

Active-set Methods for Submodular Minimization Problems

K. S. Sesh Kumar

*Imperial College London, Department of Computing
180 Queen's Gate, London SW7 2AZ*

S.KARRI@IMPERIAL.AC.UK

Francis Bach

INRIA - Sierra Project-team

Département d'Informatique de l'Ecole Normale Supérieure (UMR CNRS/ENS/INRIA)

2, rue Simone Iff

75012 Paris, France

FRANCIS.BACH@ENS.FR

Editor: Sebastian Nowozin

Abstract

We consider the submodular function minimization (SFM) and the quadratic minimization problems regularized by the Lovász extension of the submodular function. These optimization problems are intimately related; for example, min-cut problems and total variation denoising problems, where the cut function is submodular and its Lovász extension is given by the associated total variation. When a quadratic loss is regularized by the total variation of a cut function, it thus becomes a total variation denoising problem and we use the same terminology in this paper for “general” submodular functions. We propose a new active-set algorithm for total variation denoising with the assumption of an oracle that solves the corresponding SFM problem. This can be seen as local descent algorithm over ordered partitions with explicit convergence guarantees. It is more flexible than the existing algorithms with the ability for warm-restarts using the solution of a closely related problem. Further, we also consider the case when a submodular function can be decomposed into the sum of two submodular functions F_1 and F_2 and assume SFM oracles for these two functions. We propose a new active-set algorithm for total variation denoising (and hence SFM by thresholding the solution at zero). This algorithm also performs local descent over ordered partitions and its ability to warm start considerably improves the performance of the algorithm. In the experiments, we compare the performance of the proposed algorithms with state-of-the-art algorithms, showing that it reduces the calls to SFM oracles.

Keywords: discrete optimization, submodular function minimization, convex optimization, cut functions, total variation denoising.

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 (Bach, 2013), with notable applications to graph cut-based image segmentation (Boykov et al., 2001), sensor placement (Krause and Guestrin, 2011), or document summarization (Lin and Bilmes, 2011).

Let F be a normalized submodular function defined on $V = \{1, \dots, n\}$, i.e., $F : 2^V \rightarrow \mathbb{R}$ such that $F(\emptyset) = 0$ and an n -dimensional real vector u , i.e., $u \in \mathbb{R}^n$. In this paper, we consider the

submodular function minimization (SFM) problem,

$$\min_{A \subset V} F(A) - u(A), \quad (1)$$

where we use the convention $u(A) = u^\top 1_A$ and $1_A \in \{0, 1\}^n$ is the indicator vector of the set A . Note that general submodular functions can always be decomposed into a normalized submodular function, F , i.e., $F(\emptyset) = 0$ and a modular function u (see Bach, 2013).

Let f be the Lovász extension of the submodular function F . Let us consider the following continuous optimization problem

$$\min_{w \in [0, 1]^n} f(w) - u^\top w. \quad (2)$$

As a consequence of submodularity, the discrete and continuous optimization problems in Eq. 1 and Eq. 2 respectively have the same optimal solutions (Lovász, 1982). Let us consider another related continuous optimization problem

$$\min_{w \in \mathbb{R}^n} f(w) - u^\top w + \frac{1}{2} \|w\|_2^2. \quad (3)$$

If F is a cut function in a weighted undirected graph, then f is its associated total variation, hence the denomination of *total variation denoising* (TV) problem for Eq. 3, which we use in this paper—since it is equivalent to minimizing $\frac{1}{2} \|u - w\|_2^2 + f(w)$. The unique solution of the total variation denoising in Eq. 3 can be used to obtain the solution of the SFM problem in Eq. 1 by thresholding at 0. Conversely, we may obtain the optimal solution of the total variation denoising in Eq. 3 by solving a series of SFM problems using divide-and-conquer strategy.

Relationship with existing work. Generic algorithms to optimize SFM in Eq. 1 or TV in Eq. 3 problems which only access F through function values, e.g., subgradient descent or min-norm-point algorithm (Fujishige, 1984), are too slow without any assumptions (Bach, 2013), as for signal processing applications, high precision is typically required (and often the exact solution).

For decomposable problems, i.e., when $F = F_1 + \dots + F_r$, where each F_j is “simple”, some algorithms use more powerful oracles than function evaluations, improving the running times. These powerful oracles include SFM oracles that can solve the SFM problem of simple submodular function, F_j given by

$$\min_{A \subset V} F_j(A) - u_j(A), \quad (4)$$

where $u_j \in \mathbb{R}^n$. The other set of powerful oracles are total variation or TV oracles, that solve TV problems of the form

$$\min_{w \in \mathbb{R}^n} f_j(w) - u_j^\top w + \frac{1}{2} \|w\|_2^2, \quad (5)$$

where $u_j \in \mathbb{R}^n$. Note that, in general, the exact total variation oracles are at most $O(n)$ times more expensive than their respective SFM oracle as they solve all SFM problems

$$\min_{A \subset V} F_j(A) - u_j(A) + \lambda |A|, \quad (6)$$

for all $\lambda \in \mathbb{R}$, which have at most $O(n)$ unique solutions. For more details refer to Fujishige (1980) and Bach (2013). Here, $|A|$ denotes the cardinality of the set A . There does exist 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 more expensive than the corresponding SFM oracles but are still too expensive in practice.

Stobbe and Krause (2010) used SFM oracles instead of function value oracles but their algorithm remains slow in practice. However, when total variation oracles for each F_j are used, they become competitive (Komodakis et al., 2011; Kumar et al., 2015; Jegelka et al., 2013). Therefore, our goal is to design fast optimization strategies using only efficient SFM oracles for each function F_j rather than their expensive TV oracles (Kumar et al., 2015; Jegelka et al., 2013) to solve the SFM and TV denoising problems of F given by Eq. 1 and Eq. 3 respectively. An algorithm was proposed by Landrieu and Obozinski (2016) to search over partition space for solving Eq. 3 with the unary terms $(-u^\top w)$ replaced by a convex differentiable function but it applies only to functions F , which are cut functions.

In this paper, we exploit the polytope structure of these non-smooth optimization problems with exponentially many constraints, i.e., 2^n , where each face of the constraint set is indexed by an ordered partition of the underlying ground set $V = \{1, \dots, n\}$. The main insight of this paper is that given the main polytope associated with a submodular function (namely the base polytope described in Section 2) and an ordered partition, we may uniquely define a tangent cone of the polytope. Further, orthogonal projections onto the tangent cone may be done efficiently by isotonic regressions (Best and Chakravarti, 1990). The time needed is linear in the number of elements of the ordered partition used to define the tangent cone. We need SFM oracles only to check the optimality of the ordered partition. Given the orthogonal projection s onto the tangent cone, if the minimum of $F(A) - s(A)$ with respect to $A \subseteq V$ is positive then it is optimal. If it is not optimal, it gives us the violating constraints in the form of active-sets that enable us to generate a new ordered partition among the exponentially many ordered partitions.

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. This algorithm may be seen as a local descent algorithm over ordered partitions. It has the additional advantage of allowing warm restarts, which will be beneficial when we have to solve a large number of total variation denoising problems as shown in Section 5.
- 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 Jegelka et al. (2013) and Kumar et al. (2015), they are also naturally parallelizable. Given that only SFM oracles are needed, it is much more flexible than the algorithms requiring a TV oracle, and allow more applications as shown in Section 5.

2. Review of Submodular Analysis

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

$$F(A) = \sum_{i < j} a(i, j) |1_{i \in A} - 1_{j \in A}|, \quad (7)$$

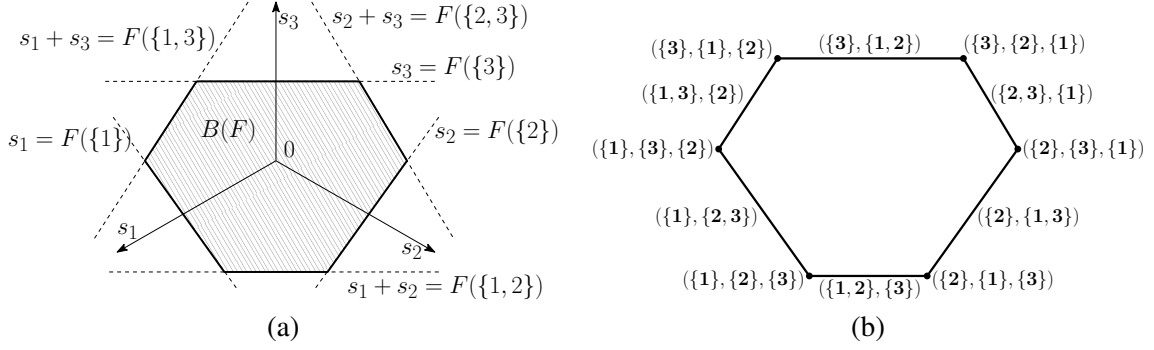


Figure 1: Base polytope for $n=3$. (a) definition from its supporting hyperplanes $\{s(A) = F(A)\}$. (b) each face (point or segment) of $B(F)$ is associated with an ordered partition.

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. 7 (Ladicky et al., 2010; Kolmogorov, 2012). However, we use cut functions as a running example to better explain our algorithm as they are most widely studied and understood among submodular functions. We now review the relevant concepts from submodular analysis (for more details, see Bach, 2013; Fujishige, 2005).

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 $[0, 1]^n$ 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 $v_1 > \dots > v_m$, $f(w)$ is equal to $f(w) = \sum_{i=1}^m v_i [F(B_i) - F(B_{i-1})]$, where $B_i = (A_1 \cup \dots \cup A_i)$. 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. 3.

This extension is piecewise linear for any set-function F . It turns out that it is convex if and only if F is submodular (Lovász, 1982). Any piecewise linear convex function may be represented as the support function of a certain polytope K , i.e., as $f(w) = \max_{s \in K} w^\top s$ (Rockafellar, 1997). For the Lovász extension of a submodular function, there is an explicit description of K , which we now review.

Base polytope. We define the *base polytope* as

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

Given that it is included in the affine hyperplane $\{s(V) = F(V)\}$, it is traditionally represented by the projection 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 $w \in \mathbb{R}^n$,

$$f(w) = \sup_{s \in B(F)} w^\top s. \quad (8)$$

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 $w_{\sigma(1)} \geq \dots \geq w_{\sigma(n)}$ 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

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

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

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

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

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 (Bach, 2013, Sec. 8):

$$\begin{aligned} \min_{w \in \mathbb{R}^n} f(w) - u^\top w + \frac{1}{2} \|w\|_2^2 &= \min_{w \in \mathbb{R}^n} \max_{s \in B(F)} s^\top w - u^\top w + \frac{1}{2} \|w\|_2^2 \text{ using Eq. 8,} \\ &= \max_{s \in B(F)} \min_{w \in \mathbb{R}^n} s^\top w - u^\top w + \frac{1}{2} \|w\|_2^2, \\ &= \max_{s \in B(F)} -\frac{1}{2} \|s - u\|_2^2, \end{aligned} \tag{9}$$

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. 1 and the TV problem in Eq. 3 are tightly connected. Indeed, given the unique solution w of the TV problem, we obtain a solution of $\min_{A \subseteq V} F(A) - u(A)$ by thresholding w at 0, i.e., by taking $A = \{i \in V, w_i \geq 0\}$ (Fujishige, 1980).

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 (Groenevelt, 1991). The extended algorithm of Jegelka et al. (2013) can reach a precision ε in $O(\log \frac{1}{\varepsilon})$ iterations but can only get the exact solution in $O(n)$ oracles. Fast efficient algorithms are proposed to solve TV problems with $O(1)$ oracles (Chambolle and Darbon, 2009; Goldfarb and Yin, 2009) but are specific to cut functions on simple graphs (chains and trees) as they exploit the weight representation given by Eq. 7. Our algorithm in Section 3 is a generalization of the divide-and-conquer strategy for solving the TV problem with general submodular functions.

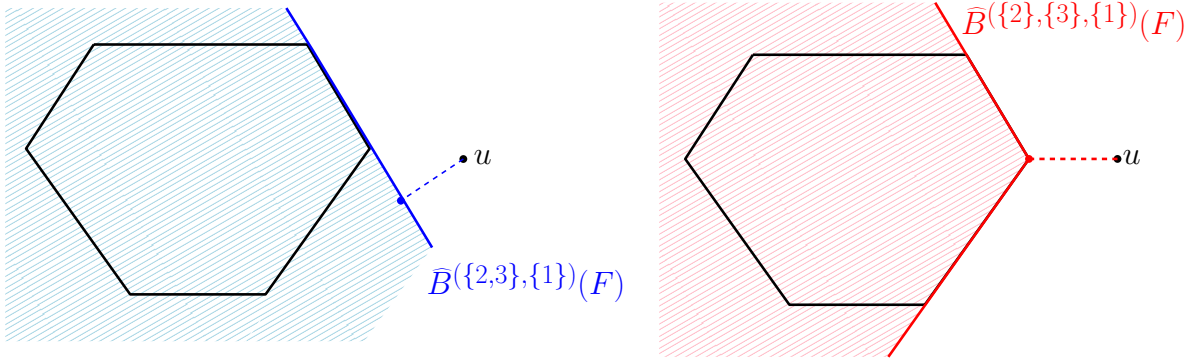


Figure 2: Projection algorithm for a single polytope: first projecting on the outer approximation $\widehat{B}^{\{\{2,3\},\{1\}\}}(F)$, with a projected element which is not in $B(F)$ (blue), then on $\widehat{B}^{\{\{2\},\{3\},\{1\}\}}(F)$, with a projected element being the projection of s onto $B(F)$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

3. 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.

3.1 Outer Approximations of $B(F)$

Supporting hyperplanes. The base polytope is defined as the intersection of half-spaces $\{s(A) \leq F(A)\}$, for $A \subseteq V$. Therefore, faces of $B(F)$ are indexed by subsets of the power set. As a consequence of submodularity (Bach, 2013; Fujishige, 2005), the faces of the base polytope $B(F)$ are characterized by “ordered partitions” $\mathcal{A} = (A_1, \dots, A_m)$ with $V = A_1 \cup \dots \cup A_m$. Then, a face of $B(F)$ is such that $s(B_i) = F(B_i)$ for all $B_i = A_1 \cup \dots \cup A_i$, $i = 1, \dots, m$. See 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 $v_1 > \dots > v_m$ 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

$$\widehat{B}^{\mathcal{A}}(F) = \left\{ s \in \mathbb{R}^n, s(V) = F(V), \forall i \in \{1, \dots, m-1\}, s(B_i) \leq F(B_i) \right\}. \quad (10)$$

Since we have relaxed all the constraints unrelated to \mathcal{A} , 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 is an upper bound on $f(w)$ since this set is an outer approximation of $B(F)$ as follows:

$$\sup_{s \in \widehat{B}^{\mathcal{A}}(F)} w^\top s = \sup_{s \in \mathbb{R}^n} \inf_{\lambda \in \mathbb{R}_+^{m-1} \times \mathbb{R}} w^\top s - \sum_{i=1}^m \lambda_i (s(B_i) - F(B_i)), \text{ using Lagrangian duality,}$$

$$\begin{aligned}
 &= \inf_{\lambda \in \mathbb{R}_+^{m-1} \times \mathbb{R}} \sup_{s \in \mathbb{R}^n} s^\top \left(w - \sum_{i=1}^m (\lambda_i + \dots + \lambda_m) 1_{A_i} \right) \\
 &\quad + \sum_{i=1}^m (\lambda_i + \dots + \lambda_m) [F(B_i) - F(B_{i-1})], \\
 &= \inf_{\lambda \in \mathbb{R}_+^{m-1} \times \mathbb{R}} \sum_{i=1}^m (\lambda_i + \dots + \lambda_m) [F(B_i) - F(B_{i-1})] \\
 &\quad \text{such that } w = \sum_{i=1}^m (\lambda_i + \dots + \lambda_m) 1_{A_i}.
 \end{aligned}$$

Thus, by defining $v_i = \lambda_i + \dots + \lambda_m$, 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}). In other words, if $w = \sum_{i=1}^m v_i 1_{A_i}$, the support function 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

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

Therefore,

$$\sup_{s \in \hat{B}^{\mathcal{A}}(F)} w^\top s = \begin{cases} f(w) & \text{if } w \in \mathcal{W}^{\mathcal{A}}, \\ \infty & \text{otherwise.} \end{cases} \quad (11)$$

3.2 Isotonic Regression for Restricted Problems

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

$$\min_{v \in \mathbb{R}^m} \sum_{i=1}^m v_i [F(B_i) - F(B_{i-1}) - u(A_i)] + \frac{1}{2} \sum_{i=1}^m |A_i| v_i^2 \text{ such that } v_1 \geq \dots \geq v_m. \quad (12)$$

This may be done by isotonic regression in complexity $O(m)$ by the weighted pool-adjacent-violator algorithm (Best and Chakravarti, 1990). 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. Because none of the constraints are tight, 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. 12 is a constrained TV denoising problem that minimizes the cost function in Eq. 3 but with the constraint that weights are compatible with the ordered partition \mathcal{A} ,

1. Given a submodular function F and an ordered partition \mathcal{A} , when the unique solution problem in Eq. 12 is such that $v_1 > \dots > v_m$, we say that \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.

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 in exactly the same way as shown in Eq. 9 in the previous section, using the definition of the support function defined by Eq. 11. 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 onto 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 $w \in \mathbb{R}^n$ obtained from Eq. 12 is optimal for the TV problem in Eq. 3 if and only if $s = u - w \in B(F)$ due to optimality conditions in Eq. 9, 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 $C_i = C \cap A_i$:

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

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 (Tarjan et al., 2006).

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. 3. Otherwise, this new strictly finer partition may not be basic, the value of the optimization problem

in Eq. 12 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.

Data: Submodular function F with SFM oracle, $u \in \mathbb{R}^n$, ordered partition \mathcal{A}

Result: primal optimal: $w \in \mathbb{R}^n$ and dual optimal: $s \in B(F)$

```

1 while True do
2   Solve Eq. 12 using isotonic regression and update  $\mathcal{A}$  with the basic ordered partition ;
3   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
       $i \in \{1, \dots, m\}$  ;
4   if  $s$  is optimal then
5     break ;
6   else
7     for  $i \in \{1, \dots, m\}$ , split the set  $A_i$  into  $C_i$  and  $A_i \setminus C_i$  in that order to get an updated
      ordered partition  $\mathcal{A}$  ;
8   end
9 end
    
```

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 (Groenvelt, 1991), that is, the isotonic regression problem is always solved without using the constraints of monotonicity, i.e., there are no merges, 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 the convergence of the algorithm in Section 3.4, we only need to show that if the optimality check fails in step (4), then step (7) introduces splits in the partition, which ensures that the isotonic regression in step (2) of the next iteration has a strictly lower value. Let us recall the isotonic regression problem solved in step (2):

$$\min_{v \in \mathbb{R}^m} \sum_{i=1}^m \left(v_i [F(B_i) - F(B_{i-1}) - u(A_i)] + \frac{1}{2} |A_i| v_i^2 \right) \quad (13)$$

$$\text{such that } v_1 \geq \dots \geq v_m. \quad (14)$$

Steps (2) 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 $i = \{1, \dots, m\}$ is given through

$$v_i |A_i| = u(A_i) - (F(B_i) - F(B_{i-1})), \quad (15)$$

which can be calculated in closed form.

The optimality check in step (4) 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

$$F(B_{i-1} \cup C_i) - F(B_{i-1}) - s(C_i) < 0. \quad (16)$$

We will show that the splits introduced by step (7) strictly reduces the function value of isotonic regression in Eq. 13, 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. 13 is equal to

$$\sum_{i=1}^m \left(v_i [F(B_{i-1} \cup C_i) - F(B_{i-1}) - u(C_i)] + v_i [F(B_i) - F(B_{i-1} \cup C_i) - u(A_i \setminus C_i)] + \frac{1}{2} v_i^2 |C_i| + \frac{1}{2} v_i^2 |A_i \setminus C_i| \right). \quad (17)$$

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., $v_1 \geq \dots \geq v_i + t > v_i - t \geq \dots \geq v_m$. The function value is given by

$$\begin{aligned} & \sum_{i=1}^m \left((v_i + t) [F(B_{i-1} \cup C_i) - F(B_{i-1}) - u(C_i)] \right. \\ & \quad \left. + (v_i - t) [F(B_i) - F(B_{i-1} \cup C_i) - u(A_i \setminus C_i)] + \frac{1}{2} (v_i + t)^2 |C_i| + \frac{1}{2} (v_i - t)^2 |A_i \setminus C_i| \right) \\ &= \sum_{i=1}^m \left((v_i [F(B_{i-1} \cup C_i) - F(B_{i-1}) - u(C_i)] + v_i [F(B_i) - F(B_{i-1} \cup C_i) - u(A_i \setminus C_i)] \right. \\ & \quad \left. + \frac{1}{2} v_i^2 |C_i| + \frac{1}{2} v_i^2 |A_i \setminus C_i|) + \frac{1}{2} t^2 |A_i| \right. \\ & \quad \left. + t (2F(B_{i-1} \cup C_i) - F(B_{i-1}) - F(B_i) - u(C_i) + u(A_i \setminus C_i) + v_i |C_i| - v_i |A_i \setminus C_i|) \right). \end{aligned}$$

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

$$\begin{aligned} & 2F(B_{i-1} \cup C_i) - F(B_{i-1}) - F(B_i) - u(C_i) + u(A_i \setminus C_i) + |C_i| v_i - |A_i \setminus C_i| v_i \\ &= 2F(B_{i-1} \cup C_i) - F(B_{i-1}) - F(B_i) - 2u(C_i) + u(A_i) + 2|C_i| v_i - |A_i| v_i \\ &= 2(F(B_{i-1} \cup C_i) - F(B_{i-1}) - u(C_i) + v_i |C_i|) \quad (\text{substituting Eq. 15}) \\ &= 2(F(B_{i-1} \cup C_i) - F(B_{i-1}) - s(C_i)) < 0 \quad (\text{as } s = u - w \text{ and Eq. 16}). \end{aligned}$$

This shows that the function strictly decreases with the splits introduced in step (7).

3.6 Discussion

Certificates of optimality. The algorithm has dual-infeasible iterates s (they only belong to $B(F)$ at convergence). Suppose that after step (3) we have $F(C) - s(C) \geq -\varepsilon$ for all $C \subset V$, where ε

shrinks as we run more iterations of the outer loop. This implies that $s \in B(F + \varepsilon 1_{\text{Card} \in (1,n)})$, i.e., $s \in B(F_\varepsilon)$ with $F_\varepsilon = F + \varepsilon 1_{\text{Card} \in (1,n)}$. Since by construction $w = u - s$, we have:

$$\begin{aligned}
 f_\varepsilon(w) - u^\top w + \frac{1}{2}\|w\|_2^2 + \frac{1}{2}\|s - u\|_2^2 &= \varepsilon \left| \max_{j \in V} w_j - \min_{j \in V} w_j \right| + f(w) - u^\top w + \|w\|^2 \\
 &= \varepsilon \left| \max_{j \in V} w_j - \min_{j \in V} w_j \right| \\
 &\quad + \sum_{i=1}^m v_i [F(B_i) - F(B_{i-1}) - u(A_i)] + \sum_{i=1}^m |A_i| v_i^2 \\
 &= \varepsilon \left| \max_{j \in V} w_j - \min_{j \in V} w_j \right| \text{ (using Eq. 15)} \\
 &= \varepsilon \text{ range}(w),
 \end{aligned}$$

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

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 $\text{range}(w) \leq \text{range}(u) + \max_{i \in V} \{F(\{i\}) + F(V \setminus \{i\}) - F(V)\}$. 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:

$$\begin{aligned}
 F(B_i) - F(B_{i-1}) &\leq \sum_{k \in A_i} F(\{k\}) \text{ and} \\
 F(B_i) - F(B_{i-1}) &\geq - \sum_{k \in A_i} F(V) - F(V \setminus \{k\}).
 \end{aligned}$$

This means that the certificate can be used in practice by replacing $\text{range}(w^*)$ by its upperbound. See experimental evaluation for a 2D total variation denoising in Appendix A.

Exact solution. If the submodular function only takes integer values and we have an approximate solution of the TV problem with gap $\varepsilon \leq \frac{1}{4n}$, then we have the optimal solution (Chakrabarty et al., 2014).

Relationship with traditional active-set algorithm. Given an ordered partition \mathcal{A} , an active-set method solves the unconstrained optimization problem in Eq. 12 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,

$$\text{Primal feasibility : } w \in \mathcal{W}^{\mathcal{A}}, \tag{18}$$

$$\text{Dual feasibility : } s \in B(F). \tag{19}$$

If Eq. 18 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 $v_i < v_{i+1}$ 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 $F = F_1 + \dots + F_r$, with r “simple” submodular functions (Bach, 2013). 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 Komodakis et al. (2011) and Jegelka et al. (2013), our framework easily extends to $r > 2$.

4.1 Reformulation as the Distance Between Two Polytopes

Following Jegelka et al. (2013), we have the primal/dual problems :

$$\begin{aligned}
 \min_{w \in \mathbb{R}^n} f_1(w) + f_2(w) - u^\top w + \frac{1}{2} \|w\|_2^2 &= \min_{w \in \mathbb{R}^n} \max_{s_1 \in B(F_1), s_2 \in B(F_2)} w^\top (s_1 + s_2) - u^\top w + \frac{1}{2} \|w\|_2^2 \\
 &= \max_{s_1 \in B(F_1), s_2 \in B(F_2)} \min_{w \in \mathbb{R}^n} (s_1 + s_2 - u)^\top w + \frac{1}{2} \|w\|_2^2 \\
 &= \max_{s_1 \in B(F_1), s_2 \in B(F_2)} -\frac{1}{2} \|s_1 + s_2 - u\|_2^2, \tag{20}
 \end{aligned}$$

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 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). We now review Alternating projections (AP) (Jegelka et al., 2013), Averaged alternating reflections (AAR) (Jegelka et al., 2013) and Dykstra’s alternating projections (DAP) (Chambolle and Pock, 2015) to show that a large number of total variation denoising problems need to be solved to obtain an optimal solution of Eq. 20. The ability to warm start and solve these total variation denoising using our algorithm in Section 3.4 can greatly improve the performance of each of these algorithms.

Alternating projections (AP). The alternating projection algorithm (Bauschke et al., 1997) 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. 20. 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 oracles for F_1 and F_2 since projection onto the base polytope is equivalent to performing TV denoising as shown in Eq. 9.

Averaged alternating reflections (AAR). The averaged alternating reflection algorithm (Bauschke et al., 2004), 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 (Jegelka et al., 2013; Kumar et al., 2015) 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

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

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 projection operation, $s_{1,t} = \Pi_{B(F_1)-u/2}(z_t)$; $s_{2,t} = \Pi_{u/2-B(F_2)}(s_{1,t})$ the sequences $s_{1,t}$ and $s_{2,t}$ converge to the nearest points on the polytopes, $B(F_1) - u/2$ and $u/2 - B(F_2)$ (Bauschke et al., 2004) respectively.

Dykstra's alternating projections (DAP). Dykstra's alternating projection algorithm (Bauschke and Borwein, 1994) 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 (Gaffke and Mathar, 1989). 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:

$$\begin{aligned} & \min_{\substack{s \in B(F_1) - \frac{u}{2} \\ s \in \frac{u}{2} - B(F_2)}} \frac{1}{2} \|s\|_2^2 \\ &= \min_{s \in \mathbb{R}^n} \frac{1}{2} \|s\|_2^2 + \iota_{B(F_1) - \frac{u}{2}}(s) + \iota_{\frac{u}{2} - B(F_2)}(s) \\ &= \min_{s \in \mathbb{R}^n} \frac{1}{2} \|s\|_2^2 + \iota_{B(F_1) - \frac{u}{2}}(s) + \iota_{B(F_2) - \frac{u}{2}}(-s) \\ &= \min_{s \in \mathbb{R}^n} \left(\frac{1}{2} \|s\|_2^2 + \max_{w_1 \in \mathbb{R}^n} w_1^\top s - f_1(w_1) + \frac{w_1^\top u}{2} + \max_{w_2 \in \mathbb{R}^n} -w_2^\top s - f_2(w_2) + \frac{w_2^\top u}{2} \right) \\ &= \max_{\substack{w_1 \in \mathbb{R}^n \\ w_2 \in \mathbb{R}^n}} \left(-f_1(w_1) - f_2(w_2) + \frac{(w_1 + w_2)^\top u}{2} + \min_{s \in \mathbb{R}^n} \left(\frac{1}{2} \|s\|_2^2 + (w_1 - w_2)^\top s \right) \right) \\ &= \max_{\substack{w_1 \in \mathbb{R}^n \\ w_2 \in \mathbb{R}^n}} -f_1(w_1) - f_2(w_2) + \frac{(w_1 + w_2)^\top u}{2} - \frac{1}{2} \|w_1 - w_2\|_2^2 \\ &= \min_{\substack{w_1 \in \mathbb{R}^n \\ w_2 \in \mathbb{R}^n}} f_1(w_1) + f_2(w_2) - \frac{(w_1 + w_2)^\top u}{2} + \frac{1}{2} \|w_1 - w_2\|_2^2, \end{aligned}$$

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

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

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

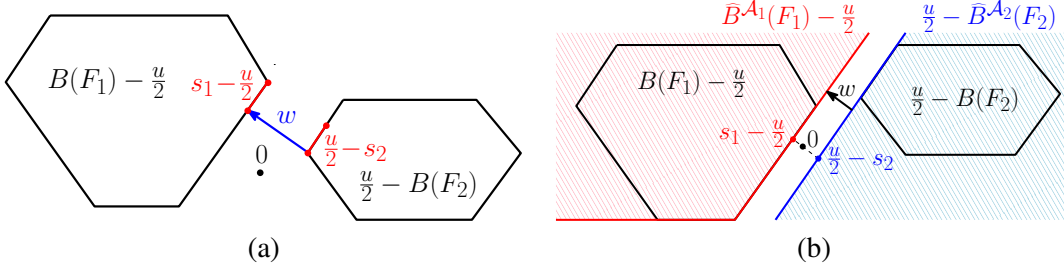


Figure 3: Closest point between two polytopes. (a) Output of Dykstra’s alternating projection algorithm for the TV problem, the pair (s_1, s_2) may not be unique while $w = s_1 + s_2 - u$ is. (b) Dykstra’s alternating projection output for outer approximations.

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 (Bauschke and Borwein, 1994); see an illustration in Figure 3-(a). This insight will be key in our algorithm to avoid cycling.

Assuming TV oracles are available for F_1 and F_2 , Jegelka et al. (2013) and Kumar et al. (2015) use alternating projection (Bauschke et al., 1997) and alternating reflection (Bauschke et al., 2004) algorithms to solve dual optimization problem in Eq. 20. Nishihara et al. (2014) gave a extensive theoretical analysis of alternating projection and showed that it converges linearly. However, these algorithms are equivalent to block *dual* coordinate descent and cannot 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 warm-starts (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. 20, we obtain the primal/dual pairs of optimization problems

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

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 B, \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) could be obtained by using convex feasibility algorithms such as alternating projections (Bauschke et al., 1997) or alternating reflections (Bauschke et al., 2004). 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 (Bauschke and Borwein, 1994). See Figure 3-(b) 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 (Bauschke and Borwein, 1994).

Translated intersecting polytopes. In our situation, we have more to work with than just the ordered partitions: *we also know the vector w* (as mentioned earlier, it is obtained cheaply from isotonic regression). Indeed, from Lemma 2.2 and Theorem 3.8 from Bauschke and Borwein (1994), 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

$$\begin{aligned}
 & \min_{\substack{s \in \widehat{B}^{\mathcal{A}_1}(F_1) - \frac{u-w}{2} \\ s \in \frac{u-w}{2} - \widehat{B}^{\mathcal{A}_2}(F_2)}} \frac{1}{2} \|s\|_2^2 & (21) \\
 &= \min_{s \in \mathbb{R}^n} \frac{1}{2} \|s\|_2^2 + \iota_{\widehat{B}^{\mathcal{A}_1}(F_1) - \frac{u-w}{2}}(s) + \iota_{\frac{u-w}{2} - \widehat{B}^{\mathcal{A}_2}(F_2)}(s), \\
 &= \min_{s \in \mathbb{R}^n} \frac{1}{2} \|s\|_2^2 + \iota_{\widehat{B}^{\mathcal{A}_1}(F_1) - \frac{u-w}{2}}(s) + \iota_{\widehat{B}^{\mathcal{A}_2}(F_2) - \frac{u-w}{2}}(-s), \\
 &= \min_{s \in \mathbb{R}^n} \left(\frac{1}{2} \|s\|_2^2 + \max_{w_1 \in \mathcal{W}^{\mathcal{A}_1}} w_1^\top s - f_1(w_1) + \frac{w_1^\top (u-w)}{2} \right. \\
 & \quad \left. + \max_{w_2 \in \mathcal{W}^{\mathcal{A}_2}} -w_2^\top s - f_2(w_2) + \frac{w_2^\top (u-w)}{2} \right), \\
 &= \max_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \left(-f_1(w_1) - f_2(w_2) + \frac{(w_1 + w_2)^\top (u-w)}{2} + \min_{s \in \mathbb{R}^n} \frac{1}{2} \|s\|_2^2 + (w_1 - w_2)^\top s \right), \\
 &= \max_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \left(-f_1(w_1) - f_2(w_2) + \frac{(w_1 + w_2)^\top (u-w)}{2} - \frac{1}{2} \|w_1 - w_2\|_2^2 \right),
 \end{aligned}$$

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

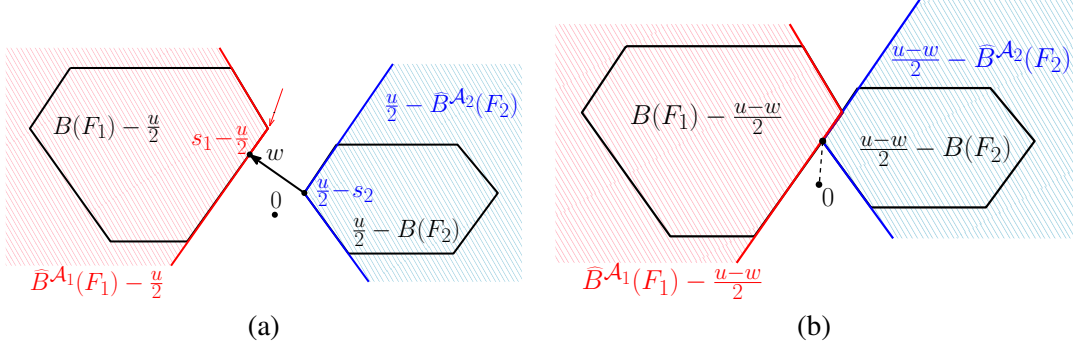


Figure 4: Translated intersecting polytopes. (a) output of our algorithm before translation. (b) Translated formulation.

$$= \min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} \left(f_1(w_1) + f_2(w_2) - \frac{(w_1 + w_2)^\top (u - w)}{2} + \frac{1}{2} \|w_1 - w_2\|_2^2 \right), \quad (22)$$

with $s = w_2 - w_1$ at optimality.

In Section 4.4 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

Data: Submodular functions F_1 and F_2 with SFM oracles, $u \in \mathbb{R}^n$, ordered partitions $\mathcal{A}_1, \mathcal{A}_2$

Result: primal optimal: $w \in \mathbb{R}^n$ and dual optimal: $s_1 \in B(F_1), s_2 \in B(F_2)$

```

1 while True do
2    $\mathcal{A} = \text{coalesce}(\mathcal{A}_1, \mathcal{A}_2)$ ;
3   Estimate  $w$  by solving  $\min_{w \in \mathcal{W}^{\mathcal{A}}} f(w) - u^\top w + \frac{1}{2} \|w\|_2^2$  using isotonic regression;
4   Projection step: Estimate  $s_1 \in \widehat{B}^{\mathcal{A}_1}(F_1)$  and  $s_2 \in \widehat{B}^{\mathcal{A}_2}(F_2)$  by projecting 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;
5   Merge the sets in  $\mathcal{A}_j$  which are tight for  $s_j, j \in \{1, 2\}$ ;
6   Check optimality of  $s_1$  and  $s_2$  as described in Section 3.3;
7   if  $s_1$  and  $s_2$  are optimal then
8     | break;
9   else
10    | for  $j \in \{1, 2\}$  and  $i_j \in \{1, \dots, m_j\}$ , split the set  $A_{j,i_j}$  into  $C_{j,i_j}$  and  $A_{j,i_j} \setminus C_{j,i_j}$  in
    | that order to get an updated ordered partition  $\mathcal{A}_j$ ;
11  end
12 end
    
```

Given two ordered partitions \mathcal{A}_1 and \mathcal{A}_2 , we obtain $s_1 \in \widehat{B}^{\mathcal{A}_1}(F_1)$ and $s_2 \in \widehat{B}^{\mathcal{A}_2}(F_2)$ 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 C, 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 for the ‘‘projection step’’ of the active-set algorithm proposed in Section 4.3 that optimizes the translated Dykstra problem in Eq. 22, i.e.,

$$\min_{\substack{w_1 \in \mathcal{W}^{\mathcal{A}_1} \\ w_2 \in \mathcal{W}^{\mathcal{A}_2}}} f_1(w_1) + f_2(w_2) - \frac{(w_1+w_2)^\top(u-w)}{2} + \frac{1}{2}\|w_1 - w_2\|^2. \quad (23)$$

The corresponding dual optimization problem is given by

$$\min_{\substack{s \in \widehat{B}^{\mathcal{A}_1}(F_1) - \frac{u-w}{2} \\ s \in \frac{u-w}{2} - \widehat{B}^{\mathcal{A}_2}(F_2)}} \frac{1}{2}\|s\|_2^2, \quad (24)$$

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. 22 given by

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

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 (Shusheng, 2000).

In our simulations, we have used the recent accelerated version of Chambolle and Pock (2015), which led to faster convergence. In order to monitor convergence of the algorithm, we compute the value of $\|u - w - s_{1,t} - s_{2,t}\|_1$, which is equal to zero at convergence. Note that the algorithm is not finitely convergent and gives only *approximate* solutions. Therefore, we introduce the approximation parameters ε_1 and ε_2 such that s_1 lies in the ε_1 -neighborhood of the base polytope of F_1 , i.e., $\min_{A \subseteq V} F_1(A) - s_1(A) \geq -\varepsilon_1$ and s_2 lies in the ε_2 -neighborhood of the base polytope of F_2 ,

i.e., $\min_{A \subseteq V} F_2(A) - s_2(A) \geq -\varepsilon_2$, respectively. See Appendix E for more details on the approximation. 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. 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. 22 using the standard active-set method (Nocedal and Wright, 2006) by solving a set of linear equations. For this purpose, we derive the equivalent optimization problems using equality constraints.

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

$$f_j(w_j) = \sum_{i_j=1}^{m_j} v_{j,i_j} \left(F_j(B_{j,i_j}) - F_j(B_{j,i_j-1}) \right) \quad (25)$$

$$w_j = \sum_{i_j=1}^{m_j} v_{j,i_j} \mathbf{1}_{A_{j,i_j}} \quad (26)$$

$$\text{with the constraints, } v_{j,1} \geq \dots \geq v_{j,m_j}. \quad (27)$$

On substituting Eq. 25, Eq. 26 and Eq. 27 in Eq. 22, we have an equivalent optimization problem:

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

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

$$\min_{\substack{x \in \mathbb{R}^{m_1+m_2} \\ D(\mathcal{A}_1, \mathcal{A}_2)x \succeq 0}} \frac{1}{2} x^\top Q(\mathcal{A}_1, \mathcal{A}_2)x + c(\mathcal{A}_1, \mathcal{A}_2)^\top x. \quad (28)$$

Here, $D(\mathcal{A}_1, \mathcal{A}_2)$ 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 computed using the function evaluations of F_1 and F_2 . Note that these evaluations need to be done only once.

Computing $Q(\mathcal{A}_1, \mathcal{A}_2)$. 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 $e(A_{1,i_1}, A_{2,i_2}) = 1_{A_{1,i_1}}^\top 1_{A_{2,i_2}}$ 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 $m_2 = 5$.

Initializing with primal feasible point. Primal active-set methods start with a primal feasible point and continue to maintain primal feasible iterates. In our case, the starting point may be obtained using the weight vector w that is estimated using isotonic regression. The vector w is compatible both with \mathcal{A}_1 and \mathcal{A}_2 , i.e., $w \in \mathcal{W}^{\mathcal{A}_1}$ and $w \in \mathcal{W}^{\mathcal{A}_2}$. Therefore, we may obtain the vectors v_1 and v_2 from w and initialize the primal feasible starting point using v_1 and v_2 .

Optimizing the quadratic program in Eq. 28 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. 28 with equality constraints.

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

$$\min_{\substack{p \in \mathbb{R}^{m_1+m_2} \\ D'p=0}} \frac{1}{2} p^\top Q(\mathcal{A}_1, \mathcal{A}_2) p + (Q(\mathcal{A}_1, \mathcal{A}_2) x_k + c(\mathcal{A}_1, \mathcal{A}_2))^\top p, \quad (29)$$

where D' is the subset of the constraints in $D(\mathcal{A}_1, \mathcal{A}_2)$, i.e., indices of constraints that are tight and x_k is a primal-feasible point. We refer the indices of the tight constraints as the *working set* and represent them by the set W in the algorithm. Therefore, the set of constraints in D' is the restriction of the constraint set $D(\mathcal{A}_1, \mathcal{A}_2)$ to the working set constraints denoted by W . The vector p gives the direction of strict descent of the cost function in Eq. 28 from feasible point x_k (Nocedal and Wright, 2006).

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 $v_{j,k_j} = v_{j,k_j+1}$. Similarly, x'_t can be computed from x_t by merging the weights v_{j,k_j} and v_{j,k_j+1} into a single weight for the merged element of the ordered partition. Finding the optimal vector p' using the quadratic program in Eq. 29 with respect to the ordered partition \mathcal{A}'_j is equivalent to solving the following unconstrained quadratic problem,

$$Q(\mathcal{A}'_1, \mathcal{A}'_2, x'_t) = \min_{p' \in \mathbb{R}^{m'_1+m'_2}} \left(\frac{1}{2} p'^\top Q(\mathcal{A}'_1, \mathcal{A}'_2) p' + (Q(\mathcal{A}'_1, \mathcal{A}'_2) x'_t + c(\mathcal{A}'_1, \mathcal{A}'_2))^\top p' \right), \quad (30)$$

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, $Q(\mathcal{A}'_1, \mathcal{A}'_2)$ (Vishnoi, 2013). We can build p from p' by repeating the values for the elements of the partition that were merged.

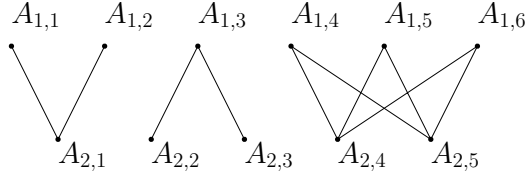


Figure 5: Bipartite graph to compute $Q(\mathcal{A}_1, \mathcal{A}_2)$ with \mathcal{A}_1 having $m_1 = 6$ components and \mathcal{A}_2 having $m_2 = 5$.

Primal active-set algorithm. We now can describe the standard primal active-set method.

Data: Laplacian matrix, $Q(\mathcal{A}_1, \mathcal{A}_2)$ and vector, $c(\mathcal{A}_1, \mathcal{A}_2)$, ordered partitions $\mathcal{A}_1, \mathcal{A}_2$

Result: $x^* \in \mathbb{R}^{m_1+m_2}$

```

1 Initialize:  $t = 0$ ;
2 Primal feasible  $x_0$  using the solution of isotonic regression  $w$ ;
3  $W_0$  with indices of rows of  $D(\mathcal{A}_1, \mathcal{A}_2)$  that are equal to 0;
4 while True do
5     Estimate primal: Solve equality constrained QP in Eq. 29 with equality constraints
        indexed by working set  $W_t$  to find optimal  $p$  ;
6     if  $p == 0$  then
7         Estimate dual:  $\lambda = D(\mathcal{A}_1, \mathcal{A}_2)^{-\top} (Q(\mathcal{A}_1, \mathcal{A}_2)x_t + c(\mathcal{A}_1, \mathcal{A}_2))$ ;
8         if  $\lambda_i \geq 0$  for all  $i \in W_t$  then
9             break ;
10        else
11            update  $j = \operatorname{argmin}_{j \in W_t} \lambda_j$  ;
12            update  $W_{t+1} = W_t \setminus \{j\}$ ;
13            update  $x_{t+1} = x_t$  ;
14        end
15    else
16        Line search: Find least  $\beta$  that retains feasibility of  $x_{t+1} = x_t + \beta p$  and find the
            blocking constraints  $B_t$  ;
17         $W_{t+1} = W_t \cup B_t$  ;
18    end
19     $t = t + 1$ 
20 end
21 return  $x^* = x_t$ 
    
```

We can estimate w_1 and w_2 from x^* , which will enable us to estimate s feasible in Eq. 24. 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. 28 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 level-sets 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 $k = 1, 2, \dots, m$. Let J_k denote the indices of the nodes of G_k in G .

$$x_{J_k}^* = \underset{\substack{x \in \mathbb{R}^{m_{1,k} + m_{2,k}} \\ D(\mathcal{A}_1, \mathcal{A}_2)_{J_k} x \succeq 0}}{\operatorname{argmin}} \frac{1}{2} x^\top Q(\mathcal{A}_1, \mathcal{A}_2)_{J_k} x + c(\mathcal{A}_1, \mathcal{A}_2)_{J_k}^\top x, \quad (31)$$

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 the base polytope of F_j into base polytopes of submodular functions formed by contracting F_j on each individual component representing the connected component k . See Appendix D 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. Here, our experiments mainly focus on the time comparisons with state-of-art methods and also show an important setting where we show the gain due to the ability to warm-start our algorithm. 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. Our algorithm leads to marginal gains for the usual non decomposable functions. However, in the non decomposable case there are many total variation problems to be solved. The ability to warm-start lends to huge improvements when compared to the usage of standard total variation oracles in this setting.

5.1 Non-decomposable Total Variation Denoising

Our experiments consider images, which are 2-dimensional grids with vertex neighborhood of size 4. The data set 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 and Darbon (2009) but our algorithms work 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. 3.

Maxflow (Boykov and Kolmogorov, 2004) 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 for the algorithm of Chambolle and Darbon (2009) 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 Chambolle and Darbon (2009). However, oracle complexity of

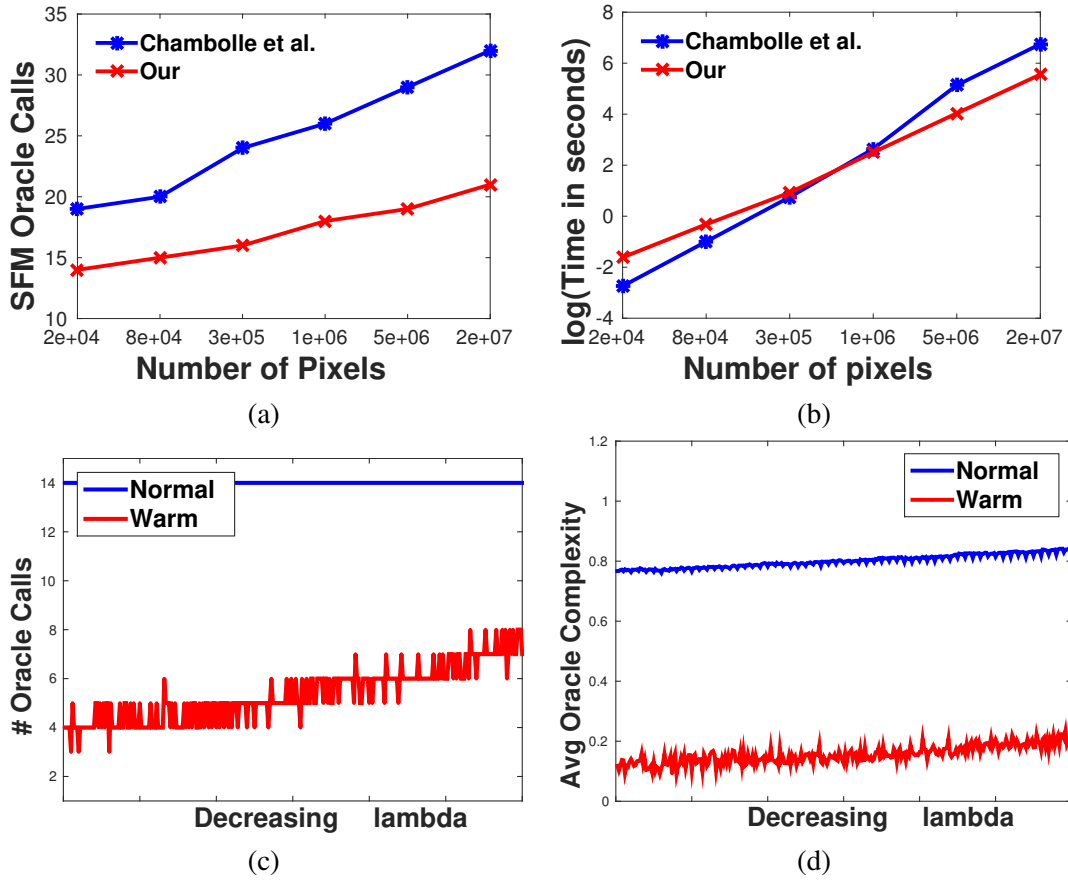


Figure 6: (a) Number of SFM oracle calls for images of various sizes, (b) Time taken for images of various sizes, (c) Number of iterations with and without warm start, (d) Average complexity of the oracle with and without warm start.

each call is higher for the our method when compared to Chambolle and Darbon (2009). 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 (Chambolle and Darbon, 2009) despite having fewer 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 fraction of pixels in the elements of the partitions that need to be queried. 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 the SFM and TV problems on a cut function defined on a 3D-grid. The 3D-grid consists of lines parallel lines in each dimension as shown in Figure 7. It can be decomposed into two functions F_1 and F_2 , where F_1 is composed on parallel 2D-grids and F_2 is composed of parallel chains. From Figure 7, the function F_1 represents all the solid edges (red and blue) whereas the function F_2 represents the dashed edges (magenta). For brevity, we refer to each 2D-grid of the function F_1 as a *frame*. The SFM oracle for the function F_1 is the maxflow-mincut (Boykov and Kolmogorov, 2004) algorithm, which may run in parallel for all frames. Similarly the SFM oracle for the function F_2 is the message passing algorithm, which may run in parallel for all chains. The corresponding TV oracles, i.e., *projection algorithm* for F_1 and F_2 may be solved using the algorithm described in Section 3.4 due to availability of the respective SFM oracles. We consider averaged alternating reflection (AAR) (Bauschke et al., 2004) by solving each projection without warm-start and counting the total number of SFM oracle calls of F_1 to solve the SFM and TV on the 3D-grid as our baseline. (SGD-P) denotes the dual subgradient based method (Komodakis et al., 2011) modified with Polyak’s rule (Poljak, 1987) to solve SFM on the 3D-grid. We show the performance of alternating projection (AP-WS), averaged alternating reflection (AAR-WS) (Bauschke et al., 2004) and Dykstra’s alternating projection (DAP-WS) (Bauschke and Borwein, 1994) using warm start of each projection with the ordered partitions. WS denotes warm start 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 data set of the Stanford bunny (max) of size $102 \times 100 \times 79$. The function F_1 represents 102 frames while F_2 represents the 7900 chains. The dimension of each frame in F_1 is 100×79 , while the length of each chain in F_2 is 102. Figure 8 (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.

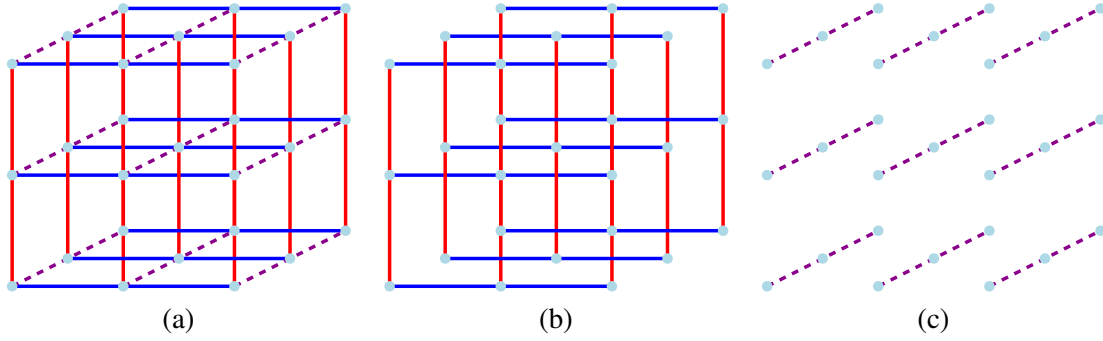


Figure 7: (a) Cut function F defined on a 3D-grid may be decomposed into: (b) F_1 represented by solid edges (red and blue) and (c) F_2 represented by dashed lines (magenta)

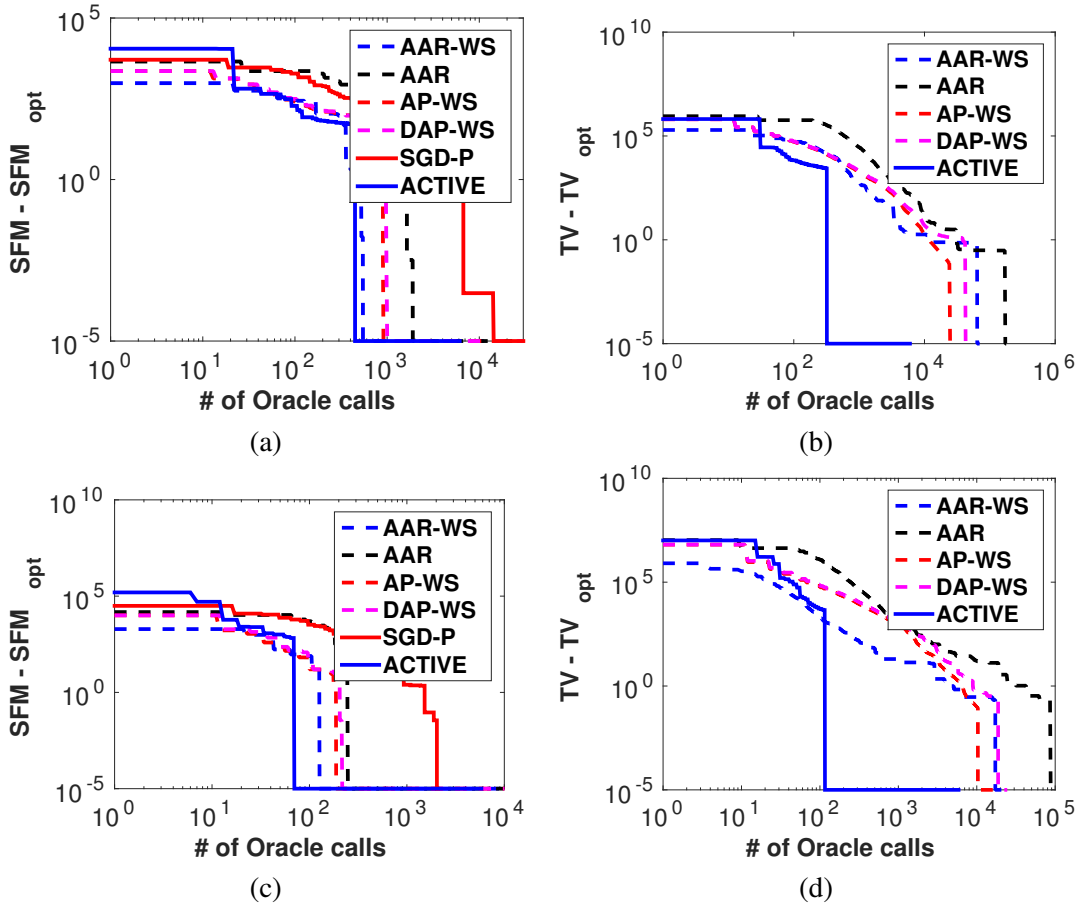


Figure 8: (a) Number of 2D SFM calls to obtain 3D SFM, (b) Number of 2D SFM calls to obtain 3D TV, (c) Number of 2D SFM calls to obtain SFM of 2D + concave function, (d) Number of 2D SFM calls to obtain TV of 2D + concave function.

Time comparisons. We also performed time comparisons between the iterative methods and the combinatorial methods on standard data sets. The standard mincut-maxflow (Boykov and Kolmogorov, 2004) on the 3D volumetric data set of the Standard bunny (max) of size $102 \times 100 \times 79$ takes 0.11 seconds while averaged alternating reflections (AAR) without warm start takes 0.38 seconds. The averaged alternating reflections with warm start (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 by Vishnoi (2013). Note that the projection on the 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 (Kumar et al., 2015). The times for (AAR), (AAR-WS) and (ACTIVE) use parallel multi-core architectures³ 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 contrast, our algorithm works on more general class of sum of submodular functions than just with cut functions.

Concave functions on cardinality. In this experiment we consider our SFM problem of sum of a 2D cut on a graph of size 5616×3744 and a super pixel based concave function on cardinality (Stobbe and Krause, 2010; Jegelka et al., 2013). The 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 8 (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 algorithm for optimizing quadratic losses regularized by Lovász extension 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.

Acknowledgments

We acknowledge support from the European Research Council grant SIERRA (project 239993). K. S. Sesh Kumar also acknowledges the support from the European Research Council grant of Prof. Vladimir Kolmogorov DOiCV(project 616160) at IST Austria. The comments of the reviewers have helped us improve the presentation significantly.

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.

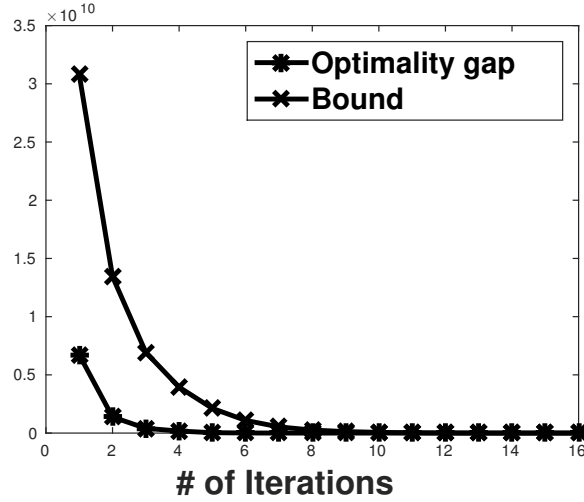


Figure 9: Certified gap vs. bound for TV denoising on a 2D image using the algorithm in Section 3.4.

Appendix A. Certificates of Optimality

We consider a total variation denoising problem on an 2D image of dimensions 384×288 using the algorithm proposed in Section 3.4 for non-decomposable functions, where we assume a 2D SFM oracle. Here, we plot the optimality gap given by $(f(w) - u^\top w + \frac{1}{2}\|w\|_2^2) - (f(w^*) - u^\top w^* + \frac{1}{2}\|w^*\|_2^2)$ and the bound, $\varepsilon \text{range}(w) + \varepsilon \text{range}(w^*)$, proposed in Section 3.6. Here, w is the solution at the end of each iteration of the algorithm and w^* is the optimal solution.

Appendix B. 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 $j = 1, 2$, $\forall i_j = (1, \dots, m_j)$, $B_{j,i_j} = (A_{j,1} \cup \dots \cup A_{j,i_j})$. The inequalities defining the outer approximation of the base polytopes are given by hyperplanes defined by

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

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 $C = \emptyset$.
- **Algorithm:** Iterate until $x = m_1$ and $y = m_2$
 1. If $|B_{1,x}| > |B_{2,y}|$ then $y := y + 1$.
 2. If $|B_{1,x}| < |B_{2,y}|$ then $x := x + 1$.

3. 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 $\mathcal{A} = (A_1, \dots, A_m)$.

Running time. The algorithm terminates in $\min(m_1, m_2)$ iterations and the checking condition for step (3) takes n iterations. Therefore, the algorithm overall takes a time of $O(\min(m_1, m_2)n)$.

Appendix C. Optimality of Algorithm for Decomposable Problems

In step (10) of the algorithms, when we split partitions, the value of the primal/dual pair of optimization algorithms

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

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 (4) 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 $\frac{1}{2}\|s_1 - u/2 + w/2\|_2^2$ 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 $\frac{1}{8}\|s_1 - s_2\|_2^2$. We have:

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

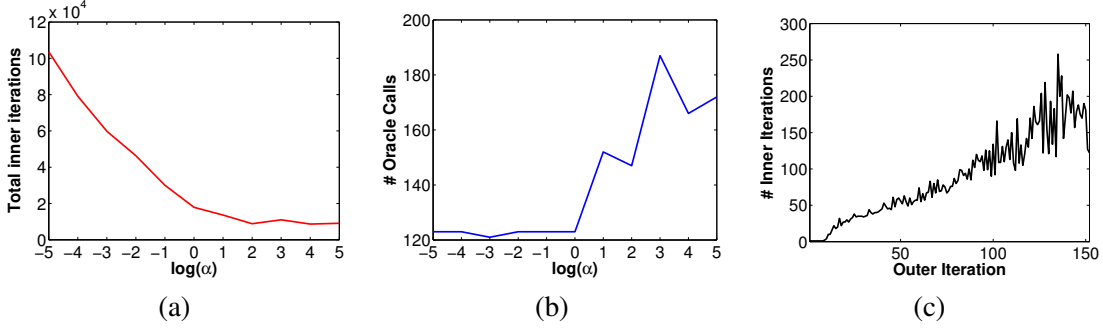


Figure 10: (a) Total number of inner iterations for varying α . (b) Total number of outer iterations for varying α . and (c) Number of inner iterations per each outer iteration for the $\alpha = 10^1$.

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

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 (5) is performing.

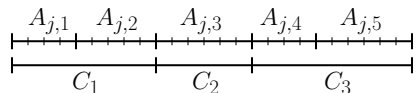
If the check is satisfied, it means that w_1 and w_2 are in fact optimal for the problem above without the restriction in compatibilities, which implies that they are the Dykstra solutions for the TV problem.

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.

Appendix D. 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 out 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 \mathcal{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

$$(s_j)_{C_k} \in \hat{B}^{A_j \cap C_k} [(F_j)_{C_k | C_1 \cup \dots \cup C_{k-1}}],$$

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

Appendix E. Approximate Dykstra Steps

The Dykstra step, i.e., step (4) 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 don't go back to the same w_1 and w_2 , which we have already encountered and avoid cycling. Figure 10 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 10-(a) shows the total number of inner iterations required to solve the TV problem. Figure 10-(b) gives the total number of SFM oracle calls required to solve the TV problem. In Figure 10-(c), we show the number of inner iterations in every outer iteration for the best α we have encountered.

References

- Maxflow dataset online. <http://vision.csd.uwo.ca/maxflow-data>.
- F. Bach. *Learning with Submodular Functions: A Convex Optimization Perspective*, volume 6 of *Foundations and Trends in Machine Learning*. NOW, 2013.
- H. H. Bauschke and J. M. Borwein. Dykstra's alternating projection algorithm for two sets. *Journal of Approximation Theory*, 79(3):418–443, 1994.
- 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.
- 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.
- M. J. Best and N. Chakravarti. Active set algorithms for isotonic regression: a unifying framework. *Mathematical Programming*, 47(1):425–439, 1990.

- 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.
- 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.
- D. Chakrabarty, P. Jain, and P. Kothari. Provable submodular minimization using Wolfe’s algorithm. In *Advances in Neural Information Processing Systems*. 2014.
- 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.
- 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.
- S. Fujishige. Lexicographically optimal base of a polymatroid with respect to a weight vector. *Mathematics of Operations Research*, 5(2):186–196, 1980.
- S. Fujishige. Submodular systems and related topics. In *Mathematical Programming at Oberwolfach II*. 1984.
- S. Fujishige. *Submodular Functions and Optimization*. Elsevier, 2005.
- N. Gaffke and R. Mathar. A cyclic projection algorithm via duality. *Metrika*, 36(1):29–54, 1989.
- D. Goldfarb and W. Yin. Parametric maximum flow algorithms for fast total variation minimization. *SIAM Journal on Scientific Computing*, 31(5):3712–3743, 2009.
- 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.
- S. Jegelka, F. Bach, and S. Sra. Reflection methods for user-friendly submodular optimization. In *Advances in Neural Information Processing Systems*, 2013.
- V. Kolmogorov. Minimizing a sum of submodular functions. *Discrete Applied Mathematics*, 160(15), 2012.
- 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.
- A. Krause and C. Guestrin. Submodularity and its applications in optimized information gathering. *ACM Transactions on Intelligent Systems and Technology*, 2(4), 2011.
- K. S. Sesh Kumar, A. Barbero, S. Jegelka, S. Sra, and F. Bach. Convex optimization for parallel energy minimization. Technical Report 01123492, HAL, 2015.
- 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.

- L. Landrieu and G. Obozinski. Cut Pursuit: fast algorithms to learn piecewise constant functions. In *Proceedings of Artificial Intelligence and Statistics*, 2016.
- H. Lin and J. Bilmes. A class of submodular functions for document summarization. In *Proceedings of NAACL/HLT*, 2011.
- L. Lovász. Submodular functions and convexity. *Mathematical programming: the state of the art, Bonn*, pages 235–257, 1982.
- R. Nishihara, S. Jegelka, and M. I. Jordan. On the convergence rate of decomposable submodular function minimization. In *Advances in Neural Information Processing Systems 27*, pages 640–648, 2014.
- J. Nocedal and S. J. Wright. *Numerical optimization*. Springer Series in Operations Research and Financial Engineering. Springer, Berlin, 2006.
- B. T. Poljak. *Introduction to optimization*. Optimization Software, 1987.
- R. T. Rockafellar. *Convex Analysis*. Princeton U. P., 1997.
- 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.
- P. Stobbe and A. Krause. Efficient minimization of decomposable submodular functions. In *Advances in Neural Information Processing Systems*, 2010.
- 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.
- N.K. Vishnoi. *$Lx = B$ - Laplacian Solvers and Their Algorithmic Applications*. Now Publishers, 2013.