Submodularity in Machine Learning
- New Directions -

Andreas Krause
Stefanie Jegelka
Network Inference

How learn who influences whom?
Summarizing Documents

How select representative sentences?
MAP inference

\[
\max_x p(x \mid z)
\]

How find the MAP labeling in discrete graphical models efficiently?
What’s common?

Formalization:

Optimize a set function $F(S)$ under constraints

generally very hard

but: structure helps!
... if $F$ is submodular, we can ...

- solve optimization problems with strong guarantees
- solve some learning problems
Outline

- What is submodularity?
  - Optimization
    - Minimization
    - Maximization
  - Learning
    - Learning for Optimization: new settings

Part I

Break

Part II

many new results! 😊
Outline

- What is submodularity?  
  many new results! 😊

- Optimization
  - Minimization: new algorithms, constraints
  - Maximization: new algorithms (unconstrained)

- Learning
  - Learning for Optimization: new settings

... and many new applications!
submodularity.org
slides, links, references, workshops, ...
Example: placing sensors

Place sensors to monitor temperature
Set functions

- finite ground set $V = \{1, 2, \ldots, n\}$
- set function $F : 2^V \to \mathbb{R}$

will assume $F(\emptyset) = 0$ (w.l.o.g.)

assume black box that can evaluate $F(A)$ for any $A \subseteq V$
Example: placing sensors

Utility $F(A)$ of having sensors at subset $A$ of all locations

A={1,2,3}: Very informative
High value $F(A)$

A={1,4,5}: Redundant info
Low value $F(A)$
Marginal gain

- Given set function $F : 2^V \rightarrow \mathbb{R}$

- Marginal gain: $\Delta_F(s \mid A) = F(\{s\} \cup A) - F(A)$

new sensor $s$
Decreasing gains: submodularity

placement A = \{1,2\}

placement B = \{1,...,5\}

Big gain

\[ A \subseteq B \]

\[ F(A \cup s) - F(A) \]

\[ \Delta(s \mid A) \]

Adding helps a lot! Adding doesn't help much!

new sensor s

+ \cdot s

+ \cdot s

small gain
Equivalent characterizations

- **Diminishing gains:** for all $A \subseteq B$

  $$F(A \cup s) - F(A) \geq F(B \cup s) - F(B)$$

- **Union-Intersection:** for all $A, B \subseteq V$

  $$F(A) + F(B) \geq A \cup B, F(A \cup B) + F(A \cap B)$$
Questions

How do I prove my problem is submodular?

Why is submodularity useful?
Example: Set cover

Node predicts values of positions with some radius

Possible locations $V$

place sensors in building

goal: cover floorplan with discs

$A \subseteq V$: $F(A) =$

“area covered by sensors placed at $A$”

Formally:

Finite set $W$, collection of $n$ subsets $S_i \subseteq W$

For $A \subseteq V$ define $F(A) = \left| \bigcup_{i \in A} S_i \right|$
Set cover is submodular

\[ A = \{s_1, s_2\} \]

\[ B = \{s_1, s_2, s_3, s_4\} \]

\[ F(A \cup \{s'\}) - F(A) \geq F(B \cup \{s'\}) - F(B) \]

B = \{s_1, s_2, s_3, s_4\}
More complex model for sensing

Joint probability distribution

\[ P(X_1, \ldots, X_n, Y_1, \ldots, Y_n) = P(Y_1, \ldots, Y_n) P(X_1, \ldots, X_n | Y_1, \ldots, Y_n) \]

\( Y_s \): temperature at location \( s \)

\( X_s \): sensor value at location \( s \)

\( X_s = Y_s + \text{noise} \)
Example: Sensor placement

Utility of having sensors at subset $A$ of all locations

$$F(A) = H(Y) - H(Y | X_A)$$

Uncertainty about temperature $Y$ before sensing

Uncertainty about temperature $Y$ after sensing

A=$\{1,2,3\}$: High value $F(A)$

A=$\{1,4,5\}$: Low value $F(A)$
Submodularity of Information Gain

\[ Y_1, \ldots, Y_m, X_1, \ldots, X_n \] discrete RVs
\[
F(A) = I(Y; X_A) = H(Y) - H(Y \mid X_A)
\]

- \( F(A) \) is NOT always submodular

If \( X_i \) are all conditionally independent given \( Y \), then \( F(A) \) is submodular!

[ Krause & Guestrin `05 ]

Proof:
“information never hurts”
Example: costs

cost: time to reach shop + price of items

ground set $V$

each item 1 $\$"

breakfast??

Market 1

Market 2

Market 3
Example: costs

cost:
  time to shop
  + price of items

\[ F(\text{coffee, sandwich}) = \text{cost(} \text{coffee} \text{) + cost(} \text{sandwich, melon} \text{)} \]

\[ = t_1 + 1 + t_2 + 2 \]

\[ = \#\text{shops} + \#\text{items} \]

submodular?
Shared fixed costs

\[ \Delta(b \mid A) = 1 + t_3 \]
\[ \Delta(b \mid B) = 1 \]

marginal cost:  \#new shops + \#new items

decreasing \(\Rightarrow\) cost is submodular!

- shops: shared fixed cost
- economies of scale
Another example: Cut functions

V={a,b,c,d,e,f,g,h}

\[ F(A) = \sum_{s \in A, t \notin A} w_{s,t} \]

Cut function is submodular!
Why are cut functions submodular?

**Submodular if** \( w \geq 0 \! \)
Closedness properties

$F_1, \ldots, F_m$ submodular functions on $V$ and $\lambda_1, \ldots, \lambda_m > 0$

Then: $F(A) = \sum_i \lambda_i F_i(A)$ is submodular

Submodularity closed under nonnegative linear combinations!

Extremely useful fact:
- $F_\theta(A)$ submodular $\implies \sum_\theta P(\theta) F_\theta(A)$ submodular!
- Multicriterion optimization
- A basic proof technique! 😊
Other closedness properties

Restriction: \( F(S) \) submodular on \( V \), \( W \) subset of \( V \)

Then \( F'(S) = F(S \cap W) \) is submodular
Other closedness properties

- **Restriction**: $F(S)$ submodular on $V$, $W$ subset of $V$
  
  Then $F'(S) = F(S \cap W)$ is submodular

- **Conditioning**: $F(S)$ submodular on $V$, $W$ subset of $V$
  
  Then $F'(S) = F(S \cup W)$ is submodular
Other closedness properties

- **Restriction**: \( F(S) \) submodular on \( V \), \( W \) subset of \( V \)
  
  Then \( F'(S) = F(S \cap W) \) is submodular

- **Conditioning**: \( F(S) \) submodular on \( V \), \( W \) subset of \( V \)
  
  Then \( F'(S) = F(S \cup W) \) is submodular

- **Reflection**: \( F(S) \) submodular on \( V \)
  
  Then \( F'(S) = F(V \setminus S) \) is submodular
Submodularity ...

discrete convexity ....

... or concavity?
Convex aspects

- convex extension
- duality
- efficient minimization

But this is only half of the story...
Concave aspects

- **submodularity:**
  \[ A \subseteq B, \ s \notin B : \]
  \[ F(A \cup s) - F(A) \geq F(B \cup s) - F(B) \]

- **concavity:**
  \[ a \leq b, \ s > 0 : \]
  \[ f(a + s) - f(a) \geq f(b + s) - f(b) \]
Submodularity and concavity

- Suppose \( g : \mathbb{N} \rightarrow \mathbb{R} \) and \( F(A) = g(|A|) \)

\[ F(A) \text{ submodular if and only if } \ldots \text{ } g \text{ is concave} \]
Maximum of submodular functions

- $F_1(A), F_2(A)$ submodular. What about

$$F(A) = \max\{ F_1(A), F_2(A) \} \quad ?$$

$F(A) = \max(F_1(A),F_2(A))$

$max(F_1,A,F_2) \text{ not submodular in general!}$
Minimum of submodular functions

Well, maybe $F(A) = \min(F_1(A), F_2(A))$ instead?

<table>
<thead>
<tr>
<th></th>
<th>$F_1(A)$</th>
<th>$F_2(A)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>{}</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>{a}</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>{b}</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>{a,b}</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

$F(\{b\}) - F(\{\}) = 0 < F(\{a,b\}) - F(\{a\}) = 1$

$\min(F_1, F_2)$ not submodular in general!
Two faces of submodular functions

- Convex aspects ➔ minimization!
- Concave aspects ➔ maximization!
What to do with submodular functions

Optimization
- Minimization
- Maximization

Learning
- Online/adaptive optim.
What to do with submodular functions

Optimization
  Minimization
  Maximization

Learning
  Online/ adaptive optim.

Minimization and maximization not the same??
Submodular minimization

\[
\min_{S \subseteq V} F(S)
\]

- Clustering
- MAP inference
- Structured sparsity regularization
- Minimum cut
Submodular minimization

\[ \min_{S \subseteq V} F(S) \]

→ submodularity and convexity
Set functions and energy functions

any set function with $|V| = n$

$F : 2^V \rightarrow \mathbb{R}$

... is a function on binary vectors!

$F : \{0, 1\}^n \rightarrow \mathbb{R}$

pseudo-boolean function
Submodularity and convexity

Lovász extension

extension

\[ f : [0, 1]^n \to \mathbb{R} \]

\[ F : \{0, 1\}^n \to \mathbb{R} \]

Lovász extension

\[ f(x) = \max_{y \in P_F} x \cdot y \]

convex

Lovász, 1982

- minimum of \( f \) is a minimum of \( F \)
- submodular minimization as convex minimization:
  polynomial time!

Grötschel, Lovász, Schrijver 1981
Submodularity and convexity

\[ F : \{0, 1\}^n \rightarrow \mathbb{R} \quad \text{extension} \quad f : [0, 1]^n \rightarrow \mathbb{R} \]

Lovász extension

\[ f(x) = \max_{y \in P_F} x \cdot y \]

convex

Lovász, 1982

- Minimum of \( f \) is a minimum of \( F \)
- Submodular minimization as convex minimization: polynomial time!
The submodular polyhedron $P_F$

$$P_F = \{ x \in \mathbb{R}^n : x(A) \leq F(A) \text{ for all } A \subseteq V \}$$

$${\color{red} x(A) = \sum_{i \in A} x_i}$$

Example: $V = \{a,b\}$

<table>
<thead>
<tr>
<th>$A$</th>
<th>$F(A)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>${}$</td>
<td>0</td>
</tr>
<tr>
<td>${a}$</td>
<td>-1</td>
</tr>
<tr>
<td>${b}$</td>
<td>2</td>
</tr>
<tr>
<td>${a,b}$</td>
<td>0</td>
</tr>
</tbody>
</table>

$${\color{red} x(\{a\}) \leq F(\{a\})}$$

$${\color{red} x(\{} ) \leq F(\{} )}$$

$${\color{red} x(\{b\}) \leq F(\{b\})}$$

$${\color{red} x(\{a,b\}) \leq F(\{a,b\})}$$
Evaluating the Lovász extension

\[ P_F = \{ x \in \mathbb{R}^n : x(A) \leq F(A) \text{ for all } A \subseteq V \} \]

Linear maximization over \( P_F \)

\[ f(x) = \max_{y \in P_F} x \cdot y \]

Exponentially many constraints!!! 😞

Computable in \( O(n \log n) \) time 😊

[Edmonds ‘70]

greedy algorithm:

- sort \( x \)
- order defines sets \( S_i = \{1, \ldots, i\} \)
- \( y_i = F(S_i) - F(S_{i-1}) \)

• Subgradient
• Separation oracle
Lovász extension: example

\[ F(a) \]
\[ F(b) \]
\[ F(a,b) \]

\[ f(x) \]

\[ A \quad F(A) \]
\[ \{\} \quad 0 \]
\[ \{a\} \quad 1 \]
\[ \{b\} \quad 0.8 \]
\[ \{a,b\} \quad 0.2 \]
Submodular minimization

\[
\min_{A \subseteq V} F(A)
\]

- minimize convex extension
- combinatorial algorithms

- ellipsoid algorithm [Grötschel et al. 81]
- subgradient method, smoothing [Stobbe & Krause 10]
- duality: minimum norm point algorithm [Fujishige & Isotani 11]
- Fulkerson prize
  Iwata, Fujishige, Fleischer 01 & Schrijver 00
- state of the art:
  \[O(n^4T + n^5\log M)\] [Iwata 03]
  \[O(n^6 + n^5T)\] [Orlin 09]

\[T = \text{time for evaluating } F\]
The minimum-norm-point algorithm

Example: \( V = \{a, b\} \)

Regularized problem

\[
\min_{x \in [0, 1]^n} f(x) + \frac{1}{2} \|x\|^2
\]

\[
u^* = \arg \min_{u \in B_F} \frac{1}{2} \|u\|^2
\]

Base polytope \( B_F \)

\[
A^* = \{ i \mid u^*(i) \leq 0 \}
\]

minimizes \( F \):

\[
A^* = \arg \min_{A \subseteq V} F(A)
\]

Fujishige ‘91, Fujishige & Isotani ‘11
The minimum-norm-point algorithm

1. find \( u^* = \arg \min_{u \in B_F} \frac{1}{2} \|u\|^2 \)
2. \( A^* = \{ i \mid u^*(i) \leq 0 \} \)

can we solve this??

yes! 😊

recall: can solve linear optimization over \( P_F \)
similar: optimization over \( B_F \) ➔ can find \( u^* \)

(Frank-Wolfe algorithm)

Fujishige ‘91, Fujishige & Isotani ‘11
Empirical comparison

Cut functions from DIMACS Challenge

Minimum norm point algorithm: usually orders of magnitude faster

[Fujishige & Isotani ’11]
Applications?
Many natural signals sparse in suitable basis. Can exploit for learning/regularization/compressive sensing...
Sparse reconstruction

\[ \min_x \| y - Mx \|^2 + \lambda \Omega(x) \]

- explain \( y \) with few columns of \( M \): few \( x_i \)

  discrete regularization on support \( S \) of \( x \)

  \[ \Omega(x) = \|x\|_0 = |S| \]

  relax to convex envelope

  \[ \Omega(x) = \|x\|_1 \]

in nature: sparsity pattern often not random...
Structured sparsity

Incorporate tree preference in regularizer?

Set function:

\[ F(T) < F(S) \]

if \( T \) is a tree and \( S \) not
\[ |S| = |T| \]

\[ F(S) = \left| \bigcup_{s \in S} \text{ancestors}(s) \right| \]
Structured sparsity

Incorporate tree preference in regularizer?

Set function:

\[ F(T) < F(S) \]

If \( T \) is a tree and \( S \) not, \( |S| = |T| \)

\[ F(S) = \left| \bigcup_{s \in S} \text{ancestors}(s) \right| \]

\[ F(T) = 3 \]
Structured sparsity

Incorporate tree preference in regularizer?

Set function:

\[ F(T) < F(S) \]

If \( T \) is a tree and \( S \) not, 
\[ |S| = |T| \]

Function \( F \) is ... 

submodular! 😊

\[ F(T) = 3 \]
Sparsity

\[
\min_x \|y - Mx\|^2 + \lambda \Omega(x)
\]

- explain \(y\) with few columns of \(M\): few \(x_i\)
- prior knowledge: patterns of nonzeros
- discrete regularization on support \(S\) of \(x\)
  \[\Omega(x) = \|x\|_0 = |S|\]
- submodular function
  \[\Omega(x) = F(S)\]
- Lovász extension
  \[\Omega(x) = f(|x|)\]
- Optimization: submodular minimization

[Bach `10]
Further connections: Dictionary Selection

$$\min_x \|y - Mx\|^2 + \lambda \Omega(x)$$

Where does the dictionary $M$ come from?

Want to learn it from data: $\{y_1, \ldots, y_n\} \subseteq \mathbb{R}^d$

Selecting a dictionary with near-max. variance reduction

$\Leftrightarrow$ Maximization of approximately submodular function

[Krause & Cevher ‘10; Das & Kempe ‘11]
Example: MAP inference

\[
\max_{\mathbf{x} \in \{0,1\}^n} \quad P(\mathbf{x} \mid \mathbf{z}) \propto \exp(-E(\mathbf{x}; \mathbf{z}))
\]

\[\Leftrightarrow \quad \min_{\mathbf{x} \in \{0,1\}^n} \quad E(\mathbf{x}; \mathbf{z})\]
Example: MAP inference

Recall: equivalence

\[ \max_{x \in \{0, 1\}^n} P(x | z) \propto \exp(-E(x; z)) \]

\[ E(e_A; z) = F(A) \]

if \( F \) is submodular (attractive potentials), then MAP inference = submodular minimization!

polynomial-time
Special cases

Minimizing general submodular functions:

- poly-time, but not very scalable

Special structure ➔ faster algorithms

- Symmetric functions
- Graph cuts
- Concave functions
- Sums of functions with bounded support
- ...

62
MAP inference

\[
\min_{\mathbf{x} \in \{0,1\}^n} E(\mathbf{x}; \mathbf{z}) = \sum_i E_i(x_i) + \sum_{ij} E_{ij}(x_i, x_j) \equiv \min_{A \subseteq V} F(A)
\]

if each \(E_{ij}\) is submodular:

\[
E_{ij}(1,0) + E_{ij}(0,1) \geq E_{ij}(0,0) + E_{ij}(1,1)
\]

then \(F\) is a graph cut function.

MAP inference = Minimum cut: fast 😊
Potential functions defined on sets of pixels with conventional guarantees and thus may produce bad results. In this paper we propose an algorithm that can compute the solution of

\[ E(x) = \sum_i E_i(x_i) + \sum_{ij} E_{ij}(x_i, x_j) \]

Pixels in one tile should have the same label

[Shotton et al. `09]
Enforcing label consistency

Pixels in a superpixel should have the same label

\[ E(x) \leq \gamma_{\text{max}} \]

concave function of cardinality \( \Rightarrow \) submodular

> 2 arguments: Graph cut ??
Higher-order functions as graph cuts?

\[ \sum_i E_i(x_i) + \sum_{ij} E_{ij}(x_i, x_j) + \sum_c E_c(x_c) \]

**General strategy:**
reduce to pairwise case by adding auxiliary variables

- works well for some particular \( E_c(x_c) \).
  [Billionet & Minoux `85, Freedman & Drineas `05, Živný & Jeavons `10, ...]

- necessary conditions complex and
  not all submodular functions equal such graph cuts
  [Živný et al.‘09]
Fast approximate minimization

- Not all submodular functions can be optimized as graph cuts
- Even if they can: possibly many extra nodes in the graph 😞

Other options?
- minimum norm algorithm
- other special cases:
  - e.g. parametric maxflow
    - [Fujishige & Iwata ‘99]

Approximate! 😊
Every submodular function can be approximated by a series of graph cut functions  
- [Jegelka, Lin & Bilmes ‘11]
Fast approximate minimization

- Not all submodular functions can be optimized as graph cuts
- Even if they can: possibly many extra nodes in the graph 😞

Approximate! 😊

decompose:
- represent as much as possible exactly by a graph
- rest: approximate iteratively by changing edge weights

solve a series of cut problems
Other special cases

- Symmetric:
  - Queyranne’s algorithm: $O(n^3)$
    \[ F(S) = F(V \setminus S) \]
    [Queyranne, 1998]

- Concave of modular:
  \[ F(S) = \sum_i g_i \left( \sum_{s \in S} w(s) \right) \]
  [Stobbe & Krause ‘10, Kohli et al, ‘09]

- Sum of submodular functions, each bounded support
  [Kolmogorov ‘12]
Submodular minimization

- unconstrained: \( \min F(A) \quad \text{s.t.} \quad A \subseteq V \)
  - nontrivial algorithms, polynomial time

- constraints: e.g. \( \min F(A) \quad \text{s.t.} \quad |A| \geq k \)
  - limited cases doable:
    - odd/even cardinality, inclusion/exclusion of a set

Special case: balanced cut

General case: \textbf{NP hard}
- hard to approximate within polynomial factors!
- \textbf{But:} special cases often still work well

[Lower bounds: Goel et al. `09, Iwata & Nagano `09, Jegelka & Bilmes `11]
Constraints

minimum...

cut  matching  path  spanning tree

ground set: edges in a graph

\[
\min_{S \in \mathcal{C}} \sum_{e \in S} w(e) \quad \Rightarrow \quad \min_{S \in \mathcal{C}} F(S)
\]
Recall: MAP and cuts

binary labeling: $x = e_A$

pairwise random field:

$E(x) = \text{Cut}(A)$

What’s the problem?

minimum cut: prefer short cut = short object boundary
Minimum cut

**implicit criterion:**
short cut = short boundary

minimize
sum of edge weights

\[ F(C) = \sum_{e \in C} w(e) \]

Minimum cooperative cut

**new criterion:**
boundary may be long if the boundary is homogeneous

minimize
submodular function of edges

\[ F(C) \]

not a sum of edge weights!
Reward co-occurrence of edges

sum of weights: use few edges

submodular cost function: use few groups $S_i$ of edges

$$F(C) = \sum_i F_i(C \cap S_i)$$

25 edges, 1 type
7 edges, 4 types
Results

Graph cut

Cooperative cut
Optimization?

- not a standard graph cut
- MAP viewpoint:
  - global, non-submodular energy function
Constrained optimization

\[
\min_{S \in \mathcal{C}} F(S)
\]

approximate optimization

- convex relaxation
- minimize surrogate function

approximation bounds dependent on \( F \):
- polynomial \( O(n) \)
- constant \( (1 + \epsilon) \)

[Goel et al.`09, Iwata & Nagano `09, Goemans et al. `09, Jegelka & Bilmes `11, Iyer et al. ICML `13, Kohli et al `13...]

Efficient constrained optimization

minimize a series of surrogate functions

1. compute linear upper bound
   \[ \hat{F}^i(S^i) = F(S^i) \]
   \[ \hat{F}^i(S) = \sum_{e \in S} w^i(S) \]

2. Solve easy sum-of-weights problem:
   \[ S'^i = \arg \min_{S \in C} \hat{F}^i(S) \]
   and repeat.

- efficient
- only need to solve sum-of-weights problems
- unifying viewpoint of submodular min and max
see Wed best student paper talk

[Jegelka & Bilmes `11, Iyer et al. ICML `13]
Submodular min in practice

- Does a special algorithm apply?
  - symmetric function?  graph cut?  .... approximately?
- Continuous methods: *convexity*
  - minimum norm point algorithm

- Other techniques  [not addressed here]
  - LP, column generation, ...
- Combinatorial algorithms: relatively high complexity

- Constraints: hard
  - majorize-minimize or relaxation
Outline

- What is submodularity?

- Optimization
  - Minimize costs
  - Maximize utility

- Learning
  - Learning for Optimization: new settings

Part I

Break!

Part II

see you in half an hour 😊