Fast Semi-differential based Submodular Function Optimization

Rishabh Iyer [1] Stefanie Jegelka [2] Jeff Bilmes [1]

[1]University of Washington, Seattle

[2]University of California, Berkeley

ICML-2013

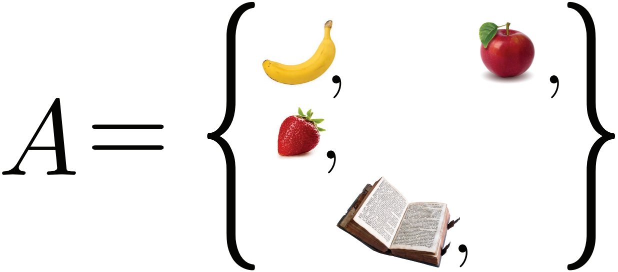
Outline

Set functions $f : 2^V \to \mathbb{R}$

$V =$

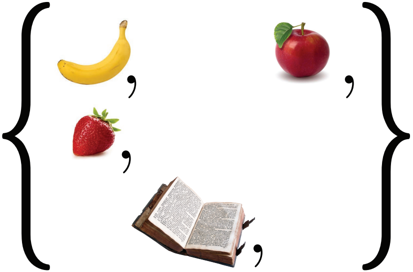
- V is a finite "ground" set of objects.

Set functions $f : 2^V \to \mathbb{R}$



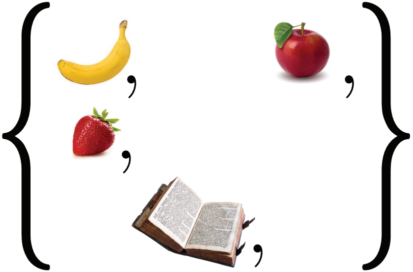
- A set function $f : 2^V \to \mathbb{R}$ produces a value for any subset $A \subseteq V$.

Set functions $f : 2^V \to \mathbb{R}$

$A =$ 

- A set function $f : 2^V \to \mathbb{R}$ produces a value for any subset $A \subseteq V$.
- For example, $f(A) = 22$,

Set functions $f : 2^V \to \mathbb{R}$

$A =$ 

- A set function $f : 2^V \to \mathbb{R}$ produces a value for any subset $A \subseteq V$.
- For example, $f(A) = 22$,
- General set function optimization can be really hard!

Submodular Set Functions

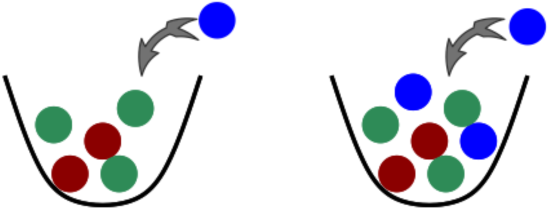
- Special class of set functions.

$$f(A \cup v) - f(A) \geq f(B \cup v) - f(B), \text{ if } A \subseteq B \tag{1}$$

Submodular Set Functions

- Special class of set functions.

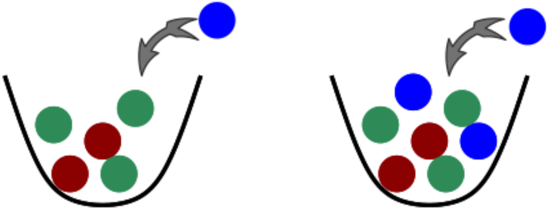
$$f(A \cup v) - f(A) \geq f(B \cup v) - f(B), \text{ if } A \subseteq B \quad (1)$$



Submodular Set Functions

- Special class of set functions.

$$f(A \cup v) - f(A) \geq f(B \cup v) - f(B), \text{ if } A \subseteq B \quad (1)$$

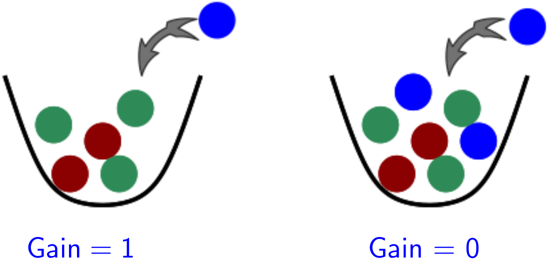


Gain = 1

Submodular Set Functions

- Special class of set functions.

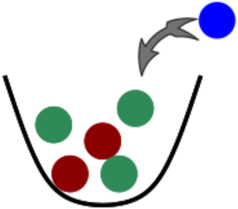
$$f(A \cup v) - f(A) \geq f(B \cup v) - f(B), \text{ if } A \subseteq B \quad (1)$$

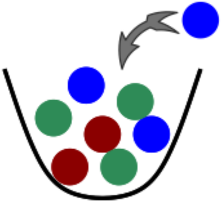


Submodular Set Functions

- Special class of set functions.

$$f(A \cup v) - f(A) \geq f(B \cup v) - f(B), \text{ if } A \subseteq B \quad (1)$$





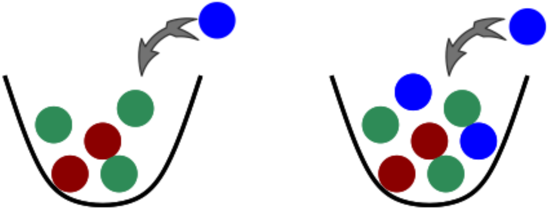
Gain $= 1$ Gain $= 0$

- Monotonicity: $f(A) \leq f(B)$, if $A \subseteq B$.

Submodular Set Functions

- Special class of set functions.

$$f(A \cup v) - f(A) \geq f(B \cup v) - f(B), \text{ if } A \subseteq B \tag{1}$$



Gain $= 1$ Gain $= 0$

- Monotonicity: $f(A) \leq f(B)$, if $A \subseteq B$.
- Modular function $f(X) = \sum_{i \in X} f(i)$ analogous to linear functions.

Submodular Function Maximization

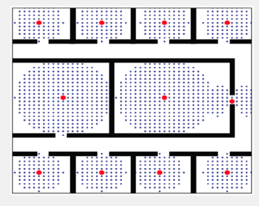
$$\text{compute } A^* \in \operatorname*{argmax}_{A \in \mathcal{C}} f(A)$$

where f is submodular, and where $\mathcal{C}$ is constraint set over which a modular function can be optimized efficiently.

Submodular Function Maximization

$$\text{compute } A^* \in \underset{A \in \mathcal{C}}{\operatorname{argmax}} f(A)$$

where f is submodular, and where $\mathcal{C}$ is constraint set over which a modular function can be optimized efficiently.

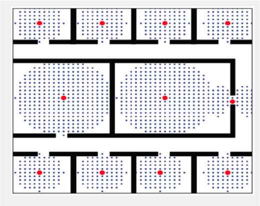


Sensor Placement
(Krause et al, 2008)

Submodular Function Maximization

$$\text{compute } A^* \in \operatorname*{argmax}_{A \in \mathcal{C}} f(A)$$

where f is submodular, and where $\mathcal{C}$ is constraint set over which a modular function can be optimized efficiently.



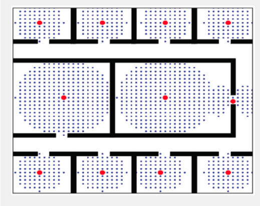
Sensor Placement
(Krause et al, 2008)

Document Summarization
(Lin & Bilmes, 2011)

Submodular Function Maximization

$$\text{compute } A^* \in \underset{A \in \mathcal{C}}{\operatorname{argmax}} f(A)$$

where f is submodular, and where $\mathcal{C}$ is constraint set over which a modular function can be optimized efficiently.



Sensor Placement (Krause et al, 2008)

Document Summarization (Lin & Bilmes, 2011)

Diversified Search (He et al 2012, Kulesza & Taskar, 2012)

Submodular Function Minimization

$$\text{compute } A^* \in \operatorname*{argmin}_{A \in \mathcal{C}} f(A)$$

where f is submodular, and where $\mathcal{C}$ is constraint set over which a modular function can be optimized efficiently.

Submodular Function Minimization

$$\text{compute } A^* \in \operatorname*{argmin}_{A \in \mathcal{C}} f(A)$$

where f is submodular, and where $\mathcal{C}$ is constraint set over which a modular function can be optimized efficiently.

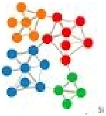
Image segmentation / MAP inference (Boykov & Jolly 2001, Jegelka & Bilmes 2011, Delong et al, 2012)

Submodular Function Minimization

$$\text{compute } A^* \in \underset{A \in \mathcal{C}}{\operatorname{argmin}} f(A)$$

where f is submodular, and where $\mathcal{C}$ is constraint set over which a modular function can be optimized efficiently.

Image segmentation / MAP inference (Boykov & Jolly 2001, Jegelka & Bilmes 2011, Delong et al, 2012)



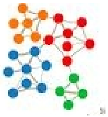
Clustering (Narasimhan & Bilmes 2011, Nagano et al, 2010)

Submodular Function Minimization

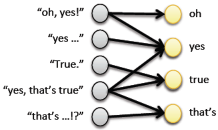
$$\text{compute } A^* \in \operatorname*{argmin}_{A \in \mathcal{C}} f(A)$$

where f is submodular, and where $\mathcal{C}$ is constraint set over which a modular function can be optimized efficiently.

Image segmentation / MAP inference (Boykov & Jolly 2001, Jegelka & Bilmes 2011, Delong et al, 2012)



Clustering (Narasimhan & Bilmes 2011, Nagano et al, 2010)



Corpus Data Subset Selection (Lin & Bilmes, 2011)

Current State of Affairs for Submodular Optimization

Current State of Affairs for Submodular Optimization

Submodular Function Minimization

- Polynomial-time but too slow $O(n^5 \times \text{FuncEvalCost} + n^6)$.
- Constrained minimization is NP-hard.
- Algorithms differ depending on the constraints.

Current State of Affairs for Submodular Optimization

Submodular Function Minimization

- Polynomial-time but too slow $O(n^5 \times \text{FuncEvalCost} + n^6)$.
- Constrained minimization is NP-hard.
- Algorithms differ depending on the constraints.

Submodular Function Maximization

- NP-hard but constant-factor approximable.
- Large class of algorithms – Local search, continuous greedy, bi-directional greedy, simulated annealing etc.

Current State of Affairs for Submodular Optimization

Submodular Function Minimization

- Polynomial-time but too slow $O(n^5 \times \text{FuncEvalCost} + n^6)$.
- Constrained minimization is NP-hard.
- Algorithms differ depending on the constraints.

Submodular Function Maximization

- NP-hard but constant-factor approximable.
- Large class of algorithms – Local search, continuous greedy, bi-directional greedy, simulated annealing etc.

Algorithms look very different!

Current State of Affairs for Submodular Optimization

Submodular Function Minimization

- Polynomial-time but too slow $O(n^5 \times \text{FuncEvalCost} + n^6)$.
- Constrained minimization is NP-hard.
- Algorithms differ depending on the constraints.

Submodular Function Maximization

- NP-hard but constant-factor approximable.
- Large class of algorithms – Local search, continuous greedy, bi-directional greedy, simulated annealing etc.

Algorithms look very different!

Which algorithm to use when?

Current State of Affairs for Submodular Optimization

Submodular Function Minimization

- Polynomial-time but too slow $O(n^5 \times \text{FuncEvalCost} + n^6)$.
- Constrained minimization is NP-hard.
- Algorithms differ depending on the constraints.

Submodular Function Maximization

- NP-hard but constant-factor approximable.
- Large class of algorithms – Local search, continuous greedy, bi-directional greedy, simulated annealing etc.

Algorithms look very different!

Which algorithm to use when?

Contribution: We present the first unifying framework for submodular minimization & maximization. Our framework is scalable to large data.

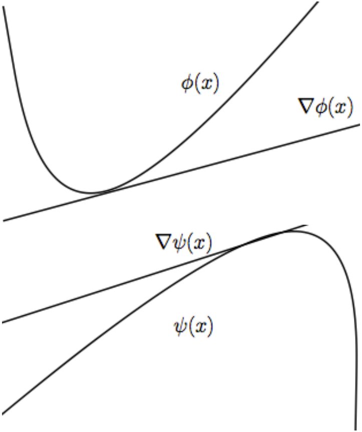
Convex/Concave and Semigradients

- A convex function ϕ has a subgradient h_y and linear lower bound:

$$\phi(y) + \langle h_y, x - y \rangle \leq \phi(x), \forall x.$$

- A concave function ψ has a supergradient g_y and linear upper bound:

$$\psi(y) + \langle g_y, x - y \rangle \geq \psi(x), \forall x.$$



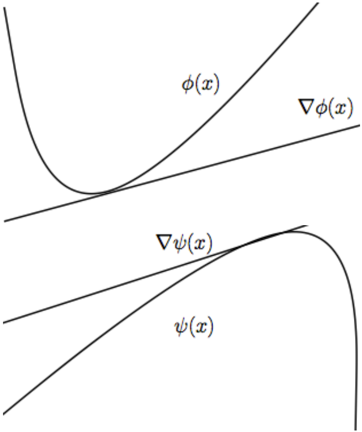
Convex/Concave and Semigradients

- A convex function ϕ has a subgradient h_y and linear lower bound:

$$\phi(y) + \langle h_y, x - y \rangle \leq \phi(x), \forall x.$$

- A concave function ψ has a supergradient g_y and linear upper bound:

$$\psi(y) + \langle g_y, x - y \rangle \geq \psi(x), \forall x.$$



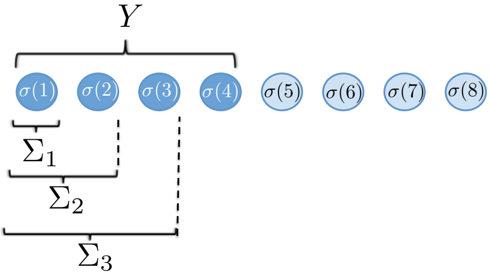
- Submodular functions have properties analogous to convexity and concavity.

Submodular Subgradients (Fujishige 1984, 2005)

- Like convex functions, submodular functions have sub-gradients. Defined at any $Y \subseteq V$.

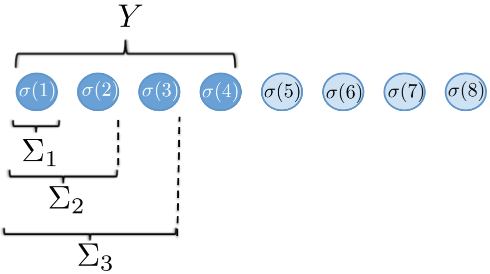
Submodular Subgradients (Fujishige 1984, 2005)

- Like convex functions, submodular functions have sub-gradients. Defined at any $Y \subseteq V$.
- Permutation σ of the ground set.



Submodular Subgradients (Fujishige 1984, 2005)

- Like convex functions, submodular functions have sub-gradients. Defined at any $Y \subseteq V$.
- Permutation σ of the ground set.

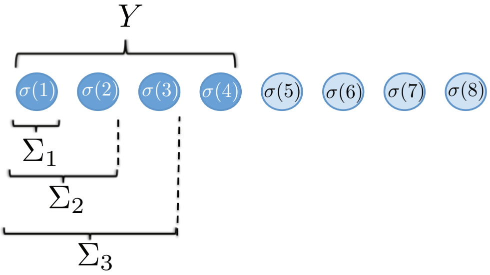


- Corresponding subgradient h_Y^σ is:

$$h_Y^\sigma(\sigma(i)) = f(\Sigma_i) - f(\Sigma_{i-1})$$

Submodular Subgradients (Fujishige 1984, 2005)

- Like convex functions, submodular functions have sub-gradients. Defined at any $Y \subseteq V$.
- Permutation σ of the ground set.



- Corresponding subgradient h_Y^σ is:

$$h_Y^\sigma(\sigma(i)) = f(\Sigma_i) - f(\Sigma_{i-1})$$

- Modular lower bound: $m_{h_Y}(X) = f(Y) + h_Y(X) - h_Y(Y) \ \leq f(X)$.

Submodular Supergradients (Iyer et al, 2013)

- Define gain of j in context of A: $f(j|A) \triangleq f(A \cup j) - f(A)$

Submodular Supergradients (Iyer et al, 2013)

- Define gain of j in context of A: $f(j|A) \triangleq f(A \cup j) - f(A)$
- Unlike convex functions, surprisingly, we show that submodular functions also have super-gradients. Defined at any $Y \subseteq V$.

Submodular Supergradients (Iyer et al, 2013)

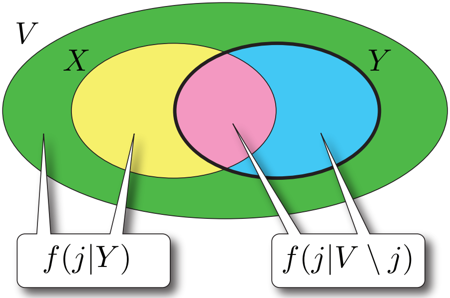
- Define gain of j in context of A: $f(j|A) \triangleq f(A \cup j) - f(A)$
- Unlike convex functions, surprisingly, we show that submodular functions also have super-gradients. Defined at any $Y \subseteq V$.
- Three of these supergradients (which we call grow, shrink, and bar) are in fact easy to obtain.

Submodular Supergradients (Iyer et al, 2013)

- Define gain of j in context of A: $f(j|A) \triangleq f(A \cup j) - f(A)$
- Unlike convex functions, surprisingly, we show that submodular functions also have super-gradients. Defined at any $Y \subseteq V$.
- Three of these supergradients (which we call grow, shrink, and bar) are in fact easy to obtain.

Grow:

$$\hat{g}_Y(j) = \begin{cases} f(j|Y) & \text{for } j \notin Y \\ f(j|V \backslash \{j\}) & \text{for } j \in Y \end{cases}$$

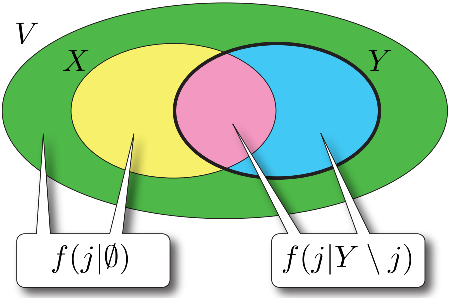


Submodular Supergradients (Iyer et al, 2013)

- Define gain of j in context of A: $f(j|A) \triangleq f(A \cup j) - f(A)$
- Unlike convex functions, surprisingly, we show that submodular functions also have super-gradients. Defined at any $Y \subseteq V$.
- Three of these supergradients (which we call grow, shrink, and bar) are in fact easy to obtain.

Shrink:

$$\breve{g}_Y(j) = \begin{cases} f(j|\emptyset) & \text{for } j \notin Y \\ f(j|Y \setminus \{j\}) & \text{for } j \in Y \end{cases}$$

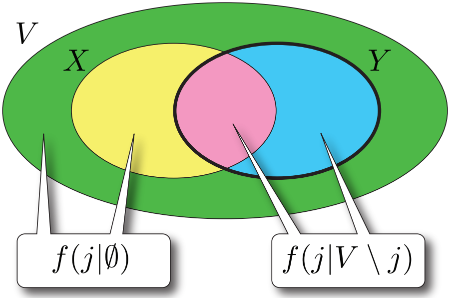


Submodular Supergradients (Iyer et al, 2013)

- Define gain of j in context of A: $f(j|A) \triangleq f(A \cup j) - f(A)$
- Unlike convex functions, surprisingly, we show that submodular functions also have super-gradients. Defined at any $Y \subseteq V$.
- Three of these supergradients (which we call grow, shrink, and bar) are in fact easy to obtain.

Bar:

$$\bar{g}_Y(j) = \begin{cases} f(j|\emptyset) & \text{for } j \notin Y \\ f(j|V\backslash\{j\}) & \text{for } j \in Y \end{cases}$$

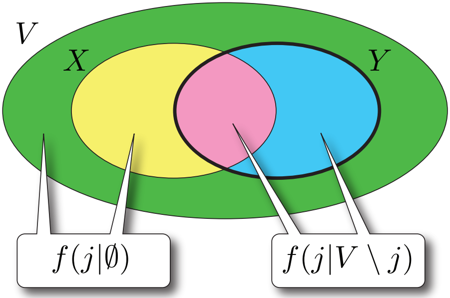


Submodular Supergradients (Iyer et al, 2013)

- Define gain of j in context of A: $f(j|A) \triangleq f(A \cup j) - f(A)$
- Unlike convex functions, surprisingly, we show that submodular functions also have super-gradients. Defined at any $Y \subseteq V$.
- Three of these supergradients (which we call grow, shrink, and bar) are in fact easy to obtain.

Bar:

$$\bar{g}_Y(j) = \begin{cases} f(j|\emptyset) & \text{for } j \notin Y \\ f(j|V\backslash\{j\}) & \text{for } j \in Y \end{cases}$$



- Modular upper bound: $m^{g_Y}(X) = f(Y) + g_Y(X) - g_Y(Y) \leq f(X)$.

Optimization Framework

Algorithm 1 Subgradient ascent [descent] algorithm for submodular maximization [minimization].

1: Start with an arbitrary X^0.

Optimization Framework

Algorithm 1 Subgradient ascent [descent] algorithm for submodular maximization [minimization].

1: Start with an arbitrary X^0.
2: **repeat**

6: **until** we have converged ($X^{i-1} = X^i$) or $i \leq T$

Optimization Framework

Algorithm 1 Subgradient ascent [descent] algorithm for submodular maximization [minimization].

1: Start with an arbitrary X^0.
2: **repeat**
3: Pick a semigradient h_{X^t} [g_{X^t}] at X^t.
6: **until** we have converged ($X^{i-1} = X^i$) or $i \leq T$

Optimization Framework

Algorithm 1 Subgradient ascent [descent] algorithm for submodular maximization [minimization].

1: Start with an arbitrary X^0.
2: **repeat**
3: Pick a semigradient h_{X^t} [g_{X^t}] at X^t.
4: $X^{t+1} \leftarrow \text{argmax}_{X \in \mathcal{C}}\, m_{h_{X^t}}(X)$ [$X^{t+1} \leftarrow \text{argmin}_{X \in \mathcal{C}}\, m^{g_{X^t}}(X)$]
5: $t \leftarrow t + 1$
6: **until** we have converged ($X^{i-1} = X^i$) or $i \leq T$

Optimization Framework

Algorithm 1 Subgradient ascent [descent] algorithm for submodular maximization [minimization].

1: Start with an arbitrary X^0.
2: **repeat**
3: Pick a semigradient h_{X^t} [g_{X^t}] at X^t.
4: $X^{t+1} \leftarrow \text{argmax}_{X \in \mathcal{C}} m_{h_{X^t}}(X)$ [$X^{t+1} \leftarrow \text{argmin}_{X \in \mathcal{C}} m^{g_{X^t}}(X)$]
5: $t \leftarrow t + 1$
6: **until** we have converged ($X^{i-1} = X^i$) or $i \leq T$

Lemma: Algorithm 1 monotonically improves the objective function value for submodular maximization and minimization at every iteration.

Unconstrained Minimization

	MMin-IIIa	MMin-IIIb	MMin-I	MMin-II
g	$\bar{g}$	$\bar{g}$	$\hat{g}$	$\check{g}$
X^0	$\emptyset$	V	$\emptyset$	V
X^c	A	B	A_+	B_+

- MMin-IIIa and IIIb are first iterations of MMin-I and MMin-II.

Unconstrained Minimization

	MMin-IIIa	MMin-IIIb	MMin-I	MMin-II
g	$\bar{g}$	$\bar{g}$	$\hat{g}$	$\check{g}$
X^0	$\emptyset$	V	$\emptyset$	V
X^c	A	B	A_+	B_+

- MMin-IIIa and IIIb are first iterations of MMin-I and MMin-II.
- A and B obtainable in $O(n)$ oracle calls.

Unconstrained Minimization

	MMin-IIIa	MMin-IIIb	MMin-I	MMin-II
g	$\bar{g}$	$\bar{g}$	$\hat{g}$	$\check{g}$
X^0	$\emptyset$	V	$\emptyset$	V
X^c	A	B	A_+	B_+

- MMin-IIIa and IIIb are first iterations of MMin-I and MMin-II.
- A and B obtainable in $O(n)$ oracle calls.
- A_+ and B_+ are local minimizers obtainable in $O(n^2)$ calls.

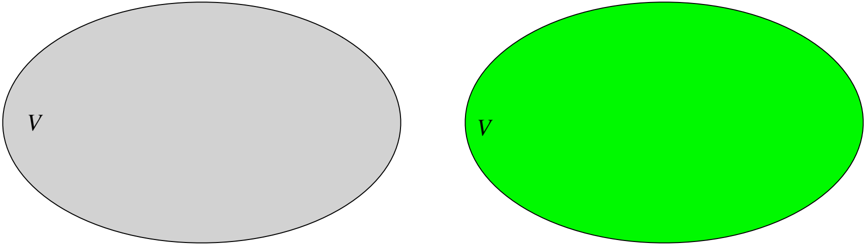
Unconstrained Minimization

	MMin-IIIa	MMin-IIIb	MMin-I	MMin-II
g	$\bar{g}$	$\bar{g}$	$\hat{g}$	$\check{g}$
X^0	$\emptyset$	V	$\emptyset$	V
X^c	A	B	A_+	B_+

- MMin-IIIa and IIIb are first iterations of MMin-I and MMin-II.
- A and B obtainable in $O(n)$ oracle calls.
- A_+ and B_+ are local minimizers obtainable in $O(n^2)$ calls.

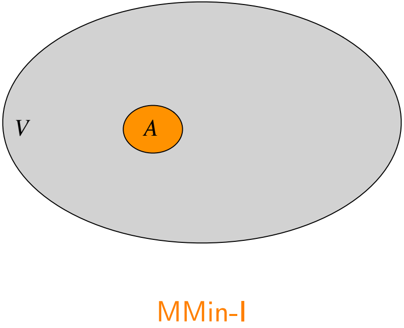
$$A \subseteq A_+ \subseteq X^* \subseteq B_+ \subseteq B$$

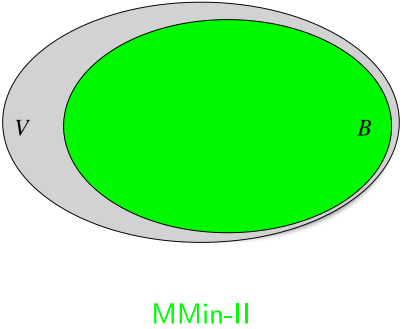
Illustrating Unconstrained Minimization



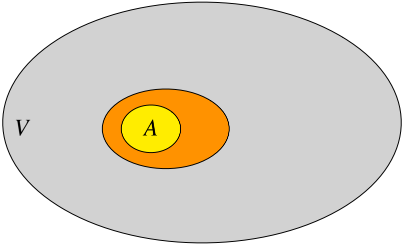
MMin-I   MMin-II

Illustrating Unconstrained Minimization

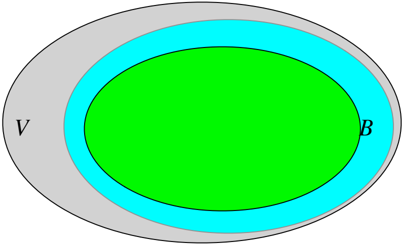




Illustrating Unconstrained Minimization

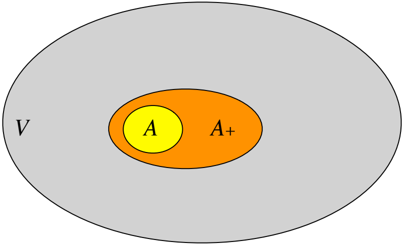


MMin-I

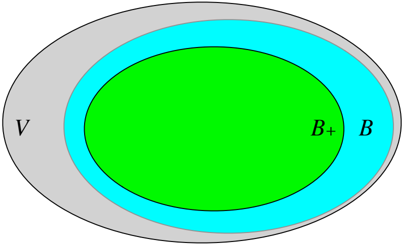


MMin-II

Illustrating Unconstrained Minimization

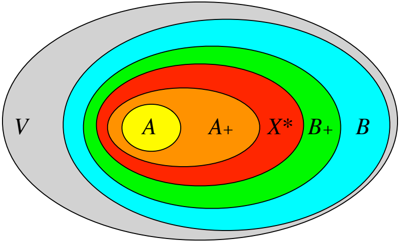


MMin-I

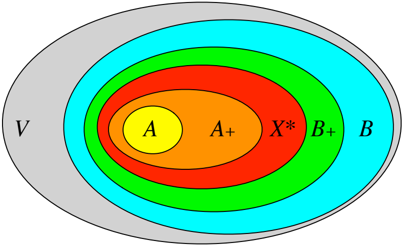


MMin-II

Illustrating Unconstrained Minimization



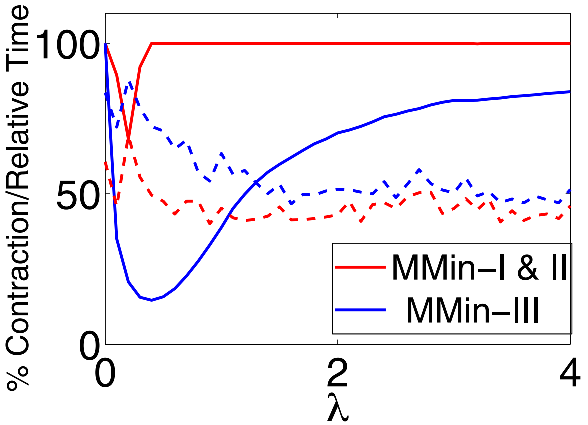
MMin-I



MMin-II

Empirical Results: Submodular Minimization

Test function: concave over modular, $\sqrt{w_1(X)} + \lambda w_2(V \backslash X)$.



Lattice reduction (solid line), and runtime reduction (dotted line).

Note: results for Bipartite Neighborhoods shown in paper.

Constrained Submodular Minimization

- Curvature of a monotone submodular function:

$$\kappa_f(X) \triangleq 1 - \min_j \frac{f(j|X \backslash j)}{f(j)}. \tag{2}$$

Constrained Submodular Minimization

- Curvature of a monotone submodular function:

$$\kappa_f(X) \triangleq 1 - \min_j \frac{f(j|X \backslash j)}{f(j)}. \quad (2)$$

Theorem

The solution $\widehat{X}$ returned by MMin-I satisfies:

$$f(\widehat{X}) \leq \frac{|X^*|}{1 + (|X^*| - 1)(1 - \kappa_f(X^*))} f(X^*) \leq \frac{1}{1 - \kappa_f(X^*)} f(X^*)$$

Constrained Submodular Minimization

- Curvature of a monotone submodular function:

$$\kappa_f(X) \triangleq 1 - \min_j \frac{f(j|X \backslash j)}{f(j)}. \tag{2}$$

Theorem

The solution $\widehat{X}$ returned by MMin-I satisfies:

$$f(\widehat{X}) \leq \frac{|X^*|}{1 + (|X^*| - 1)(1 - \kappa_f(X^*))} f(X^*) \leq \frac{1}{1 - \kappa_f(X^*)} f(X^*)$$

- Lower curvature $\Rightarrow$ Better guarantees!

Constrained Submodular Minimization

- Curvature of a monotone submodular function:

$$\kappa_f(X) \triangleq 1 - \min_j \frac{f(j|X\backslash j)}{f(j)}. \quad (2)$$

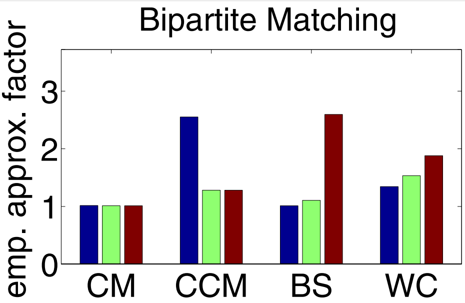
Theorem

The solution $\widehat{X}$ returned by MMin-I satisfies:

$$f(\widehat{X}) \leq \frac{|X^*|}{1 + (|X^*| - 1)(1 - \kappa_f(X^*))} f(X^*) \leq \frac{1}{1 - \kappa_f(X^*)} f(X^*)$$

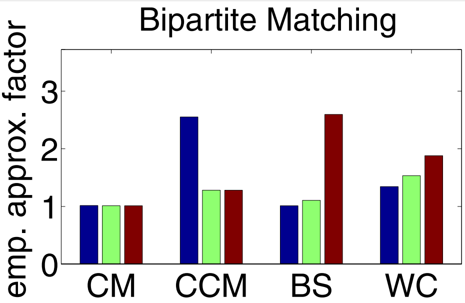
- Lower curvature $\Rightarrow$ Better guarantees!
- Improve the previous results when $\kappa_f < 1$.

Empirical Results: Constrained Submodular Minimization



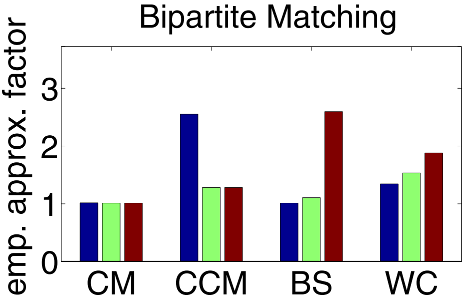
- We compare MMin-I to two other algorithms.

Empirical Results: Constrained Submodular Minimization



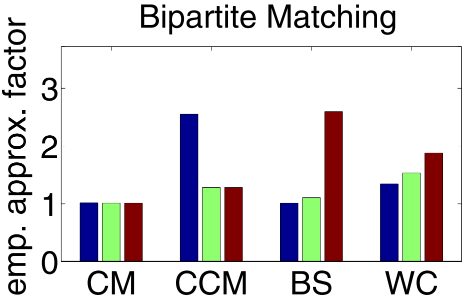
- We compare MMin-I to two other algorithms.
  1. Simple modular upper bound (MU) (i.e $\sum_{j \in X} f(j)$).

Empirical Results: Constrained Submodular Minimization



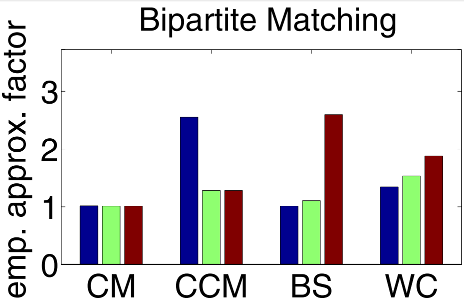
- We compare MMin-I to two other algorithms.
  1. Simple modular upper bound (MU) (i.e $\sum_{j \in X} f(j)$).
  2. More complicated Ellipsoidal Approximation (EA) Algorithm.

Empirical Results: Constrained Submodular Minimization



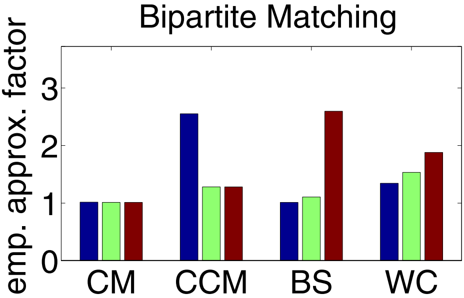
- We compare MMin-I to two other algorithms.
  1. Simple modular upper bound (MU) (i.e $\sum_{j \in X} f(j)$).
  2. More complicated Ellipsoidal Approximation (EA) Algorithm.
- Performance of MMin-I:

Empirical Results: Constrained Submodular Minimization



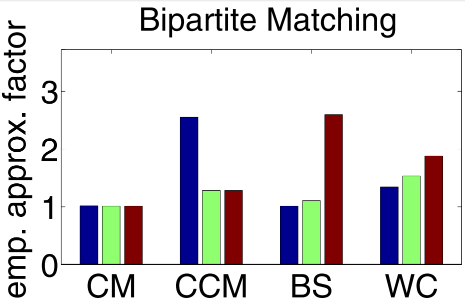
- We compare MMin-I to two other algorithms.
  1. Simple modular upper bound (MU) (i.e $\sum_{j \in X} f(j)$).
  2. More complicated Ellipsoidal Approximation (EA) Algorithm.
- Performance of MMin-I:
  1. Much better than MU.

Empirical Results: Constrained Submodular Minimization



- We compare MMin-I to two other algorithms.
  1. Simple modular upper bound (MU) (i.e $\sum_{j \in X} f(j)$).
  2. More complicated Ellipsoidal Approximation (EA) Algorithm.
- Performance of MMin-I:
  1. Much better than MU.
  2. Comparable to EA.

Empirical Results: Constrained Submodular Minimization



- We compare MMin-I to two other algorithms.
  1. Simple modular upper bound (MU) (i.e $\sum_{j \in X} f(j)$).
  2. More complicated Ellipsoidal Approximation (EA) Algorithm.
- Performance of MMin-I:
  1. Much better than MU.
  2. Comparable to EA.
- Submodular spanning tree & shortest path results given in paper.

Unconstrained Maximization

Our framework subsumes a number of state-of-the-art algorithms. For example, each of the below corresponds to subgradient ascent:

Unconstrained Maximization

Our framework subsumes a number of state-of-the-art algorithms. For example, each of the below corresponds to subgradient ascent:

- **Random Subgradient (RA/ RP):** Random subgradients (permutations) at every iteration.

Unconstrained Maximization

Our framework subsumes a number of state-of-the-art algorithms. For example, each of the below corresponds to subgradient ascent:

- **Random Subgradient (RA/ RP):** Random subgradients (permutations) at every iteration.

1/4 Approximation in Expectation!

Unconstrained Maximization

Our framework subsumes a number of state-of-the-art algorithms. For example, each of the below corresponds to subgradient ascent:

- **Random Subgradient (RA/ RP):** Random subgradients (permutations) at every iteration.

  1/4 Approximation in Expectation!

- **Randomized / Deterministic local search (RLS/DLS):** Local search based techniques naturally define subgradients.

Unconstrained Maximization

Our framework subsumes a number of state-of-the-art algorithms. For example, each of the below corresponds to subgradient ascent:

- **Random Subgradient (RA/ RP):** Random subgradients (permutations) at every iteration.

  1/4 Approximation in Expectation!

- **Randomized / Deterministic local search (RLS/DLS):** Local search based techniques naturally define subgradients.

  1/3 Approximation (FMV'07)!

Unconstrained Maximization

Our framework subsumes a number of state-of-the-art algorithms. For example, each of the below corresponds to subgradient ascent:

- **Random Subgradient (RA/ RP):** Random subgradients (permutations) at every iteration.

  1/4 Approximation in Expectation!

- **Randomized / Deterministic local search (RLS/DLS):** Local search based techniques naturally define subgradients.

  1/3 Approximation (FMV'07)!

- **Bi-directional Greedy (BG):** Bi-directional Greedy Subgradient (Buchbinder et al, 2012).

Unconstrained Maximization

Our framework subsumes a number of state-of-the-art algorithms. For example, each of the below corresponds to subgradient ascent:

- **Random Subgradient (RA/ RP):** Random subgradients (permutations) at every iteration.

  1/4 Approximation in Expectation!

- **Randomized / Deterministic local search (RLS/DLS):** Local search based techniques naturally define subgradients.

  1/3 Approximation (FMV'07)!

- **Bi-directional Greedy (BG):** Bi-directional Greedy Subgradient (Buchbinder et al, 2012).

  1/3 Approximation (BFNS'12)!

Unconstrained Maximization

Our framework subsumes a number of state-of-the-art algorithms. For example, each of the below corresponds to subgradient ascent:

- **Random Subgradient (RA/ RP):** Random subgradients (permutations) at every iteration.

  1/4 Approximation in Expectation!

- **Randomized / Deterministic local search (RLS/DLS):** Local search based techniques naturally define subgradients.

  1/3 Approximation (FMV'07)!

- **Bi-directional Greedy (BG):** Bi-directional Greedy Subgradient (Buchbinder et al, 2012).

  1/3 Approximation (BFNS'12)!

- **Randomized Greedy (RG):** Randomized variant of BG.

Unconstrained Maximization

Our framework subsumes a number of state-of-the-art algorithms. For example, each of the below corresponds to subgradient ascent:

- **Random Subgradient (RA/ RP):** Random subgradients (permutations) at every iteration.

  1/4 Approximation in Expectation!

- **Randomized / Deterministic local search (RLS/DLS):** Local search based techniques naturally define subgradients.

  1/3 Approximation (FMV'07)!

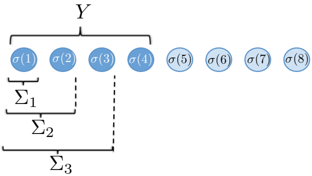
- **Bi-directional Greedy (BG):** Bi-directional Greedy Subgradient (Buchbinder et al, 2012).

  1/3 Approximation (BFNS'12)!

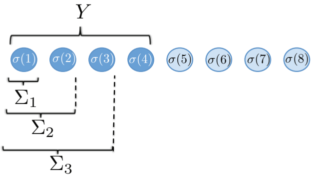
- **Randomized Greedy (RG):** Randomized variant of BG.

  1/2 Approximation in Expectation! (BFNS'12)!

Constrained Maximization and Extensions



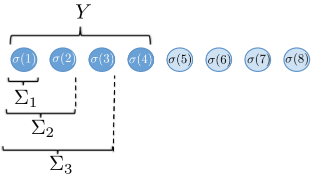
Constrained Maximization and Extensions



- Greedy subgradient for monotone submodular functions:

$$\sigma^g(i) \in \operatorname*{argmax}_{j \notin \Sigma^{\sigma^g}_{i-1} \text{ and } \Sigma^{\sigma^g}_{i-1} \cup \{j\} \in \mathcal{C}} f(j | \Sigma^{\sigma^g}_{i-1}). \tag{3}$$

Constrained Maximization and Extensions

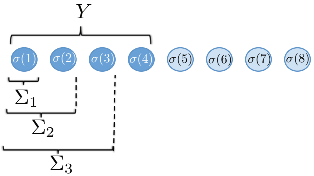


- Greedy subgradient for monotone submodular functions:

$$\sigma^g(i) \in \underset{j \notin \Sigma^{\sigma^g}_{i-1} \text{ and } \Sigma^{\sigma^g}_{i-1} \cup \{j\} \in \mathcal{C}}{\operatorname{argmax}} f(j | \Sigma^{\sigma^g}_{i-1}). \tag{3}$$

- Algorithm 1 using the subgradient h^{σ^g} exactly corresponds to the greedy algorithm.

Constrained Maximization and Extensions

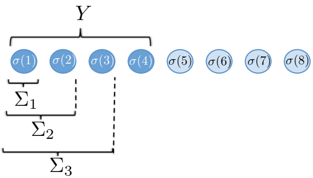


- Greedy subgradient for monotone submodular functions:

$$\sigma^g(i) \in \operatorname*{argmax}_{j \notin \Sigma^{\sigma^g}_{i-1} \text{ and } \Sigma^{\sigma^g}_{i-1} \cup \{j\} \in \mathcal{C}} f(j | \Sigma^{\sigma^g}_{i-1}). \quad (3)$$

- Algorithm 1 using the subgradient h^{σ^g} exactly corresponds to the greedy algorithm. $\Rightarrow$ $1 - 1/e$ Approximation (NWF'78)!

Constrained Maximization and Extensions



- Greedy subgradient for monotone submodular functions:

$$\sigma^g(i) \in \underset{j \notin \Sigma_{i-1}^{\sigma^g} \text{ and } \Sigma_{i-1}^{\sigma^g} \cup \{j\} \in \mathcal{C}}{\text{argmax}} f(j | \Sigma_{i-1}^{\sigma^g}). \quad (3)$$

- Algorithm 1 using the subgradient h^{σ^g} exactly corresponds to the greedy algorithm. $\Rightarrow$ $1 - 1/e$ Approximation (NWF'78)!

Generality of Algorithm MMax: For every α-approximation algorithm, there exists a schedule of subgradients obtainable in poly-time, such that Algorithm 1 (MMax) achieves an approximation factor of at least α.

Summary

- Submodular functions in machine learning.
- A generic sub-gradient ascent [super-gradient descent] framework for submodular maximization [minimization].
- The first unifying framework for general submodular optimization.
- New theoretical results for unconstrained and constrained submodular minimization.
- A novel view as a framework for submodular maximization and subsuming number of existing algorithms.
- Empirical experimental validation.

Thank You

Thank You!
Questions please.