A combinatorial algorithm minimizing submodular functions in strongly polynomial time

Alexander Schrijver[1]

Abstract. We give a strongly polynomial-time algorithm minimizing a submodular function f given by a value-giving oracle. The algorithm does not use the ellipsoid method or any other linear programming method. No bound on the complexity of the values of f is needed to be known a priori. The number of oracle calls is bounded by a polynomial in the size of the underlying set.

1. Introduction

A submodular function on a (finite) set V is a function f defined on the collection of subsets of V such that

(1) $$f(Y) + f(Z) \geq f(Y \cap Z) + f(Y \cup Z)$$

for all $Y, Z \subseteq V$.

Examples of submodular functions are the rank functions of matroids and the cut functions, which are functions f given by a directed graph $D = (V, A)$, with capacity function $c : A \to \mathbb{R}_+$, where $f(U)$ is equal to the total capacity of the arcs leaving U.

The importance of submodular functions for optimization was discovered by Edmonds [4], who found several important results on the related "polymatroids" and their intersections. For an introduction to submodular functions, with more examples, see Lovász [11], where it is also argued that submodular functions form a discrete analogue of convex functions.

Grötschel, Lovász, and Schrijver [7], [8] showed that a set U minimizing $f(U)$ can be found in strongly polynomial time, if f is given by a *value-giving oracle*, that is, an oracle that returns $f(U)$ for any given $U \subseteq V$. In [7], [8] it is assumed that f is rational-valued and that an upper bound β is known on the absolute values of the numerators and denominators of all $f(U)$. The algorithm in [7], [8] is based on the ellipsoid method, and uses therefore a heavy framework of division, rounding, and approximation; moreover, it is not practical.

The strongly polynomial-time algorithm presented in this paper is combinatorial, and no upper bound β as above is required to be known. Our algorithm is inspired by earlier work of Schönsleben [15], Lawler and Martel [10], Cunningham [2], [3], Bixby, Cunningham, and Topkis [1], and Frank [5], in particular by the combinatorial, strongly polynomial-time algorithm of [2] to minimize $r(U) - x(U)$, where r is the rank function of a matroid on V, where $x \in \mathbb{R}^V$ is a given vector, and where as usual $x(U) := \sum_{v \in U} x(v)$. Basic in Cunningham's method is to apply a lexicographic shortest path selection rule. Schönsleben [15] and Lawler and Martel [10] had shown that, for polymatroid intersection, this rule gives a polynomial bound on the number of iterations. The selection rule we give in our algorithm (the choice of t and s at the beginning of "Case 2" in Section 4 below) is in fact a simplified version of the lexicographic rule.

[1] CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands, and Department of Mathematics, University of Amsterdam, Plantage Muidergracht 24, 1018 TV Amsterdam, The Netherlands.

In [3], Cunningham extended the method to a pseudopolynomial-time algorithm for minimizing an arbitrary submodular function (for integer-valued submodular functions, it is polynomial-time in $|V|+\beta$). The reader familiar with Cunningham's papers will recognize several elements of them in the present paper.

For cut functions f and $x \in \mathbb{R}^V$, the problem of minimizing $f(U) - x(U)$ can be solved combinatorially with classical max-flow techniques (Rhys [14], Picard [12]).

Related is the combinatorial, strongly polynomial-time algorithm of Queyranne [13] that minimizes a symmetric submodular function f over the nonempty proper subsets. (f is *symmetric* if $f(U) = f(V \setminus U)$ for each $U \subseteq V$.)

The algorithm described below compares, adds, subtracts, multiplies and divides function values (among other, we solve certain systems of linear equations). One would wish to have a "fully combinatorial" algorithm, in which the function values are only compared, added, and subtracted. That is, one wishes to restrict the operations to ordered *group* operations, instead of ordered *field* operations. This requirement does not seem unreasonable for minimizing submodular functions, given what is known about such functions and related polyhedra (like the greedy method). The existence of such an algorithm is left open by this paper. (Both the algorithms of Cunningham [2] and Queyranne [13] are fully combinatorial, while that of Cunningham [3] is not.)

A useful reference for background on submodular functions is Fujishige [6]. The present paper is however self-contained.

2. Preliminaries

Let f be a submodular set function on a set V. That is, f is a real-valued function on the collection of all subsets of V, satisfying

$$(2) \qquad f(Y) + f(Z) \geq f(Y \cap Z) + f(Y \cup Z)$$

for all $Y, Z \subseteq V$. In finding the minimum value of f, we can assume $f(\emptyset) = 0$, as resetting $f(X) := f(X) - f(\emptyset)$ for all $X \subseteq V$ does not change the problem. So throughout we assume that $f(\emptyset) = 0$.

Moreover, we assume that any submodular function f on V is given by a *value-giving oracle*, that is, an oracle that returns $f(U)$ for any given subset U of V. We also assume that the numbers returned by the oracle are rational (or belong to any ordered field in which we can perform the operations algorithmically).

With a submodular function f on V, we associate the polytope B_f given by:

$$(3) \qquad B_f := \{x \in \mathbb{R}^V \mid x(U) \leq f(U) \text{ for all } U \subseteq V, x(V) = f(V)\}$$

(the 'base polytope'). Here, as usual,

$$(4) \qquad x(U) := \sum_{v \in U} x(v)$$

for $U \subseteq V$.

Consider any total order $\prec$ on V. For any $v \in V$, denote

$$(5) \qquad v_{\prec} := \{u \in V \mid u \prec v\}.$$

Define a vector $b^{\prec}$ in $\mathbb{R}^V$ by:

$$(6) \qquad b^{\prec}(v) := f(v_{\prec} \cup \{v\}) - f(v_{\prec})$$

for $v \in V$. Note that $b^{\prec}(U) = f(U)$ for each lower ideal U of $\prec$ (where a *lower ideal* of $\prec$ is a subset U of V such that if $u \in U$ and $w \prec u$ implies $w \in U$).

The vector $b^{\prec}$ can be considered as the output of the greedy method. It can be shown that $b^{\prec}$ is a vertex of B_f, and that each vertex of B_f can be obtained in this way.

However, in this paper, we do not need the geometric or algorithmic background of the $b^{\prec}$ — we only need that any $b = b^{\prec}$ constructed as above, belongs to B_f. This is not hard to derive from the submodularity of f. (Indeed, $b(U) \leq f(U)$ can be proved by induction on $|U|$: If $U = \emptyset$ it is trivial. If $U \neq \emptyset$, let v be the largest (with respect to $\prec$) element in U. Then, with the induction hypothesis and by the submodularity of f, $f(U) \geq f(U \cap v_{\prec}) + f(U \cup v_{\prec}) - f(v_{\prec}) = f(U \setminus \{v\}) + f(v_{\prec} \cup \{v\}) - f(v_{\prec}) \geq b(U \setminus \{v\}) + b(v_{\prec} \cup \{v\}) - b(v_{\prec}) = b(U \setminus \{v\}) + b(v) = b(U)$.)

3. A subroutine

In this section we describe a subroutine that is important in our algorithm. It replaces a total order $\prec$ by other total orders, thereby reducing some interval $(s,t]_{\prec}$, where

$$(7) \qquad (s,t]_{\prec} := \{v | s \prec v \preceq t\},$$

for $s, t \in V$.

Let $\prec$ be a total order on V. For any $s, u \in V$ with $s \prec u$, let $\prec^{s,u}$ be the total order on V obtained from $\prec$ by resetting $v \prec u$ to $u \prec v$ for each v satisfying $s \preceq v \prec u$. Thus in the ordering, we move u to the position just before s. (So $(s,t]_{\prec} = (s,t]_{\prec^{s,u}} \setminus \{u\}$ if $u \in (s,t]_{\prec}$.)

We first compare $b^{\prec^{s,u}}$ with $b^{\prec}$. We show that for each $v \in V$:

$$(8) \qquad \begin{aligned} &b^{\prec^{s,u}}(v) \leq b^{\prec}(v) \text{ if } s \preceq v \prec u, \\ &b^{\prec^{s,u}}(v) \geq b^{\prec}(v) \text{ if } v = u, \\ &b^{\prec^{s,u}}(v) = b^{\prec}(v) \text{ otherwise.} \end{aligned}$$

To prove this, observe that if $X \subseteq Y \subseteq V$, then for any $v \in V \setminus Y$ we have by the submodularity of f:

$$(9) \qquad f(Y \cup \{v\}) - f(Y) \leq f(X \cup \{v\}) - f(X),$$

as the union and intersection of $X \cup \{v\}$ and Y are equal to $Y \cup \{v\}$ and X respectively.

To see (8), if $s \preceq v \prec u$, then by (9),

$$(10) \qquad b^{\prec^{s,u}}(v) = f(v_{\prec^{s,u}} \cup \{v\}) - f(v_{\prec^{s,u}}) \leq f(v_{\prec} \cup \{v\}) - f(v_{\prec}) = b^{\prec}(v),$$

since $v_{\prec^{s,u}} = v_{\prec} \cup \{u\} \supset v_{\prec}$.

Similarly,

$$(11) \qquad b^{\prec^{s,u}}(u) = f(u_{\prec^{s,u}} \cup \{u\}) - f(u_{\prec^{s,u}}) \geq f(u_{\prec} \cup \{u\}) - f(u_{\prec}) = b^{\prec}(u),$$

since $u_{\prec^{s,u}} = s_{\prec} \subset u_{\prec}$.

Finally, if $v \prec s$ or $u \prec v$, then $v_{\prec^{s,u}} = v_{\prec}$, and hence $b^{\prec^{s,u}}(v) = b^{\prec}(v)$. This shows (8).

Let, for any $u \in V$, χ^u be the incidence vector of u. That is, $\chi^u(v) = 1$ if $v = u$ and $= 0$ otherwise.

Then we claim that there is a subroutine doing the following:

(12) for any $s, t \in V$ with $s \prec t$, we can find $\delta \geq 0$ and describe $b^{\prec} + \delta(\chi^t - \chi^s)$ as a convex combination of the $b^{\prec^{s,u}}$ for $u \in (s,t]_{\prec}$, in strongly polynomial time.

To describe the subroutine, we can assume that $b^{\prec} = 0$ (by replacing (temporarily) $f(U)$ by $f(U) - b^{\prec}(U)$ for each $U \subseteq V$).

By (8), the matrix $M = (b^{\prec^{s,u}}(v))_{u,v}$ with rows indexed by $u \in (s,t]_{\prec}$ and columns indexed by $v \in V$, in the order given by $\prec$, has the following, partially 'triangular', shape, where a $+$ means that the entry is ≥ 0, and a $-$ that the entry is ≤ 0:

$$
\begin{array}{c|cccccccccccccc}
 & & & & s & & & & & & & t & & & \\
\hline
 & 0 & \cdots & 0 & - & + & 0 & \cdots & \cdots & \cdots & 0 & 0 & 0 & \cdots & 0 \\
 & \vdots & & \vdots & - & - & + & \ddots & & & \vdots & \vdots & \vdots & & \vdots \\
 & \vdots & & \vdots & - & - & - & \ddots & \ddots & & \vdots & \vdots & \vdots & & \vdots \\
 & \vdots & & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & & \vdots \\
 & \vdots & & \vdots & \vdots & \vdots & \vdots & & \ddots & \ddots & 0 & 0 & \vdots & & \vdots \\
 & \vdots & & \vdots & \vdots & \vdots & \vdots & & & \ddots & + & 0 & \vdots & & \vdots \\
t & 0 & \cdots & 0 & - & - & - & \cdots & \cdots & \cdots & - & + & 0 & \cdots & 0
\end{array}
$$

As each row of M represents a vector $b^{\prec^{s,u}}$, to obtain (12) we must describe $\delta(\chi^t - \chi^s)$ as a convex combination of the rows of M, for some $\delta \geq 0$.

We call the $+$ entries in the matrix the 'diagonal' elements. Now for each row of M, the sum of its entries is 0, as $b^{\prec^{s,u}}(V) = f(V) = b^{\prec}(V) = 0$. Hence, if a 'diagonal' element $b^{\prec^{s,u}}(u)$ is equal to 0 for some $u \in (s,t]_{\prec}$, then the corresponding row of M is all-zero. So in this case we can take $\delta = 0$ in (12).

If $b^{\prec^{s,u}}(u) > 0$ for each $u \in (s,t]_{\prec}$ (that is, if each 'diagonal' element is strictly positive), then $\chi^t - \chi^s$ can be described as a nonnegative combination of the rows of M (by the sign pattern of M and since the entries in each row of M add up to 0). Hence $\delta(\chi^t - \chi^s)$ is a convex combination of the rows of M for some $\delta > 0$, yielding again (12).

4. Algorithm to minimize a submodular function f

Let f be a submodular set function on V. To minimize f, we can assume $f(\emptyset) = 0$. We assume also that $V = \{1, \ldots, n\}$.

We iteratively update a vector $x \in B_f$, given as a convex combination

(13) $$x = \lambda_1 b^{\prec_1} + \cdots + \lambda_k b^{\prec_k},$$

where the $\prec_i$ are total orders of V, and where the λ_i are positive and sum to 1. Initially, we choose an arbitrary total order $\prec$ and set $x = b^{\prec}$.

We describe the iteration. Consider the directed graph $D = (V, A)$, with

(14) $$A := \{(u,v) \mid \exists i = 1, \ldots, k : u \prec_i v\}.$$

Define

(15) $\qquad P := \{v \in V | x(v) > 0\}$ and $N := \{v \in V | x(v) < 0\}$.

Case 1: *D has no path from P to N.* Then let U be the set of vertices of D that can reach N by a directed path. So $N \subseteq U$ and $U \cap P = \emptyset$; that is, U contains all negative components of x and no positive components. Hence $x(W) \geq x(U)$ for each $W \subseteq V$. As no arcs of D enter U, U is a lower ideal of $\prec_i$, and hence $b^{\prec_i}(U) = f(U)$, for each $i = 1, \ldots, k$. Therefore for each $W \subseteq V$:

(16) $$f(W) \geq x(W) \geq x(U) = \sum_{i=1}^{k} \lambda_i b^{\prec_i}(U) = f(U);$$

so U minimizes f.

Case 2: *D has a path from P to N.* Let $d(v)$ denote the distance in D from P to v (= minimum number of arcs in a directed path from P to v). Choose $s, t \in V$ as follows.

Let t be the element in N reachable from P with $d(t)$ maximum, such that t is largest. Let s be the element with $(s,t) \in A$, $d(s) = d(t) - 1$, and s largest. Let α be the maximum of $|(s,t]_{\prec_i}|$ over $i = 1, \ldots, k$. Reorder indices such that $|(s,t]_{\prec_1}| = \alpha$.

By (12), we can find $\delta \geq 0$ and describe

(17) $$b^{\prec_1} + \delta(\chi^t - \chi^s)$$

as a convex combination of the $b^{\prec_1^{s,u}}$ for $u \in (s,t]_{\prec_1}$. Then with (13) we obtain

(18) $$y := x + \lambda_1 \delta(\chi^t - \chi^s)$$

as a convex combination of $b^{\prec_i}$ $(i = 2, \ldots, k)$ and $b^{\prec_1^{s,u}}$ $(u \in (s,t]_{\prec_1})$.

Let x' be the point on the line segment $\overline{xy}$ closest to y with $x'(t) \leq 0$. (So $x'(t) = 0$ or $x' = y$.) We can describe x' as a convex combination of $b^{\prec_i}$ $(i = 1, \ldots, k)$ and $b^{\prec_1^{s,u}}$ $(u \in (s,t]_{\prec_1})$. Moreover, if $x'(t) < 0$ then we can do without $b^{\prec_1}$.

We reduce the number of terms in the convex decomposition of x' to at most $|V|$ by linear algebra: any affine dependence of the vectors in the decomposition yields a reduction of the number of terms in the decomposition, as in the standard proof of Carathéodory's theorem (subtract an appropriate multiple of the linear expression giving the affine dependence, from the linear expression giving the convex combination, so that all coefficients remain nonnegative, and at least one becomes 0). As all $b^{\prec}$ belong to a hyperplane, this reduces the number of terms to at most $|V|$.

Then, after resetting $x := x'$, we iterate. This finishes the description of the algorithm.

5. Running time of the algorithm

We show that the number of iterations is at most $|V|^6$. Consider any iteration. Let

(19) $\qquad \beta :=$ number of $i \in \{1, \ldots, k\}$ with $|(s,t]_{\prec_i}| = \alpha$.

Let x', d', A', P', N', t', s', α', β' be the objects x, d, A, P, N, t, s, α, β in the next iteration. Then

(20) $\qquad$ for all $v \in V$, $d'(v) \geq d(v)$,

and

(21) if $d'(v) = d(v)$ for all $v \in V$, then $(d'(t'), t', s', \alpha', \beta')$ is lexicographically less than $(d(t), t, s, \alpha, \beta)$.

Since each of $d(t), t, s, \alpha, \beta$ is at most $|V|$, and since (if $d(v)$ is unchanged for all v) there are at most $|V|$ pairs $(d(t), t)$, (21) implies that in at most $|V|^4$ iterations $d(v)$ increases for some v. Any fixed v can have at most $|V|$ such increases, and hence the number of iterations is at most $|V|^6$.

Notice that

(22) for each arc $(v, w) \in A' \setminus A$ we have $s \preceq_1 w \prec_1 v \preceq_1 t$.

Indeed, as $(v, w) \notin A$ we have $w \prec_1 v$. As $(v, w) \in A'$, we have $v \prec_1^{s,u} w$ for some $u \in (s, t]_{\prec_1}$. Hence the definition of $\prec_1^{s,u}$ gives $v = u$ and $s \preceq_1 w \prec_1 u$. This shows (22).

If (20) does not hold, then $A' \setminus A$ contains an arc (v, w) with $d(w) \geq d(v) + 2$ (using that $P' \subseteq P$). By (22), $s \preceq_1 w \prec_1 v \preceq_1 t$, and so $d(w) \leq d(s) + 1 = d(t) \leq d(v) + 1$, a contradiction. This shows (20).

To prove (21), assume that $d'(v) = d(v)$ for all $v \in V$. As $x'(t') < 0$, we have $x(t') < 0$ or $t' = s$. So by our criterion for choosing t (maximizing $(d(t), t)$ lexicographically), and since $d(s) < d(t)$, we know that $d(t') \leq d(t)$, and that if $d(t') = d(t)$ then $t' \leq t$.

Next assume also that $d(t') = d(t)$ and $t' = t$. As $(s', t) \in A'$, and as (by (22)) $A' \setminus A$ does not contain any arc entering t, we have $(s', t) \in A$, and so $s' \leq s$, by the maximality of s.

Finally assume also that $s' = s$. As $(s, t]_{\prec_1^{s,u}}$ is a proper subset of $(s, t]_{\prec_1}$ for each $u \in (s, t]_{\prec_1}$, we know that $\alpha' \leq \alpha$. Moreover, if $\alpha' = \alpha$, then $\beta' < \beta$, since $\prec_1$ does not occur anymore among the linear orders making the convex combination, as $x'(t) < 0$. This proves (21).

Note. Above we have chosen t and s largest possible, in some fixed order of V. To obtain the above running time bound it only suffices to choose t and s in a consistent way. That is, if the set of choices for t is the same as in the previous iteration, then we should choose the same t — and similarly for s.

6. Ring families

Any algorithm minimizing a submodular function, can be transformed to an algorithm minimizing a submodular function defined on a ring family $\mathcal{C}$, that is, a collection $\mathcal{C}$ of subsets of V closed under union and intersection (where the submodular inequality (2) is required only for sets in $\mathcal{C}$). For this, we need to know in advance for each $v \in V$ the minimal set M_v in $\mathcal{C}$ containing v (if any), and we need to know the smallest set M in $\mathcal{C}$. This fully describes $\mathcal{C}$. (Obviously, we need to have *some* information on $\mathcal{C}$ in advance. Otherwise, it may take exponential time until the oracle will return to us any (finite) value.)

We can assume that $M = \emptyset$, $V \in \mathcal{C}$, and that $M_u \neq M_v$ for all $u \neq v$ (otherwise we can identify u and v). Thus we can represent $\mathcal{C}$ as the collection of all lower ideals of some partial order on V.

For each $v \in V$, define L_v to be the largest set in $\mathcal{C}$ not containing v (so L_v is the union of those M_u not containing v), and

$$(23) \qquad c(v) := \max\{0, f(L_v) - f(L_v \cup \{v\})\}.$$

(We can find c by $2|V|$ oracle calls.)

Then for all $X, Y \in \mathcal{C}$ with $X \subseteq Y$ one has

$$(24) \qquad f(Y) + c(Y) \geq f(X) + c(X).$$

To show this, we can assume that $Y = X \cup \{v\}$ for some $v \notin X$. So $Y \cup L_v = L_v \cup \{v\}$ and $Y \cap L_v = X$, and

$$(25) \qquad f(X) = f(Y \cap L_v) \leq f(Y) + f(L_v) - f(Y \cup L_v) \leq f(Y) + c(v).$$

This shows (24). So the function $f(X) + c(X)$ is monotone in X.

For any subset X of V let $\overline{X}$ be the smallest set in $\mathcal{C}$ containing X, and define g by

$$(26) \qquad g(X) := f(\overline{X}) + c(\overline{X})$$

for $X \subseteq V$. Then g is submodular, since for $X, Y \subseteq V$:

$$(27) \qquad \begin{aligned} g(X) + g(Y) &= f(\overline{X}) + c(\overline{X}) + f(\overline{Y}) + c(\overline{Y}) \geq f(\overline{X} \cap \overline{Y}) + f(\overline{X} \cup \overline{Y}) + c(\overline{X}) + c(\overline{Y}) \\ &= f(\overline{X} \cap \overline{Y}) + f(\overline{X} \cup \overline{Y}) + c(\overline{X} \cap \overline{Y}) + c(\overline{X} \cup \overline{Y}) \\ &\geq f(\overline{X \cap Y}) + c(\overline{X \cap Y}) + f(\overline{X \cup Y}) + c(\overline{X \cup Y}) = g(X \cap Y) + g(X \cup Y), \end{aligned}$$

where the first inequality follows from the submodularity of f, and the last inequality from (24) (note that $\overline{X} \cup \overline{Y} = \overline{X \cup Y}$ and $\overline{X} \cap \overline{Y} \supseteq \overline{X \cap Y}$).

Now with our algorithm we can find a subset U of V minimizing $g(U) - c(U)$. Then for each $T \in \mathcal{C}$ we have $f(T) = g(T) - c(T) \geq g(U) - c(U) = f(\overline{U}) + c(\overline{U}) - c(U) \geq f(\overline{U})$, as $c \geq 0$. Thus $\overline{U}$ minimizes f over $\mathcal{C}$.

Notes. Simultaneously with us, also Iwata, Fleischer, and Fujishige [9] found a non-ellipsoidal algorithm to minimize a submodular function.

I thank András Frank and Bert Gerards for stimulating discussion and suggestions, and Lisa Fleischer for helpful remarks on an earlier draft of this paper and for showing me that the number of iterations is bounded by $|V|^6$ (instead of $|V|^7$). I thank Bianca Spille for carefully reading the manuscript and for pointing out some errors. I also thank the referees for very useful suggestions improving the presentation.

References

[1] R.E. Bixby, W.H. Cunningham, D.M. Topkis, The partial order of a polymatroid extreme point, *Mathematics of Operations Research* 10 (1985) 367–378.

[2] W.H. Cunningham, Testing membership in matroid polyhedra, *Journal of Combinatorial Theory, Series B* 36 (1984) 161–188.

[3] W.H. Cunningham, On submodular function minimization, *Combinatorica* 5 (1985) 185–192.

[4] J. Edmonds, Submodular functions, matroids, and certain polyhedra, in: *Combinatorial Structures and Their Applications* (Proceedings Calgary International Conference on Combinatorial Structures and Their Applications, Calgary, Alberta, June 1969; R. Guy, H. Hanani, N. Sauer, J. Schönheim, eds.), Gordon and Breach, New York, 1970, pp. 69–87.

[5] A. Frank, Finding feasible vectors of Edmonds-Giles polyhedra, *Journal of Combinatorial Theory, Series B* 36 (1984) 221–239.

[6] S. Fujishige, *Submodular Functions and Optimization* [Annals of Discrete Mathematics, 47], North-Holland, Amsterdam, 1991.

[7] M. Grötschel, L. Lovász, A. Schrijver, The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica* 1 (1981) 169–197 [corrigendum: *Combinatorica* 4 (1984) 291–295].

[8] M. Grötschel, L. Lovász, A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer, Berlin, 1988.

[9] S. Iwata, L. Fleischer, S. Fujishige, A strongly polynomial-time algorithm for minimizing submodular functions, preprint, 1999.

[10] E.L. Lawler, C.U. Martel, Computing maximal “polymatroidal” network flows, *Mathematics of Operations Research* 7 (1982) 334–347.

[11] L. Lovász, Submodular functions and convexity, in: *Mathematical Programming — The State of the Art* (Bonn, 1982; A. Bachem, M. Grötschel, B. Korte, eds.), Springer, Berlin, 1983, pp. 234–257.

[12] J.-C. Picard, Maximal closure of a graph and applications to combinatorial problems, *Management Science* 22 (1976) 1268–1272.

[13] M. Queyranne, Minimizing symmetric submodular functions, *Mathematical Programming* 82 (1998) 3–12.

[14] J.M.W. Rhys, A selection problem of shared fixed costs and network flows, *Management Science* 17 (1970) 200–207.

[15] P. Schönsleben, *Ganzzahlige Polymatroid-Intersektions-Algorithmen*, Ph.D. Thesis, Eidgenössische Technische Hochschule Zürich, 1980.