ORIE 6334 Bridging Continuous and Discrete Optimization Dec 9, 2019

Lecture 23

Lecturer: David P. Williamson *Scribe: Sam Zhou*

In this lecture, we consider the problem of maximizing monotone submodular functions under cardinality constraints, and a more general class called matroid constraints.

1 Submodular Functions

We have a ground set of elements $E = \{e_1, \dots, e_n\} \equiv \{1, 2, \dots, n\}$.

Definition 1 *A function $f : 2^E \to \mathbf{R}_+$ is submodular if for all $S \subseteq T \subseteq E$, we have*

$$f(T \cup \{l\}) - f(T) \leq f(S \cup \{l\}) - f(S)$$

for all $l \in E \setminus T$.

By Definition 1, we see that submodular functions are scalar functions defined on subsets of E that have *decreasing marginal returns*. It can be shown that Definition 1 is equivalent to Definition 2:

Definition 2 *A function $f : 2^E \to \mathbf{R}_+$ is submodular if for any $S, T \subseteq E$, we have*

$$f(S) + f(T) \geq f(S \cup T) + f(S \cap T).$$

Definition 3 *A function $f : 2^E \to \mathbf{R}_+$ is monotone if for any $S \subseteq T \subseteq E$, we have*

$$f(S) \leq f(T).$$

Submodular functions have many applications:

- Cuts: Consider a undirected graph $G = (V, E)$, where each edge $e \in E$ is assigned with weight $w_e \geq 0$. Define the weighted cut function for subsets of E:

  $$f(S) := \sum_{e \in \delta(S)} w_e.$$

  We can see that f is submodular by showing any edge in the right-hand side of Definition 2 is also in the left-hand side.

- Influence in social networks [KKT03].

- Machine learning, algorithmic game theory, etc.

2 Maximizing Monotone Submodular Functions under Cardinality Constraints

When a submodular function $f : 2^E \to \mathbb{R}_+$ is monotone, maximizing f is easy, since the ground set E is always an optimal solution. Thus we consider maximizing monotone submodular functions under cardinality constraints:

$$\begin{array}{rl} \max & f(S) \\ s.t. & |S| \leq k \\ & S \subseteq E, \end{array} \tag{1}$$

where k is an integer satisfying $0 \leq k \leq |E|$.

In 1997, Cornuejols, Fisher and Nemhauser proposed a straightforward greedy algorithm for (1):

Algorithm 1: Greedy Algorithm.

$S \leftarrow \emptyset$;
while $|S| < k$ **do**
  $e \leftarrow \text{argmax}_{e \in E}[f(S \cup \{e\}) - f(S)]$;
  $S \leftarrow S \cup \{e\}$;
end
return S.

Theorem 1 ([CFN77]) *The greedy algorithm is a* $(1-1/e)$*-approximation algorithm for* (1).

Remark 1 $1 - 1/e \approx 0.632$.

We prove Theorem 1 by lower bounding the improvement of $f(S)$ at each iteration.

Lemma 2 *Pick any* $S \subseteq E$ *such that* $|S| < k$. *Let* $\mathcal{O}$ *denote an optimal solution to* (1), *then*

$$\max_{e \in E}[f(S \cup \{e\}) - f(S)] \geq \frac{1}{k}[f(\mathcal{O}) - f(S)]. \tag{2}$$

Proof: Let $\mathcal{O} \setminus S = \{i_1 \ldots, i_p\}$, so that $p \leq k$. Then we have

$$\begin{aligned}
f(\mathcal{O}) &\leq f(\mathcal{O} \cup S) && (3)\\
&= f(S) + \sum_{j=1}^{p} [f(S \cup \{i_1, \ldots, i_j\}) - f(S \cup \{i_1, \ldots, i_{j-1}\})] \\
&\leq f(S) + \sum_{j=1}^{p} [f(S \cup \{i_j\}) - f(S)] && (4)\\
&\leq f(S) + \sum_{j=1}^{p} \max_{e \in E} [f(S \cup \{e\}) - f(S)] && (5)\\
&= f(S) + k \max_{e \in E} [f(S \cup \{e\}) - f(S)], && (6)
\end{aligned}$$

where (3) is by the monotonicity of f, (4) and (5) are by the submodularity of f, and (6) is by $p \leq k$. □

Proof of Theorem 1: Let S^t denote the solution of the greedy algorithm at the end of iteration t. Then by Lemma 2,

$$\begin{aligned}
f(S^k) &\geq \frac{1}{k} f(\mathcal{O}) + \left(1 - \frac{1}{k}\right) f(S^{k-1}) \\
&\geq \frac{1}{k} f(\mathcal{O}) + \left(1 - \frac{1}{k}\right) \left[\frac{1}{k} f(\mathcal{O}) + \left(1 - \frac{1}{k}\right) f(S^{k-2})\right] \\
&\geq \cdots \\
&\geq \frac{f(\mathcal{O})}{k} \left[1 + \left(1 - \frac{1}{k}\right) + \left(1 - \frac{1}{k}\right)^2 + \cdots + \left(1 - \frac{1}{k}\right)^k\right] + f(\emptyset) \\
&\geq \frac{(1 - \frac{1}{k})^k}{k(1 - (1 - \frac{1}{k}))} f(\mathcal{O}) && (7)\\
&= \left(1 - \frac{1}{k}\right)^k f(\mathcal{O}) \\
&\geq \left(1 - \frac{1}{e}\right) f(\mathcal{O}), && (8)
\end{aligned}$$

where (7) is by $f(\emptyset) \geq 0$, and (8) is by the inequality $1 - x \leq e^{-x}$ for all $x \in \mathbb{R}$.

As simplistic as Algorithm 1 seems, as Feige [Fei98] pointed out, for any $\epsilon > 0$, there is no $(1 - 1/e + \epsilon)$-approximation algorithm for maximizing monotone submodular functions under cardinality constraints, unless $P = NP$.

3 Maximizing Monotone Submodular Functions under Matroid Constraints

A cardinality constraint is a special case of *matroid constraints*:

Definition 4 *Given a ground set E, a matroid $\mathcal{I}$ is a collection of subsets of E such that*

- *if $S \in \mathcal{I}$, then $S' \subseteq S \Rightarrow S' \in \mathcal{I}$;*
- *if $S, T \in \mathcal{I}$ and $|S| < |T|$, then there exists $e \in T \setminus S$ such that $S \cup \{e\} \in \mathcal{I}$.*

The elements of a matroid are called *independent sets*, whose name alludes the parallelism between independent sets and the set of linearly independent vectors in a vector space. It easy to check the set

$$\{S \subseteq E \mid |S| \leq k\}$$

is a matroid.

An independent set S is called a *base* of a matroid $\mathcal{I}$ if $\nexists S' \supseteq S$ such that $S' \in \mathcal{I}$. By the second part of Definition 4, all bases of a matroid $\mathcal{I}$ have the same cardinality. Matroids have a favorable computational property.

Fact 1 *A greedy algorithm finds a maximum weighted base of a matroid.*

An important example of matroids is the collection of edge set of the forests in a graph. The bases of this matroid are the spanning trees.

We consider maximizing monotone submodular functions under matroid constraints:

$$\begin{array}{rl} \max & f(S) \\ s.t. & S \in \mathcal{I} \\ & S \subseteq E, \end{array} \tag{9}$$

where $\mathcal{I}$ is a matroid of E. In 1978, Nemhauser, Wolsey, and Fisher proposed a local-search based algorithm [NWF78]:

Theorem 3 *A local search algorithm gives a $(1/2 - \epsilon)$-approximation algorithm for maximizing monotone submodular functions subject to matroid constraints.*

Before talking about another algorithm for (9), for a matroid S, define a polytope

$$\mathcal{P} := \{x \in \mathbb{R}^n \mid x \in \mathbb{R}^n_+, \sum_{i \in S} x_i \leq r(S),\ \forall S \subseteq E\}, \tag{10}$$

where the rank function $r : 2^E \to \mathbb{R}$ is defined as

$$r(S) = \max_{S' \subseteq S,\ S' \in \mathcal{I}} |S'|. \tag{11}$$

A useful property of $\mathcal{P}$ is that the extreme points of $\mathcal{P}$ correspond to the independent sets of $\mathcal{I}$. The algorithm we are about to present traces a continuous path in $\mathcal{P}$. To specify the algorithm, we define a *multilinear* function $F : [0,1]^n \to \mathbb{R}$, which is a continuous version of f:

$$F(x) := \sum_{S \in E} f(S) \prod_{i \in S} x_i \prod_{i \notin S} (1 - x_i). \tag{12}$$

Let $1_S \in [0,1]^n$ denote the vector that corresponds to S. It is easy to check that $F(1_S) = f(S)$. $F(x)$ is a multilinear function since it is linear in each x_i. It should be noted that in general, F is hard to evaluate, since the evaluation of F involves all the subsets of E. However, we have the following fact:

Fact 2 *We can evaluate $F(x)$ within given error by sampling.*

Lastly, for notational brevity, given $x, y \in \mathbb{R}^n$, define $x \vee y$ such that

$$(x \vee y)_i := \max\{x_i, y_i\}. \tag{13}$$

Now we are ready to present an algorithm for maximizing monotone submodular functions under cardinality constraints:

Algorithm 2: Continuous Greedy Algorithm.

$y \leftarrow 0 \in \mathbb{R}^n$;
for $t \in [0,1]$ **do**
  $w_i \leftarrow F(y(t) \vee e_i) - F(y(t))$ for each $i \in E$;
  $x(t) \leftarrow \mathrm{argmax}_{x \in \mathcal{P}} \langle w(t), x \rangle$;
  Increase $y(t)$ at rate $\frac{\mathrm{d}y(t)}{\mathrm{d}t} = x(t)$;
end
return $y(1) = \int_0^1 x(t)\mathrm{d}t$.

Here the computation of $x(t)$ uses Fact 1, the correspondence between the extreme points of $\mathcal{P}$ and the bases of $\mathcal{I}$, and the fact that there always exists an extreme point solution to a linear optimization problem over a polytope. We can obtain a polynomial time algorithm by discretizing the time steps of $y(t)$ in Algorithm 2.

Theorem 4 ([CCPV11]) *Let $\mathcal{O}$ denote an optimal solution to* (9)*, the continuous greedy algorithm returns $y(1) \in \mathcal{P}$ such that*

$$F(y(1)) \geq (1 - \frac{1}{e}) f(\mathcal{O}).$$

Remark 2 *We can obtain an* $(1 - 1/e)$*-approximation solution to* (9) *by checking the extreme points of the face of $\mathcal{P}$ that $y(1)$ lies in.*

As for Theorem 1, to prove Theorem 4, we first present a result that is useful to lower bounding the growth rate of $f(y(t))$:

Lemma 5 *For all* $y \in [0,1]^n$,

$$f(\mathcal{O}) \leq F(y) + \sum_{i \in \mathcal{O}} [F(y \vee e_i) - F(y)].$$

Proof: For all $R \subseteq E$, let $\mathcal{O} \setminus R = \{i_1, \ldots, i_p\}$, then we have

$$\begin{aligned} f(\mathcal{O}) &\leq f(\mathcal{O} \cup R) &(14)\\ &= f(R) + \sum_{j=1}^{p} [f(R \cup \{i_1, \ldots, i_j\}) - f(R \cup \{i_1, \ldots, i_{j-1}\})] \\ &\leq f(R) + \sum_{j=1}^{p} [f(R \cup \{i_j\}) - f(R)] &(15)\\ &= f(R) + \sum_{i \in \mathcal{O}} [f(R \cup \{i\}) - f(R)], &(16) \end{aligned}$$

where (14) is by the monotonicity of f, (15) is by the submodularity of f and (16) is by the observation that $f(R \cup \{i\}) - f(S) = 0$ when $i \in R$. For given $y \in [0,1]^n$, consider drawing R by random sampling: $i \in R$ with probability y_i. Then each $S \subseteq E$ has probability $\prod_{i \in S} x_i \prod_{i \notin S} (1 - x_i)$ to be chosen, which gives

$$E[f(R)] = \sum_{S \in E} f(S) \prod_{i \in S} y_i \prod_{i \notin S} (1 - y_i) = F(y).$$

By the same argument, we have

$$E[f(R \cup \{i\})] = F(y \vee e_i),$$

and (14)-(16) shows the Lemma is true. □

Proof of Theorem 4: Since $x(t) \in \mathcal{P}$ for any $t \in [0,1]$, we have

$$y(1) = \int_0^1 x(t) \mathrm{d}t \in \mathcal{P}$$

by the convexity of $\mathcal{P}$. Compute

$$\begin{aligned} \frac{\mathrm{d}F(y(t))}{\mathrm{d}t} &= \sum_{i \in E} \left(\frac{\mathrm{d}y_i(t)}{\mathrm{d}t} \cdot \frac{\partial F(y)}{\partial y_i} \Big|_{y=y(t)} \right) \\ &= \sum_{i \in E} \left(x_i(t) \cdot \frac{\partial F(y)}{\partial y_i} \Big|_{y=y(t)} \right) \\ &= \sum_{i \in E} \left(x_i(t) \cdot \frac{F(y(t) \vee e_i) - F(y(t))}{1 - y_i(t)} \right) &(17)\\ &\geq \sum_{i \in E} \left(x_i(t) w_i(t) \right), &(18) \end{aligned}$$

where (17) is by the linearity of $F(y)$ in y_i, and (18) is by the definition of $w(t)$.

Let $1_{\mathcal{O}} \in [0,1]^n$ denote the vector that corresponds to an optimal solution $\mathcal{O}$, then by the definition of $x(t)$ and Lemma 5,

$$\begin{aligned}\langle w(t), x(t)\rangle &\geq \langle w(t), 1_{\mathcal{O}}\rangle \\ &= \sum_{i\in\mathcal{O}} [F(y(t) \vee e_i) - F(y(t))] \\ &\geq f(\mathcal{O}) - F(y(t)).\end{aligned}$$

Hence

$$\frac{\mathrm{d}F(y(t))}{\mathrm{d}t} \geq f(\mathcal{O}) - F(y(t)).$$

This implies that $F(y(t))$ dominates $\phi(t) : [0,1] \to \mathbb{R}^n$ subject to

$$\frac{\mathrm{d}\phi(t)}{\mathrm{d}t} = f(\mathcal{O}) - \phi(y(t)). \tag{19}$$

Solve (19), we get

$$\phi(t) = (1 - e^{-t}) f(\mathcal{O}),$$

and

$$F(y(1)) \geq \phi(1) = (1 - e^{-1}) f(\mathcal{O}).$$

Remark 3 *Buchbinder, Feldman and Schwartz gives a nice summary of maximizing submodular functions in [BFS16]. In 2012, Filmus and Ward [FW12] proposed a local search based* $(1-1/e)$*-approximation algorithm for maximizing monotone submodular functions under matroid constraints.*

References

[BFS16] Niv Buchbinder, Moran Feldman, and Roy Schwartz. Comparing apples and oranges: Query trade-off in submodular maximization. *Mathematics of Operations Research*, 42(2):308–329, 2016.

[CCPV11] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[CFN77] Gerard Cornuejols, Marshall L Fisher, and George L Nemhauser. Exceptional paper—location of bank accounts to optimize float: An analytic study of exact and approximate algorithms. *Management science*, 23(8):789–810, 1977.

[Fei98] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[FW12] Yuval Filmus and Justin Ward. A tight combinatorial algorithm for submodular maximization subject to a matroid constraint. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 659–668. IEEE, 2012.

[KKT03] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[NWF78] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.