Active-set Methods for Submodular Minimization Problems

K. S. Sesh Kumar, Francis Bach

Active-set Methods for Submodular Minimization Problems

K. S. Sesh Kumar
INRIA-Sierra project-team
Département d'Informatique
de l'Ecole Normale Supérieure
Paris, France
sesh-kumar.karri@inria.fr

Francis Bach
INRIA-Sierra project-team
Département d'Informatique
de l'Ecole Normale Supérieure
Paris, France
francis.bach@inria.fr

November 18, 2016

Abstract

We consider submodular optimization problems such as submodular function minimization(SFM) and the quadratic problems regularized by the Lovász extension; for cut functions, this corresponds respectively to graph cuts and total variation (TV) denoising. Given a submodular function with an SFM oracle, we propose a new active-set algorithm for total variation denoising, which is more flexible than existing ones; the algorithm may be seen as a local descent algorithm over ordered partitions with explicit convergence guarantees. For functions that decompose into the sum of two functions F_1 and F_2 with efficient SFM oracles, we propose a new active-set algorithm for total variation denoising (and hence for SFM by thresholding the solution at zero). This algorithm also optimizes over ordered partitions and improves empirically over existing ones based on TV or SFM oracles for F_1 and F_2.

1 Introduction

Submodular optimization problems such as total variation denoising and submodular function minimization are convex optimization problems which are common in computer vision, signal processing and machine learning [2], with notable applications to graph cut-based image segmentation [9], sensor placement [21], or document summarization [25].

In this paper, we consider a normalized submodular function F defined on $V = \{1, \dots, n\}$, i.e., $F : 2^V \to \mathbb{R}$ such that $F(\varnothing) = 0$ and an n-dimensional real vector u, i.e., $u \in \mathbb{R}^n$. We aim at minimizing with respect to $w \in \mathbb{R}^n$:

$$f(w) - u^\top w + \tfrac{1}{2}\|w\|_2^2, \tag{1}$$

where f is the Lovász extension of F. If F is a cut function in a weighted undirected graph, then f is its total variation, hence the denomination of *total variation denoising* problem for Eq. (1), which we use in this paper—since it is equivalent to minimizing $\frac{1}{2}\|u - w\|_2^2 + f(w)$.

We also consider the general submodular function minimization (SFM) problem:

$$\min_{w \in [0,1]^n} f(w) - u^\top w = \min_{A \subseteq V} F(A) - u(A), \tag{2}$$

where we use the convention $u(A) = u^\top 1_A$, with $1_A \in \{0,1\}^n$ is the indicator vector of the set A. Our goal in this paper is to propose iterative algorithms to solve these two problems given certain oracles on the submodular function F. Note that our algorithms minimize general submodular functions as any submodular

function can be decomposed into a normalized submodular function, F, i.e., $F(\varnothing) = 0$ and a modular function, u(see [2]).

Relationship with existing work. Generic algorithms to optimize SFM in Eq. (2) or TV in Eq. (1) problems which only access F through function values (e.g., subgradient descent or min-norm-point algorithm) are too slow without any assumptions [2], as for signal processing applications, high precision is typically required (and often the exact solution).

For decomposable problems, i.e., when $F = F_1 + \cdots + F_r$, where each F_j is "simple", the algorithms use more powerful oracles than function evaluations improving the running times. [30] used SFM oracles instead of function value oracles but they still remain slow in practice. However, when total variation oracles for each F_j are used, they become competitive [20, 22, 18]. Note that, in general, the exact total variation oracles are at most $O(n)$ times expensive than their SFM oracles. However, there are a subclass of submodular functions (cut functions and other submodular functions that can be written in form of cuts) whose total variation oracles are only $O(1)$ times expensive than the corresponding SFM oracles but are still too expensive in practice.Therefore, the goal is to design fast optimization strategies using efficient SFM oracles for each function F_j rather than their expensive TV oracles [22, 18] to solve the SFM and TV denoising problems of F given by Eq. (2) and Eq. (1) respectively. An algorithm was proposed by [24] to search over partition space for solving Eq. (1) with the unary terms $(-u^\top w)$ replaced by a convex differentiable function but restricting to cut functions.

In this paper, we exploit the polytope structure of these non-smooth optimization problems, where each face is indexed by an ordered partition of the underlying ground set $V = \{1, \dots, n\}$. The main insight of this paper is that once given a face of the main polytope associated with the submodular function (namely the base polytope described in Section 2) and its tangent cone, orthogonal projections may be done in linear time by isotonic regressions. We will only need SFM oracles, i.e., the minimization of $F(A) - s(A)$ with respect to $A \subseteq V$ for all possible $s \in \mathbb{R}^n$, to check optimality of this partition and/or generate a new partition.

Contributions. We make two main contributions:

- Given a submodular function F with an SFM oracle, we propose a new active-set algorithm for total variation denoising in Section 3, which is more efficient and flexible than existing ones (e.g., it allows warm restarts). This algorithm may be seen as a local descent algorithm over ordered partitions.
- Given a decomposition of $F = F_1 + F_2$, with available SFM oracles for each F_j, we propose an active-set algorithm for total variation denoising in Section 4 (and hence for SFM by thresholding the solution at zero). These algorithms optimize over ordered partitions (one per function F_j). Following [18, 22], they are also naturally parallelizable. Given that only SFM oracles are needed, it is much more flexible, and allow more applications as shown in Section 5.

2 Review of Submodular Analysis

A set-function $F : 2^V \to \mathbb{R}$ is submodular if $F(A) + F(B) \geqslant F(A \cup B) + F(A \cap B)$ for any subsets A, B of V. Our main motivating examples in this paper are cuts in a weighted undirected graph with weight function $a : V \times V \to \mathbb{R}_+$, which can be defined as

$$F(A) = \sum_{i<j} a(i,j) |1_{i \in A} - 1_{j \notin A}|, \tag{3}$$

where $i, j \in V$. Note that there are other submodular functions on which our algorithm works, e.g., concave function on the cardinality set, which *cannot* be represented in the form of Eq. (3) [23, 19]. However, we use cut functions as a running example to better explain our algorithm as they are most widely studied and

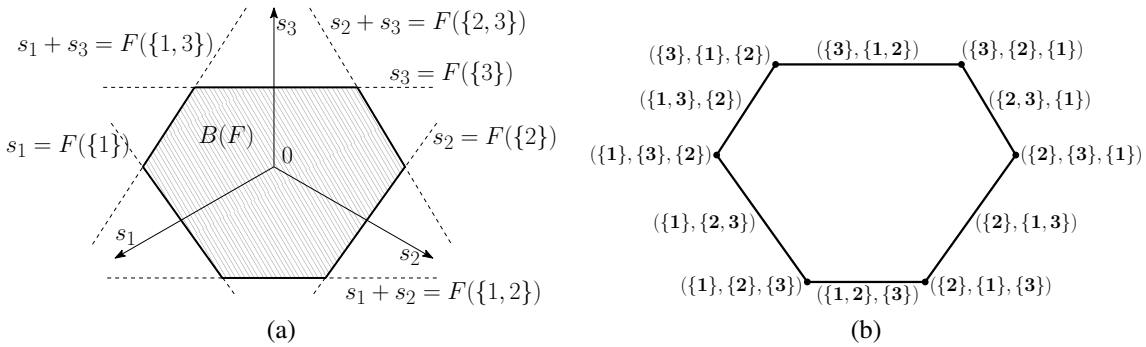


Figure 1: Base polytope for $n=3$. (a) definition from its supporting hyperplanes $\{s(A) = F(A)\}$. (b) each face (point or segment) of $B(F)$ is associated with an ordered partition.

understood among submodular functions. We now review the relevant concepts from submodular analysis (for more details, see [2, 14]).

Lovász extension and convexity. The power set 2^V is naturally identified with the vertices $\{0,1\}^n$ of the hypercube in n dimensions (going from $A \subseteq V$ to $1_A \in \{0,1\}^n$). Thus, any set-function may be seen as a function f on $\{0,1\}^n$. It turns out that f may be extended to the full hypercube $[0,1]^n$ (and then to $\mathbb{R}^n$) by piecewise-linear interpolation, and then to the whole vector space $\mathbb{R}^n$.

Given a vector $w \in \mathbb{R}^n$, and given its *unique* level-set representation as $w = \sum_{i=1}^m v_i 1_{A_i}$, with $(A_1, \dots, A_m)$ a partition of V and $v_1 > \cdots > v_m$, $f(w)$ is equal to $f(w) = \sum_{i=1}^m v_i \big[F(B_i) - F(B_{i-1})\big]$, where $B_i = (A_1 \cup \ldots \cup A_i)$. For cut functions, the Lovász extension happens to be equal to the *total variation*, $f(w) = \sum_{i<j} a(i,j)|w_i - w_j|$, hence our denomination total variation denoising for the problem in Eq. (1).

This extension is piecewise linear for any set-function F. It turns out that it is convex if and only if F is submodular [26]. Any piecewise linear convex function may be represented as the support function of a certain polytope K, i.e., as $f(w) = \max_{s \in K} w^\top s$ [28]. For the Lovász extension of a submodular function, we have an explicit description of K, which we now review.

Base polytope. We define the *base polytope* as

$$B(F) = \big\{ s \in \mathbb{R}^n, \ s(V) = F(V), \ \forall A \subset V, s(A) \leqslant F(A) \big\}.$$

Given that it is included in the affine hyperplane $\{s(V) = F(V)\}$, it is traditionally represented projected on that hyperplane (see Figure 1 (a)). A key result in submodular analysis is that the Lovász extension is the support function of $B(F)$, that is, for any $w \in \mathbb{R}^n$,

$$f(w) = \sup_{s \in B(F)} w^\top s. \tag{4}$$

The maximizers above may be computed in closed form from an ordered level-set representation of w using a greedy algorithm, which (a) first sorts the elements of w in decreasing order such that $w_{\sigma(1)} \geq \ldots \geq w_{\sigma(n)}$ where σ represents the order of the elements in V; and (b) computes $s_{\sigma(k)} = F(\{\sigma(1), \dots, \sigma(k)\}) - F(\{\sigma(1), \dots, \sigma(k-1)\})$.

SFM as a convex optimization problem. Another key result of submodular analysis is that minimizing a submodular function F (i.e., minimizing the Lovász extension f on $\{0,1\}^n$), is equivalent to minimizing the Lovász extension f on the full hypercube $[0,1]^n$ (a convex optimization problem). Moreover, with convex duality we have

$$\min_{A \subseteq V} F(A) - u(A) \quad = \quad \min_{w \in \{0,1\}^n} f(w) - u^\top w$$

$$
\begin{aligned}
&= \min_{w \in [0,1]^n} f(w) - u^\top w \\
&= \min_{w \in [0,1]^n} \max_{s \in B(F)} s^\top w - u^\top w \\
&= \max_{s \in B(F)} \min_{w \in [0,1]^n} s^\top w - u^\top w \\
&= \max_{s \in B(F)} \sum_{i=1}^n \min\{s_i - u_i, 0\}.
\end{aligned}
$$

This dual problem allows to obtain certificates of optimality for the primal-dual pairs $w \in [0,1]^n$ and $s \in B(F)$ using the quantity

$$\text{gap}(w,s) := f(w) - u^\top w - \min_{i=1}^n \{s_i - u_i, 0\},$$

which is always non-negative. It is equal to zero only at optimal and the corresponding (w,s) form the optimal primal-dual pairs.

Total variation denoising as projection onto the base polytope. A consequence of the representation of f as a support function leads to the following primal/dual pair [2, Sec. 8]:

$$
\begin{aligned}
\min_{w \in \mathbb{R}^n} f(w) - u^\top w + \tfrac{1}{2}\|w\|_2^2 &= \min_{w \in \mathbb{R}^n} \max_{s \in B(F)} s^\top w - u^\top w + \tfrac{1}{2}\|w\|_2^2 \text{ using Eq. (4)}, \\
&= \max_{s \in B(F)} \min_{w \in \mathbb{R}^n} s^\top w - u^\top w + \tfrac{1}{2}\|w\|_2^2, \\
&= \max_{s \in B(F)} -\tfrac{1}{2}\|s - u\|_2^2, \qquad (5)
\end{aligned}
$$

with $w = u - s$ at optimality. Thus the TV problem is equivalent to the orthogonal projection of u onto $B(F)$.

From TV denoising to SFM. The SFM problem in Eq. (2) and the TV problem in Eq. (1) are tightly connected. Indeed, given the unique solution w of the TV problem, then we obtain a solution of $\min_{A \subseteq V} F(A) - u(A)$ by thresholding w at 0, i.e., by taking $A = \{i \in V, w_i \geqslant 0\}$ [13].

Conversely, one may solve the TV problem by an appropriate sequence of SFM problems. The original divide-and-conquer algorithm may involve $O(n)$ SFM problems [17]. The extended algorithm of [18] can reach a precision ε in $O(\log \frac{1}{\varepsilon})$ but can only get the exact solution in $O(n)$ oracles. Fast efficient algorithms are proposed to solve TV problems with $O(1)$ oracles [11, 16] but are specific to cut functions on simple graphs (chains and trees) as they exploit its weight representation given by Eq. (3). Our algorithm in Section 3 is a generalization of the divide-and-conquer strategy for solving the TV problem with general submodular functions.

3 Ordered Partitions and Isotonic Regression

The main insight of this paper is (a) to consider the detailed face structure of the base polytope $B(F)$ and (b) to notice that for the outer approximation of $B(F)$ based on the tangent cone to a certain face, the orthogonal projection problem (which is equivalent to constrained TV denoising) may be solved efficiently using a simple algorithm, originally proposed to solve isotonic regression in linear time. This allows an explicit efficient local search over ordered partitions.

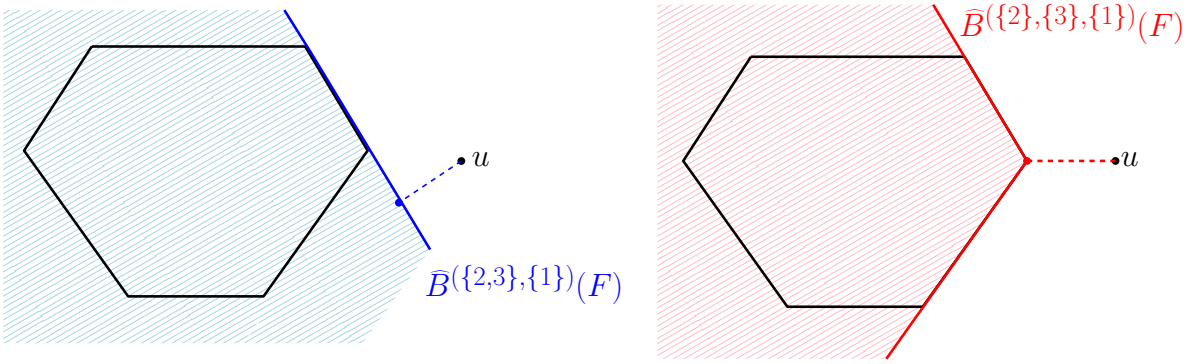


Figure 2: Projection algorithm for a single polytope: first projecting on the outer approximation $\widehat{B}^{(\{2,3\},\{1\})}(F)$, with a projected element which is not in $B(F)$ (blue), then on $\widehat{B}^{(\{2\},\{3\},\{1\})}(F)$, with a projected element being the projection of s onto $B(F)$ (red).

3.1 Outer approximations of $B(F)$

Supporting hyperplanes. The base polytope is defined as the intersection of half-spaces $\{s(A) \leqslant F(A)\}$, for $A \subseteq V$. Therefore, faces of $B(F)$ are indexed by subsets of the power set. As a consequence of submodularity [2, 14], the faces of the base polytope $B(F)$ are characterized by "ordered partitions" $\mathcal{A} = (A_1, \dots, A_m)$ with $V = A_1 \cup \cdots \cup A_m$. Then, a face of $B(F)$ is such that $s(B_i) = F(B_i)$ for all $B_i = A_1 \cup \cdots \cup A_i$, $i = 1, \dots, m$. See the Figure 1 (b) for the enumeration of faces for $n = 3$ based on an enumeration of all ordered partitions. Such ordered partitions are associated to vectors $w = \sum_{i=1}^m v_i 1_{A_i}$ with $v_1 > \ldots > v_m$ with all solutions of $\max_{s \in B(F)} w^\top s$ being on the corresponding face.

From a face of $B(F)$ defined by the ordered partition $\mathcal{A}$, we may define its *tangent cone* $\widehat{B}^{\mathcal{A}}(F)$ at this face as the set

$$\widehat{B}^{\mathcal{A}}(F) = \Big\{ s \in \mathbb{R}^n, s(V) = F(V), \forall i \in \{1, \dots, m-1\}, s(B_i) \leqslant F(B_i) \Big\}. \tag{6}$$

These are outer approximations of $B(F)$, as illustrated in Figure 2 for two ordered partitions.

Support function. We may compute the support function of $\widehat{B}^{\mathcal{A}}(F)$, which should be an upper bound on $f(w)$ since this set is an outer approximation of $B(F)$ as follows:

$$\begin{aligned}
\sup_{s \in \widehat{B}^{\mathcal{A}}(F)} w^\top s &= \sup_{s \in \mathbb{R}^n} \inf_{\lambda \in \mathbb{R}_+^{m-1} \times \mathbb{R}} w^\top s - \sum_{i=1}^m \lambda_i (s(B_i) - F(B_i)) \\
&\qquad \text{(using Lagrangian duality)}, \\
&= \inf_{\lambda \in \mathbb{R}_+^{m-1} \times \mathbb{R}} \sup_{s \in \mathbb{R}^n} s^\top \Big(w - \sum_{i=1}^m (\lambda_i + \cdots + \lambda_m) 1_{A_i} \Big) \\
&\qquad + \sum_{i=1}^m (\lambda_i + \cdots + \lambda_m) \big[F(B_i) - F(B_{i-1}) \big], \\
&= \inf_{\lambda \in \mathbb{R}_+^{m-1} \times \mathbb{R}} \sum_{i=1}^m (\lambda_i + \cdots + \lambda_m) \big[F(B_i) - F(B_{i-1}) \big] \\
&\qquad \text{such that } w = \sum_{i=1}^m (\lambda_i + \cdots + \lambda_m) 1_{A_i}.
\end{aligned}$$

Thus, by defining $v_i = \lambda_i + \cdots + \lambda_m$, which are decreasing, the support function is finite for w having ordered level sets corresponding to the ordered partition $\mathcal{A}$ (we then say that w is *compatible* with $\mathcal{A}$), i.e., $w = \sum_{i=1}^m v_i 1_{A_i}$; the support functions is equal to the Lovász extension $f(w)$. Otherwise, when w is not compatible with $\mathcal{A}$, the support function is infinite.

Let us now denote $\mathcal{W}^{\mathcal{A}}$ as a set of all weight vectors w that are compatible with the ordered partition $\mathcal{A}$. This can be defined as

$$\mathcal{W}^{\mathcal{A}} = \{w \in \mathbb{R}^n \mid \exists v \in \mathbb{R}^m, w = \sum_{i=1}^m v_i 1_{A_i}, v_1 \geq \ldots \geq v_m\}.$$

Therefore,

$$\sup_{s \in \widehat{B}^{\mathcal{A}}(F)} w^\top s = \begin{cases} f(w) & \text{if } w \in \mathcal{W}^{\mathcal{A}}, \\ \infty & \text{otherwise.} \end{cases} \tag{7}$$

3.2 Isotonic regression for restricted problems

Given an ordered partition $\mathcal{A} = (A_1, \ldots, A_m)$ of V, we consider the original TV problem restricted to w in $\mathcal{W}^{\mathcal{A}}$. Since on this constraint set $f(w) = \sum_{i=1}^m v_i \big[F(B_i) - F(B_{i-1})\big]$ is a linear function, this is equivalent to

$$\min_{v \in \mathbb{R}^m} \sum_{i=1}^m v_i \big[F(B_i) - F(B_{i-1}) - u(A_i)\big] + \tfrac{1}{2} \textstyle\sum_{i=1}^m |A_i| v_i^2 \text{ such that } v_1 \geqslant \cdots \geqslant v_m. \tag{8}$$

This may be done by isotonic regression in complexity $O(m)$ by the weighted pool-adjacent-violator algorithm [7]. Typically the solution v will have some values that are equal to each other, which corresponds to merging some sets A_i. If these merges are made, we now obtain a *basic ordered partition*[1] such that our optimal w has *strictly decreasing* values. Primal stationarity leads to explicit values of v given by $v_i = u(A_i)/|A_i| - (F(B_i) - F(B_{i-1}))/|A_i|$, i.e., given $\mathcal{A}$, the exact solution of the TV problem may be obtained in closed form.

Dual interpretation. Eq. (8) is a constrained TV denoising problem that minimises the cost function in Eq. (1) but with the constraint that weights are compatible with the ordered partition $\mathcal{A}$, i.e. $\min_{w \in \mathcal{W}^{\mathcal{A}}} f(w) - u^\top w + \frac{1}{2}\|w\|_2^2$. The dual of the problem can be derived exactly the same way as shown in Eq. (5) in the previous section, using the definition of the support function defined by Eq. (7). The corresponding dual is given by $\max_{s \in \widehat{B}^{\mathcal{A}}(F)} -\frac{1}{2}\|s-u\|_2^2$, with the relationship $w = u - s$ at optimality. Thus, this corresponds to projecting u on the outer approximation of the base polytope, $\widehat{B}^{\mathcal{A}}(F)$, which only has m constraints instead of the $2^n - 1$ constraints defining $B(F)$. See an illustration in Figure 2.

3.3 Checking optimality of a basic ordered partition

Given a basic ordered partition $\mathcal{A}$, the associated $w \in \mathbb{R}^n$ is optimal for the TV problem in Eq. (1) if and only if $s = u - w \in B(F)$ due to optimality conditions in Eq. (5), which can be checked by minimizing the submodular function $F - s$. For a basic partition, a more efficient algorithm is available.

By repeated application of submodularity, we have for all sets $C \subseteq V$, if $C_i = C \cap A_i$:

[1] Given a submodular function F and an ordered partition $\mathcal{A}$, when the unique solution problem in Eq. (8) is such that $v_1 > \cdots > v_m$, we say that we $\mathcal{A}$ is a *basic ordered partition* for $F - u$. Given any ordered partition, isotonic regression allows to compute a coarser partition (obtained by partially merging some sets) which is basic.

$$
\begin{aligned}
F(C) - s(C) &= F(V \cap C) - \sum_{i=1}^{m} s(C_i) \text{ (as } s \text{ is a modular function)}, \\
&= F(B_m \cap C) - \sum_{i=1}^{m} s(C_i) \\
&\quad + \sum_{i=1}^{m-1} F(B_i \cap C) - F(B_i \cap C) \text{ (as } B_m = V), \\
&= \sum_{i=1}^{m} F(B_i \cap C) - F(B_{i-1} \cap C) - s(C_i) \\
&\qquad \text{(let } B_0 = \varnothing \text{ and as } F(\varnothing) = 0), \\
&= \sum_{i=1}^{m} F((B_{i-1} \cup A_i) \cap C) - F(B_{i-1} \cap C) - s(C_i) \\
&\qquad \text{(since } B_i = B_{i-1} \cup A_i), \\
&= \sum_{i=1}^{m} F((B_{i-1} \cap C) \cup (A_i \cap C)) - F(B_{i-1} \cap C) - s(C_i), \\
&= \sum_{i=1}^{m} F((B_{i-1} \cap C) \cup C_i) - F(B_{i-1} \cap C) - s(C_i), \\
&\geqslant \sum_{i=1}^{m} \big[F(B_{i-1} \cup C_i) - F(B_{i-1}) - s(C_i)\big] \\
&\qquad \text{(as } (B_{i-1} \cap C) \subseteq B_{i-1} \text{ and due to submodularity of } F).
\end{aligned}
$$

Moreover, we have $s(A_i) = F(B_i) - F(B_{i-1})$, which implies $s(B_i) = F(B_i)$ for all $i \in \{1, \dots, m\}$, and thus all subproblems $\min_{C_i \subseteq A_i} F(B_{i-1} \cup C_i) - F(B_{i-1}) - s(C_i)$ have non-positive values. This implies that we may check optimality by solving these m subproblems: s is optimal if and only if all of them have zero values. This leads to smaller subproblems whose overall complexity is less than a single SFM oracle call. Moreover, for cut functions, it may be solved by a single oracle call on a graph where some edges have been removed [31].

Given all sets C_i, we may then define a new ordered partition by splitting all A_i for which $F(B_{i-1} \cup C_i) - F(B_{i-1}) - s(C_i) < 0$. If no split is possible, the pair (w, s) is optimal for Eq. (1). Otherwise, this new strictly finer partition may not be basic, the value of the optimization problem in Eq. (8) is strictly lower as shown in Section 3.5 (and leads to another basic ordered partition), which ensures the finite convergence of the algorithm.

3.4 Active-set algorithm

This leads to the novel active-set algorithm below.

- **Input**: Submodular function F with SFM oracle, $u \in \mathbb{R}^n$, ordered partition $\mathcal{A}$
- **Algorithm**: iterate until convergence
  (a) Solve Eq. (8) by isotonic regression.
  (b) Merge the sets with equal values of v_i to define a new ordered partition $\mathcal{A}$. Define $w = \sum_{i=1}^{m} v_i 1_{A_i}$ and $s = u - w$.
  (c) Check optimality by solving $\min_{C_i \subseteq A_i} F(B_{i-1} \cup C_i) - F(B_{i-1}) - s(C_i)$ for $i \in \{1, \dots, m\}$.

(d) If s not optimal, for all C_i which are different from $\varnothing$ and A_i, add the new set $B_{i-1} \cup C_i$ in the ordered partition $\mathcal{A}$.

– **Output**: $w \in \mathbb{R}^n$ and $s \in B(F)$.

Relationship with divide-and-conquer algorithm. When starting from the trivial ordered partition $\mathcal{A} = (V)$, then we exactly obtain a parallel version of the divide-and-conquer algorithm [17], that is, the isotonic regression problem in (a) is always solved without using the constraints of monotonicity, i.e., there are no merges in (b), and thus in (c), it is not necessary to re-solve the problems where nothing has changed. This shows that the number of iterations is then less than n. The key added benefits in our formulation is the possibility of warm-starting, which can be very useful for building paths of solutions with different weights on the total variation. This is also useful for decomposable functions where many TV oracles are needed with close-by inputs. See experiments in Section 5.

3.5 Proof of convergence

In order to prove convergence of the algorithm, we only need to show that if the optimality check fails in step (c), then step (d) introduces splits in the partition, which ensures that the isotonic regression in step (a) of the next iteration has a strictly lower value. Let us recall the isotonic regression problem solved in step (a):

$$\min_{v \in \mathbb{R}^m} \sum_{i=1}^{m} \Big(v_i \big[F(B_i) - \quad F(B_{i-1}) - u(A_i) \big] + \tfrac{1}{2} |A_i| v_i^2 \Big) \tag{9}$$

$$\text{such that } v_1 \geqslant \cdots \geqslant v_m. \tag{10}$$

Steps (a-b) ensures that the ordered partition $\mathcal{A}$ is a basic ordered partition warranting that the inequality constraints are strict, i.e., no two partitions have the same value v_i and the values v_i for each element of the partition $i = \{1, \ldots, m\}$ is given through

$$v_i |A_i| = u(A_i) - (F(B_i) - F(B_{i-1})), \tag{11}$$

which can be calculated in closed form.

The optimality check in step (c) decouples into checking the optimality in each subproblem as shown in Section 3.3. If the optimality test fails, then there is a subset of C_i of A_i for some of elements of the partition $\mathcal{A}$ such that $F(B_{i-1} \cup C_i) - F(B_{i-1}) - s(C_i) < 0$. We will show that the splits introduced by step (d) strictly reduces the function value of isotonic regression in Eq. (9), while maintaining the feasibility of the problem. The splits modify the cost function of the isotonic regression as follows, as the objective function in Eq. (9) is equal to

$$\sum_{i=1}^{m} \Big(v_i \big[F(B_{i-1} \cup C_i) - F(B_{i-1}) - u(C_i) \big] + v_i \big[F(B_i) - F(B_{i-1} \cup C_i) - u(A_i \setminus C_i) \big] \\ + \tfrac{1}{2} v_i^2 |C_i| + \tfrac{1}{2} v_i^2 |A_i \setminus C_i| \Big). \tag{12}$$

Let us assume a positive $t \in \mathbb{R}$, which is small enough. The direction that the isotonic regression moves is $v_i + t$ for the partition corresponding to C_i and $v_i - t$ for the partition corresponding to $A_i \setminus C_i$ maintaining the feasibility of the isotonic regression problem, i.e., $v_1 \geqslant \ldots \geqslant v_i + t > v_i - t \geqslant \ldots \geqslant v_m$. The function value is given by

$$
\begin{aligned}
&\sum_{i=1}^{m} \Big((v_i + t)\big[F(B_{i-1} \cup C_i) - F(B_{i-1}) - u(C_i)\big] + (v_i - t)\big[F(B_i) - F(B_{i-1} \cup C_i) - u(A_i \setminus C_i)\big] \\
&\qquad + \tfrac{1}{2}(v_i + t)^2 |C_i| + \tfrac{1}{2}(v_i - t)^2 |A_i \setminus C_i| \Big) \\
= \;& \sum_{i=1}^{m} \Big(\big(v_i\big[F(B_{i-1} \cup C_i) - F(B_{i-1}) - u(C_i)\big] + v_i\big[F(B_i) - F(B_{i-1} \cup C_i) - u(A_i \setminus C_i)\big] \\
&\qquad + \tfrac{1}{2} v_i^2 |C_i| + \tfrac{1}{2} v_i^2 |A_i \setminus C_i|\big) \\
&\qquad + t\,\big(2F(B_{i-1} \cup C_i) - F(B_{i-1}) - F(B_i) - u(C_i) + u(A_i \setminus C_i) + v_i |C_i| - v_i |A_i \setminus C_i|\big) \\
&\qquad + \tfrac{1}{2} t^2 |A_i| \Big).
\end{aligned}
$$

From this we can compute the directional derivative of the function at $t = 0$, which is given by

$$
\begin{aligned}
& 2F(B_{i-1} \cup C_i) - F(B_{i-1}) - F(B_i) - u(C_i) + u(A_i \setminus C_i) + |C_i| v_i - |A_i \setminus C_i| v_i \\
= \;& 2F(B_{i-1} \cup C_i) - F(B_{i-1}) - F(B_i) - 2u(C_i) + u(A_i) + 2|C_i| v_i - |A_i| v_i \\
= \;& 2\big(F(B_{i-1} \cup C_i) - F(B_{i-1}) - u(C_i) + v_i |C_i|\big) \text{ (substituting Eq. (11))} \\
= \;& 2\big(F(B_{i-1} \cup C_i) - F(B_{i-1}) - s(C_i)\big) < 0 \text{ (as } s = u - w \text{ and Eq. (3.5)).}
\end{aligned}
$$

This shows that the function strictly decreases with the splits introduced in step (d).

3.6 Discussion

Certificates of optimality. The algorithm has dual-infeasible iterates s (they only belong to $B(F)$ at convergence). However, after step (c), we have that for all $C \subset V$, $F(C) - s(C) \geqslant -\varepsilon$. This implies that $s \in B(F + \varepsilon 1_{\mathrm{Card} \in (1,n)})$, i.e., $s \in B(F_\varepsilon)$ with $F_\varepsilon = F + \varepsilon 1_{\mathrm{Card} \in (1,n)}$. Since by construction $w = u - s$, we have:

$$
\begin{aligned}
f_\varepsilon(w) - u^\top w + \tfrac{1}{2}\|w\|_2^2 + \tfrac{1}{2}\|s - u\|_2^2 &= \varepsilon \big| \max_{j \in V} w_j - \min_{j \in V} w_j \big| + f(w) - u^\top w + \|w\|^2 \\
&= \varepsilon \big| \max_{j \in V} w_j - \min_{j \in V} w_j \big| \\
&\qquad + \sum_{i=1}^{m} v_i \big[F(B_i) - F(B_{i-1}) - u(A_i)\big] + \sum_{i=1}^{m} |A_i| v_i^2 \\
&= \varepsilon \big| \max_{j \in V} w_j - \min_{j \in V} w_j \big| \text{ (using Eq. (11))} \\
&= \varepsilon \operatorname{range}(w),
\end{aligned}
$$

where $\operatorname{range}(w) = \max_{k \in V} w_k - \min_{k \in V} w_k$. This means that w is approximately optimal for $f(w) - u^\top w + \frac{1}{2}\|w\|_2^2$ with *certified gap* less than $\varepsilon \operatorname{range}(w) + \varepsilon \operatorname{range}(w^*)$.

Maximal range of an active-set solution. For any ordered partition $\mathcal{A}$, and the optimal value of w (which we know in closed form), we have $\operatorname{range}(w) \leqslant \operatorname{range}(u) + \max_{i \in V} \big\{F(\{i\}) + F(V \setminus \{i\}) - F(V)\big\}$. Indeed, for the u part of the expression, this is because values of w are averages of values of u; for the F part of the expression, we always have by submodularity:

$$
F(B_i) - F(B_{i-1}) \;\leqslant\; \sum_{k \in A_i} F(\{k\}) \text{ and}
$$

$$F(B_i) - F(B_{i-1}) \quad \geqslant \quad - \sum_{k \in A_i} F(V) - F(V \backslash \{k\}).$$

This means that the certificate can be used in practice by replacing $\mathrm{range}(w^*)$ by its upperbound.

Exact solution. If the submodular function only takes integer values and we have an approximate solution of the TV problem with gap $\varepsilon \leqslant \frac{1}{4n}$, then we have the optimal solution [10].

Relationship with traditional active-set algorithm. Given an ordered partition $\mathcal{A}$, an active-set method solves the unconstrained optimization problem in Eq. (8) to obtain a value of v using the primary stationary conditions. The corresponding primal value $w = \sum_{i=1}^m v_i 1_{A_i}$ and dual value $s = u - w$ are optimal, if and only if,

$$\text{Primal feasibility : } \; w \in \mathcal{W}^{\mathcal{A}}, \tag{13}$$

$$\text{Dual feasibility : } \; s \in B(F). \tag{14}$$

If Eq. (13) is not satisfied, a move towards the optimal w is performed to ensure primal feasibility by performing line search, i.e., two consecutive sets A_i and A_{i+1} with increasing values of v, i.e., $v_i < v_{i+1}$ are merged and a potential w is computed until primal feasibility is met. Then dual feasibility is checked and potential splits are proposed.

In our approach, we consider a different strategy which is more direct and does many merges simultaneously by using *isotonic regression*. Our method explicitly moves from ordered partitions to ordered partitions and computes an optimal vector w, which is always feasible.

4 Decomposable Problems

Many interesting problems in signal processing and computer vision naturally involve submodular functions F that decompose into $F = F_1 + \cdots + F_r$, with r "simple" submodular functions [2]. For example, a cut function in a 2D grid decomposes into a function F_1 composed of cuts along vertical lines and a function F_2 composed of cuts along horizontal lines. For both of these functions, SFM oracles may be solved in $O(n)$ by message passing. For simplicity, in this paper, we consider the case $r = 2$ functions, but following [20, 18], our framework easily extends to $r > 2$.

4.1 Reformulation as the distance between two polytopes

Following [18], we have the primal/dual problems :

$$\begin{aligned}
\min_{w \in \mathbb{R}^n} f_1(w) + f_2(w) - u^\top w + \tfrac{1}{2}\|w\|_2^2 &= \min_{w \in \mathbb{R}^n} \max_{s_1 \in B(F_1),\, s_2 \in B(F_2)} w^\top (s_1 + s_2) - u^\top w + \tfrac{1}{2}\|w\|_2^2 \\
&= \max_{s_1 \in B(F_1),\, s_2 \in B(F_2)} \min_{w \in \mathbb{R}^n} (s_1 + s_2 - u)^\top w + \tfrac{1}{2}\|w\|_2^2 \\
&= \max_{s_1 \in B(F_1),\, s_2 \in B(F_2)} -\tfrac{1}{2}\|s_1 + s_2 - u\|_2^2, \qquad (15)
\end{aligned}$$

with $w = u - s_1 - s_2$ at optimality.

This is the projection of u on the sum of the base polytopes $B(F_1) + B(F_2) = B(F)$. Further, this may be interpreted as finding the distance between two polytopes $B(F_1) - u/2$ and $u/2 - B(F_2)$. Note that these

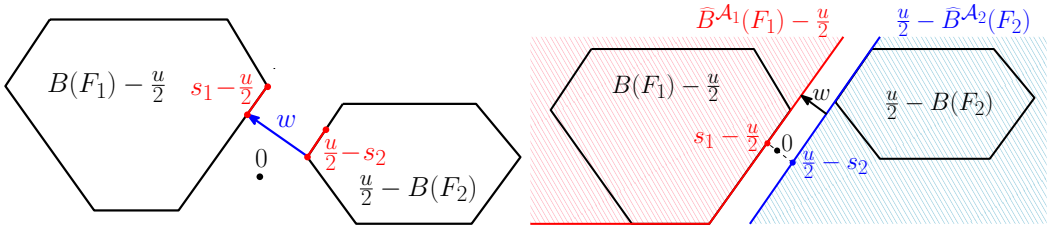


Figure 3: Closest point between two polytopes. Top: output of Dykstra's alternating projection algorithm for the TV problem, the pair (s_1, s_2) may not be unique while $w = s_1 + s_2 - u$ is. Bottom: Dykstra's alternating projection output for outer approximations.

two polytopes typically do not intersect (they will if and only if $w = 0$ is the optimal solution of the TV problem, which is an uninteresting situation).

Alternating projections (AP). The alternating projection algorithm [4] was proposed to solve the convex feasibility problem, i.e., to obtain a feasible point in the intersection of two polytopes. It is equivalent to performing block coordinate descent on the dual derived in Eq. (15). Let us denote the projection onto a polytope K as Π_K, i.e., $\Pi_K(y) = \operatorname{argmax}_{x \in K} -\|x - y\|_2^2$. Therefore, alternating projections lead to the following updates for our problem.

$$z_t = \Pi_{u/2 - B(F_2)} \Pi_{B(F_1) - u/2}(z_{t-1}),$$

where z_0 is an arbitrary starting point. Thus each of these steps require TV oracle for F_1 and F_2 since projection onto the base polytope is equivalent to performing TV denoising as shown in Eq. (5).

Averaged alternating reflections (AAR). The averaged alternating reflection algorithm [5], which is also known as Douglas-Rachford splitting can be used to solve convex feasibility problems. It is observed to converge quicker than alternating projection [18, 22] in practice. We now introduce a reflection operator for the polytope K as R_K, i.e., $R_K = 2\Pi_K - I$, where I is the identity operator. Therefore, reflection of t on a polytope K is given by $R_K(t) = 2\Pi_K(t) - t$. The updates of each iteration of the averaged alternating reflections, which starts with an auxiliary sequence z_0 initialized to 0 vector, are given by

$$z_t = \tfrac{1}{2}(I + R_{u/2 - B(F_2)} R_{B(F_1) - u/2})(z_{t-1}).$$

In the feasible case, i.e., intersecting polytopes, the sequence z_t weakly converges to a point in the intersection of the polytopes. However, in our case, we have non intersecting polytopes which leads to a converging sequence of z_t with AP but a diverging sequence of z_t with AAR. However, when we project z_t by using the following projection operation, i.e, $s_{1,t} = \Pi_{B(F_1) - u/2}(z_t)$; $s_{2,t} = \Pi_{u/2 - B(F_2)}(s_{1,t})$ the sequences $s_{1,t}$ and $s_{2,t}$ converge to nearest points on the polytopes, $B(F_1) - u/2$ and $u/2 - B(F_2)$ [5].

Dykstra's alternating projections (DAP). Dykstra's alternating projection algorithm [3] retrieves a convex feasible point closest to an arbitrary point, which we assume to be 0. It can also be used and has a form of primal descent interpretation, i.e., as coordinate descent for a well-formulated primal problem [15]. Let us denote ι_K as the indicator function of a convex set K. In our case we consider finding the nearest points on the polytopes $B(F_1) - u/2$ and $u/2 - B(F_2)$ closest to 0, which can be formally written as:

$$\begin{aligned}
\min_{\substack{s \in B(F_1) - \frac{u}{2} \\ s \in \frac{u}{2} - B(F_2)}} \frac{1}{2}\|s\|_2^2 &= \min_{s \in \mathbb{R}^n} \frac{1}{2}\|s\|_2^2 + \iota_{B(F_1) - \frac{u}{2}}(s) + \iota_{\frac{u}{2} - B(F_2)}(s) \\
&= \min_{s \in \mathbb{R}^n} \frac{1}{2}\|s\|_2^2 + \iota_{B(F_1) - \frac{u}{2}}(s) + \iota_{B(F_2) - \frac{u}{2}}(-s)
\end{aligned}$$

$$
\begin{aligned}
&= \min_{s\in\mathbb{R}^n} \frac{1}{2}\|s\|_2^2 + \max_{w_1\in\mathbb{R}^n} w_1^\top s - f_1(w_1) + \frac{w_1^\top u}{2} + \max_{w_2\in\mathbb{R}^n} -w_2^\top s - f_2(w_2) + \frac{w_2^\top u}{2} \\
&= \max_{\substack{w_1\in\mathbb{R}^n \\ w_2\in\mathbb{R}^n}} -f_1(w_1) - f_2(w_2) + \frac{(w_1+w_2)^\top u}{2} + \min_{s\in\mathbb{R}^n} \frac{1}{2}\|s\|_2^2 + (w_1-w_2)^\top s \\
&= \max_{\substack{w_1\in\mathbb{R}^n \\ w_2\in\mathbb{R}^n}} -f_1(w_1) - f_2(w_2) + \frac{(w_1+w_2)^\top u}{2} - \frac{1}{2}\|w_1-w_2\|_2^2 \\
&= \min_{\substack{w_1\in\mathbb{R}^n \\ w_2\in\mathbb{R}^n}} f_1(w_1) + f_2(w_2) - \frac{(w_1+w_2)^\top u}{2} + \frac{1}{2}\|w_1-w_2\|_2^2,
\end{aligned}
$$

where $s = w_2 - w_1$ at optimality. The block coordinate descent gives

$$
\begin{aligned}
w_{1,t} &= \operatorname{prox}_{f_1 - u/2}(w_{2,t-1}) = (I - \Pi_{B(F_1)-u/2})(w_{2,t-1}), \\
s_{1,t} &= w_{2,t-1} - w_{1,t}, \\
w_{2,t} &= \operatorname{prox}_{f_2 - u/2}(w_{1,t}) = (I - \Pi_{B(F_2)-u/2})(w_{1,t}), \\
s_{2,t} &= w_{1,t} - w_{2,t},
\end{aligned}
$$

where I is the identity matrix. This is exactly the same as Dykstra's alternating projection steps.

We have implemented it, and it behaves similar to alternating projections, but it still requires TV oracles for projection (see experiments in Section 5). There is however a key difference: while alternating projections and alternating reflections always converge to a pair of closest points, Dykstra's alternating projection algorithm converges to a *specific* pair of points, namely the pair closest to the initialization of the algorithm [3]; see an illustration in Figure 3 (top). This insight will be key in our algorithm to avoid cycling.

Assuming TV oracles are available for F_1 and F_2, [18, 22] use alternating projection [4] and alternating reflection [5] algorithms. However, these algorithms are equivalent to block *dual* coordinate descent and cannot be be cast explicitly as descent algorithms for the primal TV problem. On the other hand, Dykstra's alternating projection is a descent algorithm on the primal, which enables local search over partitions. Complex TV oracles are often implemented by using SFM oracles recursively with the divide-and-conquer strategy on the individual functions. Using our algorithm in Section 3.4, they can be made more efficient using warmstarts (see experiments in Section 5).

4.2 Local search over partitions using active-set method

Given our algorithm for a single function, it is natural to perform a local search over two partitions $\mathcal{A}_1$ and $\mathcal{A}_2$, one for each function F_1 and F_2, and consider in the primal formulation a weight vector w compatible with both $\mathcal{A}_1$ and $\mathcal{A}_2$; or, equivalently, in the dual formulation, two outer approximations $\widehat{B}^{\mathcal{A}_1}(F_1)$ and $\widehat{B}^{\mathcal{A}_2}(F_2)$. That is, given the ordered partitions $\mathcal{A}_1$ and $\mathcal{A}_2$, using a similar derivation as in Eq. (15), we obtain the primal/dual pairs of optimization problems

$$
\max_{\substack{s_1\in\widehat{B}^{\mathcal{A}_1}(F_1) \\ s_2\in\widehat{B}^{\mathcal{A}_2}(F_2)}} -\tfrac{1}{2}\|u - s_1 - s_2\|_2^2 = \min_{\substack{w\in\mathcal{W}^{\mathcal{A}_1} \\ w\in\mathcal{W}^{\mathcal{A}_2}}} f_1(w) + f_2(w) - u^\top w + \tfrac{1}{2}\|w\|_2^2,
$$

with $w = u - s_1 - s_2$ at optimality.

Primal solution by isotonic regression. The primal solution w is unique by strong convexity. Moreover, it has to be compatible with both $\mathcal{A}_1$ and $\mathcal{A}_2$, which is equivalent to being compatible with the *coalesced*

ordered partition $\mathcal{A} = \text{coalesce}(\mathcal{A}_1, \mathcal{A}_2)$ defined as the coarsest ordered partition compatible by both. As shown in Appendix A, $\mathcal{A}$ may be found in time $O(\min(m_1, m_2)n)$.

Given $\mathcal{A}$, the primal solution w of the subproblem may be found by isotonic regression like in Section 3.2 in time $O(m)$ where m is the number of sets in $\mathcal{A}$. However, finding the optimal dual variables s_1 and s_2 turns out to be more problematic. We know that $s_1 + s_2 = u - w$ and that $s_1 + s_2 \in \widehat{B}^{\mathcal{A}}(F)$, but the split of $s_1 + s_2$ into (s_1, s_2) is unknown.

Obtaining dual solutions. Given ordered partitions $\mathcal{A}_1$ and $\mathcal{A}_2$, a unique well-defined pair (s_1, s_2) can be obtained by using convex feasibility algorithms such as alternating projections [4] or alternating reflections [5]. However, the result would depend in non understood ways on the initialization, and we have observed cycling of the active-set algorithm. Using Dykstra's alternating projection algorithm allows us to converge to a unique well-defined pair (s_1, s_2) that will lead to a provably non-cycling algorithm.

When running the Dykstra's alternating projection algorithm starting from 0 on the polytopes $\widehat{B}^{\mathcal{A}_1}(F_1) - u/2$ and $u/2 - \widehat{B}^{\mathcal{A}_2}(F_2)$, if w is the unique distance vector between the two polytopes, then the iterates converge to the projection of 0 onto the convex sets of elements in the two polytopes that achieve the minimum distance [3]. See Figure 3 (bottom) for an illustration. This algorithm is however slow to converge when the polytopes do not intersect. Note that $w \neq 0$ in most of our situations and convergence is hard to monitor because primal iterates of the Dykstra's alternating projection diverge [3].

Translated intersecting polytopes. In our situation, we want to reach the solution *while knowing the vector* w (as mentioned earlier, it is obtained cheaply from isotonic regression). Indeed, from Lemma 2.2 and Theorem 3.8 from [3], given this vector w, we may translate the two polytopes and now obtain a formulation where the two polytopes do intersect; that is we aim at projecting 0 on the (non-empty) intersection of $\widehat{B}^{\mathcal{A}_1}(F_1) - u/2 + w/2$ and $u/2 - w/2 - \widehat{B}^{\mathcal{A}_2}(F_2)$. See Figure 4. We also refer to this as the *translated Dykstra problem*[2] in the rest of the paper. This is equivalent to solving the following optimization problem

$$
\begin{aligned}
\min_{\substack{s \in \widehat{B}^{\mathcal{A}_1}(F_1) - \frac{u-w}{2} \\ s \in \frac{u-w}{2} - \widehat{B}^{\mathcal{A}_2}(F_2)}} \tfrac{1}{2}\|s\|_2^2 &= \min_{s \in \mathbb{R}^n} \tfrac{1}{2}\|s\|_2^2 + \iota_{\widehat{B}^{\mathcal{A}_1}(F_1) - \frac{u-w}{2}}(s) + \iota_{\frac{u-w}{2} - \widehat{B}^{\mathcal{A}_2}(F_2)}(s), \\
&= \min_{s \in \mathbb{R}^n} \tfrac{1}{2}\|s\|_2^2 + \iota_{\widehat{B}^{\mathcal{A}_1}(F_1) - \frac{u-w}{2}}(s) + \iota_{\widehat{B}^{\mathcal{A}_2}(F_2) - \frac{u-w}{2}}(-s), \\
&= \min_{s \in \mathbb{R}^n} \Bigg(\tfrac{1}{2}\|s\|_2^2 + \max_{w_1 \in \mathcal{W}^{\mathcal{A}_1}} w_1^\top s - f_1(w_1) + \tfrac{w_1^\top (u-w)}{2} \\
&\qquad + \max_{w_2 \in \mathcal{W}^{\mathcal{A}_2}} -w_2^\top s - f_2(w_2) + \frac{w_2^\top (u-w)}{2} \Bigg), \\
&= \max_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \Bigg(- f_1(w_1) - f_2(w_2) + \frac{(w_1 + w_2)^\top (u-w)}{2} \\
&\qquad + \min_{s \in \mathbb{R}^n} \frac{1}{2}\|s\|_2^2 + (w_1 - w_2)^\top s \Bigg), \\
&= \max_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \Bigg(- f_1(w_1) - f_2(w_2) + \frac{(w_1 + w_2)^\top (u-w)}{2} \\
&\qquad - \frac{1}{2}\|w_1 - w_2\|_2^2 \Bigg),
\end{aligned}
$$

[2]We refer to finding a Dykstra solution for translated intersecting polytopes as translated Dykstra problem

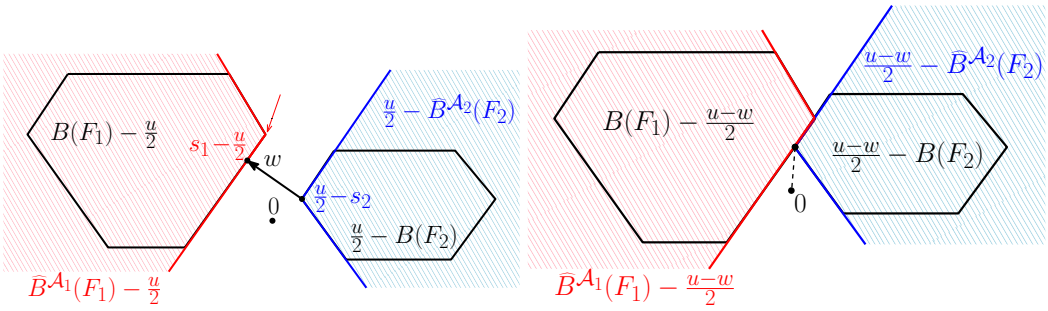


Figure 4: Translated intersecting polytopes. Top: output of our algorithm before translation. Bottom: Translated formulation.

$$= \min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \left(f_1(w_1) + f_2(w_2) - \frac{(w_1 + w_2)^\top (u - w)}{2} + \frac{1}{2}\|w_1 - w_2\|_2^2 \right), \tag{16}$$

with $s = w_2 - w_1$ at optimality.

In Section 4.3 we propose algorithms to solve the above optimization problems. Assuming that we are able to solve this step efficiently, we now present our active-set algorithm for decomposable functions below.

4.3 Active-set algorithm for decomposable functions

- **Input**: Submodular function F_1 and F_2 with SFM oracles, $u \in \mathbb{R}^n$, ordered partitions $\mathcal{A}_1, \mathcal{A}_2$
- **Algorithm**: iterate until convergence (i.e., $\varepsilon_1 + \varepsilon_2$ small enough)
  (a) Find $\mathcal{A} = \text{coalesce}(\mathcal{A}_1, \mathcal{A}_2)$ and run isotonic regression to minimize $f(w) - u^\top w + \frac{1}{2}\|w\|_2^2$ such that w is compatible with $\mathcal{A}$.
  (b) Find the projection of 0 onto the intersection of $\widehat{B}^{\mathcal{A}_1}(F_1) - u/2 + w/2$ and $u/2 - w/2 - \widehat{B}^{\mathcal{A}_2}(F_2)$ using any of the algorithms described in Section 4.4.
  (c) Merge the sets in $\mathcal{A}_j$ which are tight for s_j, $j \in \{1, 2\}$.
  (d) Check optimality by solving $\min_{C_{j,i_j} \subseteq A_{j,i_j}} F_j(B_{j,i_j-1} \cup C_{j,i_j}) - F_j(B_{j,i+j-1}) - s_j(C_{j,i_j})$ for $i_j \in \{1, \dots, m_j\}$, Monitor ε_1 and ε_2 such that $F_j(C_j) - s_j(C_j) \geqslant -\varepsilon_j$, $j = 1, 2$.
  (e) If both s_1 and s_2 not optimal, for all C_{j,i_j} which are different from $\varnothing$ and A_{j,i_j}, split partitions.
- **Output**: $w \in \mathbb{R}^n$ and $s_1 \in B(F_1)$, $s_2 \in B(F_2)$.

Given two ordered partitions $\mathcal{A}_1$ and $\mathcal{A}_2$, we obtain $s_1 \in \hat{B}^{\mathcal{A}_1}(F_1)$ and $s_2 \in \hat{B}^{\mathcal{A}_2}(F_2)$ as described in the following section. The solution $w = u - s_1 - s_2$ is optimal if and only if both $s_1 \in B(F_1)$ and $s_2 \in B(F_2)$. When checking the optimality described in Section 3.3, we split the partition. As shown in Appendix B, either (a) $\|w\|_2^2$ strictly increases at each iteration, or (b) $\|w\|_2^2$ remains constant but $\|s_1 - s_2\|_2^2$ strictly increases. This implies that the algorithm is finitely convergent.

4.4 Optimizing the "translated Dykstra problem"

In this section, we describe algorithms to solve step (b) of the active-set algorithm proposed in Section 4.3 that optimizes the translated Dykstra problem in Eq. (16), i.e.,

$$\min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} f_1(w_1) + f_2(w_2) - \tfrac{(w_1+w_2)^\top (u-w)}{2} + \tfrac{1}{2}\|w_1 - w_2\|^2. \tag{17}$$

The corresponding dual optimization problem is given by

$$\min_{\substack{s \in \widehat{B}^{\mathcal{A}_1}(F_1) - \frac{u-w}{2} \\ s \in \frac{u-w}{2} - \widehat{B}^{\mathcal{A}_2}(F_2)}} \tfrac{1}{2}\|s\|_2^2 \tag{18}$$

with the optimality condition $s = w_2 - w_1$. Note that the only link to submodularity is that f_1 and f_2 are linear functions on $\mathcal{W}^{\mathcal{A}_1}$ and $\mathcal{W}^{\mathcal{A}_2}$, respectively. The rest of this section primarily deals with optimizing a quadratic program and we present two algorithms in Section 4.4.1 and Section 4.4.2.

4.4.1 Accelerated Dykstra's algorithm

In this section, we find the projection of the origin onto the intersection of the translated base polytopes obtained by solving the optimization problem in Eq. (16) given by

$$\min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} f_1(w_1) + f_2(w_2) - \tfrac{(w_1+w_2)^\top (u-w)}{2} + \tfrac{1}{2}\|w_1 - w_2\|^2,$$

using Dykstra's alternating projection. It can be solved using the following Dykstra's iterations:

$$\begin{aligned}
s_{1,t} &= \Pi_{\widehat{B}^{\mathcal{A}_1}(F_1)}(u/2 - w/2 + w_{2,t-1}), \\
w_{1,t} &= u/2 - w/2 + w_{2,t-1} - s_{1,t}, \\
s_{2,t} &= \Pi_{\widehat{B}^{\mathcal{A}_2}(F_2)}(u/2 - w/2 + w_{1,t}), \\
w_{2,t} &= u/2 - w/2 + w_{1,t} - s_{2,t},
\end{aligned}$$

with Π_C denoting the orthogonal projection onto the sets C, solved here by isotonic regression. Note that the value of the auxiliary variable w_2 can be warm-started. The algorithm converges linearly for polyhedral sets [29].

In our simulations, we have used the recent accelerated version of [12], which led to faster convergence. In order to monitor convergence, we compute the value of $\|u - w - s_{1,t} - s_{2,t}\|_1$ which is equal to zero at convergence. The optimization problem can also be decoupled into smaller optimization problems by using the knowledge of the face of the base polytopes on which s_1 and s_2 lie. See details in the Appendix D. This is still slow to converge in practice and therefore we present an active-set method in the next section.

4.4.2 Primal active-set method

In this section, we find the projection of the origin onto the intersection of the translated base polytopes given by Eq. (16) using the standard active-set method [27] by solving a set of linear equations. For this purpose, we derive the equivalent optimization problems using equality constraints.

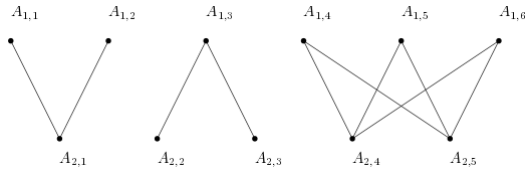


Figure 5: Bipartite graph to estimate $Q(\mathcal{A}_1, \mathcal{A}_2)$ with $\mathcal{A}_1$ having $m_1 = 6$ components and $\mathcal{A}_2$ having $m_2 = 5$.

The ordered partition, $\mathcal{A}_j$ is given by $(A_{j,1}, \ldots, A_{j,m_j})$, where m_j is the number of elements in the ordered partitions. Let B_{j,i_j} be defined as $(A_{j,1} \cup \ldots \cup A_{j,i_j})$. Therefore,

$$
\begin{aligned}
f_j(w_j) &= \sum_{i_j=1}^{m_j} v_{j,i_j} \Big(F_j(B_{j,i_j}) - F_j(B_{j,i_j-1}) \Big) && (19)\\
w_j &= \sum_{i_j=1}^{m_j} v_{j,i_j} 1_{A_{j,i_j}} && (20)\\
&\text{with the constraints, } v_{j,1} \geq \ldots \geq v_{j,m_j}. && (21)
\end{aligned}
$$

On substituting Eq. (19), Eq. (20) and Eq. (21) in Eq. (16), we have and equivalent optimization problem:

$$
\begin{aligned}
\min_{\substack{v_{1,1} \geq \ldots \geq v_{1,m_1} \\ v_{2,1} \geq \ldots \geq v_{2,m_2}}} & \sum_{i_1=1}^{m_1} \Big(F_1(B_{1,i_1}) - F_1(B_{1,i_1-1}) - \frac{u(A_{1,i_1}) - w(A_{1,i_1})}{2} \Big) v_{1,i_1} \\
+ & \sum_{i_2=1}^{m_2} \Big(F_2(B_{2,i_2}) - F_2(B_{2,i_2-1}) - \frac{u(A_{2,i_2}) - w(A_{2,i_2})}{2} \Big) v_{2,i_2} \\
+ & \sum_{i_1=1}^{m_1} \frac{1}{2} |A_{1,i_1}| v_{1,i_1}^2 + \sum_{i_2=1}^{m_2} \frac{1}{2} |A_{2,i_2}| v_{2,i_2}^2 \\
- & \sum_{i_1=1}^{m_1} \sum_{i_2=1}^{m_2} v_{1,i_1} v_{2,i_2} 1_{A_{1,i_1}}^\top 1_{A_{2,i_2}}.
\end{aligned}
$$

This can be written as a quadratic program in $x = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$ with inequality constraints in the following form

$$
\min_{\substack{x \in \mathbb{R}^{m_1+m_2} \\ D(\mathcal{A}_1, \mathcal{A}_2) x \succcurlyeq 0}} \frac{1}{2} x^\top Q(\mathcal{A}_1, \mathcal{A}_2) x + c(\mathcal{A}_1, \mathcal{A}_2)^\top x, \tag{22}
$$

where $D(\mathcal{A}_1, \mathcal{A}_2)$ is a sparse matrix of size $(m_1 + m_2 - 2) \times (m_1 + m_2)$, which is a block diagonal matrix containing the difference or first order derivative matrices of sizes $m_1 - 1 \times m_1$ and $m_2 - 1 \times m_2$ as the blocks and $c(\mathcal{A}_1, \mathcal{A}_2)$ is a linear vector that can be estimated using the functions evaluations of F_1 and F_2. Note that these evaluations need to be done only once.

Estimating $Q(\mathcal{A}_1, \mathcal{A}_2)$. Let us consider a bipartite graph, $G = (\mathcal{A}_1, \mathcal{A}_2, E)$, with $m_1 + m_2$ nodes representing the ordered partitions of $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively. The weight of the edge between each element of

ordered partitions of $\mathcal{A}_1$, represented by A_{1,i_1} and each element of ordered partitions of $\mathcal{A}_2$, represented by A_{2,i_2} is the number of elements of the ground set V that lie in both these partitions and can be written as $e(A_{1,i_1}, A_{2,i_2}) = 1^\top_{A_{1,i_1}} 1_{A_{2,i_2}}$ for all $e \in E$. The matrix $Q(\mathcal{A}_1, \mathcal{A}_2)$ represents the Laplacian matrix of the graph G. Figure 5 shows a sample bipartite graph with $m_1 = 6$ and $m_2 = 5$.

Optimizing the quadratic program in Eq. (22) by using active-set methods is equivalent to finding the face of the constraint set on which the optimal solution lies. For this purpose, we need to be able to solve the quadratic program in Eq. (22) with equality constraints.

Equality constraint QP. Let us now consider the following quadratic program with equality constraints

$$\min_{\substack{p \in \mathbb{R}^{m_1+m_2} \\ D'p=0}} \frac{1}{2} p^\top Q(\mathcal{A}_1, \mathcal{A}_2) p + \big(Q(\mathcal{A}_1, \mathcal{A}_2) x_k + c(\mathcal{A}_1, \mathcal{A}_2)\big)^\top p, \tag{23}$$

where D' is the subset of the constraints in $D(\mathcal{A}_1, \mathcal{A}_2)$, i.e. indices of its rows that are tight and x_k is a primal-feasible point. The vector p gives the direction of strict descent of the cost function in Eq. (22) from feasible point x_k [27].

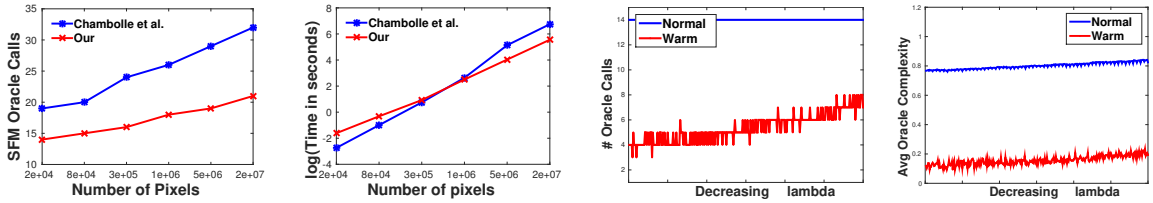
Without loss of generality, let us assume that the equality constraints are $v_{j,k_j} = v_{j,k_j+1}$ for any k_j in $[0, m_j)$. Let $\mathcal{A}'_j$ be the new ordered partition formed by merging A_{j,k_j} and A_{j,k_j+1} as $v_{j,k_j} = v_{j,k_j+1}$. Finding the optimal vector p using the quadratic program in Eq. (23) with respect to the ordered partition $\mathcal{A}'_j$ is equivalent to solving the following unconstrained quadratic problem,

$$\mathcal{Q}(\mathcal{A}'_1, \mathcal{A}'_2, x_t) = \min_{p' \in \mathbb{R}^{m'_1+m'_2}} \left(\frac{1}{2} p'^\top Q(\mathcal{A}'_1, \mathcal{A}'_2) p' + \big(Q(\mathcal{A}'_1, \mathcal{A}'_2) x_t + c(\mathcal{A}'_1, \mathcal{A}'_2)\big)^\top p' \right), \tag{24}$$

where m'_j is the number of elements of the ordered partition $\mathcal{A}'_j$. This can be estimated by solving a linear system using conjugate gradient descent. The complexity of each iteration of the conjugate gradient is given by $O((m'_1 + m'_2)k)$ where k is the number of non-zero elements in the sparse matrix, $Q(\mathcal{A}'_1, \mathcal{A}'_2)$ [32]. We can build p from p' by repeating the values for the elements of the partition that were merged.

Primal active-set algorithm. We now describe the standard active-set method to solve the quadratic program in Eq. (16).

- **Input**: Laplacian matrix, $Q(\mathcal{A}_1, \mathcal{A}_2)$ and vector, $c(\mathcal{A}_1, \mathcal{A}_2)$, ordered partitions $\mathcal{A}_1, \mathcal{A}_2$, w.
- **Algorithm**: Iterate on t until both primal and dual feasibility conditions in Eq. (17) and Eq. (18) respectively are satisfied.
- **Initialize**: x_0 using w, Working set WS_0 with tight sets of $\mathcal{A}_j$. Estimate $\mathcal{A}'_1$, $\mathcal{A}'_2$ from WS_0.
  (a) Solve $\mathcal{Q}(\mathcal{A}'_1, \mathcal{A}'_2, x_t)$ in Eq. (24) to find optimal p.
  (b) **Check Primal Feasibility**: If $p == 0$
    1. **Estimate Dual**:
       $\lambda = D(\mathcal{A}_1, \mathcal{A}_2)^{-\top} \big(Q(\mathcal{A}_1, \mathcal{A}_2) x_t + c(\mathcal{A}_1, \mathcal{A}_2)\big)$.
    2. **Check Dual Feasibility**:
       if $\lambda_i \geq 0$ for all in $i \in WS_t$
       * return Optimal $x^* = x_t$.
    3. else



(a) (b) (c) (d)

Figure 6: (a) Number of SFM oracle calls for images of various sizes, (b) Time taken for images of various sizes, (c) Number of iterations with and without warm start, (d) Average complexity of the oracle with and without warm start.

- ∗ **Most violated Constraint**:
  update $j = \text{argmin}_{j \in WS_t} \lambda_j$.
- ∗ **Update Working Set**:
  update $WS_{t+1} = WS_t \setminus \{j\}$.
  update $\mathcal{A}'_1, \mathcal{A}'_2$ from WS_{t+1}.
- ∗ update $x_{t+1} = x_t$
- ∗ Goto step (a).

4. endif

(c) else

1. **Line Search**: Find least α that retains feasibility of $x_{t+1} = x_t + \alpha p$ and find the blocking constraints B_t.
2. **Update Working Set**: $WS_{t+1} = WS_t \cup B_t$ and update $\mathcal{A}'_1, \mathcal{A}'_2$ from WS_{t+1}.
3. Goto step (a).

(d) endif

– **Output**: $x^* \in \mathbb{R}^{m_1+m_2}$.

We can estimate w_1 and w_2 from x^*, which will enable us to estimate s feasible in Eq. (18). Therefore we can estimate the dual variable $s_1 \in B^{\mathcal{A}_1}(F_1)$ and $s_2 \in B^{\mathcal{A}_2}(F_2)$ using s.

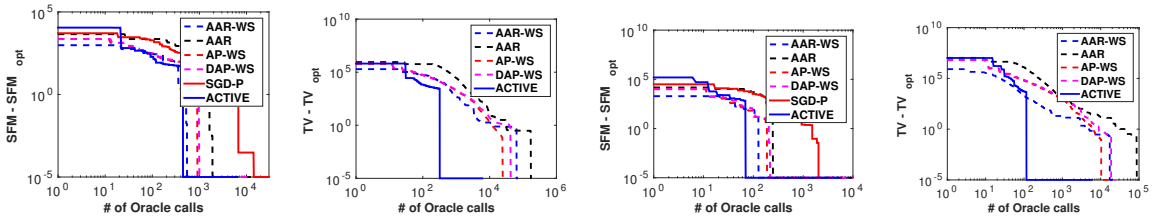
4.5 Decoupled problem

In our context, the quadratic program in Eq. (22) can be decoupled into smaller optimization problems. Let us consider the bipartite graph $G = (\mathcal{A}_1, \mathcal{A}_2, E)$ of which Q is the Laplacian matrix. The number of connected components of the graph, G is equal to the number of levelsets of w.

Let m be the total number of connected components in G. These connected components define a partition on the ground set V and a total order on elements of the partition can be obtained using the levels sets of w. Let k denote the index of each bipartite subgraph of G represented by $G_k = (\mathcal{A}_{1,k}, \mathcal{A}_{2,k}, E_k)$, where $k = 1, 2, \ldots, m$. Let J_k denote the indices of the nodes of G_k in G.

$$x^*_{J_k} = \underset{\substack{x \in \mathbb{R}^{m_{1,k}+m_{2,k}} \\ D(\mathcal{A}_1,\mathcal{A}_2)_k x \succcurlyeq 0}}{\text{argmin}} \frac{1}{2} x^\top Q(\mathcal{A}_1, \mathcal{A}_2)_{J_k J_k} x + c(\mathcal{A}_1, \mathcal{A}_2)^\top_{J_k} x, \tag{25}$$

where $m_{j,k}$ is size of $\mathcal{A}_{j,k}$. Therefore, $m_{1,k} + m_{2,k}$ is the total number of nodes in the subgraph G_k. Note that this is exactly equivalent to decomposition of base polytope of F_j into base polytopes of submodular



(a) (b) (c) (d)

Figure 7: (a) Number of 2D SFM calls to obtain 3D SFM, (b) Number of 2D SFM calls to obtain 3D TV, (c) Number of 2D SFM calls to obtain SFM of 2D + concave function, (d) Number of 2D SFM calls to obtain TV of 2D + concave function.

functions formed by contracting F_j on each individual components representing the connected component k. See Appendix C for more details.

5 Experiments

In this section, we show the results of the algorithms proposed on various problems. We first consider the problem of solving total variation denoising for a non decomposable function using active-set methods in Section 5.1, specifically cut functions. In Section 5.2, we consider cut functions on a 3D grid decomposed into a function of the 2D grid and a function of chains. We then consider a 2D grid and a concave function on cardinality, which is not a cut function.

5.1 Non decomposable total variation denoising

Our experiments consider images, which are 2-dimensional grids with 4-neighborhood. The dataset comprises of 6 different images of varying sizes. We consider a large image of size 5616×3744 and recursively scale into a smaller image of half the width and half the height maintaining the aspect ratio. Therefore, the size of each image is four times smaller than the size of the previous image. We restrict to anisotropic uniform-weighted total variation to compare with Chambolle et al. [11] but our algorithms works as well with weighted total variation, which is standard in computer vision, and on any graph with SFM oracles. Therefore, the unweighted total variation is

$$f(w) \;=\; \lambda \sum_{i \sim j} |w_i \;-\; w_j|,$$

where λ is a regularizing constant for solving the total variation problem in Eq. (1).

Maxflow [8] is used as the SFM oracle for checking the optimality of the ordered partitions. Figure 6(a) shows the number of SFM oracle calls required to solve the TV problem for images of various sizes. Note that in [11] each SFM oracle call optimizes smaller problems sequentially, while each SFM oracle call in our method optimizes several independent smaller problems in parallel. Therefore, our method has lesser number of oracle calls than [11]. However, oracle complexity of each call is higher for the our method when compared to [11]. Figure 6(b) shows the time required for each of the methods to solve the TV problem to convergence. We have an optimized code and only use the oracle as plugin which takes about 80-85 percent of the running time. This is primarily the reason our approach takes more time than [11] inspite of having lesser oracle calls for small images.

Figure 6(c) also shows the ability to warm start by using the output of a related problem, i.e., when computing the solution for several values of λ (which is typical in practice). In this case, we use optimal ordered partitions of the problem with larger λ to warm start the problem with smaller λ. It can be observed that warm start of the algorithm requires lesser number of oracle calls to converge than using the initialization with trivial ordered partition. Warm start also largely helps in reducing the burden on the SFM oracle. With warm starts the number of ordered partitions does not change much over iterations. Hence, it suffices to query only ordered partitions that have changed. To analyze this we define *oracle complexity* as the ratio of pixels in the elements of the partitions that need to be queried with the full set. Oracle complexity is averaged over iterations to understand the average burden on the oracle per iteration. With warm starts this reduces drastically, which can be observed in Figure 6(d).

5.2 Decomposable total variation denoising and SFM

Cut functions. In the decomposable case, we consider a 3D-grid that decomposes into a function F_1 composed of 2D grids and a function F_2 composed of chains. For each of these functions, the corresponding *SFM oracle* is a maxflow-mincut [8] and message passing algorithm on chains respectively. The corresponding *projection algorithm* can be solved by using the algorithm described in Section 3.4. We consider averaged alternating reflection (AAR) [5] by solving each projection without warmstart and counting the total number of 2D SFM oracle calls to solve the SFM and TV on the 3D-grid as our baseline. (SGD-P) denotes the dual subgradient based method [20] modified with Polyak's [6] rule to solve SFM on the 3D-grid. We show the performance of alternating projection (AP-WS), averaged alternating reflection (AAR-WS) [5] and Dykstra's alternating projection (DAP-WS) [3] using warmstart of each projection with the ordered partitions. WS denotes warmstart variant of each of the algorithm. The performance of the active-set algorithm proposed in Section 4.3 with inner loop solved using the primal active-set method proposed in Section 4.4.2 is represented by (ACTIVE).

In our experiments, we consider the 3D volumetric dataset of the Stanford bunny [1] of size $102 \times 100 \times 79$. F_1 denotes 102 2D frame of size 100×79 and F_2 represents the $100 \times 79 = 7900$ chains of length 102. Figure 7 (a) and (b) show that (AP-WS), (AAR-WS), (DAP-WS) and (ACTIVE) require relatively less number of oracle calls when compared to when compared to AAR or SGD-P. Note that we count 2D SFM oracle calls as they are more expensive than the SFM oracles on chains.

Time comparisons. We also performed time comparisons between the iterative methods and the combinatorial methods on standard datasets. The standard mincut-maxflow [8] on the 3D volumetric dataset of the Standard bunny [1] of size $102 \times 100 \times 79$ takes 0.11 seconds while averaged alternating reflections (AAR) without warmstart takes 0.38 seconds. The averaged alternating reflections with warmstart (AAR-WS) takes 0.21 seconds and the active-set method (ACTIVE) takes 0.38 seconds. The main bottleneck in the active-set method is the inversion of the Laplacian matrix and it could considerably improve by using methods suggested in [32]. Note that the projection on base polytopes of F_1 and F_2 can be parallelized by projecting onto each of the 2D frame of F_1 and each line of F_2 respectively [22]. The times for (AAR), (AAR-WS) and (ACTIVE) use parallel multicore architectures[3] while the combinatorial algorithm only uses a single core. Note that cut functions on grid structures are only a small subclass of submodular functions with such efficient combinatorial algorithms. In constrast, our algorithm works on more general class of sum of submodular functions than just cut functions.

Concave functions on cardinality. In this experiment we consider SFM problem of sum of a 2D cut on a graph of size 5616×3744 and a super pixel based concave function on cardinality [30, 18]. The

[3] 20 core, Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz with 100Gigabytes of memory. We only use up to 16 cores of the machine to ensure accurate timings

unary potentials of each pixel is calculated using the Gaussian mixture model of the color features. The edge weight $a(i,j) = \exp(-\|y_i - y_j\|^2)$, where y_i denotes the RGB values of the pixel i. In order to evaluate the concave function, regions R_j are extracted via superpixels and, for each R_j, defining the function $F_2(S) = |S||R_j \setminus S|$. We use 200 and 500 regions. Figure 7 (c) and (d) shows that (AP-WS), (AAR-WS), (DAP-WS) and (ACTIVE) algorithms converge for solving TV quickly by using only SFM oracles and relatively less number of oracle calls. Note that we count 2D SFM oracle calls.

6 Conclusion

In this paper, we present an efficient active-set algroithm for optimizing quadratic losses regularized by Lovász extesion of a submodular function using the SFM oracle of the function. We also present an active-set algorithms to minimize sum of "simple" submodular functions using SFM oracles of the individual "simple" functions. We also show that these algorithms are competitive to the existing state-of-art algorithms to minimize submodular functions.

Acknowledgements

We acknowledge support from the European Research Council grant SIERRA (project 239993).

References

[1] Maxflow dataset online. `http://vision.csd.uwo.ca/maxflow-data`.

[2] F. Bach. *Learning with Submodular Functions: A Convex Optimization Perspective*, volume 6 of *Foundations and Trends in Machine Learning*. NOW, 2013.

[3] H. H. Bauschke and J. M. Borwein. Dykstra's alternating projection algorithm for two sets. *Journal of Approximation Theory*, 79(3):418–443, 1994.

[4] H. H. Bauschke, J. M. Borwein, and A. S. Lewis. The method of cyclic projections for closed convex sets in Hilbert space. *Contemporary Mathematics*, 204:1–38, 1997.

[5] H. H. Bauschke, P. L. Combettes, and D. Luke. Finding best approximation pairs relative to two closed convex sets in Hilbert spaces. *Journal of Approximation Theory*, 127(2):178–192, 2004.

[6] D. P. Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.

[7] M. J. Best and N. Chakravarti. Active set algorithms for isotonic regression: a unifying framework. *Mathematical Programming*, 47(1):425–439, 1990.

[8] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[9] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[10] D. Chakrabarty, P. Jain, and P. Kothari. Provable submodular minimization using Wolfe's algorithm. In *Advances in Neural Information Processing Systems*. 2014.

[11] A. Chambolle and J. Darbon. On total variation minimization and surface evolution using parametric maximum flows. *International Journal of Computer Vision*, 84(3):288–307, 2009.

[12] A. Chambolle and T. Pock. A remark on accelerated block coordinate descent for computing the proximity operators of a sum of convex functions. Technical Report 01099182, HAL, 2015.

[13] S. Fujishige. Lexicographically optimal base of a polymatroid with respect to a weight vector. *Mathematics of Operations Research*, 5(2):186–196, 1980.

[14] S. Fujishige. *Submodular Functions and Optimization*. Elsevier, 2005.

[15] N. Gaffke and R. Mathar. A cyclic projection algorithm via duality. *Metrika*, 36(1):29–54, 1989.

[16] D. Goldfarb and W. Yin. Parametric maximum flow algorithms for fast total variation minimization. *SIAM Journal on Scientific Computing*, 31(5):3712–3743, 2009.

[17] H. Groenevelt. Two algorithms for maximizing a separable concave function over a polymatroid feasible region. *European Journal of Operational Research*, 54(2):227–236, 1991.

[18] S. Jegelka, F. Bach, and S. Sra. Reflection methods for user-friendly submodular optimization. In *Advances in Neural Information Processing Systems*, 2013.

[19] V. Kolmogorov. Minimizing a sum of submodular functions. *Discrete Applied Mathematics*, 160(15), 2012.

[20] N. Komodakis, N. Paragios, and G. Tziritas. MRF energy minimization and beyond via dual decomposition. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, 33(3):531–552, 2011.

[21] A. Krause and C. Guestrin. Submodularity and its applications in optimized information gathering. *ACM Transactions on Intelligent Systems and Technology*, 2(4), 2011.

[22] K. S. S. Kumar, A. Barbero, S. Jegelka, S. Sra, and F. Bach. Convex optimization for parallel energy minimization. Technical Report 01123492, HAL, 2015.

[23] L. Ladicky, C. Russell, P. Kohli, and P. H. S. Torr. Graph cut based inference with co-occurrence statistics. In *Proceedings of the 11th European Conference on Computer Vision*, 2010.

[24] L. Landrieu and G. Obozinski. Cut Pursuit: fast algorithms to learn piecewise constant functions. In *Proceedings of Artificial Intelligence and Statistics*, 2016.

[25] H. Lin and J. Bilmes. A class of submodular functions for document summarization. In *Proceedings of NAACL/HLT*, 2011.

[26] L. Lovász. Submodular functions and convexity. *Mathematical programming: the state of the art, Bonn*, pages 235–257, 1982.

[27] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer Series in Operations Research and Financial Engineering. Springer, Berlin, 2006.

[28] R. T. Rockafellar. *Convex Analysis*. Princeton U. P., 1997.

[29] X. Shusheng. Estimation of the convergence rate of Dykstras cyclic projections algorithm in polyhedral case. *Acta Mathematicae Applicatae Sinica (English Series)*, 16(2):217–220, 2000.

[30] P. Stobbe and A. Krause. Efficient minimization of decomposable submodular functions. In *Advances in Neural Information Processing Systems*, 2010.

[31] R. Tarjan, J. Ward, B. Zhang, Y. Zhou, and J. Mao. Balancing applied to maximum network flow problems. In *European Symp. on Algorithms (ESA)*, pages 612–623, 2006.

[32] N. Vishnoi. *Lx = B - Laplacian Solvers and Their Algorithmic Applications*. Now Publishers, 2013.

A Algorithms for coalescing partitions

The basic interpretation in coalescing two ordered partitions is as follows. Given an ordered partition $\mathcal{A}_1$ and $\mathcal{A}_2$ with m_1 and m_2 elements in the partitions respectively, we define for each $j = 1, 2$, $\forall i_j = (1, \ldots, m_j)$, $B_{j,i_j} = (A_{j,1} \cup \ldots \cup A_{j,i_j})$.

The inequalities defining the outer approximation of the base polytopes are given by hyperplanes defined by

$$\forall i_j = (1, \ldots, m_j), s_j(B_{j,i_j}) \leq F_j(B_{j,i_j})$$

. The hyperplanes defined by common sets of both these partitions, defines the coalesced ordered partitions. The following algorithm performs coalescing between these partitions.

- **Input**: Ordered partitions $\mathcal{A}_1$ and $\mathcal{A}_2$.
- **Initialize**: $x = 1$, $y = 1$, $z = 1$ and $C = \varnothing$.
- **Algorithm**: Iterate until $x = m_1$ and $y = m_2$ with $m = z$
  (a) If $|B_{1,x}| > |B_{2,y}|$ then $y := y + 1$.
  (b) If $|B_{1,x}| < |B_{2,y}|$ then $x := x + 1$.
  (c) If $|B_{1,x}| == |B_{2,y}|$ then
    - If $B_{1,x} == B_{2,y}$ then
      * $A_z = (B_{1,x} \setminus C)$,
      * $C = B_{1,x}$, and
      * $z := z + 1$.
- **Output**: $m = z$, ordered partitions $\mathcal{A} = (A_1, \ldots, A_m)$.

B Optimality of algorithm for decomposable problems

In step (d) of the algorithms, when we split partitions, the value of the primal/dual pair of optimization algorithms

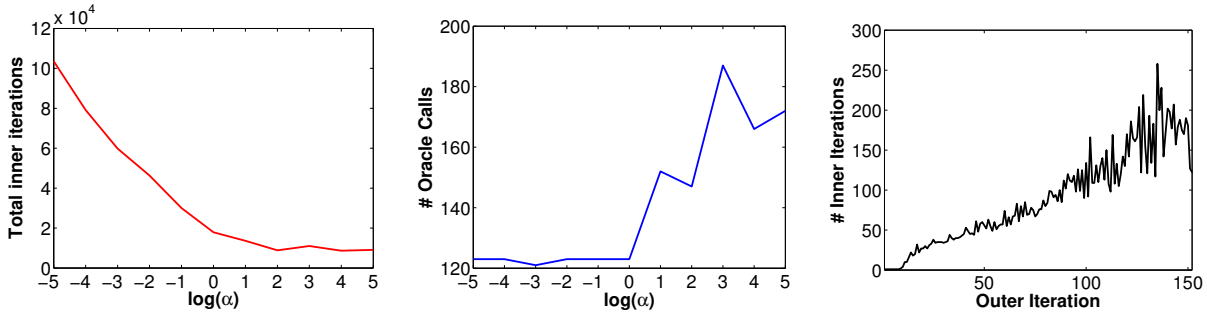
$$\max_{\substack{s_1 \in \widehat{B}^{\mathcal{A}_1}(F_1) \\ s_2 \in \widehat{B}^{\mathcal{A}_2}(F_2)}} -\tfrac{1}{2}\|u - s_1 - s_2\|_2^2 = \min_{\substack{w \in \mathcal{W}^{\mathcal{A}_1} \\ w \in \mathcal{W}^{\mathcal{A}_2}}} f_1(w) + f_2(w) - u^\top w + \tfrac{1}{2}\|w\|_2^2,$$

cannot increase. This is because, when splitting, the constraint set for the minimization problem only gets bigger. Since at optimality, we have $w = u - s_1 - s_2$, $\|w\|_2$ cannot decrease, which shows the first statement.

Now, if $\|w\|_2$ remains constant after an iteration, then it has to be the same (and not only have the same norm), because the optimal s_1 and s_2 can only move in the direction orthogonal to w.

In step (b) of the algorithm, we project 0 on the (non-empty) intersection of $\widehat{B}^{\mathcal{A}_1}(F_1) - u/2 + w/2$ and $u/2 - w/2 - \widehat{B}^{\mathcal{A}_2}(F_2)$. This corresponds to minimizing $\frac{1}{2}\|s_1 - u/2 + w/2\|^2$ such that $s_1 \in \widehat{B}^{\mathcal{A}_1}(F_1)$ and $s_2 = u - w - s_1 \in \widehat{B}^{\mathcal{A}_2}(F_2)$. This is equivalent to minimizing $\frac{1}{8}\|s_1 - s_2\|^2$. We have:

$$\max_{\substack{s_1 \in \widehat{B}^{\mathcal{A}_1}(F_1) \\ s_2 \in \widehat{B}^{\mathcal{A}_2}(F_2) \\ s_1 + s_2 = u - w}} -\frac{1}{8}\|s_1 - s_2\|_2^2 \quad = \quad \min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \max_{\substack{s_1 \in \mathbb{R}^n \\ s_2 \in \mathbb{R}^n \\ s_1 + s_2 = u - w}} \left(-\frac{1}{8}\|s_1 - s_2\|_2^2 + f_1(w_1) + f_2(w_2) \right.$$
$$\left. -w_1^\top s_1 - w_2^\top s_2 \right)$$



(a) (b) (c)

Figure 8: (a) Total number of inner iterations for varying α. (b) Total number of outer iterations for varying α. and (c) Number of inner iterations per each outer iteration for the $\alpha = 10^1$.

$$
\begin{aligned}
&= \min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \max_{s_2 \in \mathbb{R}^n} \Big(-\frac{1}{8}\|u - w - 2s_2\|_2^2 + f_1(w_1) + f_2(w_2) \\
&\qquad\qquad -w_1^\top (u - w - s_2) - w_2^\top s_2 \Big) \\
&= \min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \max_{s_2 \in \mathbb{R}^n} \Big(-\frac{1}{8}\|u - w\|_2^2 - \frac{1}{2}\|s_2\|_2^2 + \frac{1}{2} s_2^\top (u - w) \\
&\qquad\qquad + f_1(w_1) + f_2(w_2) - w_1^\top (u - w - s_2) - w_2^\top s_2 \Big) \\
&= \min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} -w_1^\top (u - w) + f_1(w_1) + f_2(w_2) - \frac{1}{8}\|u - w\|_2^2 \\
&\qquad\qquad + \max_{s_2 \in \mathbb{R}^n} -\frac{1}{2}\|s_2\|_2^2 + s_2^\top \big(\tfrac{u-w}{2} + w_1 - w_2 \big) \\
&= \min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \Big(-w_1^\top (u - w) + f_1(w_1) + f_2(w_2) - \frac{1}{8}\|u - w\|_2^2 \\
&\qquad\qquad + \frac{1}{2}\| \tfrac{u-w}{2} + w_1 - w_2 \|_2^2 \Big) \\
&= \min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \Big(-w_1^\top (u - w) + f_1(w_1) + f_2(w_2) + \frac{1}{2}\|w_1 - w_2\|_2^2 \\
&\qquad\qquad + \frac{1}{2}(u - w)^\top (w_1 - w_2) \Big) \\
&= \min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \Big(f_1(w_1) + f_2(w_2) - \frac{1}{2}(u - w)^\top (w_1 + w_2) \\
&\qquad\qquad + \frac{1}{2}\|w_1 - w_2\|_2^2 \Big),
\end{aligned}
$$

Thus s_1 and s_2 are dual to certain vectors w_1 and w_2, which minimize a decoupled formulation in f_1 and f_2. To check optimality, like in the single function case, it decouples over the constant sets of w_1 and w_2, which is exactly what step (c) is performing.

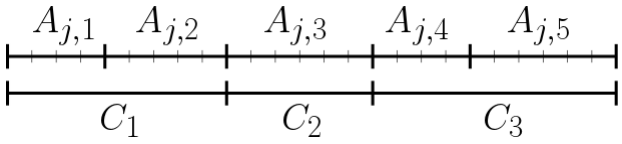
If the check is satisfied, it means that w_1 and w_2 are in fact optimal for the problem above without the restriction in compatibilities, which implies that they are the Dykstra solutions for the TV problem.

If the check is not satisfied, then the same reasoning as for the one function case, leads directions of descent for the new primal problem above. Hence it decreases; since its value is equal to $-\frac{1}{8}\|s_1 - s_2\|_2^2$, the value of $\|s_1 - s_2\|_2^2$ must increase, hence the second statement.

C Decoupled problems.

Given that we deal with polytopes, knowing w implies that we know the faces on which we have to looked for. It turns outs that for base polytopes, these faces are products of base polytopes for modified functions (a similar fact holds for their outer approximations).

Given the ordered partition $\mathcal{A}'$ defined by the level sets of w (which have to be finer than $\mathcal{A}_1$ and $\mathcal{A}_2$), we know that we may restrict $\widehat{B}^{\mathcal{A}_j}(F_j)$ to elements s such that $s(B) = F(B)$ for all sup-level sets B of w (which have to be unions of contiguous elements of A_j); see an illustration below.



More precisely, if $C_1, \dots, C_{m'}$ are constant sets of w ordered with decreasing values. Then, we may search for s_j independently for each subvector $(s_j)_{C_k} \in \mathbb{R}^{C_k}$, $k \in \{1, \dots, m'\}$ and with the constraint that

$$(s_j)_{C_k} \in \hat{B}^{\mathcal{A}_j \cap C_k}\big[(F_j)_{C_k | C_1 \cup \cdots \cup C_{k-1}}\big],$$

where $\mathcal{A}_j \cap C_k$ is the ordered partition obtained from $\mathcal{A}_j$ once restricted onto C_k and the submodular function is the so-called contraction of F on C_k given $C_1 \cup \cdots \cup C_{k-1}$, defined as $S \mapsto F_j(S \cup C_1 \cup \cdots \cup C_{k-1}) - F(C_1 \cup \cdots \cup C_{k-1})$. Thus this corresponds to solving m different smaller subproblems.

D Choice of α

The Dykstra step, i.e., step (b) of the algorithm proposed in Section 4.4.1 is not finitely convergent. Therefore, it needs to be solved approximately. For this purpose, we introduce a parameter α to approximately solve the Dykstra step such that $\|s_1 + s_2 - u + w\|_1 \leq \alpha(\epsilon_1 + \epsilon_2)$. Let ϵ be defined as $\alpha(\epsilon_1 + \epsilon_2)$. This shows that the s_1 and s_2 are ϵ-accurate. Therefore, α must be chosen in such a way that we avoid cycling in our algorithm. However, another alternative is to warm start the dykstra step with w_1 and w_2 of the previous iteration. This ensures we dont go back to the same w_1 and w_2, which we have already encountered and avoid cycling. Figure 8 shows the performance of our algorithm for a simple problem of 100×100 2D-grid with 4-neighborhood and uniform weights on the edges with varying α. Figure 8-(a) shows the total number of inner iterations required to solve the TV problem. Figure 8-(b) gives the total number of SFM oracle calls required to solve the TV problem. In Figure 8-(c), we show the number of inner iterations in every outer iteration for the best α we have encountered.