Min-Norm Algorithm

Progress in the past weeks:

1. How to find min-norm (up to the first 2 steps in the min-norm algorithm of matlab)
2. Why does the min-norm point give the minimum value of F

1. Minimum-Norm Point Algorithm and Its Application in Submodular Functions

The Minimum-Norm-Point Algorithm Applied to Submodular Function Minimization and Linear Programming

Satoru FUJISHIGE , Takumi HAYASHI†, and Shigueo ISOTANI

September, 2006

2.1. Description of the minimum-norm-point algorithm

Consider the n-dimensional Euclidean space $\mathbf{R}^n$. Suppose that we are given a finite set P of points p_i $(i \in I)$ in $\mathbf{R}^n$. The problem is to find the minimum-norm point x^* in the convex hull $\hat{P}$ of points p_i $(i \in I)$.

Wolfe's algorithm [15] is given as follows.

The Minimum-Norm-Point Algorithm

Input: A finite set P of points p_i $(i \in I)$ in $\mathbf{R}^n$.
Output: The minimum-norm point x^* in the convex hull $\hat{P}$ of the points p_i $(i \in I)$.
Step 1: Choose any point p in P and put $S := \{p\}$ and $\hat{x} := p$.
Step 2: Find a point $\hat{p}$ in P that minimizes the linear function $\langle \hat{x}, p \rangle = \sum_{k=1}^{n} \hat{x}(k)p(k)$ in $p \in P$. Put $S := S \cup \{\hat{p}\}$.
If $\langle \hat{x}, \hat{p} \rangle = \langle \hat{x}, \hat{x} \rangle$, then return $x^* = \hat{x}$ and halt;
else go to Step 3.
Step 3: Find the minimum-norm point y in the affine hull of points in S.
If y lies in the relative interior of the convex hull of S, then put $\hat{x} := y$ and go to Step 2.
Step 4: Let z be the point that is the nearest to y among the intersection of the convex hull of S and the line segment $[y, \hat{x}]$ between y and $\hat{x}$. Also let $S' \subset S$ be the unique subset of S such that z lies in the relative interior of the convex hull of S'. Put $S := S'$ and $\hat{x} := z$.
Go to Step 3.
(End)

-What is submodular function?

youtube.com/watch?v=Y3u_hvxayDY

Submodularity: Theory and Applications I by Stephanie Jegelka

Part 2 of this series of lectures is given by Andreas Krause. He implemented the minimum-norm-point algorithm in Matlab ("Submodular Function Optimization" under mathworks.com).

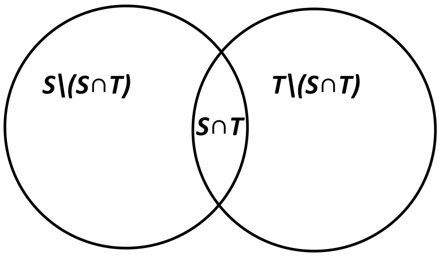
According to the lecture, there are 2 equivalent forms of definitions for submodular functions:

1. A function applicable to all subsets of a set *V* where if $A \subseteq B \subseteq V$ and *e* is a single element then:

$$F(A \cup \{e\}) - F(A) \geq F(B \cup \{e\}) - F(B)$$

2. A function .... where for any 2 subsets *S* and *T* of *V*:

$$F(S) + F(T) \geq F(S \cap T) + F(S \cup T)$$



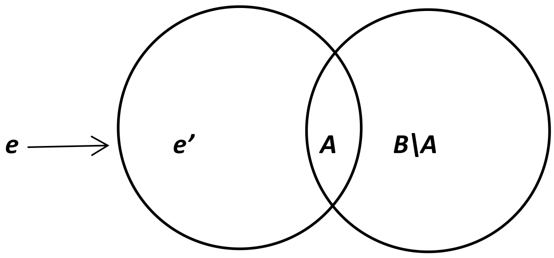
Proof that 2 implies 1: Take *A* as $S \cap T$, *B* as *T* and *e* as $S \setminus S \cap T$. Then we have:

$$F(A \cup \{e\}) + F(B) \geq F(A) + F(B \cup \{e\}) \quad \square$$

My idea for the proof that 1 implies 2: a "quasi-induction" that "integrates points"

1) When $S \setminus (S \cap T)$ is empty or is a single point, 2 is true.

2) If for a set function *F*, 1 is true and 2 is true for $S \setminus (S \cap T) = e'$ and we wish to expand *e'* by one point, *e*:



We have $F(e' \cup A) + F(B) \geq F(A) + F(e' \cup B) \geq F(A) + F(e' \cup B)$ (1)

And by 1 we have $F(\{e\} \cup e' \cup A) - F(e' \cup A) \geq F(\{e \cup e' \cup B\}) - F(e' \cup B)$ (2)

Add (1) and (2) up and 2 is valid for a larger set that represents $S \setminus (S \cap T)$.

Using similar idea we might prove another equivalent definition:

Theorem 44.1. *A set function f on S is submodular if and only if*

(44.3) $$f(U \cup \{s\}) + f(U \cup \{t\}) \geq f(U) + f(U \cup \{s,t\})$$

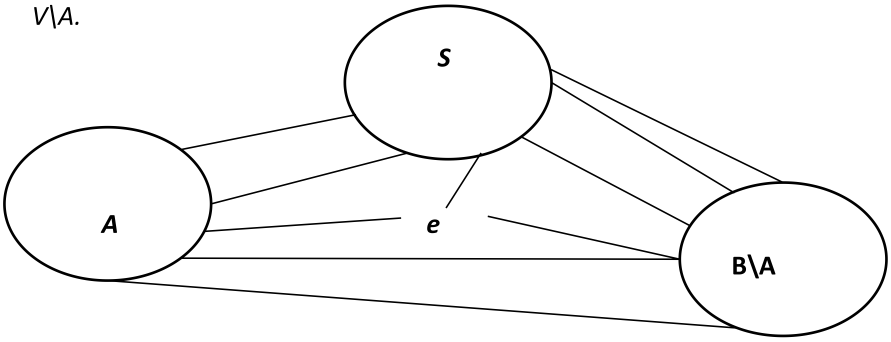
for each $U \subseteq S$ and distinct $s, t \in S \setminus U$.

From "Combinatorial Optimization: Polyhedra and Efficiency", by Alexander Schrijver, Volume A.

For the introduction today, we will only use definition 1.

Some examples of submodular functions:

1. Given a set of vertices V and edges, $F(A)$ denotes the total number of edges between A and $V\backslash A$.



There are edges connecting these 4 components. Let $G(P,Q)$ denote the number of edges between components P and Q, then:

$$F(A) = G(A,\{e\}) + G(A, B \setminus A) + G(A,S)$$
$$F(B) = G(A,S) + G(B \setminus A, S) + G(A,\{e\}) + G(B \setminus A, \{e\})$$
$$F(A \cup \{e\}) = G(A, B \setminus A) + G(B \setminus A, \{e\}) + G(A,S) + G(S,\{e\})$$
$$F(B \cup \{e\}) = G(A,S) + G(B \setminus A, S) + G(S,\{e\})$$

$$G(B \setminus A, \{e\}) + G(S,\{e\}) - G(A,\{e\}) \geq G(S,\{e\}) - G(A,\{e\}) - G(B \setminus A, \{e\})$$

2. Let M be a positive semi-definite matrix. Define $F(\emptyset) = 0$ and $F(S)$ be the log determinant of the matrix formed by taking all the s rows and columns of M, where s is an element of S.
This can be proven by checking the Cholesky decomposition of M and its submatrices.

-Lovasz Extension

Considering minimization (and probably maximization), we can assume $F(\emptyset) = 0$. This is because submodular function only places restrictions on the “marginal gain” of function value when an element is integrated into a set. We can change the “starting value” of the “base set”.

Lovasz’ extension:

$$x = [x_1, x_2, \cdots, x_n]^T$$
$$0 \leq x_{S_1} \leq x_{S_2} \leq \cdots \leq x_{S_n} \leq 1$$
$$f(x) = x_{S_1} F\left(\{S_j\}_{j=1}^{n}\right) + \sum_{k=2}^{n} \left(x_{S_k} - x_{S_{k-1}}\right) F\left(\{S_j\}_{j=k}^{n}\right) + \left(1 - x_{S_n}\right) F(\emptyset) \quad (3)$$

When $F(\emptyset) = 0$ this is equivalent to:

$$f(x) = \sum_{k=1}^{n} x_{S_k} \left[F\left(\{S_j\}_{j=k}^{n}\right) - F\left(\{S_j\}_{j=k}^{n} \setminus \{S_k\}\right)\right] \quad (4)$$

And also equivalent to:

$$f(x) = max \sum_{k=1}^{n} x_k y_k \text{, where } \sum_{j \in S} y_j \leq F(S) \forall S \subseteq \{1,2,\cdots,n\},\ \boldsymbol{\sum_{j=1}^{n} y_j = F(\{1,2,\cdots,n\})} \quad (5)$$

The set of vectors $[y_1, \cdots, y_n]^T$ that satisfy the inequalities is called a *polytope*. The set of vectors that satisfy the inequalities plus the equality in bold is called a *Base polytope*.

To see (3) is equivalent to (4), we may rewrite x_{S_k} as $x_{S_1} + \sum_{j=2}^{k} x_{S_j} - x_{S_{j-1}}$, and rewrite $F(\{S_k\}_{k=1}^{n})$ as $\sum_{j=k}^{n} F\left(\{S_p\}_{p=j}^{n}\right) - F\left(\{S_p\}_{p=j}^{n} \setminus \{S_j\}\right)$, i.e. the sum of *"marginal gains"*.

To see (4) is equivalent to (5), we may notice that all entries in *x* are non-negative, and thus we should maximize $y^T x$ by maximizing the product between the largest entry of *x* and the corresponding entry in *y*, i.e. $x_{S_n} y_{S_n}$, and then $x_{S_{n-1}} y_{S_{n-1}}$ and so on. This process of maximization is called *greedy algorithm*.

According to the lecture video, minimizing $||y||$ is equivalent to minimizing $\{F(p) | p \in P(\{k\}_{k=1}^{n})\}$, where *P* denotes the power set. I have not figured out why.

Possible reference? A paper written by Mr. Satoru Fujishige in 1980.
The context in this paper is a bit different from ours. It discusses a "rank function", which requires non-negative "marginal gain".

MATHEMATICS OF OPERATIONS RESEARCH
Vol. 5, No. 2, May 1980
Printed in U.S.A.

LEXICOGRAPHICALLY OPTIMAL BASE OF A POLYMATROID WITH RESPECT TO A WEIGHT VECTOR*†

SATORU FUJISHIGE

University of Tsukuba

Let (E, ρ) be a polymatroid with a ground set E and a rank function ρ. A base $\mathbf{x} = (x(e))_{e \in E}$ of polymatroid (E, ρ) is called a lexicographically optimal base of (E, ρ) with respect to a weight vector $\mathbf{w} = (w(e))_{e \in E}$ if the $|E|$-tuple of the numbers $x(e)/w(e)$ $(e \in E)$ arranged in order of increasing magnitude is lexicographically maximum among all $|E|$-tuples of numbers $y(e)/w(e)$ $(e \in E)$ arranged in the same manner for all bases $y = (y(e))_{e \in E}$ of (E, ρ). We give theorems that characterize the relationship between weight vectors and lexicographically optimal bases and point out that a lexicographically optimal base minimizes among all bases a quadratic objective function defined in terms of the associated weight vector. Also, we present an algorithm for finding the (unique) lexicographically optimal base with respect to a given weight vector. Furthermore, we consider the problem of determining the set of weight vectors with respect to which a given base is lexicographically optimal and provide an algorithm for solving it, which is useful for the sensitivity analysis of the optimal base with regard to the variation of the weight vector. The algorithms proposed in the present paper efficiently solve the problem, treated by N. Megiddo, of finding a lexicographically optimal flow in a network with multiple sources and sinks, which is a special case of the problem considered here.

Possible inference: minimax theorem.

It a well-known fact of optimization and game theory that $\min_A \max_B f(A, B) \geq \max_B \min_A f(A, B)$. Strong duality is related to the *minimax theorem* (see `https://en.wikipedia.org/wiki/Minimax_theorem`), which states that in fact

$$\min_{A} \max_{B} f(A, B) = \max_{B} \min_{A} f(A, B)$$

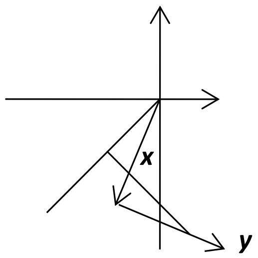
in the case that $f(A, \cdot)$ is concave in B and $f(\cdot, B)$ is convex in A.

(MA4270, Professor Jonathan Mark Scarlett)

This implies that $\min_x \max_y \langle x, y\rangle = \max_y \min_x \langle x, y\rangle = \max_{y=[y_i,\cdots,y_n]^T} \sum_{y_i<0} y_i$ for y in the base polytope. When $\|y\|$ decreases, the negative terms move towards the center. Thus we kind of have the same objective in both the minimization of $\|y\|$ and the maximization of the sum of the negative terms of *y*.

However, as can be seen in the algorithm, the minimum-norm-point algorithm actually creates some deviation when reducing $\|y\|$, which might not reduce $\sum_{y_i<0} y_i$.

An example in 2-D:
Sum of the negative terms is the product between [1,1] and *x*. When *x* moves in direction *y* perpendicular to it, its norm increases whereas the product between [1,1] and it decreases.



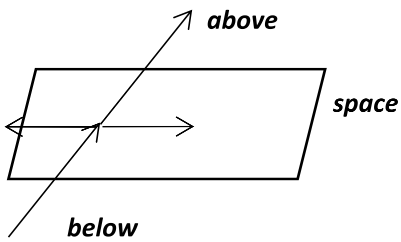
2. Implementation of the Minimum-Norm-Point Algorithm in Matlab

In matlab, $\hat{p} = [p_1, \cdots, p_n]^T$ is taken to be *f(-x)*, where *f* denotes the Lovasz Extension. This makes sense as it minimizes $x_{S_k} p_{S_k}$ for large *k*, and maximizes this product for small *k*.

What is an affine hull?
For *n* points in the polyhedron, $x_1,\ldots,x_n$, name x_k-x_1 as d_k, then the affine hull formed by $x_1,\ldots,x_n$ equals to the set of vectors $\{x_1+d\}$ where *d* is in the vector space formed by $d_2,\ldots,d_n$.

Another equivalent definition is $\{\sum_{k=1}^n a_k x_k | \sum_{k=1}^n a_k = 1\}$. It can be verified that any vector that can be written in one form can be written in another.

Thus in step 3, the minimum-norm point in the affine hull is *x-the projection of x onto the vector space formed by* $d_2,\ldots,d_n$.



Finally, how to find *z*?
We may assume that all the points in *S* are vertices on the boundary, because a point $\hat{p}$ in the interior of a convex hull does not minimize $\langle \hat{x}, \hat{p}\rangle$.

Thus $x_1, d_2, \ldots, d_n$ are linearly independent, and each index x_1 and d_k must be unique such that the linear combination of $x_1, d_2, \ldots, d_n$ add to $\hat{x}$ or y.

We may just find the minimum "fraction" of movement from $\hat{x}$ to y where one of the indices drop to 0.