Fast Semi-differential based Submodular function Optimization

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

[1]University of Washington, Seattle; [2]University of California, Berkeley

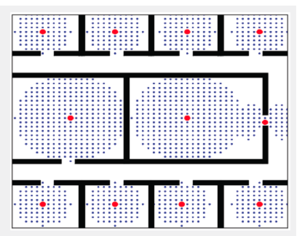
Overview

- A generic sub-gradient ascent [super-gradient descent] framework for submodular maximization [minimization].
- New theoretical results for submodular minimization.
- A novel view as a framework for submodular maximization.
- Empirical experimental validation.

Submodular Optimization in Machine Learning

Maximization:

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



Sensor Placement

Summarization

Minimization:

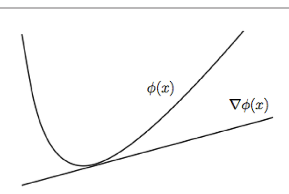
$$A^* \in \operatorname{argmin}_{A \in \mathcal{C}} f(A)$$

MAP Inference

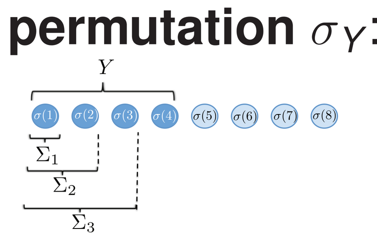
Clustering

Convexity, Concavity & Submodular Semigradients

Subgradients:



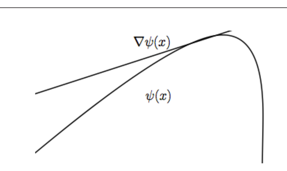
- Akin to convexity.
- Denote a **permutation** σ_Y:



- $h_Y(\sigma_Y(i)) = f(\Sigma_i) - f(\Sigma_{i-1})$
- Modular Lower bound:

$$m_{h_Y}(X) = f(Y) + h_Y(X) - h_Y(Y) \leq f(X)$$

Supergradients:



- Akin to concavity.
- Three specific supergradients:

$$\underbrace{\begin{aligned}\hat{g}_Y(j) &= f(j|V\backslash\{j\}) \\ \breve{g}_Y(j) &= f(j|Y\backslash\{j\}) \\ \bar{g}_Y(j) &= f(j|V\backslash\{j\})\end{aligned}}_{\text{for } j \in Y} \quad \underbrace{\begin{aligned}\hat{g}_Y(j) &= f(j|Y) \\ \breve{g}_Y(j) &= f(j|\emptyset) \\ \bar{g}_Y(j) &= f(j|\emptyset)\end{aligned}}_{\text{for } j \notin Y.}$$

- Modular Upper bound:

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

Semigradient Descent Algorithmic 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: $X^{t+1} \leftarrow \operatorname{argmax}_{X \in \mathcal{C}} m_{h_{X^t}}(X)$ [$X^{t+1} \leftarrow \operatorname{argmin}_{X \in \mathcal{C}} m^{g_{X^t}}(X)$]
5: $t \leftarrow t+1$
6: **until** we have converged ($X^{i-1} = X^i$) or $i \leq T$

Lemma 1
Algorithm 1 monotonically improves the objective function value at every iteration.

Define the **curvature** of a monotone submodular function κ_f as:

$$\kappa_f = 1 - \min_{j \in V} \frac{f(j|V\backslash j)}{f(j)} \qquad (1)$$

Submodular Function Minimization (MMin)

- Use supergradients $\hat{g}$, $\breve{g}$ and $\bar{g}$ in algorithm 1.

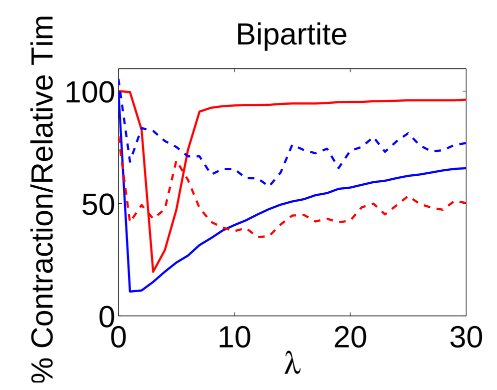
	MMin-IIIa	MMin-IIIb	MMin-I	MMin-II
g	$\bar{g}$	$\bar{g}$	$\hat{g}$	$\breve{g}$
X^0	$\emptyset$	V	$\emptyset$	V
X^c	A	B	A_+	B_+

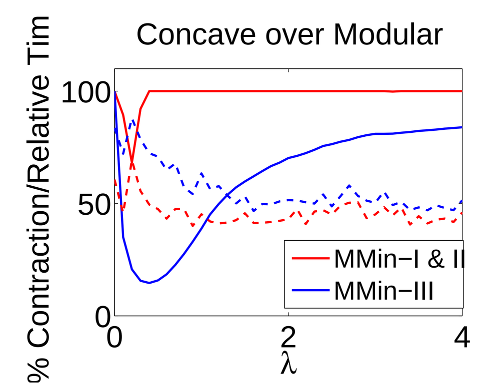
Unconstrained minimization ($\mathcal{C} = 2^V$)

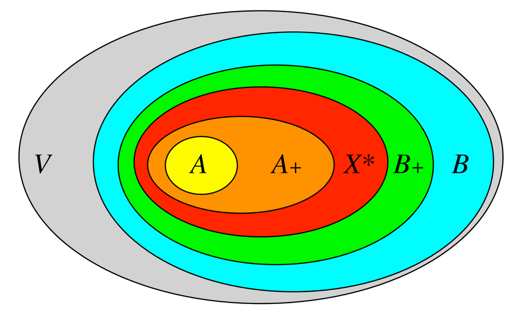
- Define $A = \{j : f(j|\emptyset) < 0\}$ and $B = \{j : f(j|V - \{j\}) > 0\}$.
- Known that for every optimizer $X^* : A \subseteq X^* \subseteq V$.

Theorem 1: MMin-IIIa and MMin-IIIb obtain sets $X^c = A$ and $X^c = B$ respectively. Furthermore, in $O(n^2)$ oracle calls, MMin-I and II obtain sets A_+ and B_+, which are local minimizers of f and satisfy $A \subseteq A_+ \subseteq X^* \subseteq B_+ \subseteq B$.

- Emperical results tested on Concave over modular: $\sqrt{w_1(X)} + \lambda w_2(V \backslash X)$ and Bipartite Neighborhoods: $\sqrt{w_1(\Gamma(X))} + \lambda w_2(V \backslash X)$.







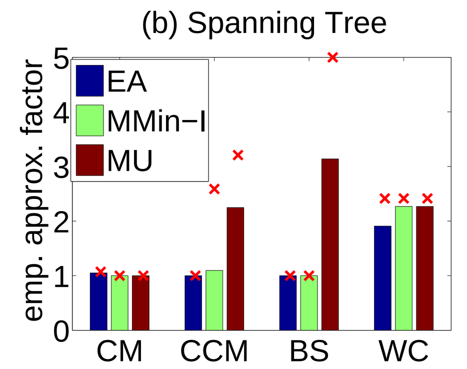
Constrained minimization (monotone)

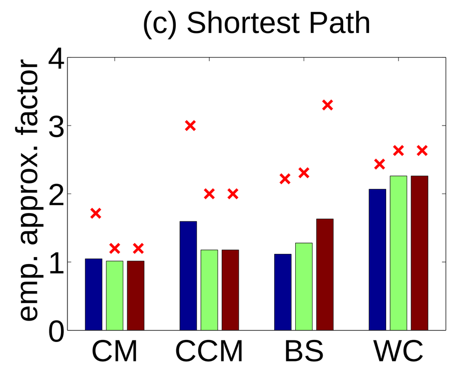
Theorem 2: The set A_+ obtained through MMin-I under constraints $\mathcal{C}$ obtains:

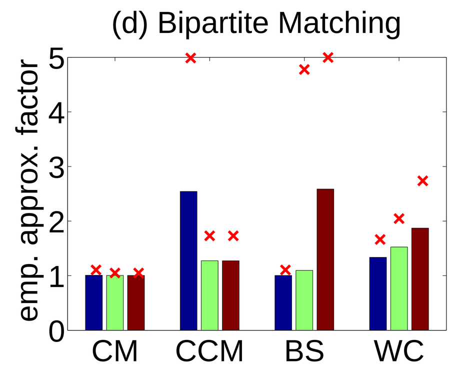
$$f(A_+) \leq \frac{|X^*|}{1 + (|X^*| - 1)(1 - \kappa_f)} f(X^*) \leq \frac{1}{1 - \kappa_f} f(X^*)$$

Constraints	Our Approximation Bounds
Spanning Tree / Perfect Matchings	$\frac{n}{1+(n-1)(1-\kappa_f)}$
Cardinality/ Matroid	$\frac{k}{1+(k-1)(1-\kappa_f)}$
Minimum s-t path	$\frac{n}{1+(n-1)(1-\kappa_f)}$
Minimum s-t cuts	$\frac{m}{1+(m-1)(1-\kappa_f)}$

- Emperical results:
  - (a) Concave over Modular (CM),
  - (b) Clustered concave over Modular (CCM)
  - (c) Best Set function ($f(X) = I(|X \cap R| \geq 1) + \sum_{j \in R \backslash X} w_j$) and
  - (d) Worst case-function (WC): $f(X) = \min\{|X \cap \bar{R}|, |X|, \alpha\}$.







Submodular Function Maximization (MMax)

- We choose the permutation based subgradients in Algorithm 1.
- Different choices of subgradients subsume a number of known submodular maximization algorithms.

Unconstrained maximization

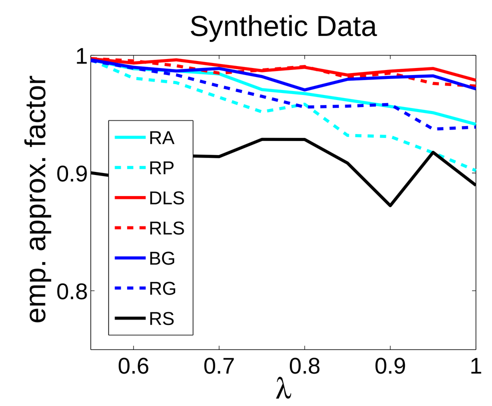
- **Random Subgradient (RA/ RP):** Random subgradients (permutations) at every iteration.
- **Randomized / Deterministic local search (RLS/DLS):** Local search based techniques naturally define subgradients.
- **Bi-directional Greedy (BG):** Bi-directional Greedy Subgradient (Buchbinder et al, 2012).
- **Randomized Greedy (RG):** Randomized variant of BG.

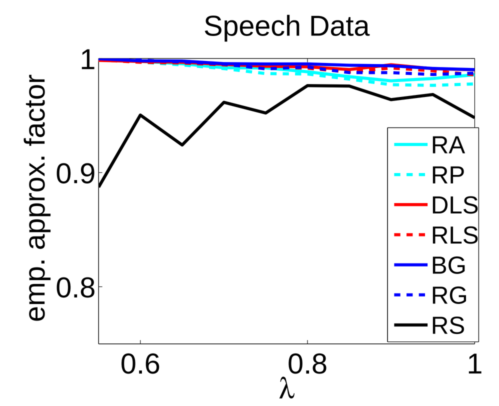
Subgradient	Approximation guarantee	Hardness
RA/RP	1/4	1/2
RLS/DLS	1/3	1/2
BG	1/3	1/2
RG	1/2	1/2

- Emperical results on synthetic data and TIMIT data, using a submodular function:

$$f(X) = \sum_{i \in V} \sum_{j \in X} s_{ij} - \lambda \sum_{i,j \in X} s_{ij} \qquad (2)$$

- Compare against a baseline (RS).





Constrained maximization (monotone)

- Greedy subgradient:

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

- Algorithm 1 using the subgradient h^{σ^g} exactly corresponds to the greedy algorithm!
- Obtains a $\frac{1}{\kappa_f}(1 - e^{-\kappa_f} \geq 1 - 1/e$ approximation!
- Similar analysis extends to Knapsack constraints.

Generality of Algorithm 1 for submodular Maximization

The subgradient algorithm for maximization is more general:

Theorem 3: For every α-approximation algorithm, there exists a schedule of subgradients obtainable in poly-time, such that Algorithm 1 acheives an approximation factor of atleast α.