Submodular Functions, Optimization, and Applications to Machine Learning

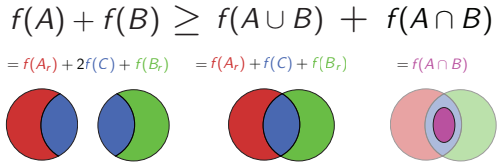
— Spring Quarter, Lecture 19 —

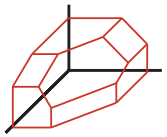
http://j.ee.washington.edu/~bilmes/classes/ee596b_spring_2014/

Prof. Jeff Bilmes

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

June 4th, 2014





Cumulative Outstanding Reading

- Good references for today: Schrijver-2003, Oxley-1992/2011, Welsh-1973, Goemans-2010, Cunningham-1984, Edmonds-1969, Choquet-1955, Grabisch/Marichal/Mesiar/Pap "Aggregation Functions", Lovász-1983, Bach-2011.
- Read Tom McCormick's overview paper on SFM `http://people.commerce.ubc.ca/faculty/mccormick/sfmchap8a.pdf`
- Read chapters 1 - 4 from Fujishige book.
- Matroid properties `http://www-math.mit.edu/~goemans/18433S09/matroid-notes.pdf`
- Read lecture 14 slides on lattice theory at our web page (`http://j.ee.washington.edu/~bilmes/classes/ee596b_spring_2014/`)
- Wolfe "Finding the Nearest Point in a Polytope", 1976.
- Fujishige & Isotani, "A Submodular Function Minimization Algorithm Based on the Minimum-Norm Base", 2009.

Sources for Today's Lecture

- "Submodular Function Maximization", Krause and Golovin.
- Chekuri, Vondrak, Zenklusen, "Submodular Function Maximization via the Multilinear Relaxation and Contention Resolution Schemes", 2011 (a recent paper (appeared yesterday) that, among other things, has a nice up-to-date summary on all the results on submodular max).
- Minoux, "Accelerated Greedy Algorithms for Maximizing Submodular Set Functions", 1977.
- Feige, Mirrokni, Vondrak, "Maximizing non-monotone submodular functions", 2007.
- Fujishige, "Submodular Functions and Optimization", 2005.
- Fujishige, "Submodular Systems and Related Topics", 1984.
- Fisher, Nemhauser, Wolsey, "An Analysis of Approximations for Maximizing Submodular Set Functions - II", 1978.
- Lin & Bilmes, "A Class Of Submodular Functions for Document Summarization", 2011.

Other readings

- J. Vondrak, "Submodularity and curvature: the optimal algorithm" in RIMS Kokyuroku Bessatsu B23, Kyoto, 2010.
- M. Conforti and G. Cornuéjols. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the Rado-Edmonds theorem. Discrete Applied Math, 7(3):251-274, 1984.

Announcements, Assignments, and Reminders

- Weekly Office Hours: Wednesdays, 5:00-5:50, or by skype or google hangout (email me).

Class Road Map - IT-I

- L1 (3/31): Motivation, Applications, & Basic Definitions
- L2: (4/2): Applications, Basic Definitions, Properties
- L3: More examples and properties (e.g., closure properties), and examples, spanning trees
- L4: proofs of equivalent definitions, independence, start matroids
- L5: matroids, basic definitions and examples
- L6: More on matroids, System of Distinct Reps, Transversals, Transversal Matroid, Matroid and representation
- L7: Dual Matroids, other matroid properties, Combinatorial Geometries
- L8: Combinatorial Geometries, matroids and greedy, Polyhedra, Matroid Polytopes,
- L9: From Matroid Polytopes to Polymatroids.
- L10: Polymatroids and Submodularity
- L11: More properties of polymatroids, SFM special cases
- L12: polymatroid properties, extreme points polymatroids,
- L13: sat, dep, supp, exchange capacity, examples
- L14: Lattice theory: partially ordered sets; lattices; distributive, modular, submodular, and boolean lattices; ideals and join irreducibles.
- L15: Supp, Base polytope, polymatroids and entropic Venn diagrams, exchange capacity,
- L16: proof that minimum norm point yields min of submodular function, and the lattice of minimizers of a submodular function, Lovasz extension
- L17: Lovasz extension, Choquet Integration, more properties/examples of Lovasz extension, convex minimization and SFM.
- L18: Lovasz extension examples and structured convex norms, The Min-Norm Point Algorithm detailed.
- L19: symmetric submodular function minimization, maximizing monotone submodular function w. card constraints.
- L20: maximizing monotone submodular function w. other constraints, non-monotone maximization.

Finals Week: June 9th-13th, 2014.

Symmetric Submodular Functions

- Given: $\breve{f} : 2^E \to \mathbb{R}$, if $\breve{f}$ is submodular and also has the property that $\breve{f}(A) = \breve{f}(E \setminus A)$ for all A, then $\breve{f}$ is said to be symmetric submodular

Symmetric Submodular Functions

- Given: $\breve{f} : 2^E \to \mathbb{R}$, if $\breve{f}$ is submodular and also has the property that $\breve{f}(A) = \breve{f}(E \setminus A)$ for all A, then $\breve{f}$ is said to be symmetric submodular
- Given any non-symmetric submodular function f, we can always symmetrize it, $f_{\text{symmetric}}(A) = f(A) + f(E \setminus A)$.

Symmetric Submodular Functions

- Given: $\breve{f} : 2^E \to \mathbb{R}$, if $\breve{f}$ is submodular and also has the property that $\breve{f}(A) = \breve{f}(E \setminus A)$ for all A, then $\breve{f}$ is said to be symmetric submodular
- Given any non-symmetric submodular function f, we can always symmetrize it, $f_{\text{symmetric}}(A) = f(A) + f(E \setminus A)$.
- *Symmetrize* and *normalize* f as $f \to \breve{f}$ via the operation: $\breve{f}(A) = f(A) + f(E \setminus A) - f(E)$, so that $\breve{f}(\emptyset) = 0$ if $f(\emptyset) = 0$.

Symmetric Submodular Functions

- Given: $\breve{f} : 2^E \to \mathbb{R}$, if $\breve{f}$ is submodular and also has the property that $\breve{f}(A) = \breve{f}(E \setminus A)$ for all A, then $\breve{f}$ is said to be symmetric submodular
- Given any non-symmetric submodular function f, we can always symmetrize it, $f_{\text{symmetric}}(A) = f(A) + f(E \setminus A)$.
- *Symmetrize* and *normalize* f as $f \to \breve{f}$ via the operation: $\breve{f}(A) = f(A) + f(E \setminus A) - f(E)$, so that $\breve{f}(\emptyset) = 0$ if $f(\emptyset) = 0$.
- Such an $\breve{f}$ is also non-negative since

$$2\breve{f}(A) = \breve{f}(A) + \breve{f}(E \setminus A) \geq \breve{f}(\emptyset) + \breve{f}(E) = 2\breve{f}(\emptyset) \geq 0 \quad (19.1)$$

Symmetric Submodular Functions

- Given: $\breve{f} : 2^E \to \mathbb{R}$, if $\breve{f}$ is submodular and also has the property that $\breve{f}(A) = \breve{f}(E \setminus A)$ for all A, then $\breve{f}$ is said to be symmetric submodular
- Given any non-symmetric submodular function f, we can always symmetrize it, $f_{\text{symmetric}}(A) = f(A) + f(E \setminus A)$.
- *Symmetrize* and *normalize* f as $f \to \breve{f}$ via the operation: $\breve{f}(A) = f(A) + f(E \setminus A) - f(E)$, so that $\breve{f}(\emptyset) = 0$ if $f(\emptyset) = 0$.
- Such an $\breve{f}$ is also non-negative since

$$2\breve{f}(A) = \breve{f}(A) + \breve{f}(E \setminus A) \geq \breve{f}(\emptyset) + \breve{f}(E) = 2\breve{f}(\emptyset) \geq 0 \qquad (19.1)$$

- Equivalence class: $f \to \breve{f}$ same up to modular shift since $\breve{f} = \breve{g}$ if $f = g + m$ with m modular $\Rightarrow$ consider only polymatroidal f.

Symmetric Submodular Functions

- Given: $\breve{f} : 2^E \to \mathbb{R}$, if $\breve{f}$ is submodular and also has the property that $\breve{f}(A) = \breve{f}(E \setminus A)$ for all A, then $\breve{f}$ is said to be symmetric submodular
- Given any non-symmetric submodular function f, we can always symmetrize it, $f_{\text{symmetric}}(A) = f(A) + f(E \setminus A)$.
- *Symmetrize* and *normalize* f as $f \to \breve{f}$ via the operation: $\breve{f}(A) = f(A) + f(E \setminus A) - f(E)$, so that $\breve{f}(\emptyset) = 0$ if $f(\emptyset) = 0$.
- Such an $\breve{f}$ is also non-negative since

$$2\breve{f}(A) = \breve{f}(A) + \breve{f}(E \setminus A) \geq \breve{f}(\emptyset) + \breve{f}(E) = 2\breve{f}(\emptyset) \geq 0 \qquad (19.1)$$

- Equivalence class: $f \to \breve{f}$ same up to modular shift since $\breve{f} = \breve{g}$ if $f = g + m$ with m modular $\Rightarrow$ consider only polymatroidal f.
- Combinatorial mutual information function, so $\breve{f}(A) = I_f(A; V \setminus A)$ where $I_f(A; B) = f(A) + f(B) - f(A \cup B) - f(A \cap B)$.

Symmetric Submodular Functions

- Given: $\breve{f} : 2^E \to \mathbb{R}$, if $\breve{f}$ is submodular and also has the property that $\breve{f}(A) = \breve{f}(E \setminus A)$ for all A, then $\breve{f}$ is said to be symmetric submodular
- Given any non-symmetric submodular function f, we can always symmetrize it, $f_{\text{symmetric}}(A) = f(A) + f(E \setminus A)$.
- *Symmetrize* and *normalize* f as $f \to \breve{f}$ via the operation: $\breve{f}(A) = f(A) + f(E \setminus A) - f(E)$, so that $\breve{f}(\emptyset) = 0$ if $f(\emptyset) = 0$.
- Such an $\breve{f}$ is also non-negative since

$$2\breve{f}(A) = \breve{f}(A) + \breve{f}(E \setminus A) \geq \breve{f}(\emptyset) + \breve{f}(E) = 2\breve{f}(\emptyset) \geq 0 \qquad (19.1)$$

- Equivalence class: $f \to \breve{f}$ same up to modular shift since $\breve{f} = \breve{g}$ if $f = g + m$ with m modular $\Rightarrow$ consider only polymatroidal f.
- Combinatorial mutual information function, so $\breve{f}(A) = I_f(A; V \setminus A)$ where $I_f(A; B) = f(A) + f(B) - f(A \cup B) - f(A \cap B)$.
- Example: $f(A) = H(X_A) =$ entropy, then $\breve{f} = I(X_A; X_{E \setminus A}) =$ symmetric mutual information.

Separators of submodular function via symmetrized version

- Such a symmetrized submodular function measures a form of "dependence" between A and $\bar{A} \triangleq E \setminus A$.

Separators of submodular function via symmetrized version

- Such a symmetrized submodular function measures a form of "dependence" between A and $\bar{A} \triangleq E \setminus A$.

Theorem 19.3.1

We are given an f that is normalized & submodular. If $\exists A$ s.t. $\breve{f}(A) \triangleq f(A) + f(\bar{A}) - f(E) = 0$ then f is "decomposable" w.r.t. A — this means $f(B) = f(B \cap A) + f(B \cap \bar{A})$, $\forall B$.

Separators of submodular function via symmetrized version

- Such a symmetrized submodular function measures a form of "dependence" between A and $\bar{A} \triangleq E \setminus A$.

Theorem 19.3.1

We are given an f that is normalized & submodular. If $\exists A$ s.t. $\breve{f}(A) \triangleq f(A) + f(\bar{A}) - f(E) = 0$ then f is "decomposable" w.r.t. A — this means $f(B) = f(B \cap A) + f(B \cap \bar{A})$, $\forall B$.

Proof.

...

Separators of submodular function via symmetrized version

- Such a symmetrized submodular function measures a form of "dependence" between A and $\bar{A} \triangleq E \setminus A$.

Theorem 19.3.1

We are given an f that is normalized & submodular. If $\exists A$ s.t. $\breve{f}(A) \triangleq f(A) + f(\bar{A}) - f(E) = 0$ then f is "decomposable" w.r.t. A — this means $f(B) = f(B \cap A) + f(B \cap \bar{A})$, $\forall B$.

Proof.

- By submodularity (subadditivity for non-intersecting sets), we have:

$$f(B) = f\Big((B \cap A) \cup (B \cap \bar{A})\Big) \leq f(B \cap A) + f(B \cap \bar{A}) \tag{19.2}$$

...

Separators of submodular function via symmetrized version

- Such a symmetrized submodular function measures a form of "dependence" between A and $\bar{A} \triangleq E \setminus A$.

Theorem 19.3.1

We are given an f that is normalized & submodular. If $\exists A$ s.t. $\breve{f}(A) \triangleq f(A) + f(\bar{A}) - f(E) = 0$ then f is "decomposable" w.r.t. A — this means $f(B) = f(B \cap A) + f(B \cap \bar{A})$, $\forall B$.

Proof.

- By submodularity (subadditivity for non-intersecting sets), we have:

$$f(B) = f\Big((B \cap A) \cup (B \cap \bar{A})\Big) \leq f(B \cap A) + f(B \cap \bar{A}) \tag{19.2}$$

- Hence, $f(B) \leq f(B \cap A) + f(B \cap \bar{A})$.

...

Separators of submodular function via symmetrized version

. . . proof of Theorem 19.3.1 cont.

□

Separators of submodular function via symmetrized version

. . . proof of Theorem 19.3.1 cont.

- By submodularity

$f(B) - f(B \cap A) - f(B \cap \bar{A})$

□

Separators of submodular function via symmetrized version

. . . proof of Theorem 19.3.1 cont.

- By submodularity

$$f(B) - f(B \cap A) - f(B \cap \bar{A}) \geq f(A \cup B) - f(A) - f(B \cap \bar{A}) \quad (19.3)$$

□

Separators of submodular function via symmetrized version

. . . proof of Theorem 19.3.1 cont.

- By submodularity

$$f(B) - f(B \cap A) - f(B \cap \bar{A}) \geq f(A \cup B) - f(A) - f(B \cap \bar{A}) \quad (19.3)$$

$$\geq f((A \cup B) \cup \bar{A}) - f(A) - f(\bar{A}) \quad (19.4)$$

□

Separators of submodular function via symmetrized version

. . . proof of Theorem 19.3.1 cont.

- By submodularity

$$f(B) - f(B \cap A) - f(B \cap \bar{A}) \geq f(A \cup B) - f(A) - f(B \cap \bar{A}) \quad (19.3)$$

$$\geq f((A \cup B) \cup \bar{A}) - f(A) - f(\bar{A}) \quad (19.4)$$

$$= f(E) - f(A) + f(\bar{A}) = 0 \quad (19.5)$$

□

Separators of submodular function via symmetrized version

. . . proof of Theorem 19.3.1 cont.

- By submodularity

$$f(B) - f(B \cap A) - f(B \cap \bar{A}) \geq f(A \cup B) - f(A) - f(B \cap \bar{A}) \quad (19.3)$$
$$\geq f((A \cup B) \cup \bar{A}) - f(A) - f(\bar{A}) \quad (19.4)$$
$$= f(E) - f(A) + f(\bar{A}) = 0 \quad (19.5)$$

- Eqn. (19.3) follows since $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$,

□

Separators of submodular function via symmetrized version

. . . proof of Theorem 19.3.1 cont.

- By submodularity

$$f(B) - f(B \cap A) - f(B \cap \bar{A}) \geq f(A \cup B) - f(A) - f(B \cap \bar{A}) \quad (19.3)$$
$$\geq f((A \cup B) \cup \bar{A}) - f(A) - f(\bar{A}) \quad (19.4)$$
$$= f(E) - f(A) + f(\bar{A}) = 0 \quad (19.5)$$

- Eqn. (19.3) follows since $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$, and Eqn. (19.4) follows since $B \cap \bar{A} = (A \cup B) \cap \bar{A}$ and $f(A \cup B) + f(\bar{A}) \geq f((A \cup B) \cup \bar{A}) + f((A \cup B) \cap \bar{A})$.

□

Separators of submodular function via symmetrized version

. . . proof of Theorem 19.3.1 cont.

- By submodularity

$$f(B) - f(B \cap A) - f(B \cap \bar{A}) \geq f(A \cup B) - f(A) - f(B \cap \bar{A}) \quad (19.3)$$
$$\geq f((A \cup B) \cup \bar{A}) - f(A) - f(\bar{A}) \quad (19.4)$$
$$= f(E) - f(A) + f(\bar{A}) = 0 \quad (19.5)$$

- Eqn. (19.3) follows since $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$, and Eqn. (19.4) follows since $B \cap \bar{A} = (A \cup B) \cap \bar{A}$ and $f(A \cup B) + f(\bar{A}) \geq f((A \cup B) \cup \bar{A}) + f((A \cup B) \cap \bar{A})$.
- Hence, both $f(B) \geq f(B \cap A) + f(B \cap \bar{A})$ (from above) and $f(B) \leq f(B \cap A) + f(B \cap \bar{A})$ (previous slide).

□

Separators of submodular function via symmetrized version

- Again, let $\breve{f}$ be the symmetrized version of f.

Separators of submodular function via symmetrized version

- Again, let $\breve{f}$ be the symmetrized version of f.
- Definition: If $\breve{f}(A) = 0$, then any $A' \subseteq A$ and $\bar{A}' \subseteq E \setminus A$ are "independent" w.r.t. submodular g, and A is called a separator.

Separators of submodular function via symmetrized version

- Again, let $\breve{f}$ be the symmetrized version of f.
- Definition: If $\breve{f}(A) = 0$, then any $A' \subseteq A$ and $\bar{A}' \subseteq E \setminus A$ are "independent" w.r.t. submodular g, and A is called a separator.
- random variables: $X_A \perp\!\!\!\perp X_B \Rightarrow X_{A'} \perp\!\!\!\perp X_{B'}$ $\forall$ $A' \subseteq A$ and $B' \subseteq B$.

Separators of submodular function via symmetrized version

- Again, let $\breve{f}$ be the symmetrized version of f.
- Definition: If $\breve{f}(A) = 0$, then any $A' \subseteq A$ and $\bar{A}' \subseteq E \setminus A$ are "independent" w.r.t. submodular g, and A is called a separator.
- random variables: $X_A \perp\!\!\!\perp X_B \Rightarrow X_{A'} \perp\!\!\!\perp X_{B'}$ $\forall$ $A' \subseteq A$ and $B' \subseteq B$.
- Set of separators of $\breve{f}$ is closed under intersection, union, and complementation. Hence, the separators partition E.

Separators of submodular function via symmetrized version

- Again, let $\breve{f}$ be the symmetrized version of f.
- Definition: If $\breve{f}(A) = 0$, then any $A' \subseteq A$ and $\bar{A}' \subseteq E \setminus A$ are "independent" w.r.t. submodular g, and A is called a separator.
- random variables: $X_A \perp\!\!\!\perp X_B \Rightarrow X_{A'} \perp\!\!\!\perp X_{B'}$ $\forall$ $A' \subseteq A$ and $B' \subseteq B$.
- Set of separators of $\breve{f}$ is closed under intersection, union, and complementation. Hence, the separators partition E.
- In following slides, $\breve{f}$ is symmetrized & normalized version of f.

Review

Next slide is from Lecture 4.

Many (Equivalent) Definitions of Submodularity

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B), \quad \forall A, B \subseteq V \tag{19.6}$$

$$f(j|S) \geq f(j|T), \ \forall S \subseteq T \subseteq V, \text{ with } j \in V \setminus T \tag{19.7}$$

$$f(C|S) \geq f(C|T), \forall S \subseteq T \subseteq V, \text{ with } C \subseteq V \setminus T \tag{19.8}$$

$$f(j|S) \geq f(j|S \cup \{k\}), \ \forall S \subseteq V \text{ with } j \in V \setminus (S \cup \{k\}) \tag{19.9}$$

$$f(A \cup B|A \cap B) \leq f(A|A \cap B) + f(B|A \cap B), \quad \forall A, B \subseteq V \tag{19.10}$$

$$f(T) \leq f(S) + \sum_{j \in T \setminus S} f(j|S) - \sum_{j \in S \setminus T} f(j|S \cup T - \{j\}), \ \forall S, T \subseteq V \tag{19.11}$$

$$f(T) \leq f(S) + \sum_{j \in T \setminus S} f(j|S), \ \forall S \subseteq T \subseteq V \tag{19.12}$$

$$f(T) \leq f(S) - \sum_{j \in S \setminus T} f(j|S \setminus \{j\}) + \sum_{j \in T \setminus S} f(j|S \cap T) \ \forall S, T \subseteq V \tag{19.13}$$

$$f(T) \leq f(S) - \sum_{j \in S \setminus T} f(j|S \setminus \{j\}), \ \forall T \subseteq S \subseteq V \tag{19.14}$$

Minimization of a Symmetric Submodular Functions

- Minimizing symmetric submodular functions can be done in strongly polynomial time $O(n^3)$. The algorithm by Nagamochi & Ibaracki 1992 for graph cuts shown by Queyranne in 1995 to work for sym. SFM.

Minimization of a Symmetric Submodular Functions

- Minimizing symmetric submodular functions can be done in strongly polynomial time $O(n^3)$. The algorithm by Nagamochi & Ibaracki 1992 for graph cuts shown by Queyranne in 1995 to work for sym. SFM.
- The algorithm finds (as a subroutine) MA (maximum adjacency) or a maximum back orders (not same as greedy order).

Choose v_1 arbitrarily ;
$W_1 \leftarrow (v_1)$ /* The first of an ordered list W_i. */ ;
for $i \leftarrow 1 \dots |V| - 1$ **do**
  Choose $v_{i+1} \in \operatorname{argmin}_{u \in V \setminus W_i} f(W_i | \{u\})$;
  $W_{i+1} \leftarrow (W_i, v_{i+1})$; /* Append v_{i+1} to end of W_i */

Minimization of a Symmetric Submodular Functions

- Minimizing symmetric submodular functions can be done in strongly polynomial time $O(n^3)$. The algorithm by Nagamochi & Ibaracki 1992 for graph cuts shown by Queyranne in 1995 to work for sym. SFM.
- The algorithm finds (as a subroutine) MA (maximum adjacency) or a maximum back orders (not same as greedy order).

Choose v_1 arbitrarily ;
$W_1 \leftarrow (v_1)$ /* The first of an ordered list W_i. */ ;
for $i \leftarrow 1 \dots |V| - 1$ **do**
  Choose $v_{i+1} \in \operatorname{argmin}_{u \in V \setminus W_i} f(W_i | \{u\})$;
  $W_{i+1} \leftarrow (W_i, v_{i+1})$; /* Append v_{i+1} to end of W_i */

- Note algorithm operates on non-symmetric function f. If f is already symmetric and normalized, then $f = \breve{f}$.

Minimization of a Symmetric Submodular Functions

- Minimizing symmetric submodular functions can be done in strongly polynomial time $O(n^3)$. The algorithm by Nagamochi & Ibaracki 1992 for graph cuts shown by Queyranne in 1995 to work for sym. SFM.
- The algorithm finds (as a subroutine) MA (maximum adjacency) or a maximum back orders (not same as greedy order).

Choose v_1 arbitrarily ;
$W_1 \leftarrow (v_1)$ /* The first of an ordered list W_i. */ ;
for $i \leftarrow 1 \dots |V| - 1$ **do**
  Choose $v_{i+1} \in \operatorname{argmin}_{u \in V \setminus W_i} f(W_i | \{u\})$;
  $W_{i+1} \leftarrow (W_i, v_{i+1})$; /* Append v_{i+1} to end of W_i */

- Note algorithm operates on non-symmetric function f. If f is already symmetric and normalized, then $f = \breve{f}$.
- The final ordered set $W_n = (v_1, v_2, \dots, v_n)$ is special in that the last two nodes (v_{n-1}, v_n) serve as a surrogate minimizer for a special case.

Pendent pair

- A ordered pair of elements (t, u) is called a pendent pair if u is a minimizer amongst all sets that separate u and t.

Pendent pair

- A ordered pair of elements (t, u) is called a pendent pair if u is a minimizer amongst all sets that separate u and t.
- That is (t, u) is a pendent pair if

$$\{u\} \in \operatorname*{argmin}_{A \subseteq V : u \in A, t \notin A} \breve{f}(A) \tag{19.6}$$

Pendent pair

- A ordered pair of elements (t, u) is called a pendent pair if u is a minimizer amongst all sets that separate u and t.
- That is (t, u) is a pendent pair if

$$\{u\} \in \operatorname*{argmin}_{A \subseteq V : u \in A, t \notin A} \breve{f}(A) \tag{19.6}$$

- That is,

$$\breve{f}(\{u\}) \leq \breve{f}(A) \;\; \forall A \text{ s.t. } t \notin A \ni u \tag{19.7}$$

Pendent pair

- A ordered pair of elements (t, u) is called a pendent pair if u is a minimizer amongst all sets that separate u and t.
- That is (t, u) is a pendent pair if

$$\{u\} \in \operatorname*{argmin}_{A \subseteq V : u \in A, t \notin A} \breve{f}(A) \tag{19.6}$$

- That is,

$$\breve{f}(\{u\}) \leq \breve{f}(A) \;\; \forall A \text{ s.t. } t \notin A \ni u \tag{19.7}$$

Theorem 19.3.2

In the ordered set $W = (v_1, \ldots, v_n)$ generated by the MA algorithm, then (v_{n-1}, v_n) is a pendent pair.

Pendent pair

- A ordered pair of elements (t, u) is called a pendent pair if u is a minimizer amongst all sets that separate u and t.
- That is (t, u) is a pendent pair if

$$\{u\} \in \operatorname*{argmin}_{A \subseteq V : u \in A, t \notin A} \breve{f}(A) \tag{19.6}$$

- That is,

$$\breve{f}(\{u\}) \leq \breve{f}(A) \;\; \forall A \text{ s.t. } t \notin A \ni u \tag{19.7}$$

Theorem 19.3.2

In the ordered set $W = (v_1, \ldots, v_n)$ generated by the MA algorithm, then (v_{n-1}, v_n) is a pendent pair.

- Interestingly, this algorithm is the same as maximum cardinality search (MCS), when f represents a graph cut function (recall, MCS is used to efficiently test graph chordality).

Minimization of a Symmetric Submodular Functions

- Now, given a pendent pair (t, u) there are two cases.

Minimization of a Symmetric Submodular Functions

- Now, given a pendent pair (t, u) there are two cases.
- Either: The global minimizer, say X^* of $\breve{f}$ is such that $t \notin X^* \ni u$ or we, by symmetry, can w.l.o.g. choose the minimizer so that both $\{t, u\} \in X^*$.

Minimization of a Symmetric Submodular Functions

- Now, given a pendent pair (t, u) there are two cases.
- Either: The global minimizer, say X^* of $\breve{f}$ is such that $t \notin X^* \ni u$ or we, by symmetry, can w.l.o.g. choose the minimizer so that both $\{t, u\} \in X^*$.
- We store the score (min value) in the first case,

Minimization of a Symmetric Submodular Functions

- Now, given a pendent pair (t, u) there are two cases.
- Either: The global minimizer, say X^* of $\breve{f}$ is such that $t \notin X^* \ni u$ or we, by symmetry, can w.l.o.g. choose the minimizer so that both $\{t, u\} \in X^*$.
- We store the score (min value) in the first case, then, consider a new element "tu" and clustered ground set $V' = V \setminus \{t, u\} \cup \{tu\}$, and new symmetric submodular function $f' : 2^{V'} \to \mathbb{R}$ with

$$\breve{f}'(X) = \begin{cases} \breve{f}(X) & \text{if } tu \notin X \\ \breve{f}(X \cup \{t, u\} \setminus \{tu\}) & \text{if } tu \in X \end{cases} \tag{19.8}$$

Minimization of a Symmetric Submodular Functions

- Now, given a pendent pair (t, u) there are two cases.
- Either: The global minimizer, say X^* of $\breve{f}$ is such that $t \notin X^* \ni u$ or we, by symmetry, can w.l.o.g. choose the minimizer so that both $\{t, u\} \in X^*$.
- We store the score (min value) in the first case, then, consider a new element "tu" and clustered ground set $V' = V \setminus \{t, u\} \cup \{tu\}$, and new symmetric submodular function $f' : 2^{V'} \to \mathbb{R}$ with

$$\breve{f}'(X) = \begin{cases} \breve{f}(X) & \text{if } tu \notin X \\ \breve{f}(X \cup \{t, u\} \setminus \{tu\}) & \text{if } tu \in X \end{cases} \tag{19.8}$$

- We then find a new pendent pair on f' using the above algorithm, store the new min value, and merge, and repeat.

Minimization of a Symmetric Submodular Functions

- Now, given a pendent pair (t, u) there are two cases.
- Either: The global minimizer, say X^* of $\breve{f}$ is such that $t \notin X^* \ni u$ or we, by symmetry, can w.l.o.g. choose the minimizer so that both $\{t, u\} \in X^*$.
- We store the score (min value) in the first case, then, consider a new element "tu" and clustered ground set $V' = V \setminus \{t, u\} \cup \{tu\}$, and new symmetric submodular function $f' : 2^{V'} \to \mathbb{R}$ with

$$\breve{f}'(X) = \begin{cases} \breve{f}(X) & \text{if } tu \notin X \\ \breve{f}(X \cup \{t, u\} \setminus \{tu\}) & \text{if } tu \in X \end{cases} \tag{19.8}$$

- We then find a new pendent pair on f' using the above algorithm, store the new min value, and merge, and repeat.
- We do this n times. We take the min over all of the stored values.

Minimization of a Symmetric Submodular Functions

- Now, given a pendent pair (t, u) there are two cases.
- Either: The global minimizer, say X^* of $\breve{f}$ is such that $t \notin X^* \ni u$ or we, by symmetry, can w.l.o.g. choose the minimizer so that both $\{t, u\} \in X^*$.
- We store the score (min value) in the first case, then, consider a new element "tu" and clustered ground set $V' = V \setminus \{t, u\} \cup \{tu\}$, and new symmetric submodular function $f' : 2^{V'} \to \mathbb{R}$ with

$$\breve{f}'(X) = \begin{cases} \breve{f}(X) & \text{if } tu \notin X \\ \breve{f}(X \cup \{t, u\} \setminus \{tu\}) & \text{if } tu \in X \end{cases} \tag{19.8}$$

- We then find a new pendent pair on f' using the above algorithm, store the new min value, and merge, and repeat.
- We do this n times. We take the min over all of the stored values.
- The pendent pair corresponding to the min element, say (t', u') will (most probability) correspond to nested clusters, so we use the original ground elements corresponding to u'.

Minimization of a Symmetric Submodular Functions

Theorem 19.3.3

The final resultant u' when expanded to original ground elements minimizes the symmetric submodular function f in $O(n^3)$ time.

- This has become known as Queyranne's algorithm for symmetric submodular function minimization.
- This was done in 1995 and it is said that this result, at that time, rekindled the efforts to find general combinatorial SFM.
- The actual algorithm was originally developed by Nagamochi and Ibaraki for a simple algorithm for finding graph cut. Queyranne showed it worked for any symmetric submodular function.
- Hence, it seems reasonable that symmetric SFM is faster than general SFM (although this question is still unknown).
- Quoting Fujishige from NIPS 2012, he said that he "hopes general purpose SFM is $O(n^4)$" ☺.

Maximization of Submodular Functions

- We spent much time on submodular function minimization (SFM) and saw this can be done in polynomial time.

Maximization of Submodular Functions

- We spent much time on submodular function minimization (SFM) and saw this can be done in polynomial time.
- Submodular maximization is also quite useful.

Maximization of Submodular Functions

- We spent much time on submodular function minimization (SFM) and saw this can be done in polynomial time.
- Submodular maximization is also 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

- We spent much time on submodular function minimization (SFM) and saw this can be done in polynomial time.
- Submodular maximization is also 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

- We spent much time on submodular function minimization (SFM) and saw this can be done in polynomial time.
- Submodular maximization is also 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 either

Maximization of Submodular Functions

- We spent much time on submodular function minimization (SFM) and saw this can be done in polynomial time.
- Submodular maximization is also 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 either
  - Find the maximum under some constraint

Maximization of Submodular Functions

- We spent much time on submodular function minimization (SFM) and saw this can be done in polynomial time.
- Submodular maximization is also 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 either
  - Find the maximum under some constraint
  - Find the maximum for a non-polymatroid submodular function

Maximization of Submodular Functions

- We spent much time on submodular function minimization (SFM) and saw this can be done in polynomial time.
- Submodular maximization is also 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 either
  - Find the maximum under some constraint
  - Find the maximum for a non-polymatroid submodular function
  - Do both.

Maximization of Submodular Functions

- We spent much time on submodular function minimization (SFM) and saw this can be done in polynomial time.
- Submodular maximization is also 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 either
  - Find the maximum under some constraint
  - Find the maximum for a non-polymatroid submodular function
  - Do both.
- 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 $E_1, E_2, \ldots, E_m$ 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.,

$$\text{minimize} |X| \text{ subject to } f(X) \geq |E| \tag{19.9}$$

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.,

$$\text{minimize} |X| \text{ subject to } f(X) \geq |E| \tag{19.9}$$

- 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.,

$$\text{minimize} |X| \text{ subject to } f(X) \geq |E| \tag{19.9}$$

- 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:

$$\max f(X) \text{ subject to } |X| \leq k \tag{19.10}$$

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:

$$\max f(X) \text{ subject to } |X| \leq k \tag{19.10}$$

- 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 $i = 0 \ldots (k-1)$:

$$S_{i+1} = S_i \cup \left\{ \operatorname*{argmax}_{v \in V \setminus S_i} f(S_i \cup \{v\}) \right\} \tag{19.11}$$

The Greedy Algorithm for Submodular Max

A bit more precisely:

Algorithm 2: The Greedy Algorithm

Set $S_0 \leftarrow \emptyset$;
for $i \leftarrow 0 \dots |E| - 1$ **do**
  Choose v_i as follows:
  $v_i \in \left\{ \text{argmax}_{v \in V \setminus S_i} f(\{v\}|S_i) \right\} = \left\{ \text{argmax}_{v \in V \setminus S_i} f(S_i \cup \{v\}) \right\}$;
  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 19.4.1

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 19.4.1

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 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 19.4.1

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 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 $\epsilon > 0$.