)$ in sorted priority queue.
- Priority queue, $O(1)$ to find max, $O(\log n)$ to insert in right place.
- Once we choose a max v, then set $S_{i+1} \leftarrow S_i + v$.
- For $v \notin S_{i+1}$ we have $f(v|S_{i+1}) \leq f(v|S_i)$ by submodularity.

Minoux's Accelerated Greedy for Submodular Functions

- At stage i in the algorithm, we have a set of gains $f(v|S_i)$ for all $v \notin S_i$. Store these values $\alpha_v \leftarrow f(v|S_i)$ in sorted priority queue.
- Priority queue, $O(1)$ to find max, $O(\log n)$ to insert in right place.
- Once we choose a max v, then set $S_{i+1} \leftarrow S_i + v$.
- For $v \notin S_{i+1}$ we have $f(v|S_{i+1}) \leq f(v|S_i)$ by submodularity.
- Therefore, if we find a v' such that $f(v'|S_{i+1}) \geq \alpha_v$ for all $v \neq v'$, then since

$$f(v'|S_{i+1}) \geq \alpha_v = f(v|S_i) \geq f(v|S_{i+1}) \tag{12.34}$$

we have the true max, and we need not re-evaluate gains of other elements again.

Minoux's Accelerated Greedy for Submodular Functions

- At stage i in the algorithm, we have a set of gains $f(v|S_i)$ for all $v \notin S_i$. Store these values $\alpha_v \leftarrow f(v|S_i)$ in sorted priority queue.
- Priority queue, $O(1)$ to find max, $O(\log n)$ to insert in right place.
- Once we choose a max v, then set $S_{i+1} \leftarrow S_i + v$.
- For $v \notin S_{i+1}$ we have $f(v|S_{i+1}) \leq f(v|S_i)$ by submodularity.
- Therefore, if we find a v' such that $f(v'|S_{i+1}) \geq \alpha_v$ for all $v \neq v'$, then since

$$f(v'|S_{i+1}) \geq \alpha_v = f(v|S_i) \geq f(v|S_{i+1}) \tag{12.34}$$

  we have the true max, and we need not re-evaluate gains of other elements again.
- Strategy is: find the $\text{argmax}_{v' \in V \setminus S_{i+1}} \alpha_{v'}$, and then compute the real $f(v'|S_{i+1})$. If it is greater than all other α_v's then that's the next greedy step. Otherwise, replace $\alpha_{v'}$ with its real value, resort $(O(\log n))$, and repeat.

Minoux's Accelerated Greedy for Submodular Functions

- Minoux's algorithm is exact, in that it has the same guarantees as does the $O(n^2)$ greedy Algorithm 4 (this means it will return either the same answers, or answers that have the $1 - 1/e$ guarantee).

Minoux's Accelerated Greedy for Submodular Functions

- Minoux's algorithm is exact, in that it has the same guarantees as does the $O(n^2)$ greedy Algorithm 4 (this means it will return either the same answers, or answers that have the $1 - 1/e$ guarantee).
- In practice: Minoux's trick has enormous speedups ($\approx 700\times$) over the standard greedy procedure due to reduced function evaluations and use of good data structures (priority queue).

Minoux's Accelerated Greedy for Submodular Functions

- Minoux's algorithm is exact, in that it has the same guarantees as does the $O(n^2)$ greedy Algorithm 4 (this means it will return either the same answers, or answers that have the $1 - 1/e$ guarantee).
- In practice: Minoux's trick has enormous speedups ($\approx 700\times$) over the standard greedy procedure due to reduced function evaluations and use of good data structures (priority queue).
- When choosing a of size k, naïve greedy algorithm is $O(nk)$ but accelerated variant at the very best does $O(n + k)$, so this limits the speedup.

Minoux's Accelerated Greedy for Submodular Functions

- Minoux's algorithm is exact, in that it has the same guarantees as does the $O(n^2)$ greedy Algorithm 4 (this means it will return either the same answers, or answers that have the $1 - 1/e$ guarantee).
- In practice: Minoux's trick has enormous speedups ($\approx 700\times$) over the standard greedy procedure due to reduced function evaluations and use of good data structures (priority queue).
- When choosing a of size k, naïve greedy algorithm is $O(nk)$ but accelerated variant at the very best does $O(n + k)$, so this limits the speedup.
- Algorithm has been rediscovered (I think) independently (CELF - cost-effective lazy forward selection, Leskovec et al., 2007)

Minoux's Accelerated Greedy for Submodular Functions

- Minoux's algorithm is exact, in that it has the same guarantees as does the $O(n^2)$ greedy Algorithm 4 (this means it will return either the same answers, or answers that have the $1 - 1/e$ guarantee).
- In practice: Minoux's trick has enormous speedups ($\approx 700\times$) over the standard greedy procedure due to reduced function evaluations and use of good data structures (priority queue).
- When choosing a of size k, naïve greedy algorithm is $O(nk)$ but accelerated variant at the very best does $O(n + k)$, so this limits the speedup.
- Algorithm has been rediscovered (I think) independently (CELF - cost-effective lazy forward selection, Leskovec et al., 2007)
- Can be used used for "big data" sets (e.g., social networks, selecting blogs of greatest influence, document summarization, etc.).

Priority Queue

- Use a priority queue Q as a data structure: operations include:

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 $\alpha \in \mathbb{R}$.

$$\text{insert}(Q, (v, \alpha)) \tag{12.35}$$

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 $\alpha \in \mathbb{R}$.

$$\text{insert}(Q, (v, \alpha)) \tag{12.35}$$

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

$$(v, \alpha) \leftarrow \text{pop}(Q) \tag{12.36}$$

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 $\alpha \in \mathbb{R}$.

$$\text{insert}(Q, (v, \alpha)) \tag{12.35}$$

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

$$(v, \alpha) \leftarrow \text{pop}(Q) \tag{12.36}$$

  - Query the value of the max item in the queue

$$\max(Q) \in \mathbb{R} \tag{12.37}$$

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 $\alpha \in \mathbb{R}$.
$$\text{insert}(Q, (v, \alpha)) \tag{12.35}$$
  - Pop the item (v, α) with maximum value α off the queue.
$$(v, \alpha) \leftarrow \text{pop}(Q) \tag{12.36}$$
  - Query the value of the max item in the queue
$$\max(Q) \in \mathbb{R} \tag{12.37}$$
- 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

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 $\alpha \in \mathbb{R}$.

$$\text{insert}(Q, (v, \alpha)) \tag{12.35}$$

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

$$(v, \alpha) \leftarrow \text{pop}(Q) \tag{12.36}$$

  - Query the value of the max item in the queue

$$\max(Q) \in \mathbb{R} \tag{12.37}$$

- 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 2: 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:

$$S^* \in \operatorname*{argmin}_{S \subseteq V} |S| \text{ such that } f(S) \geq \alpha \tag{12.38}$$

where α is a "cover" requirement.

(Minimum) Submodular Set Cover

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

$$S^* \in \operatorname*{argmin}_{S \subseteq V} |S| \text{ such that } f(S) \geq \alpha \tag{12.38}$$

  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:

$$S^* \in \operatorname*{argmin}_{S \subseteq V} |S| \text{ such that } f'(S) \geq f'(V) \tag{12.39}$$

(Minimum) Submodular Set Cover

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

$$S^* \in \operatorname*{argmin}_{S \subseteq V} |S| \text{ such that } f(S) \geq \alpha \tag{12.38}$$

  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:

$$S^* \in \operatorname*{argmin}_{S \subseteq V} |S| \text{ such that } f'(S) \geq f'(V) \tag{12.39}$$

- Note that this immediately generalizes standard set cover, in which case $f(A)$ is the cardinality of the union of sets indexed by A.

(Minimum) Submodular Set Cover

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

$$S^* \in \operatorname*{argmin}_{S \subseteq V} |S| \text{ such that } f(S) \geq \alpha \tag{12.38}$$

  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:

$$S^* \in \operatorname*{argmin}_{S \subseteq V} |S| \text{ such that } f'(S) \geq f'(V) \tag{12.39}$$

- 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

$$|S^{\mathsf{G}}| \leq |S^*| H(\max_{s \in V} f(\{s\})) = |S^*| O(\log_e(\max_{s \in V} f(\{s\}))) \qquad (12.40)$$

where H is the harmonic function, i.e., $H(d) = \sum_{i=1}^{d} (1/i)$.

(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

$$|S^{\mathsf{G}}| \leq |S^*| H(\max_{s \in V} f(\{s\})) = |S^*| O(\log_e(\max_{s \in V} f(\{s\}))) \tag{12.40}$$

  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:

$$|S^{\mathsf{G}}| \leq |S^*| \left(1 + \log_e \frac{f(V)}{f(V) - f(S_{T-1})}\right) \tag{12.41}$$

  wehre S_T is the final greedy solution that occurs at step T.

(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

$$|S^{\mathsf{G}}| \leq |S^*| H(\max_{s \in V} f(\{s\})) = |S^*| O(\log_e(\max_{s \in V} f(\{s\}))) \tag{12.40}$$

  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:

$$|S^{\mathsf{G}}| \leq |S^*| \left(1 + \log_e \frac{f(V)}{f(V) - f(S_{T-1})}\right) \tag{12.41}$$

  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.

Summary: Monotone Submodular Maximization

- Only makes sense when there is a constraint.

Summary: Monotone Submodular Maximization

- Only makes sense when there is a constraint.
- We discussed cardinality constraint

Summary: Monotone Submodular Maximization

- Only makes sense when there is a constraint.
- We discussed cardinality constraint
- Generalizes the max k-cover problem, and also similar to the set cover problem.

Summary: Monotone Submodular Maximization

- Only makes sense when there is a constraint.
- We discussed cardinality constraint
- Generalizes the max k-cover problem, and also similar to the set cover problem.
- Simple greedy algorithm gets $1 - e^{-\ell/k}$ approximation, where k is size of optimal set we compare against, and ℓ is size of set greedy algorithm chooses.

Summary: Monotone Submodular Maximization

- Only makes sense when there is a constraint.
- We discussed cardinality constraint
- Generalizes the max k-cover problem, and also similar to the set cover problem.
- Simple greedy algorithm gets $1 - e^{-\ell/k}$ approximation, where k is size of optimal set we compare against, and ℓ is size of set greedy algorithm chooses.
- Submodular cover: min. $|S|$ s.t. $f(S) \geq \alpha$.

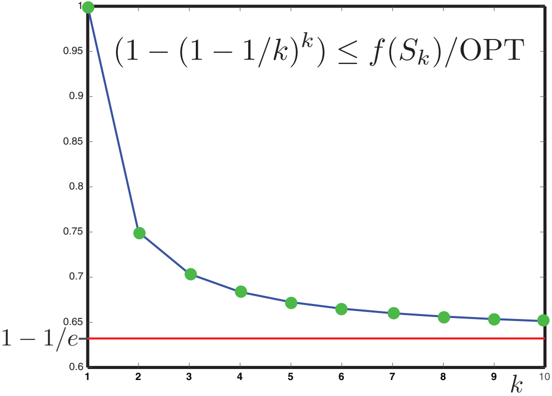
Summary: Monotone Submodular Maximization

- Only makes sense when there is a constraint.
- We discussed cardinality constraint
- Generalizes the max k-cover problem, and also similar to the set cover problem.
- Simple greedy algorithm gets $1 - e^{-\ell/k}$ approximation, where k is size of optimal set we compare against, and ℓ is size of set greedy algorithm chooses.
- Submodular cover: min. $|S|$ s.t. $f(S) \geq \alpha$.
- Minoux's accelerated greedy trick.

The Greedy Algorithm: $1 - 1/e$ intuition.

- At step $i < k$, greedy chooses v_i to maximize $f(v|S_i)$.
- Let S^* be optimal solution (of size k) and OPT $= f(S^*)$. By submodularity, we will show:

$$\exists v \in V \setminus S_i : f(v|S_i) = f(S_i + v|S_i) \geq \frac{1}{k}(\text{OPT} - f(S_i)) \qquad (12.21)$$



Equation (12.30) will show that Equation (12.21) $\Rightarrow$:

$$\begin{aligned}
&\text{OPT} - f(S_{i+1}) \\
&\qquad \leq (1-1/k)(\text{OPT} - f(S_i)) \\
\Rightarrow \;& \text{OPT} - f(S_k) \\
&\qquad \leq (1-1/k)^k \text{OPT} \\
&\qquad \leq 1/e\text{OPT} \\
\Rightarrow \;& \text{OPT}(1-1/e) \leq f(S_k)
\end{aligned}$$

Randomized greedy

- How can we produce a randomized greedy strategy, one where each greedy sweep produces a set that, on average, has a $1 - 1/e$ guarantee?

Randomized greedy

- How can we produce a randomized greedy strategy, one where each greedy sweep produces a set that, on average, has a $1 - 1/e$ guarantee?
- Suppose the following holds:

$$E[f(a_{i+1}|A_i)] \geq \frac{f(OPT) - f(A_i)}{k} \tag{12.42}$$

where $A_i = (a_1, a_2, \ldots, a_i)$ are the first i elements chosen by the strategy.

Randomized greedy

- How can we produce a randomized greedy strategy, one where each greedy sweep produces a set that, on average, has a $1 - 1/e$ guarantee?
- Suppose the following holds:

$$E[f(a_{i+1}|A_i)] \geq \frac{f(OPT) - f(A_i)}{k} \tag{12.42}$$

where $A_i = (a_1, a_2, \ldots, a_i)$ are the first i elements chosen by the strategy.

- See problem 5, homework 4.