Submodular Functions, Optimization, and Applications to Machine Learning
— Spring Quarter, Lecture 15 —
http://www.ee.washington.edu/people/faculty/bilmes/classes/ee596b_spring_2016/

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

May 23rd, 2016

\[ f(A) + f(B) \geq f(A \cup B) + f(A \cap B) \]

Cumulative Outstanding Reading

- Read chapters 2 and 3 from Fujishige’s book.
- Read chapter 1 from Fujishige’s book.
Announcements, Assignments, and Reminders

- Homework 3, available at our assignment dropbox (https://canvas.uw.edu/courses/1039754/assignments), due (electronically) Monday (5/2) at 11:55pm.
- Homework 1, available at our assignment dropbox (https://canvas.uw.edu/courses/1039754/assignments), due (electronically) Friday (4/8) at 11:55pm.
- Weekly Office Hours: Mondays, 3:30-4:30, or by skype or google hangout (set up meeting via our discussion board (https://canvas.uw.edu/courses/1039754/discussion_topics)).

Class Road Map - IT-I

- L1(3/28): Motivation, Applications, & Basic Definitions
- L3(4/4): Info theory exs, more apps, definitions, graph/combinatorial examples, matrix rank example, visualization
- L4(4/6): Graph and Combinatorial Examples, matrix rank, Venn diagrams, examples of proofs of submodularity, some useful properties
- L5(4/11): Examples & Properties, OtherDefs., Independence
- L6(4/13): Independence, Matroids, Matroid Examples, matroid rank is submodular
- L8(4/20): Transversals, Matroid and representation, Dual Matroids
- L10(4/27): Matroid and Greedy, Polyhedra, Matroid Polytopes
- L11(5/2): From Matroids to Polymatroids, Polymatroids
- L12(5/4): Polymatroids, Polymatroids and Greedy
- L13(5/9): Polymatroids and Greedy; Possible Polytopes; Extreme Points; Polymatroids, Greedy, and Cardinality Constrained Maximization
- L14(5/11): Cardinality Constrained Maximization; Curvature; Submodular Max w. Other Constraints
- L15(5/16): Submodular Max w. Other Constraints, Most Violated ≤ , Matroids cont., Closure/Sat,
- L16(5/18): Closure/Sat, Fund. Circuit/Dep, Min-Norm Point and SFM, Min-Norm Point Algorithm, Proof that min-norm gives optimal.
- L17(5/23).
- L18(5/25).
- L19(6/1).
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:
  \[
  \exists v \in V \setminus S_i : f(v|S_i) = f(S_i + v|S_i) \geq \frac{1}{k}(OPT - f(S_i)) \quad (15.1)
  \]

Equation (??) will show that Equation (15.1) implies:

\[
OPT - f(S_{i+1}) \leq (1 - 1/k)(OPT - f(S_i))
\]

\[
\Rightarrow OPT - f(S_k) \leq (1 - 1/k)^kOPT
\]

\[
\leq 1/eOPT
\]

\[
\Rightarrow OPT(1 - 1/e) \leq f(S_k)
\]

Priority Queue

- Use a priority queue $Q$ as a data structure: operations include:
  - Insert an item $(v, \alpha)$ into queue, with $v \in V$ and $\alpha \in \mathbb{R}$.
    \[
    \text{INSERT}(Q, (v, \alpha)) \quad (15.14)
    \]
  - Pop the item $(v, \alpha)$ with maximum value $\alpha$ off the queue.
    \[
    (v, \alpha) \leftarrow \text{POP}(Q) \quad (15.15)
    \]
  - Query the value of the max item in the queue
    \[
    \text{MAX}(Q) \in \mathbb{R} \quad (15.16)
    \]

- On next slide, we call a popped item “fresh” if the value $(v, \alpha)$ 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 3: 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. \hspace{1em} INSERT($Q, f(v)$)
4. repeat
5. \hspace{2em} $(v, \alpha) \leftarrow \text{POP}(Q)$;
6. \hspace{2em} if $\alpha$ not “fresh” then
7. \hspace{3em} recompute $\alpha \leftarrow f(v|S_i)$
8. \hspace{2em} if (popped $\alpha$ in line 5 was “fresh”) OR ($\alpha \geq \text{MAX}(Q)$) then
9. \hspace{3em} Set $S_{i+1} \leftarrow S_i \cup \{v\}$;
10. \hspace{3em} $i \leftarrow i + 1$;
11. \hspace{2em} else
12. \hspace{3em} 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:

\[
S^* \in \arg\min_{S \subseteq V} |S| \text{ such that } f(S) \geq \alpha \tag{15.14}
\]

where $\alpha$ is a “cover” requirement.

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

\[
S^* \in \arg\min_{S \subseteq V} |S| \text{ such that } f'(S) \geq f'(V) \tag{15.15}
\]

- 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

$$|S^G| \leq |S^*| H(\max_{s \in V} f(\{s\})) = |S^*| O(\log_e (\max_{s \in V} f(\{s\}))) \quad (15.14)$$

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:

$$|S^G| \leq |S^*| \left(1 + \log_e \frac{f(V)}{f(V) - f(S^* - 1)}\right) \quad (15.15)$$

where $S_T$ 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 $\alpha$ is the desired cover constraint.

**Curvature of a Submodular function**

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

$$c \triangleq 1 - \min_{S, j \notin S : f(j|\emptyset) \neq 0} \frac{f(j|S)}{f(j|\emptyset)} = 1 - \min_{j : f(j|\emptyset) \neq 0} \frac{f(j|V \setminus \{j\})}{f(j|\emptyset)} \quad (15.17)$$

- 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 \( \frac{1}{1 + c} \) approximation to \( \max \{ f(S) : S \in \mathcal{I} \} \) when \( f \) has total curvature \( c \).
- Hence, greedy subject to matroid constraint is a \( \max(\frac{1}{1 + c}, \frac{1}{2}) \) approximation algorithm, and if \( c < 1 \) then it is better than \( \frac{1}{2} \) (e.g., with \( c = \frac{1}{4} \) then we have a 0.8 algorithm).

For \( k \)-uniform matroid (i.e., \( k \)-cardinality constraints), then approximation factor becomes \( \frac{1}{c}(1 - e^{-c}) \).

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

\[
S_{i+1} = S_i \cup \left\{ \arg\max_{v \in V \setminus S_i : S_i + v \in \bigcap_{i=1}^p I_i} f(S_i \cup \{v\}) \right\} \tag{15.17}
\]

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

**Theorem 15.2.2**

*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 \( f(S_i) \geq \frac{1}{p+1} \max_{|S| \leq i, S \in \bigcap_{i=1}^p I_i} f(S) \), 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.
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.

I have ... as an example of public ownership

je le ai ... comme exemple de propriété publique
One possible alignment, a matching, with score as sum of edge weights.

I have … as an example of public ownership

je le ai … comme exemple de propriété publique

Edges incident to English words constitute an edge partition

I have … as an example of public ownership

je le ai … comme exemple de propriété publique

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

I have ... as an example of public ownership

je le ai ... comme exemple de propriété publique

- 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.

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

. . . the ... of public ownership
. . . le ... de propriété publique

. . . the ... of public ownership
. . . le ... de propriété publique
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

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.
Submodular Welfare: Submodular Max over matroid partition

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

$$E' = E \cup E \cup \cdots \cup E$$

(15.2)

- Let $E^{(i)} \subseteq E'$ 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

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

(15.3)
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 \( \frac{1}{2} \) 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 \( e = "salmon roll" \).
- 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
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 \( E \cup \overline{E} \cup \overline{E} \cup \overline{E} \cup \overline{E} \cup \overline{E} \).
- 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
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 \( E \cup E \cup E \cup E \cup E \cup E \). 
- 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
Submodular Max w. Other Constraints

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.

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

$$S_{i+1} = S_i \cup \left\{ \arg \max_{v \in V \setminus S_i} \left( f(S_i \cup \{v\}) - f(S_i) \right) \right\}$$

(15.4)

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

$$S_{i+1} = S_i \cup \left\{ \arg \max_{v \in V \setminus S_i} \frac{f(S_i \cup \{v\}) - f(S_i)}{c(v)} \right\}$$

(15.5)

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.
- 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 $(\frac{1}{3} - \frac{\varepsilon}{n})$ approximation for maximizing non-monotone non-negative submodular functions, with most $O(\frac{1}{\varepsilon} n^3 \log n)$ 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).
- The following interesting result is true for any submodular function:

**Lemma 15.3.1**

*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 15.3.1**

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 \( \frac{1}{2} \) approximation algorithm for unconstrained non-monotone non-negative submodular maximization.
- Buchbinder, Feldman, Naor, Schwartz 2012. Recall \( [a]_+ = \max(a, 0) \).

**Algorithm 4**: Randomized Linear-time non-monotone submodular max

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

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.
- 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 and polyhedral approaches

- We’ve spent much time discussing SFM and the polymatroidal polytope, and in general polyhedral approaches for SFM.
- Most of the approaches for submodular max have not used such an approach, probably due to the difficulty in computing the “concave extension” of a submodular function (the convex extension is easy, namely the Lovász extension).
- A paper by Chekuri, Vondrak, and Zenklusen (2011) make some progress on this front using multilinear extensions.

Multilinear extension

Definition 15.3.2

For a set function $f : 2^V \rightarrow \mathbb{R}$, define its multilinear extension $F : [0, 1]^V \rightarrow \mathbb{R}$ by

$$F(x) = \sum_{S \subseteq V} f(S) \prod_{i \in S} x_i \prod_{j \in V \setminus S} (1 - x_j) \quad (15.6)$$

- Note that $F(x) = Ef(\hat{x})$ where $\hat{x}$ is a random binary vector over $\{0, 1\}^V$ with elements independent w. probability $x_i$ for $\hat{x}_i$.
- While this is defined for any set function, we have:

Lemma 15.3.3

Let $F : [0, 1]^V \rightarrow \mathbb{R}$ be multilinear extension of set function $f : 2^V \rightarrow \mathbb{R}$, then

- If $f$ is monotone non-decreasing, then $\frac{\partial F}{\partial x_i} \geq 0$ for all $i \in V$, $x \in [0, 1]^V$.
- If $f$ is submodular, then $\frac{\partial^2 F}{\partial x_i \partial x_j} \leq 0$ for all $i, j \in V$, $x \in [0, 1]^V$. 

Moreover, we have

**Lemma 15.3.4**

Let $F : [0, 1]^V \rightarrow \mathbb{R}$ be multilinear extension of set function $f : 2^V \rightarrow \mathbb{R}$, then

- If $f$ is monotone non-decreasing, then $F$ is non-decreasing along any line of direction $d \in \mathbb{R}^E$ with $d \geq 0$.
- If $f$ is submodular, then $F$ is concave along any line of direction $d \geq 0$, and is convex along any line of direction $1_v - 1_w$ for any $v, w \in V$.

Another connection between submodularity and convexity/concavity but note, unlike the Lovász extension, this function is neither.

---

**Submodular Max and polyhedral approaches**

- Basic idea: Given a set of constraints $\mathcal{I}$, we form a polytope $P_\mathcal{I}$ such that $\{1_I : I \in \mathcal{I}\} \subseteq P_\mathcal{I}$.
- We find $\max_{x \in P_\mathcal{I}} F(x)$ where $F(x)$ is the multi-linear extension of $f$, to find a fractional solution $x^*$.
- We then round $x^*$ to a point on the hypercube, thus giving us a solution to the discrete problem.
In the recent paper by Chekuri, Vondrak, and Zenklusen, they show:

1) constant factor approximation algorithm for max \{F(x) : x \in P\} for any down-monotone solvable polytope \(P\) and \(F\) multilinear extension of any non-negative submodular function.

2) A randomized rounding (pipage rounding) scheme to obtain an integer solution

3) An optimal \((1 - 1/e)\) instance of their rounding scheme that can be used for a variety of interesting independence systems, including \(O(1)\) knapsacks, \(k\) matroids and \(O(1)\) knapsacks, a \(k\)-matchoid and \(\ell\) sparse packing integer programs, and unsplittable flow in paths and trees.

Also, Vondrak showed that this scheme achieves the \(\frac{1}{e}(1 - e^{-c})\) curvature based bound for any matroid, which matches the bound we had earlier for uniform matroids with standard greedy.

In general, one needs to do Monte-Carlo methods to estimate the multilinear extension (so further approximations would apply).
Theorem 15.4.1

Let $f$ be a polymatroid function defined on subsets of $E$. For any $x \in \mathbb{R}^E_+$, and any $P_f^+$-basis $y^x \in \mathbb{R}^E_+$ of $x$, the component sum of $y^x$ is

$$y^x(E) = \text{rank}(x) = \max \left( y(E) : y \leq x, y \in P_f^+ \right) = \min \left( x(A) + f(E \setminus A) : A \subseteq E \right) \quad (15.10)$$

As a consequence, $P_f^+$ is a polymatroid, since r.h.s. is constant w.r.t. $y^x$.

Taking $E \setminus B = \text{supp}(x)$ (so elements $B$ are all zeros in $x$), and for $b \notin B$ we make $x(b)$ is big enough, the r.h.s. min has solution $A^* = B$. We recover submodular function from the polymatroid polyhedron via the following:

$$\text{rank}\left( \frac{1}{\epsilon} 1_{E \setminus B} \right) = f(B) = \max \left\{ y(B) : y \in P_f^+ \right\} \quad (15.11)$$

In fact, we will ultimately see a number of important consequences of this theorem (other than just that $P_f^+$ is a polymatroid).
Corollary 15.4.2

We have that:

\[
\max \{ y(E) : y \in P_{\text{ind. set}}(M), y \leq x \} = \min \{ r_M(A) + x(E \setminus A) : A \subseteq E \}
\]

(15.30)

where \( r_M \) is the matroid rank function of some matroid.

Consider Theorem ??, the matroid case is now a special case, where we have that:

\[
P_r^+ = \{ x \in \mathbb{R}^E : x \geq 0, x(A) \leq r_M(A), \forall A \subseteq E \}
\]

(15.7)

Suppose we have any \( x \in \mathbb{R}^E \) such that \( x \notin P_r^+ \).

Hence, there must be a set of \( \mathcal{W} \subseteq 2^V \), each member of which corresponds to a violated inequality, i.e., equations of the form \( x(A) > r_M(A) \) for \( A \in \mathcal{W} \).

The most violated inequality when \( x \) is considered w.r.t. \( P_r^+ \) corresponds to the set \( A \) that maximizes \( x(A) - r_M(A) \), i.e., the most violated inequality is valuated as:

\[
\max \{ x(A) - r_M(A) : A \in \mathcal{W} \} = \max \{ x(A) - r_M(A) : A \subseteq E \}
\]

(15.8)

Since \( x \) is modular and \( x(E \setminus A) = x(E) - x(A) \), we can express this via a min as in:

\[
\min \{ r_M(A) + x(E \setminus A) : A \subseteq E \}
\]

(15.9)
Consider

\[ P_f^+ = \{ x \in \mathbb{R}^E : x \geq 0, x(A) \leq f(A), \forall A \subseteq E \} \quad (15.10) \]

Suppose we have any \( x \in \mathbb{R}_E^+ \) such that \( x \notin P_f^+ \).

Hence, there must be a set of \( \mathcal{W} \subseteq 2^V \), each member of which corresponds to a violated inequality, i.e., equations of the form \( x(A) > r_M(A) \) for \( A \in \mathcal{W} \).

\[ \mathcal{W} = \{\{1\}\{2\}\} \quad \mathcal{W} = \{\{2\}, \{1, 2\}\} \quad \mathcal{W} = \{\{1, 2\}\} \]

The most violated inequality when \( x \) is considered w.r.t. \( P_f^+ \) corresponds to the set \( A \) that maximizes \( x(A) - f(A) \), i.e., the most violated inequality is valuated as:

\[ \max \{ x(A) - f(A) : A \in \mathcal{W} \} = \max \{ x(A) - f(A) : A \subseteq E \} \quad (15.11) \]

Since \( x \) is modular and \( x(E \setminus A) = x(E) - x(A) \), we can express this via a min as in:

\[ \min \{ f(A) + x(E \setminus A) : A \subseteq E \} \quad (15.12) \]

More importantly, \( \min \{ f(A) + x(E \setminus A) : A \subseteq E \} \) is a form of submodular function minimization, namely, \( \min \{ f(A) - x(A) : A \subseteq E \} \) for a submodular \( f \) and \( x \in \mathbb{R}_E^+ \), consisting of a difference of polymatroid and modular function (so \( f - x \) is no longer necessarily monotone, nor positive).

We will ultimately answer how general this form of SFM is.
The following three slides are review from lecture 6.

### Matroids, other definitions using matroid rank $r : 2^V \rightarrow \mathbb{Z}^+$

**Definition 15.5.3 (closed/flat/subspace)**

A subset $A \subseteq E$ is **closed** (equivalently, a **flat** or a **subspace**) of matroid $M$ if for all $x \in E \setminus A$, $r(A \cup \{x\}) = r(A) + 1$.

Definition: A **hyperplane** is a flat of rank $r(M) - 1$.

**Definition 15.5.4 (closure)**

Given $A \subseteq E$, the **closure** (or **span**) of $A$, is defined by $\text{span}(A) = \{b \in E : r(A \cup \{b\}) = r(A)\}$.

Therefore, a closed set $A$ has $\text{span}(A) = A$.

**Definition 15.5.5 (circuit)**

A subset $A \subseteq E$ is **circuit** or a **cycle** if it is an inclusionwise-minimal dependent set (i.e., if $r(A) < |A|$ and for any $a \in A$, $r(A \setminus \{a\}) = |A| - 1$).
A set is independent if and only if it contains no circuit. Therefore, it is not surprising that circuits can also characterize a matroid.

**Theorem 15.5.3 (Matroid by circuits)**

Let $E$ be a set and $C$ be a collection of subsets of $E$ that satisfy the following three properties:

1. **(C1)**: $\emptyset \notin C$
2. **(C2)**: if $C_1, C_2 \in C$ and $C_1 \subseteq C_2$, then $C_1 = C_2$.
3. **(C3)**: if $C_1, C_2 \in C$ with $C_1 \neq C_2$, and $e \in C_1 \cap C_2$, then there exists a $C_3 \in C$ such that $C_3 \subseteq (C_1 \cup C_2) \setminus \{e\}$.

Several circuit definitions for matroids.

**Theorem 15.5.3 (Matroid by circuits)**

Let $E$ be a set and $C$ be a collection of nonempty subsets of $E$, such that no two sets in $C$ are contained in each other. Then the following are equivalent.

1. $C$ is the collection of circuits of a matroid;
2. if $C, C' \in C$, and $x \in C \cap C'$, then $(C \cup C') \setminus \{x\}$ contains a set in $C$;
3. if $C, C' \in C$, and $x \in C \cap C'$, and $y \in C \setminus C'$, then $(C \cup C') \setminus \{x\}$ contains a set in $C$ containing $y$;

Again, think about this for a moment in terms of linear spaces and matrices, and spanning trees.
**Fundamental circuits in matroids**

**Lemma 15.5.1**

Let \( I \in \mathcal{I}(M) \), and \( e \in E \), then \( I \cup \{e\} \) contains at most one circuit in \( M \).

**Proof.**

- Suppose, to the contrary, that there are two distinct circuits \( C_1, C_2 \) such that \( C_1 \cup C_2 \subseteq I \cup \{e\} \).
- Then \( e \in C_1 \cap C_2 \), and by (C2), there is a circuit \( C_3 \) of \( M \) s.t. \( C_3 \subseteq (C_1 \cup C_2) \setminus \{e\} \subseteq I \).
- This contradicts the independence of \( I \).

In general, let \( C(I, e) \) be the unique circuit associated with \( I \cup \{e\} \) (commonly called the fundamental circuit in \( M \) w.r.t. \( I \) and \( e \)).

**Matroids: The Fundamental Circuit**

- Define \( C(I, e) \) be the unique circuit associated with \( I \cup \{e\} \) (the fundamental circuit in \( M \) w.r.t. \( I \) and \( e \), if it exists).
- If \( e \in \text{span}(I) \setminus I \), then \( C(I, e) \) is well defined (\( I + e \) creates one circuit).
- If \( e \in I \), then \( I + e = I \) doesn’t create a circuit. In such cases, \( C(I, e) \) is not really defined.
- In such cases, we define \( C(I, e) = \{e\} \), and we will soon see why.
- If \( e \notin \text{span}(I) \), then \( C(I, e) = \emptyset \), since no circuit is created in this case.
Lemma 15.5.2

Let $B(D)$ be the set of bases of any set $D$. Then, given matroid $\mathcal{M} = (E, \mathcal{I})$, and any loop-free (i.e., no dependent singleton elements) set $D \subseteq E$, we have:

$$\bigcup_{B \in B(D)} B = D.$$  \hspace{1cm} (15.13)

Proof.

- Define $D' \triangleq \bigcup_{B \in B(D)} B$, and suppose $\exists d \in D$ such that $d \notin D'$.
- Hence, $\forall B \in B(D)$ we have $d \notin B$, and $B + c$ must contain a single circuit for any $B$, namely $C(B, d)$.
- Then choose $d' \in C(B, d)$ with $d' \neq d$.
- Then $B + d - d'$ is independent size $|B|$ subset of $D$ and hence spans $D$, and thus is a $d$-containing member of $B(D)$, contradicting $d \notin D'$.

The $\text{sat}$ function = Polymatroid Closure

- Thus, in a matroid, closure (span) of a set $A$ are all items that $A$ spans (eq. that depend on $A$).
- We wish to generalize closure to polymatroids.
- Consider $x \in P_f$ for polymatroid function $f$.
- Again, recall, tight sets are closed under union and intersection, and therefore form a distributive lattice.
- That is, we saw in Lecture 7 that for any $A, B \in \mathcal{D}(x)$, we have that $A \cup B \in \mathcal{D}(x)$ and $A \cap B \in \mathcal{D}(x)$, which can constitute a join and meet.
- Recall, for a given $x \in P_f$, we have defined this tight family as

$$\mathcal{D}(x) = \{ A : A \subseteq E, x(A) = f(A) \}$$  \hspace{1cm} (15.14)
The \textit{sat} function = Polymatroid Closure

- Now given $x \in P^+_f$:
  \begin{equation}
  \mathcal{D}(x) = \{A : A \subseteq E, x(A) = f(A)\} \tag{15.15}
  \end{equation}
  \begin{equation}
  = \{A : f(A) - x(A) = 0\} \tag{15.16}
  \end{equation}

- Since $x \in P^+_f$ and $f$ is presumed to be polymatroid function, we see $f'(A) = f(A) - x(A)$ is a non-negative submodular function, and $\mathcal{D}(x)$ are the zero-valued minimizers (if any) of $f'(A)$.
- The zero-valued minimizers of $f'$ are thus closed under union and intersection.
- In fact, this is true for all minimizers of a submodular function as stated in the next theorem.

### Minimizers of a Submodular Function form a lattice

**Theorem 15.6.1**

For arbitrary submodular $f$, the minimizers are closed under union and intersection. That is, let $\mathcal{M} = \text{arg\ min}_{X \subseteq E} f(X)$ be the set of minimizers of $f$. Let $A, B \in \mathcal{M}$. Then $A \cup B \in \mathcal{M}$ and $A \cap B \in \mathcal{M}$.

**Proof.** 

Since $A$ and $B$ are minimizers, we have $f(A) = f(B) \leq f(A \cap B)$ and $f(A) = f(B) \leq f(A \cup B)$.

By submodularity, we have

\begin{equation}
  f(A) + f(B) \geq f(A \cup B) + f(A \cap B) \tag{15.17}
  \end{equation}

Hence, we must have $f(A) = f(B) = f(A \cup B) = f(A \cap B)$.

Thus, the minimizers of a submodular function form a lattice, and there is a maximal and a minimal minimizer of every submodular function.
The sat function $= \text{Polymatroid Closure}$

- Matroid closure is generalized by the unique maximal element in $\mathcal{D}(x)$, also called the polymatroid closure or $\text{sat}$ (saturation function).
- For some $x \in P_f$, we have defined:

$$
\text{cl}(x) \overset{\text{def}}{=} \text{sat}(x) = \bigcup \{ A : A \in \mathcal{D}(x) \} \\
= \bigcup \{ A : A \subseteq E, x(A) = f(A) \}
$$

(15.18)

$$
= \{ e : e \in E, \forall \alpha > 0, x + \alpha e \notin P_f \}
$$

(15.19)

(15.20)

- Hence, $\text{sat}(x)$ is the maximal (zero-valued) minimizer of the submodular function $f_x(A) \triangleq f(A) - x(A)$.
- Eq. (15.20) says that $\text{sat}$ consists of any point $x$ that is $P_f$ saturated (any additional positive movement, in that dimension, leaves $P_f$).
- We’ll revisit this in a few slides.
- First, we see how $\text{sat}$ generalizes matroid closure.

Consider matroid $(E, I) = (E, r)$, some $I \in \mathcal{I}$. Then $1_I \in P_r$ and

$\mathcal{D}(1_I) = \{ A : 1_I(A) = r(A) \}$

(15.21)

and

$$
\text{sat}(1_I) = \bigcup \{ A : A \subseteq E, A \in \mathcal{D}(1_I) \} = \bigcup \{ A : A \subseteq E, 1_I(A) = r(A) \}
$$

(15.22)

(15.23)

(15.24)

- Notice that $1_I(A) = |I \cap A| \leq |I|$.
- Intuitively, consider an $A \supset I \in \mathcal{I}$ that doesn’t increase rank, meaning $r(A) = r(I)$. If $r(A) = |I \cap A| = r(I \cap A)$, as in Eqn. (15.24), then $A$ is in $I$’s span, so should get $\text{sat}(1_I) = \text{span}(I)$.
- We formalize this next.
The sat function = Polymatroid Closure

Lemma 15.7.1 (Matroid sat : \( \mathbb{R}_+^E \rightarrow 2^E \) is the same as closure.)

\[ \text{For } I \in \mathcal{I}, \text{ we have } \text{sat}(1_I) = \text{span}(I) \]  
(15.25)

Proof.

- For \( 1_I(I) = |I| = r(I) \), so \( I \in D(1_I) \) and \( I \subseteq \text{sat}(1_I) \). Also, \( I \subseteq \text{span}(I) \).
- Consider some \( b \in \text{span}(I) \setminus I \).
- Then \( I \cup \{ b \} \in D(1_I) \) since \( 1_I(I \cup \{ b \}) = |I| = r(I \cup \{ b \}) = r(I) \).
- Thus, \( b \in \text{sat}(1_I) \).
- Therefore, \( \text{sat}(1_I) \supseteq \text{span}(I) \).

... proof continued.

- Now, consider \( b \in \text{sat}(1_I) \setminus I \).
- Choose any \( A \in D(1_I) \) with \( b \in A \), thus \( b \in A \setminus I \).
- Then \( 1_I(A) = |A \cap I| = r(A) \).
- Now \( r(A) = |A \cap I| \leq |I| = r(I) \).
- Also, \( r(A \cap I) = |A \cap I| \) since \( A \cap I \in \mathcal{I} \).
- Hence, \( r(A \cap I) = r(A) = r((A \cap I) \cup (A \setminus I)) \) meaning \( (A \setminus I) \subseteq \text{span}(A \cap I) \subseteq \text{span}(I) \).
- Since \( b \in A \setminus I \), we get \( b \in \text{span}(I) \).
- Thus, \( \text{sat}(1_I) \subseteq \text{span}(I) \).
- Hence \( \text{sat}(1_I) = \text{span}(I) \).  

Prof. Jeff Bilmes
F56/77 (pg.65/87)
The sat function $= \text{Polymatroid Closure}$

- Now, consider a matroid $(E, r)$ and some $C \subseteq E$ with $C \not\in \mathcal{I}$, and consider $1_C$. Is $1_C \in P_r$? No, it might not be a vertex, or even a member, of $P_r$.
- span$(\cdot)$ operates on more than just independent sets, so $\text{span}(C)$ is perfectly sensible.
- Note $\text{span}(C) = \text{span}(B)$ where $\mathcal{I} \ni B \in \mathcal{B}(C)$ is a base of $C$.
- Then we have $1_B \leq 1_C \leq 1_{\text{span}(C)}$, and that $1_B \in P_r$. We can then make the definition:

$$\text{sat}(1_C) \triangleq \text{sat}(1_B) \text{ for } B \in \mathcal{B}(C)$$

(15.26)

In which case, we also get $\text{sat}(1_C) = \text{span}(C)$ (in general, could define $\text{sat}(y) = \text{sat}($P-basis$(y))$).
- However, consider the following form

$$\text{sat}(1_C) = \bigcup \{A : A \subseteq E, |A \cap C| = r(A)\}$$

(15.27)

Exercise: is $\text{span}(C) = \text{sat}(1_C)$? Prove or disprove it.

The sat function, span, and submodular function minimization

- Thus, for a matroid, $\text{sat}(1_I)$ is exactly the closure (or span) of $I$ in the matroid. I.e., for matroid $(E, r)$, we have $\text{span}(I) = \text{sat}(1_B)$.
- Recall, for $x \in P_f$ and polymatroidal $f$, $\text{sat}(x)$ is the maximal (by inclusion) minimizer of $f(A) - x(A)$, and thus in a matroid, $\text{span}(I)$ is the maximal minimizer of the submodular function formed by $r(A) - 1_I(A)$.
- Submodular function minimization can solve “span” queries in a matroid or “sat” queries in a polymatroid.
We are given an $x \in P_f^+$ for submodular function $f$.

Recall that for such an $x$, $\text{sat}(x)$ is defined as

$$\text{sat}(x) = \bigcup \{A : x(A) = f(A)\} \quad (15.28)$$

We also have stated that $\text{sat}(x)$ can be defined as:

$$\text{sat}(x) = \left\{ e : \forall \alpha > 0, x + \alpha 1_e \notin P_f^+ \right\} \quad (15.29)$$

We next show more formally that these are the same.

Let's start with one definition and derive the other.

$$\text{sat}(x) \overset{\text{def}}{=} \left\{ e : \forall \alpha > 0, x + \alpha 1_e \notin P_f^+ \right\} \quad (15.30)$$

$$= \left\{ e : \forall \alpha > 0, \exists A \text{ s.t. } (x + \alpha 1_e)(A) > f(A) \right\} \quad (15.31)$$

$$= \left\{ e : \forall \alpha > 0, \exists A \ni e \text{ s.t. } (x + \alpha 1_e)(A) > f(A) \right\} \quad (15.32)$$

This last bit follows since $1_e(A) = 1 \iff e \in A$. Continuing, we get

$$\text{sat}(x) = \left\{ e : \forall \alpha > 0, \exists A \ni e \text{ s.t. } x(A) + \alpha > f(A) \right\} \quad (15.33)$$

Given that $x \in P_f^+$, meaning $x(A) \leq f(A)$ for all $A$, we must have

$$\text{sat}(x) = \left\{ e : \forall \alpha > 0, \exists A \ni e \text{ s.t. } x(A) = f(A) \right\} \quad (15.34)$$

$$= \left\{ e : \exists A \ni e \text{ s.t. } x(A) = f(A) \right\} \quad (15.35)$$

So now, if $A$ is any set such that $x(A) = f(A)$, then we clearly have

$$\forall e \in A, e \in \text{sat}(x), \text{ and therefore that } \text{sat}(x) \supseteq A \quad (15.36)$$
sat, as tight polymatroidal elements

...and therefore, with sat as defined in Eq. (??),

$$\text{sat}(x) \supseteq \bigcup \{ A : x(A) = f(A) \}$$  \hspace{1cm} (15.37)

On the other hand, for any $e \in \text{sat}(x)$ defined as in Eq. (15.35), since $e$ is itself a member of a tight set, there is a set $A \ni e$ such that $x(A) = f(A)$, giving

$$\text{sat}(x) \subseteq \bigcup \{ A : x(A) = f(A) \}$$  \hspace{1cm} (15.38)

Therefore, the two definitions of sat are identical.

---

Another useful concept is saturation capacity which we develop next.

For $x \in P_f$, and $e \in E$, consider finding

$$\max \{ \alpha : \alpha \in \mathbb{R}, x + \alpha 1_e \in P_f \}$$  \hspace{1cm} (15.39)

This is identical to:

$$\max \{ \alpha : (x + \alpha 1_e)(A) \leq f(A), \forall A \ni \{ e \} \}$$  \hspace{1cm} (15.40)

since any $B \subseteq E$ such that $e \notin B$ does not change in a $1_e$ adjustment, meaning $(x + \alpha 1_e)(B) = x(B)$.

Again, this is identical to:

$$\max \{ \alpha : x(A) + \alpha \leq f(A), \forall A \ni \{ e \} \}$$  \hspace{1cm} (15.41)

or

$$\max \{ \alpha : \alpha \leq f(A) - x(A), \forall A \ni \{ e \} \}$$  \hspace{1cm} (15.42)
Submodular Max w. Other Constraints  Most Violated ≤  Matroids cont.  Closure/Sat  Closure/Sat  Fund. Circuit/Dep

### Saturation Capacity

- The max is achieved when
  \[
  \alpha = \hat{c}(x; e) \overset{\text{def}}{=} \min \{ f(A) - x(A), \forall A \supseteq \{e\} \} \tag{15.43}
  \]
- \( \hat{c}(x; e) \) is known as the **saturation capacity** associated with \( x \in P_f \) and \( e \).
- Thus we have for \( x \in P_f \),
  \[
  \hat{c}(x; e) \overset{\text{def}}{=} \min \{ f(A) - x(A), \forall A \ni e \} \tag{15.44}
  = \max \{ \alpha : \alpha \in \mathbb{R}, x + \alpha 1_e \in P_f \} \tag{15.45}
  \]
- We immediately see that for \( e \in E \setminus \text{sat}(x) \), we have that \( \hat{c}(x; e) > 0 \).
- Also, for \( e \in \text{sat}(x) \), we have that \( \hat{c}(x; e) = 0 \).
- Note that any \( \alpha \) with \( 0 \leq \alpha \leq \hat{c}(x; e) \) we have \( x + \alpha 1_e \in P_f \).
- We also see that computing \( \hat{c}(x; e) \) is a form of submodular function minimization.

### Dependence Function

- Tight sets can be restricted to contain a particular element.
- Given \( x \in P_f \), and \( e \in \text{sat}(x) \), define
  \[
  \mathcal{D}(x, e) = \{ A : e \in A \subseteq E, x(A) = f(A) \}
  = \mathcal{D}(x) \cap \{ A : A \subseteq E, e \in A \} \tag{15.46}
  \]
- Thus, \( \mathcal{D}(x, e) \subseteq \mathcal{D}(x) \), and \( \mathcal{D}(x, e) \) is a sublattice of \( \mathcal{D}(x) \).
- Therefore, we can define a unique minimal element of \( \mathcal{D}(x, e) \) denoted as follows:
  \[
  \text{dep}(x, e) = \begin{cases} 
  \bigcap \{ A : e \in A \subseteq E, x(A) = f(A) \} & \text{if } e \in \text{sat}(x) \\
  \emptyset & \text{else}
  \end{cases} \tag{15.48}
  \]
- I.e., \( \text{dep}(x, e) \) is the minimal element in \( \mathcal{D}(x) \) that contains \( e \) (the minimal \( x \)-tight set containing \( e \)).

Prof. Jeff Bilmes  EE596b/Spring 2016/Submodularity - Lecture 15 - May 23rd, 2016  F63/77 (pg.73/87)
• Given some \( x \in P_f \),
• The picture on the right summarizes the relationships between the lattices and sublattices.
• Note, \( \bigcap_e \text{dep}(x, e) = \text{dep}(x) \).

Given \( x \in P_f \), recall distributive lattice of tight sets
\( \mathcal{D}(x) = \{ A : x(A) = f(A) \} \)
We had that \( \text{sat}(x) = \bigcup \{ A : A \in \mathcal{D}(x) \} \) is the “1” element of this lattice.
Consider the “0” element of \( \mathcal{D}(x) \), i.e., \( \text{dry}(x) \overset{\text{def}}{=} \bigcap \{ A : A \in \mathcal{D}(x) \} \)
We can see \( \text{dry}(x) \) as the elements that are necessary for tightness.
That is, we can equivalently define \( \text{dry}(x) \) as
\[
\text{dry}(x) = \{ e' : x(A) < f(A), \forall A \not\ni e' \} \quad (15.49)
\]
This can be read as, for any \( e' \in \text{dry}(x) \), any set that does not contain \( e' \) is not tight for \( x \) (any set \( A \) that is missing any element of \( \text{dry}(x) \) is not tight).
Perhaps, then, a better name for \( \text{dry} \) is \( \text{ntight}(x) \), for the necessary for tightness (but we’ll actually use neither name).
Note that \( \text{dry} \) need not be the empty set. Exercise: give example.
An alternate expression for \( \text{dep} = \text{dry} \)

- Now, given \( x \in P_f \), and \( e \in \text{sat}(x) \), recall distributive sub-lattice of \( e \)-containing tight sets \( \mathcal{D}(x, e) = \{ A : e \in A, x(A) = f(A) \} \).
- We can define the "1" element of this sub-lattice as \( \text{sat}(x, e) \) with
  \[
  \text{sat}(x, e) \overset{\text{def}}{=} \bigcup \{ A : A \in \mathcal{D}(x, e) \}.
  \]
- Analogously, we can define the "0" element of this sub-lattice as \( \text{dry}(x, e) \) with
  \[
  \text{dry}(x, e) \overset{\text{def}}{=} \bigcap \{ A : A \in \mathcal{D}(x, e) \}.
  \]
- We can see \( \text{dry}(x, e) \) as the elements that are necessary for \( e \)-containing tightness, with \( e \in \text{sat}(x) \).
- That is, we can view \( \text{dry}(x, e) \) as
  \[
  \text{dry}(x, e) = \{ e' : x(A) < f(A), \forall A \not\ni e', e \in A \} \tag{15.50}
  \]
- This can be read as, for any \( e' \in \text{dry}(x, e) \), any \( e \)-containing set that does not contain \( e' \) is not tight for \( x \).
- But actually, \( \text{dry}(x, e) = \text{dep}(x, e) \), so we have derived another expression for \( \text{dep}(x, e) \) in Eq. (15.50).

Dependence Function and Fundamental Matroid Circuit

- Now, let \( (E, \mathcal{I}) = (E, r) \) be a matroid, and let \( I \in \mathcal{I} \) giving \( 1_I \in P_f \).
  We have \( \text{sat}(1_I) = \text{span}(I) = \text{closure}(I) \).
- Given \( e \in \text{sat}(1_I) \setminus I \) and then consider an \( A \ni e \) with \( |I \cap A| = r(A) \).
- Then \( I \cap A \) serves as a base for \( A \) (i.e., \( I \cap A \) spans \( A \)) and any such \( A \) contains a circuit (i.e., we can add \( e \in A \setminus I \) to \( I \cap A \) w/o increasing rank).
- Given \( e \in \text{sat}(1_I) \setminus I \), and consider \( \text{dep}(1_I, e) \), with
  \[
  \text{dep}(1_I, e) = \bigcap \{ A : e \in A \lneq E, 1_I(A) = r(A) \} \tag{15.51}
  \]
  \[
  = \bigcap \{ A : e \in A \lneq E, |I \cap A| = r(A) \} \tag{15.52}
  \]
  \[
  = \bigcap \{ A : e \in A \lneq E, r(A) - |I \cap A| = 0 \} \tag{15.53}
  \]
- By SFM lattice, \( \exists \) a unique minimal \( A \ni e \) with \( |I \cap A| = r(A) \).
- Thus, \( \text{dep}(1_I, e) \) must be a circuit since if it included more than a circuit, it would not be minimal in this sense.
Dependence Function and Fundamental Matroid Circuit

- Therefore, when \( e \in \text{sat}(1_I) \setminus I \), then \( \text{dep}(1_I, e) = C(I, e) \) where \( C(I, e) \) is the unique circuit contained in \( I + e \) in a matroid (the fundamental circuit of \( e \) and \( I \) that we encountered before).
- Now, if \( e \in \text{sat}(1_I) \cap I \) with \( I \in \mathcal{I} \), we said that \( C(I, e) \) was undefined (since no circuit is created in this case) and so we defined it as \( C(I, e) = \{ e \} \).
- In this case, for such an \( e \), we have \( \text{dep}(1_I, e) = \{ e \} \) since all such sets \( A \ni e \) with \( |I \cap A| = r(A) \) contain \( e \), but in this case no cycle is created, i.e., \( |I \cap A| \geq |I \cap \{ e \}| = r(e) = 1 \).
- We are thus free to take subsets of \( I \) as \( A \), all of which must contain \( e \), but all of which have rank equal to size.
- Also note: in general for \( x \in P_f \) and \( e \in \text{sat}(x) \), we have \( \text{dep}(x, e) \) is tight by definition.

Summary of \( \text{sat}, \text{and dep} \)

- For \( x \in P_f \), \( \text{sat}(x) \) (span, closure) is the maximal saturated \((x\text{-tight})\) set w.r.t. \( x \). i.e., \( \text{sat}(x) = \{ e : e \in E, \forall \alpha > 0, x + \alpha 1_e \notin P_f \} \). That is,
  \[
  \text{cl}(x) \overset{\text{def}}{=} \text{sat}(x) \triangleq \bigcup \{ A : A \in \mathcal{D}(x) \} = \bigcup \{ A : A \subseteq E, x(A) = f(A) \} = \{ e : e \in E, \forall \alpha > 0, x + \alpha 1_e \notin P_f \}
  \]

- For \( e \in \text{sat}(x) \), we have \( \text{dep}(x, e) \subseteq \text{sat}(x) \) (fundamental circuit) is the minimal \((x\text{-tight})\) set w.r.t. \( x \) containing \( e \). i.e.,
  \[
  \text{dep}(x, e) = \begin{cases} 
  \bigcap \{ A : e \in A \subseteq E, x(A) = f(A) \} & \text{if } e \in \text{sat}(x) \\
  \emptyset & \text{else} 
  \end{cases}
  = \{ e' : \exists \alpha > 0, \text{ s.t. } x + \alpha (1_e - 1_{e'}) \in P_f \}
  \]
Dependence Function and exchange

- For \( e \in \text{span}(I) \setminus I \), we have that \( I + e \notin I \). This is a set addition restriction property.
- Analogously, for \( e \in \text{sat}(x) \), any \( x + \alpha e \notin P_f \) for \( \alpha > 0 \). This is a vector increase restriction property.
- Recall, we have \( C(I, e) \setminus e' \in I \) for \( e' \in C(I, e) \). I.e., \( C(I, e) \) consists of elements that when removed recover independence.
- In other words, for \( e \in \text{span}(I) \setminus I \), we have that
  \[
  C(I, e) = \{ a \in E : I + e - a \in I \}
  \]  \hspace{1cm} (15.58)
- I.e., an addition of \( e \) to \( I \) stays within \( I \) only if we simultaneously remove one of the elements of \( C(I, e) \).
- But, analogous to the circuit case, is there an exchange property for \( \text{dep}(x, e) \) in the form of vector movement restriction?
- We might expect the vector \( \text{dep}(x, e) \) property to take the form:
  a positive move in the \( e \)-direction stays within \( P_f^+ \) only if we simultaneously take a negative move in one of the \( \text{dep}(x, e) \) directions.

Dependence Function and exchange in 2D

- \( \text{dep}(x, e) \) is set of neg. directions we must move if we want to move in pos. \( e \) direction, starting at \( x \) and staying within \( P_f \).
- Viewable in 2D, we have for \( A, B \subseteq E \), \( A \cap B = \emptyset \):

  \[\begin{array}{c}
  \text{Left: } A \cap \text{dep}(x, e) = \emptyset, \text{ and we can’t move further in } (e) \text{ direction, and moving in any negative } a \in A \text{ direction doesn’t change that. Notice no dependence between } (e) \text{ and any element in } A. \\
  \text{Right: } A \subseteq \text{dep}(x, e), \text{ and we can’t move further in the } (e) \text{ direction, but we can move further in } (e) \text{ direction by moving in some } a \in A \text{ negative direction. Notice dependence between } (e) \text{ and elements in } A. 
  \end{array}\]
We can move neither in the (e) nor the (a) direction, but we can move in the (e) direction if we simultaneously move in the -(a) direction.

In 3D, we have:

\[ \text{dep}(x, e) = \{ a : a \in E, \exists \alpha > 0 : x + \alpha (1_e - 1_a) \in P_f \} \]  

(15.59)

We next show this formally...

The derivation for \( \text{dep}(x, e) \) involves turning a strict inequality into a non-strict one with a strict explicit slack variable \( \alpha \):  

\[
\text{dep}(x, e) = \text{ntight}(x, e) = \\
= \{ e' : x(A) < f(A), \forall A \ni e', e \in A \} \\
= \{ e' : \exists \alpha > 0, \text{ s.t. } \alpha \leq f(A) - x(A), \forall A \ni e', e \in A \} \\
= \{ e' : \exists \alpha > 0, \text{ s.t. } \alpha 1_e(A) \leq f(A) - x(A), \forall A \ni e', e \in A \} \\
= \{ e' : \exists \alpha > 0, \text{ s.t. } \alpha (1_e(A) - 1_e'(A)) \leq f(A) - x(A), \forall A \ni e', e \in A \} \\
= \{ e' : \exists \alpha > 0, \text{ s.t. } x(A) + \alpha (1_e(A) - 1_e'(A)) \leq f(A), \forall A \ni e', e \in A \} \\

(15.60 - 15.65)

Now, \( 1_e(A) - 1_e'(A) = 0 \) if either \( \{ e, e' \} \subseteq A \), or \( \{ e, e' \} \cap A = \emptyset \).

Also, if \( e' \in A \) but \( e \notin A \), then 

\[ x(A) + \alpha (1_e(A) - 1_e'(A)) = x(A) - \alpha \leq f(A) \text{ since } x \in P_f. \]
thus, we get the same in the above if we remove the constraint $A \not\ni e', e \in A$, that is we get

$$\text{dep}(x, e) = \{e' : \exists \alpha > 0, \text{ s.t. } x(A) + \alpha(1_e - 1_{e'}) \leq f(A), \forall A\}$$

(15.66)

This is then identical to

$$\text{dep}(x, e) = \{e' : \exists \alpha > 0, \text{ s.t. } x + \alpha(1_e - 1_{e'}) \in P_f\}$$

(15.67)

Compare with original, the minimal element of $D(x, e)$, with $e \in \text{sat}(x)$:

$$\text{dep}(x, e) = \begin{cases} \bigcap \{A : e \in A \subseteq E, x(A) = f(A)\} & \text{if } e \in \text{sat}(x) \\ \emptyset & \text{else} \end{cases}$$

(15.68)

---

**Summary of Concepts**

- Most violated inequality $\max \{x(A) - f(A) : A \subseteq E\}$
- Matroid by circuits, and the fundamental circuit $C(I, e) \subseteq I + e$.
- Minimizers of submodular functions form a lattice.
- Minimal and maximal element of a lattice.
- $x$-tight sets, maximal and minimal tight set.
- $\text{sat function} \& \text{Closure}$
- Saturation Capacity
- $e$-containing tight sets
- $\text{dep function} \& \text{fundamental circuit of a matroid}$
Summary important definitions so far: tight, dep, & sat

- **x-tight sets:** For $x \in P_f$, $\mathcal{D}(x) = \{ A \subseteq E : x(A) = f(A) \}$.
- **Polymatroid closure/maximal x-tight set:** For $x \in P_f$, 
  
  $\text{sat}(x) = \bigcup \{ A : A \in \mathcal{D}(x) \} = \{ e : e \in E, \forall \alpha > 0, x + \alpha 1_e \notin P_f \}$.

- **Saturation capacity:** for $x \in P_f$, 
  
  $0 \leq \hat{c}(x; e) = \min \{ f(A) - x(A) | \forall A \ni e \} = \max \{ \alpha : \alpha \in \mathbb{R}, x + \alpha 1_e \in P_f \}$.

- **Recall:** $\text{sat}(x) = \{ e : \hat{c}(x; e) = 0 \}$ and $E \setminus \text{sat}(x) = \{ e : \hat{c}(x; e) > 0 \}$.

- **e-containing x-tight sets:** For $x \in P_f$, 
  
  $\mathcal{D}(x, e) = \{ A : e \in A \subseteq E, x(A) = f(A) \} \subseteq \mathcal{D}(x)$.

- **Minimal e-containing x-tight set/polymatroidal fundamental circuit:** For $x \in P_f$, 
  
  $\text{dep}(x, e) = \begin{cases} 
  \bigcap \{ A : e \in A \subseteq E, x(A) = f(A) \} & \text{if } e \in \text{sat}(x) \\
  \emptyset & \text{else} \\
  \{ e' : \exists \alpha > 0, \text{ s.t. } x + \alpha (1_e - 1_{e'}) \in P_f \} & \text{if } e \notin \text{sat}(x) \end{cases}$