DC-Programming versus $\ell_0$-Superiorization for Discrete Tomography

Aviv Gibali and Stefania Petra

Abstract

In this paper we focus on the reconstruction of sparse solutions to underdetermined systems of linear equations with variable bounds. The problem is motivated by sparse and gradient-sparse reconstruction in binary and discrete tomography from limited data. To address the $\ell_0$-minimization problem we consider two approaches: DC-programming and $\ell_0$-superiorization. We show that $\ell_0$-minimization over bounded polyhedra can be equivalently formulated as a DC program. Unfortunately, standard DC algorithms based on convex programming often get trapped in local minima. On the other hand, $\ell_0$-superiorization yields comparable results at significantly lower costs.

1 Introduction

We consider the $\ell_0$-minimization problem

$$\min \|Dx\|_0 \quad \text{subject to} \quad x \in \mathcal{P} \subset \mathbb{R}^n,$$

over a polyhedron $\mathcal{P}$. Matrix $D \in \{0, 1\}^{n\times n}$ is a diagonal matrix that selects entries in $x$, $\|x\|_0 := \|\text{supp}(x)\| := |\{i \in \{1, 2, \ldots, n\} : x_i \neq 0\}|$ counts the nonzeros entries and is typically coined $\ell_0$-norm. $\|\cdot\|_0$ is not a norm and problem (1.1) is known to be NP-hard. See e.g. [Nat95] for the case of affine linear constraints.
\[ \mathcal{P} = \{ x \in \mathbb{R}^n : Ax = b \} \] with \( A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m \). In this work we focus on polyhedra of the form
\[ \mathcal{P} = \{ x \in \mathbb{R}^n : Ax = b, Cx \geq d \}, \] described by linear equalities and inequalities, with \( A \in \mathbb{R}^{m \times n}, C \in \mathbb{R}^{k \times n}, b \in \mathbb{R}^m, d \in \mathbb{R}^k \).

### 1.1 Motivation

Our work is motivated by the image reconstruction problem from few linear measurements, e.g. discrete tomography [HK99], where typically there are known bounds on the image values. Moreover, we aim to reconstruct an image that exhibits some structure, e.g. sparsity/gradient-sparsity.

Recovering a structured signal/image from few linear measurements is a central point in both compressed sensing (CS) [FR13] and discrete tomography [HK99]. In CS the signal structure is described by means of a low complexity model. The most basic model is signal sparsity and the associated \( \ell_0 \)-minimization problem writes
\[
\min \| x \|_0 \quad \text{subject to} \quad Ax = b. \tag{1.3}
\]
If \( x \) is sparse, i.e. \( x \in \Sigma_s := \{ x : \| x \|_0 \leq s \} \) and the measurement matrix \( A \in \mathbb{R}^{m \times n} \) satisfies certain conditions, like e.g. being well-conditioned when restricted to \( \Sigma_s \), see [FR13], one can consider instead of (1.3) the easier \( \ell_1 \)-minimization problem
\[
\min \| x \|_1 \quad \text{subject to} \quad Ax = b. \tag{1.4}
\]
The theory of CS implies that if \( m \geq C s \log(n/s) \) and the entries of \( A \) follow a normal distribution, i.e. \( a_{ij} \sim \mathcal{N}(0, 1) \), the problem (1.3) can be solved in polynomial time via (1.4). If one could solve (1.3) directly, then \( m \geq 2s \) measurements would suffice to recover any \( s \)-sparse solution. Interestingly, if one approximates the \( \ell_0 \)-norm by an \( \ell_p \)-quasi-norm, where \( 0 < p < 1 \), then one can recover a \( s \)-sparse vector via \( \ell_p \)-minimization
\[
\min \| x \|_p \quad \text{subject to} \quad Ax = b, \tag{1.5}
\]
as soon as \( m \geq C_1(p) \cdot s + C_2(p) \cdot s \cdot \log(n/s) \), see [CS08]. Hence, although non-convex metrics are generally more challenging to minimize, they have the advantage to enable reconstructing a sparse signal from substantially fewer measurements than the convex \( \ell_1 \)-minimization counterpart. This has been (empirically) observed [YLHX15] also for other \( \ell_0 \)-norm substitutes like e.g.
\[
\min \| x \|_1 - \| x \|_2 \quad \text{subject to} \quad Ax = b. \tag{1.6}
\]
1.2 Objectives

Solution approaches for (1.3) are typically approximated by related concave programs like (1.5) or (1.6), which are usually solved by DC-programming [THPD05], see [Man96, YLHX15]. In DC-programming, see Sect. 3, one solves a sequence of easier (convex) problems of the same type. Similarly, in a superiorization [Cen15, CZ13] framework, see Sect. 4, one successively applies a basic algorithm, e.g. feasibility seeking, that is bounded perturbation resilient to find a “better” feasible solution in terms of a cost function. Our objective is to apply both frameworks to problems (1.1), (1.3), or a related problem like e.g. (1.5) for some \( p \in (0, 1) \). In particular are interested in:

- How and why the two approaches differ?
- If superiorization is applicable to sparse recovery and if yes, how?

2 Preliminaries

2.1 Basic Definitions and Notation

The extended real line is denoted by \( \mathbb{R} := \mathbb{R} \cup \{ \pm \infty \} \). For \( n \in \mathbb{N} \), we use the shorthands \([n] = \{1, 2, \ldots, n\}\) and \([n]_0 = \{0, 1, 2, \ldots, n\}\). Vectors \( x \in \mathbb{R}^n \) are column vectors and indexed by superscripts. \( x^\top \) denotes the transposed vector \( x \) and \( \langle x, y \rangle \) or \( x^\top y \) the Euclidean inner product. \( \top \) stands for the transpose. To save space, however, we will sometimes simply write e.g. \( z = (x, y) \) instead of correctly denoting \( z = ((x)^\top, (y)^\top)^\top \), for \( z = (\vec{x}, y) \). \( \mathbb{1} = (1, 1, \ldots, 1)^\top \) denotes the all one-vector whose dimension will always be clear from the context. The dimension of a vector \( x \) we denote by \( \dim(x) \). The (quasi)-norms \( \|x\|_p = \left( \sum_{i \in [n]} |x_i|^p \right)^{\frac{1}{p}} \), \( p > 0 \) will be called \( \ell_p \)-(quasi-)norms. \( \|x\|_\infty := \max_{i \in [n]} |x_i| \) is the maximum norm. The \( \ell_0 \)-measure (not a norm!) stands for the cardinality of the support of \( x \), i.e. \( \|x\|_0 := |\text{supp}(x)| \), with \( \text{supp}(x) = \{ i \in [n] : x_i \neq 0 \} \). The sign vector \( \text{sign}(x) \in \mathbb{R}^n \) is defined component-wise as

\[
(\text{sign}(x))_i = \begin{cases} 
1, & x_i > 0, \\
0, & x_i = 0, \\
-1, & x_i < 0. 
\end{cases} \tag{2.1}
\]

Note that \( \|x\|_0 = \sum_{i \in [n]} \text{sign}(|x_i|) \). The Euclidean unit ball is denoted by \( B(0) \), while we write \( B_\infty(0) := \{ x : \|x\|_\infty \leq 1 \} \) for the unit ball w.r.t. the max-norm.

For some matrix \( A \in \mathbb{R}^{m \times n} \), the nullspace is denoted by \( \text{N}(A) \) and its range by \( \text{R}(A) \).
We consider images $u(x), x \in \Omega \subset \mathbb{R}^2$ discretized as follows. $\Omega$ is assumed to be a rectangle covered by a regular grid graph $G = (V, E)$ of size $|V| = n$. Accordingly, we identify $V = \prod_{i \in [2]} [n_i]_0 \subset \mathbb{Z}^2, n_i \in \mathbb{N}$. Thus, vertices $v \in V$ are indexed by $(i, j)^T \in \mathbb{Z}^2$ with ranges $i \in [n_1]_0, j \in [n_2]_0$, and

$$n = n_1n_2.$$  

(2.2)

As a result, discretization of $u(x), x \in \Omega$, yields the vector $u \in \mathbb{R}^n$, where we keep the symbol $u$ for simplicity.

Consider the one-dimensional discrete derivative operator

$$\partial : \mathbb{R}^m \to \mathbb{R}^{m-1}, \quad \partial_{i,j} = \begin{cases} -1, & i = j, \\ +1, & j = i + 1, \\ 0, & \text{otherwise}. \end{cases}$$  

(2.3)

Forming corresponding operators $\partial_1, \partial_2$ for each coordinate, conforming to the ranges of $i, j$ such that $(i, j) \in V$, we obtain the discrete gradient operator

$$\nabla = \left( \partial_1 \otimes I_2 \right) \left( I_1 \otimes \partial_2 \right) \in \mathbb{R}^{p \times n},$$  

(2.4)

where $\otimes$ denotes the Kronecker product and $I_i, i = 1, 2$, are identity matrices with appropriate dimensions. The *anisotropic* discretized total variation (TV) is given by

$$\text{TV}(u) := \|\nabla u\|_1.$$  

(2.5)

The image gradient sparsity is given by $\|\nabla u\|_0$.

For simplicity we will consider as basic feasible set the polyhedron $\mathcal{P} = \{ x \in \mathbb{R}^n : Ax = b, Cx \geq d \}$, with $A \in \mathbb{R}^{m \times n}, C \in \mathbb{R}^{k \times n}, b \in \mathbb{R}^m, d \in \mathbb{R}^k$ assumed to be non-empty. This is a particular closed and convex set, which for the applications we have in mind can also be assumed to be bounded.

We denote the class of *proper convex and lower semicontinuous* (lsc) functions by

$$\mathcal{F}_0(\mathbb{R}^n) := \{ f : \mathbb{R}^n \to \overline{\mathbb{R}} : f \text{ is proper, convex and lsc } \}. $$  

(2.6)

Hence, the *indicator function* $\delta_\mathcal{P} : \mathbb{R}^n \to \overline{\mathbb{R}}$ of $\mathcal{P}$ defined as

$$\delta_\mathcal{P}(x) = \begin{cases} \infty, & x \in \mathcal{P}, \\ 0, & x \notin \mathcal{P}, \end{cases}$$

is lsc since $\mathcal{P}$ is closed. The *epigraph* of $f : \mathbb{R}^n \to \mathbb{R}$ is the set of points lying on or above its graph, that is

$$\text{epi}(f) := \{(x, \alpha) \in \mathbb{R}^n \times \mathbb{R} : f(x) \leq \alpha \} \subseteq \mathbb{R}^{n+1}.$$  

(2.7)
For a function \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) the function \( f^* : \mathbb{R}^n \rightarrow \overline{\mathbb{R}} \) defined by
\[
f^*(p) := \sup_{x \in \mathbb{R}^n} \{ \langle p, x \rangle - f(x) \}
\]
is called the (Fenchel) conjugate of \( f \). The conjugate \( f^* \) is the pointwise supremum of the affine functions \( p \mapsto \langle p, x \rangle - \alpha \) parametrized by \((x, \alpha) \in \text{epi}(f)\), which implies that \( f^* \) is lsc convex.

A mapping \( F : \mathbb{R}^n \rightarrow \mathbb{R}^n \) is called \( L \)-Lipschitz continuous for some \( L \geq 0 \) if
\[
\| F(x) - F(y) \| \leq L \| x - y \|, \text{ for all } x, y \in \mathbb{R}^n.
\]

The unit sphere in \( \mathbb{R}^n \) is defined as
\[
S^{n-1} = \{ g \in \mathbb{R}^n \mid \| g \| = 1 \}. \quad (2.10)
\]
The set of vertices of the unit hypercube in \( \mathbb{R}^n \) is denoted by \( G = \{-1, 1\}^n \).

For any \( g \in S^{n-1} \) we define \( |g_i| = \max \{ |g_k| : k \in [n] \} \). The set of univariate positive infinitesimal functions is
\[
P = \{ z(\lambda) \in \mathbb{R} : z(\lambda) > 0, \lambda > 0, \lambda^{-1}z(\lambda) \rightarrow_{\lambda \rightarrow 0} 0 \}. \quad (2.11)
\]

Given a set of points \( S = \{x_i\}_{i=1}^N \subset \mathbb{R}^n \), the convex hull of \( S \), denoted by \( \text{conv}(S) \) is the set of all convex combinations of its points, i.e.
\[
\text{conv}(S) = \left\{ \sum_{i \in [N]} \alpha_i x_i : \alpha_i \geq 0, \sum_{i \in [N]} \alpha_i = 1, x_i \in S \right\}. \quad (2.12)
\]

Let us recall the orthogonal (metric) projection operator \( P_C : \mathbb{R}^n \rightarrow C \) onto a non-empty, closed and convex set \( C \subseteq \mathbb{R}^n \). For each point \( x \in \mathbb{R}^n \), there exists a unique nearest point, denoted by \( P_C(x) \). That is,
\[
\| x - P_C(x) \| \leq \| x - y \| \text{ for all } y \in C. \quad (2.13)
\]
The metric projection \( P_C \) is characterized [GR84, Section 3] by the following two properties:
\[
P_C(x) \in C \quad (2.14)
\]
and
\[
\langle x - P_C(x), P_C(x) - y \rangle \geq 0 \text{ for all } x \in \mathbb{R}^n, y \in C, \quad (2.15)
\]
and if \( C \) is a hyperplane, then (2.15) becomes an equality.

The subdifferential set denoted by \( \partial f(x) \) of \( f : \mathbb{R}^n \rightarrow \overline{\mathbb{R}} \) at a point \( x \in \mathbb{R}^n \) where \( f(x) \in \mathbb{R} \), is defined as
\[
\partial f(x) = \{ \xi \in \mathbb{R}^n : f(z) \geq f(x) + \langle \xi, z - x \rangle, \forall z \in \mathbb{R}^n \}. \quad (2.16)
\]
We agree that \( \partial f(x) = \emptyset \) if \( f(x) \) is not finite. Elements of the subdifferential set are called subgradients.
2.2 Equivalence of $\ell_0$- and $\ell_p$-Minimization over Bounded Polyhedra

We consider here a variation of problem (1.1)

$$\min \|Dx\|_0 \quad \text{subject to} \quad x \in \mathcal{P} \cap B_\infty(0)$$

(2.17)

along with the associated $\ell_p$-minimization problem

$$\min \|Dx\|_p \quad \text{subject to} \quad x \in \mathcal{P} \cap B_\infty(0).$$

(2.18)

Since we intersect $\mathcal{P} = \{x \in \mathbb{R}^n : Ax = b, Cx \geq d\}$ with the $B_\infty(0)$ ball our feasible set becomes a bounded polyhedron, hence a polytope.

Along the lines of [FM11] we establish the equivalence of (2.17) and (2.18) for a sufficiently small $p > 0$. As in [FM11] we rewrite the objective of (2.17) as a step function and reformulate (2.18) as a concave optimization problem. Using that $\|Dx\|_0 = \sum_{i \in [n]} D_{ii} \text{sign}(|x_i|)$ and introducing an auxiliary variable $y_i$ for $|x_i|$ we rewrite (2.17) as

$$\min (D1)^\top \text{sign}(y) \quad \text{subject to} \quad x \in \mathcal{P}, -y \leq x \leq y, -1 \leq x \leq 1.$$  

(2.19)

Similarly we can rewrite (2.18) as

$$\min (D1)^\top y^p \quad \text{subject to} \quad x \in \mathcal{P}, -y \leq x \leq y, -1 \leq x \leq 1, \quad \text{where the } p \text{-power should be understood component-wise, i.e. } y^p := (y_1^p, \ldots, y_n^p) \in \mathbb{R}^n.$$  

We define now a subset of the bounded feasible region of the two above problems as follows

$$\mathcal{F} := \{(x, y) \in \mathbb{R}^{2n} : x \in \mathcal{P}, -y \leq x \leq y, -1 \leq x \leq 1, 0 \leq y \leq 1\}.$$  

(2.21)

We can consider further w.l.o.g. $\mathcal{F}$ as the feasible set of both minimization problems (2.19) and (2.20), since in view of the inequalities $-y \leq x \leq y, \|x\|_\infty \leq 1$ the inequalities $0 \leq t \leq 1$ do not influence the solutions $x$.

Observe now that problem

$$\min (D1)^\top y^p \quad \text{subject to} \quad (x, y) \in \mathcal{F}$$

is a concave problem with a polytope as feasible set. Hence it’s infimum is attained at a vertex of $\mathcal{F}$.

**Lemma 2.1.** [Roc70, Corollary 32.3.3 and 32.3.4] Let $f : \mathbb{R}^n \to \mathbb{R}$ be a concave function, and let $C \subset \mathbb{R}^n$ be a nonempty polyhedral convex set contained in $\text{dom } f := \{x : f(x) < \infty\}$. Suppose that $C$ contains no lines, and that $f$ is bounded below on $C$. Then the infimum of $f$ relative to $C$ is attained at one of the (finitely many) extreme points of $C$. 


Theorem 2.2. The $\ell_0$-norm minimization problem (2.17) is equivalent to the $\ell_p$-norm minimization problem (2.18) for some $p_0 \leq 1$. Furthermore, there exists a vertex of $\mathcal{F}$ that is an exact solution of the $\ell_0$-norm minimization problem (2.17), or equivalently of (2.19), and is a global solution of the $\ell_p$-norm minimization problem (2.18), or equivalently of the concave problem (2.20), for some $p_0 = 1/q_0$ where $q_0 \in \mathbb{N}$.

Proof. Note that the objective function of (2.20) is concave for $y \geq 0$, $p := 1/q < 1$, and that

$$0 \leq (D\|)^T y^p = (D\|)^T y^{1/q} \leq (D\|)^T \text{sign}(y), \quad \text{for } 0 \leq y \leq 1. \quad (2.22)$$

Since $(D\|)^T y^{1/q} \geq 0$ holds on the bounded polyhedral set $\mathcal{F}$, it follows by Lem. 2.1 that $y \mapsto (D\|)^T y^{1/q}$ has a vertex $(x(q), y(q))$ of $\mathcal{F}$ as a solution for each $q \in \mathbb{N}$. Since $\mathcal{F}$ has a finite number of vertices, one vertex, say $(\tilde{x}, \tilde{y})$, will repeatedly solve (2.20) for some increasing infinite sequence in a subset $Q := \{q_0, q_1, q_2, \ldots \} \subset \mathbb{N}$. Hence for $q_i \in Q$

$$(D\|)^T \tilde{y}^{\frac{1}{q_i}} = \min_{(x,y) \in \mathcal{F}} (D\|)^T y^{\frac{1}{q_i}} \leq \inf_{(x,y) \in \mathcal{F}} (D\|)^T \text{sign}(y), \quad (2.23)$$

where the last inequality above follows from (2.22). Letting $i \to \infty$ it follows from (2.23) that

$$(D\|)^T \text{sign}(\tilde{y}) = \lim_{i \to \infty} (D\|)^T \tilde{y}^{\frac{1}{q_i}} \leq \inf_{(x,y) \in \mathcal{F}} (D\|)^T \text{sign}(y).$$

Since $(\tilde{x}, \tilde{y}) \in \mathcal{F}$, it follows that $(\tilde{x}, \tilde{y})$ solves (2.19). Furthermore $(\tilde{x}, \tilde{y})$ is a vertex of $\mathcal{F}$. \hfill $\square$

From the proof of Thm. 2.2, in particular (2.22), we also obtain the following result. This will be further used in order to design a descent direction for the superiorization algorithm in Section 4.

Proposition 2.3. Let $p > 0$, not necessary $p < 1$ and $0 \leq y \leq 1$. Then

$$\phi_p(y) := (D\|)^T y^p \leq \|Dy\|_0 =: \phi_0(y). \quad (2.24)$$

Moreover, a nonascent direction for $\phi_p$ is a nonascent direction for $\phi_0$.

Further we note that $\phi_p$ above is differentiable and a nonascent direction for $\phi_p$ and $\phi_0$ is the gradient of $\phi_p = pDy^{p-1}$.

Remark 2.1. Alternatives to the $\ell_p$-quasinorms exists throughout the literature, such as,

$$|x|/(|x| + \alpha), \quad \text{or} \quad 1 - \exp(-|x|/\alpha),$$

for some small $\alpha > 0$. These functions are particular instances of $f_\alpha : \mathbb{R} \to \mathbb{R}$, of the form $f_\alpha(x) = r_\alpha(|x|)$, where $r_\alpha(0) = 0$, $r_\alpha$ is increasing and concave on $\mathbb{R}_+$, $r'_\alpha$ and $t \mapsto r_\alpha(t)/t$ is non-increasing. Then $\sum_{i=1}^n f_\alpha(x_i)$ can be used as a proxy for the $\ell_p$-quasinorm and for $\ell_0$ for an appropriate choice of parameter $\alpha > 0$. 

2.3 Sparse Regularization for Binary Tomography

In this section we recast the sparse binary tomography problem as (2.17). We consider the problem of reconstructing a vectorized binary image $u \in \{0,1\}^n$ from a limited number of tomographic projections

$$Mu = q.$$ (2.25)

Each pixel is associated with some unknown binary variable $u_i \in \{0,1\}$. Each entry in $q \in \mathbb{R}^m$, called tomographic measurement or single projection, corresponds to the integrated gray values of $u$ along the single ray, see Fig. 2.1. Hence the integral can be split into the sum of products $M_{ij}u_j$, where each matrix entry $M_{ij} \geq 0$ corresponds to the length of the intersection of the $i$-th ray with the $j$-th pixel. If ray $i$ and pixel $j$ do not intersect then $M_{ij} = 0$. Stacking all equations for all the rays together leads to the linear equations in (2.25) and provides the following representation of the reconstruction problem

$$Mu = q, \quad u \in \{0,1\}^n.$$ (2.26)

The binary reconstruction problem (2.26) is NP-complete for more than two projections. Our approach here is to relax the constraints and to move the combinatorial difficulty into the objective. This leads us to

$$\min \|u\|_0 \quad \text{subject to} \quad Mu = q, \quad u \in [0,1]^n.$$ (2.27)
Consequently, we obtained a convex feasible set of the form (1.2), with $A := M$, $b := q$, $C := I \in \mathbb{R}^{n \times n}$, $d := 0 \in \mathbb{R}^n$ and $m$ equals the number of rays. Moreover $\|u\|_\infty \leq 1$ and our polyhedron 

$$\mathcal{P} = \{u: Mu = q, u \in [0, 1]^n\}$$

is bounded.

**Remark 2.2.** Note that problem (2.27) is equivalent to

$$\min \|u\|_p^p \quad \text{subject to} \quad Mu = q, \ u \in [0, 1]^n$$

for a small but finite $p < 1$. In view of the non-negativity of $u$ it is not necessary to introduce auxiliary variables for $|u|$ as done in the previous section.

### 2.4 Gradient-Sparse Regularization for Discrete Tomography

Our objective in this section is to introduce a second regularization model from discrete tomography and to recast it as (2.17). Like previously, we wish to reconstruct a vectorized image $u \in \mathbb{R}^n$ from a limited number of tomographic projections (2.25). We now have a small range of signal values in $u$ due to few materials only (e.g. bone, fat, water etc.). In particular, we have an upper bound on the maximal absolute value of $u$,

$$u_i \in [0, u_b], \forall i \in [n].$$

Moreover, we assume that the image to be reconstructed is gradient-sparse, i.e. $\|\nabla u\|_0 \ll n$. This leads us to the following reconstruction problem

$$\min \|\nabla u\|_0 \quad \text{subject to} \quad Mu = q, \ u \geq 0, \|u\|_\infty \leq u_b. \quad (2.28)$$

Rescaling, with $\tilde{u} := u/u_b$ and $\tilde{M} := u_b M$, leads to

$$\min \|\nabla \tilde{u}\|_0 \quad \text{subject to} \quad \tilde{M} \tilde{u} = q, \ \tilde{u} \geq 0, \|\tilde{u}\|_\infty \leq 1. \quad (2.29)$$

We further simplify notation by setting $u \leftarrow \tilde{u}$. We note that (2.29) is equivalent to

$$\min_{u,z} \|z\|_0 \quad \text{subject to} \quad Mu = q, \ \nabla u = z, \ u \in [0, 1]^n, \quad (2.30)$$

that can be rewritten as

$$\min_{u,z} \|z\|_0 \quad \text{subject to} \quad \begin{pmatrix} M & 0 \\ \nabla & -I \end{pmatrix} \begin{pmatrix} u \\ z \end{pmatrix} = \begin{pmatrix} q \\ 0 \end{pmatrix}, \ u \in [0, 1]^n. \quad (2.31)$$

Using that $\|z\|_\infty = \|\nabla u\|_\infty \leq 1$, in view of the definition of $z$ and the given bounds on $u$, we can recast (2.31) as

$$\min_x \|Dx\|_0 \quad \text{subject to} \quad Ax = b, Cx \geq d, \|x\|_\infty \leq 1, \quad (2.32)$$
where
\[
A := \begin{pmatrix} M & 0 \\ \nabla & -I \end{pmatrix}, \quad b := \begin{pmatrix} q \\ 0 \end{pmatrix}, \quad x := \begin{pmatrix} u \\ z \end{pmatrix}, \quad D := \begin{pmatrix} 0 & 0 \\ I_{\dim(z)} & 0 \end{pmatrix}
\]
and
\[
C := \begin{pmatrix} I_{\dim(u)} & 0 \\ 0 & 0 \end{pmatrix}, \quad d := \begin{pmatrix} 0 \\ 0 \end{pmatrix}.
\]

3 DC-Programming

We have seen in Section 2.2 that one can replace \(\ell_0\)-minimization problem over bounded polyhedra with an concave minimization problem. A concave program is a special instance of a DC-program. A DC-program is a problem of the form
\[
\min_{x \in \mathbb{R}^n} f(x), \quad f(x) = g(x) - h(x),
\]
with \(g, h \in \mathcal{F}_0(\mathbb{R}^n)\) proper, convex, lsc. We emphasize that problem (3.1) is non-convex, since \(f\) is decomposed into a convex part \(g\) and into a concave part \(h\) and in general non-convex. Such an \(f\) will be called DC function.

As discussed earlier, the conditions for global optimality in DC-programs do not yield efficient general algorithms.

3.1 DC-Algorithm

There are some popular techniques for the global optimization of (3.1) - among them, branch-and-bound and cutting planes algorithms. Here we omit discussion of global optimization techniques for DC-programming, refer to the overview [HT99] and focus instead on an convex-based approach to local optimization. The following basic algorithm [PDEB86, PDTH97] computes a non-increasing sequence \(\{f(x^{(k)})\}\) converging to a stationary point by iteratively minimizing \(g\) plus an affine upper bound of \(h\). It is known as the simplified DC Algorithm (DCA).

Given \(x := x^{(k)}\), the first step of each iteration computes an upper bound of \(f\) by determining the closest affine upper bound of the concave component \(h\). The second
Algorithm 1: DC Algorithm (DCA)

**Input:** Good initialization $x$.

**Output:** Approximate local minimizer $\hat{x}$ of problem (3.1).

begin

repeat

\[ y^{(k)} \in \partial h(x^{(k)}) \]
\[ x^{(k+1)} \in \arg\min_x \{ g(x) - \left( h(x^{(k)}) + \langle y^{(k)}, x - x^{(k)} \rangle \right) \} = \arg\min_x \{ g(x) - \langle y^{(k)}, x \rangle \} \]

until some convergence criteria is met at $x^{(K)}$.

\[ \hat{x} \leftarrow x^{(K)} \]

end

step updates $x$ by minimizing this upper bound.

\[ h(x) + h^*(y) \geq \langle x, y \rangle \Leftrightarrow -h(x) \leq -\langle x, y \rangle + h^*(y), \quad \forall y \]

As already mentioned, the sequence of iterates is non-increasing. Denote $x := x^{(k)}$, $y := y^{(k)}$, $x' := x^{(k+1)}$ and

\[
\begin{pmatrix}
y \\
x'
\end{pmatrix} = \begin{pmatrix}
\partial h(x) \\
\partial g^*(y)
\end{pmatrix}.
\]
Then

\[ \inf_x f(x) \leq f(x') \leq g(x') - h(x') \leq g(x') - h(x) \tag{3.4a} \]

since \( y \in \partial h(x) \)

\[ = g(x') - \langle y, x' \rangle + \langle y, x \rangle - h(x) \tag{3.4b} \]

use \( x' = \text{argmin}_{\tilde{x}} g(\tilde{x}) - \langle y, \tilde{x} \rangle \)

\[ \leq g(x) - \langle y, x \rangle + \langle y, x \rangle - h(x) \tag{3.4c} \]

\[ = g(x) - h(x) \tag{3.4d} \]

Algorithm 1 generalizes subgradient optimization of convex functions to local optimization of DC functions. Accordingly, basic concepts of convex optimization like duality and KKT conditions were extended to DC functions [Tol78]. [PDTH97, PDTH06] guarantees convergence of the sequence \( \{x^{(k)}\} \) generated by Algorithm 1 to a critical point by starting with \( x^{(0)} \in \text{dom}(g) \).

**Remark 3.1.** The sequences \( \{x^{(k)}\} \) and \( \{y^{(k)}\} \) are well defined if and only if

\[ \text{dom} \partial g \subset \text{dom} \partial h \quad \text{and} \quad \text{dom} \partial h^* \subset \text{dom} \partial g^*. \]

**Theorem 3.1.** [PDTH06, Thm. 3.7] Suppose that the sequences \( \{x^{(k)}\} \) and \( \{y^{(k)}\} \) are defined by the DCA Algorithm 1 and assume DCA is well defined. Then every limit point of the sequence \( x^{(k)} \) (resp. \( y^{(k)} \)) is a critical point of \( g - h \) (resp. \( h^* - g^* \)).

### 3.2 DC-Programming for \( \ell_p \)-Minimization

Recall \( \ell_p \)-minimization, \( p < 1 \) over bounded polyhedra from (2.18), that was rewritten as a concave optimization problem in (2.20).

The DC Algorithm 1 applied to (2.20), also called **Successive Linearization Algorithm** (SLA), consists of linearizing the differentiable concave objective function of problem (2.20) around a current point \((x^{(k)}, y^{(k)})\) and solving the resulting linear program. The algorithm terminates in a finite number of steps at a stationary point after adding the constraint \( y \geq \epsilon \mathbf{1} \) to the minimization problem (2.20) for some small \( \epsilon > 0 \) to ensure the differentiability of the objective function of (2.20). Hence, we consider

\[ \mathcal{F}_\epsilon := \mathcal{F} \cap \{y : y \geq \epsilon \mathbf{1}\}, \quad \mathcal{F} := \{(x, y) \in \mathbb{R}^{2n} : x \in \mathcal{P}, -y \leq x \leq y, -\mathbf{1} \leq x \leq \mathbf{1}, 0 \leq y \leq 1\} \tag{3.5} \]

and

\[ \min_{\bar{x}, y} \bar{x}^T Dy^{p} \quad \text{subject to} \quad (x, y) \in \mathcal{F}_\epsilon. \tag{3.6} \]
Algorithm 2: DCA for (2.20) / Successive Linearization Algorithm (SLA)

**Input:** Choose $p \in (0, 1)$, $\epsilon$ small, $(x^{(0)}, y^{(0)})$.

**Output:** Approximated stationary point of (2.20).

**for** $k = 1, \ldots,$ **do**

\[
(x^{(k+1)}, y^{(k+1)}) = \arg \min_{(x,y) \in \mathcal{F}_\epsilon} ((Dy^{(k)})^{p-1})^\top y \\
\text{Stop when } ((Dy^{(k)})^{p-1})^\top (y^{(k)} - y^{(k+1)}) = 0 \\
\text{Otherwise set } k \leftarrow k + 1
\]

With $g(x, y) := \delta_{\mathcal{F}_\epsilon} (x, y)$ and $h(x, y) := 2^\top Dy^p$ we now obtain the iteration in Algorithm 2 below. By [Man96, Thm. 4.2] additionally have the following finite termination result for Algorithm 2.

**Proposition 3.2.** Algorithm 2 generates a finite sequence $(x^{(k)}, y^{(k)})_{k \in \mathbb{N}}$ with strictly decreasing objective function values for the $\ell_p$-minimization problem (2.20) with $p \in (0, 1)$, and terminating at an iteration $K \in \mathbb{N}$ satisfying the following minimum principle necessary optimality condition

\[
\left(\left(Dy^{(K)}\right)^{1/q-1}\right)^\top (y - y^{(K)}) \geq 0, \quad (x, y) \in \mathcal{F}_\epsilon, \quad (3.7)
\]

which states that $(x^{(K)}, y^{(K)})$ is a stationary point of (3.6).

### 3.3 DC-Programming for $\ell_{1-2}$-Minimization

An solution approach [YLHX15] for (1.3) is to replace the $\ell_0$-minimization by minimization of a non-convex yet Lipschitz continuous metric $\ell_{1-2}$ and consider

\[
\min \|x\|_1 - \|x\|_2 \quad \text{subject to } \quad Ax = b. \quad (3.8)
\]

Similarly we can consider for the minimization of (1.1) a proxy

\[
\min \|Dx\|_1 - \|Dx\|_2 \quad \text{subject to } \quad x \in \mathcal{P} \subset \mathbb{R}^n, \quad (3.9)
\]

with $\mathcal{P}$ from (1.2) or $\mathcal{P} = \{x \in \mathbb{R}^n : Ax = b, Cx \geq d, \|x\|_{\infty} \leq 1\}$. See Fig. 3.1 for an illustration of different sparsity measures. It is immediate to recast (3.9) as a difference of two convex functions

\[
\min \underbrace{\|x\|_1 + \delta_{\mathcal{P}}(x)}_{=: g(x)} - \underbrace{\|x\|_2}_{=: h(x)}. \quad (3.10)
\]
Figure 3.1: Contours of four sparsity metrics: \( \ell_1 \), \( \ell_{1-2} \), \( \ell_2 \) and \( \ell_1 - \alpha \ell_2^2 \), \( \alpha \in \{0.1, 0.3\} \). The level curves of \( \ell_{1-2} \) tend to the axes as the values get small and are more similar to \( \ell_0 \). In Sect. 2.2 we have shown that minimizing \( \ell_p \), for \( p < 1 \) small but finite, over bounded polyhedra is equivalent to \( \ell_0 \)-minimization. Our numerical experiments will show however that \( \ell_1 - \alpha \ell_2^2 \)-minimization seems to be (numerically) more efficient, since it better avoids local minima.

Using that
\[
\partial h(x) = \begin{cases} 
\frac{x}{\|x\|_2}, & \text{if } x \neq 0, \\
B_1(0), & \text{if } x = 0,
\end{cases}
\]
we can specialize Algorithm 1 for (3.10) to the following iteration summarized in Algorithm 3 below.

\begin{algorithm}
\caption{DC Algorithm (DCA) for \( \ell_{1-2} \)-Minimization.}
\begin{algorithmic}
\Input Good initialization \( x \).
\Output Approximate local minimizer \( \hat{x} \) of problem (3.10).
\Begin
\Repeat
\State \( y^{(k)} = \begin{cases} 
\frac{x^{(k)}}{\|x^{(k)}\|_2}, & \text{if } x^{(k)} \neq 0, \\
0, & \text{if } x^{(k)} = 0,
\end{cases} \)
\State \( x^{(k+1)} \in \arg\min_{x \in \mathcal{P}} \{ \|x\|_1 - \langle y^{(k)}, x \rangle \} \)
\Until \text{some convergence criteria is met at } x^{(K)}.
\State \( \hat{x} \leftarrow x^{(K)} \).
\End
\end{algorithmic}
\end{algorithm}

\section{3.4 DC-Programming for \( \ell_1 - \ell_2^2 \) Minimization}

We note that a DC function \( f \) has infinitely many DC decompositions. E.g. if \( f = g - h \), then \( f = (g+k) - (h+k) \) for every \( k \in \mathcal{F}_0(\mathbb{R}^n) \). The primal DC corresponding to the two DC decompositions of the objective function \( f \) are identical, but their dual programs \( h^* - g^* \) are quite different. Hence the DCA relative to these DC decompositions is also different. In other words, there are as many DCAs as there
are DC decompositions of the objective function $f$. It is useful to find a suitable DC decomposition of $f$ since it may have an important influence on the efficiency of the DCA. It is common to make the DC components of the primal objective function $f = g - h$ strongly convex. A straightforward approach is to choose above $k(x) = \frac{1}{2} \|x\|_2^2$. If $f(x) = (\|x\|_1 + \frac{1}{2} \|x\|_2^2) - (\|x\|_2 + \frac{1}{2} \|x\|_2^2)$ like in Sect. 3.3, then DC Algorithm 1 will require to solve in each iteration a quadratic program. We prefer however to solve linear programs instead and consider the problem

$$\min \|Dx\|_1 - \alpha \|Dx\|_2^2 \quad \text{subject to} \quad x \in \mathcal{P} \subseteq \mathbb{R}^n,$$

(3.12)

with $\mathcal{P}$ from (1.2) or $\mathcal{P} = \{x \in \mathbb{R}^n : Ax = b, Cx \geq d, \|x\|_\infty \leq 1\}$. We recast (3.12) as a DC-program

$$\min \left\{ \|x\|_1 + \delta_{\mathcal{P}}(x) - \alpha \|x\|_2^2 \right\}.$$

(3.13)

Using that

$$\partial h(x) = \nabla h(x) = 2\alpha x$$

(3.14)

we can specialize Algorithm 1 for (3.13) to Algorithm 4 below.

Algorithm 4: DC Algorithm (DCA) for $\ell_1 - \alpha \ell_2^2$-Minimization.

Input: Good initialization $x$, $\alpha \in (0,1)$.

Output: Approximate local minimizer $\hat{x}$ of problem (3.13).

begin

repeat

$\quad x^{(k+1)} \in \arg \min_{x \in \mathcal{P}} \left\{ \|x\|_1 - (2\alpha x^{(k)}, x) \right\}$

until some convergence criteria is met at $x^{(K)}$.

$\hat{x} \leftarrow x^{(K)}$.

end

4 Superiorization

4.1 Superiorization: Basic Concepts.

Superiorization is a recently introduced methodology which gains increasing interest and recognition, as evidenced by the dedicated special issue entitled: “Superiorization: Theory and Applications”, in the journal Inverse Problems [CHE34]. The state of current research on superiorization can best be appreciated from the “Superiorization and Perturbation Resilience of Algorithms: A Bibliography compiled and continuously updated by Yair Censor”[Cenml]. In addition, [Her14], [Cen15] and
[RP17, Section 4] are recent reviews of interest. Research works in this bibliography include a variety of reports ranging from new applications to new mathematical results of the foundation of superiorization.

This methodology is heuristic and its goal is to find certain good, or superior, solutions to optimization problems. More precisely, suppose that we want to solve a certain optimization problem, for example, minimization of a convex function under constraints (for an approach which considers the superiorization methodology in a much broader form, see [RP17, Section 4]) then, solving the full problem can be rather demanding from the computational point of view, but solving part of it, say the feasibility part (namely, finding a point which satisfies all the constraints) is, in many cases, less demanding. Suppose further that our algorithmic scheme which solves the feasibility problem is perturbation resilient, that is, it converges to a solution of the feasibility problem despite perturbations which may appear in the algorithmic steps due to noise, computational errors, and so on.

Under these assumptions, the superiorization methodology claims that there is an advantage in considering perturbations in an active way during the performance of the scheme which tries to solve the feasibility part. What is this advantage? It may simply be a solution (or an approximation solution) to the feasibility problem which is found faster thanks to the perturbations; it may also be a feasible solution \( x' \) which is better than (or superior) feasible solutions \( x \) which would have been obtained without the perturbations, where this “superiority” is measured with respect to some given cost/merit function \( \phi \), namely \( \phi(x') \leq \phi(x) \) is required (and hopefully \( \phi(x') \) will be much smaller than \( \phi(x) \)).

Observe that in the case of convex optimization problem, when the objective \( \phi \) is a convex function, the perturbation can be chosen in a non-ascending direction, as the following example: take \( s^{(k,n)} \in \partial \phi(y^{(k,n)}) \) where \( \partial \phi(z) \) is the subgradient set of \( \phi \) at \( z \), and define \( v^{(k,n)} = -s^{(k,n)}/\|s^{(k,n)}\| \) if \( 0 \notin \partial \phi(y^{(k,n)}) \), and \( v^{(k,n)} = 0 \), if \( 0 \in \partial \phi(y^{(k,n)}) \); See for example [BDHK07, CDH14].

Before we introduce Censor et al. superiorization “without gradients” [CHS16], which is more general and relevant to our results, we wish to present the superiorization methodology in the convex settings. It is clear that in the non-convex case, which is of our interest in this paper, the major question is how to choose a nonascent direction for \( \phi \) in Algorithm 5 below.

Let \( \phi : \mathbb{R}^n \to \mathbb{R} \) be a real-valued convex continuous function and let for simplicity of presentation, assume here that \( \Gamma = \mathbb{R}^n \). The Superiorized Version of the Basic Algorithm \( \mathfrak{K} \) that we consider here is based on [CZ15, Algorithm 4.1] and is presented in Algorithm 5.
Algorithm 5: Superiorized Version of the Basic Algorithm \( \mathcal{A} \)

Input: \( N \in \mathbb{N} \) and \( y^{(0)} \in \mathbb{R}^n \) a user-chosen vector.
Output: A solution to problem \( \mathcal{T} \) which is superior with respect to \( \phi \).

begin
   Given a current iteration vector \( y^{(k)} \) pick an \( N_k \in \{1, 2, \ldots, N\} \):
repeat
   \( y^{(k,0)} \leftarrow y^{(k)} \).
   Pick \( 0 < \beta_{k,n} \leq 1 \) such that \( \sum_{k=0}^{\infty} \sum_{n=0}^{N_k-1} \beta_{k,n} < \infty \).
   Pick a \( v^{(k,n)} \) that is a direction of non-ascend for \( \phi \) at \( y^{(k,n)} \).
   Calculate the perturbed iterate \( y^{(k,n+1)} \leftarrow y^{(k,n)} + \beta_{k,n} v^{(k,n)} \).
   set \( n \leftarrow n + 1 \)
until \( n < N_k \)
Exit the loop with the vector \( y^{(k,N_k)} \).
Calculate \( y^{(k+1)} \leftarrow \mathcal{A}(y^{(k,N_k)}) \) and set \( k \leftarrow k + 1 \).

4.2 Basic Algorithms for SFP

The Convex Feasibility Problem (CFP) stands at the core of the modeling of many inverse problems in various areas of mathematics and the physical sciences; for example in sensor networks, in radiation therapy treatment planning, in color imaging and in adaptive filtering, see e.g., [CAP88, Byr99, CDH10] and references therein. The CFP formulation is given next.

**Problem 4.1. The Convex Feasibility Problem (CFP).**
For \( i = 1, \ldots, p \) let \( C_i \subseteq \mathbb{R}^n \) be closed and convex sets. The CFP is:

\[
\text{find a point } x^* \in C := \bigcap_{i=1}^{p} C_i. \tag{4.1}
\]

In 1994, Censor and Elfving [CE94] introduced the Split Convex Feasibility Problem (SFP). This reformulation was employed for solving an inverse problem in intensity-modulated radiation therapy (IMRT) treatment planning, see [CBMAT06]. The problem formulates as follows.

**Problem 4.2. The Split Convex Feasibility Problem (SFP).**
Let \( \mathbb{R}^n \) and \( \mathbb{R}^m \) be two Euclidean spaces. Let \( C \subseteq \mathbb{R}^n \) and \( Q \subseteq \mathbb{R}^m \) be two non-empty, closed and convex sets, in addition given a bounded linear operator \( A : \mathbb{R}^n \rightarrow \mathbb{R}^m \), the SFP is:

\[
\text{find a point } x^* \in C \text{ such that } y^* = Ax^* \in Q. \tag{4.2}
\]

A useful tool which is used frequently for solving CFPs as well as SFPs is the class of projection methods, (see, e.g., [Byr08, Ceg12, CZ97]). These are iterative
algorithms that use projections onto sets, relying on the principle that when a family of sets is present, then projections onto the given individual sets are easier to perform than projections onto other sets (intersections, image sets under some transformation, etc.) that are derived from the given individual sets. See illustrations in Figure 4.1 which is taken from [CEH01]. Their main advantage, which makes them successful in real-world applications, is computational. They commonly are able to handle huge-size problems of dimensions beyond which more sophisticated methods cease to be efficient or even applicable due to memory requirements.

Figure 4.1: Different projection methods for the linear case. The figure is reproduced from [CEH01].

In order to discuss a more general case, we assume (standard assumption) that the set $C$ (also $Q$ for the SFP) can be represented as a sublevel set of a convex and subdifferential function $c : \mathbb{R}^n \rightarrow \mathbb{R}$, that is

$$C = \{x \in \mathbb{R}^n : c(x) \leq 0\}. \quad (4.3)$$

Next, we define the set

$$\tilde{C} = \{y \in \mathbb{R}^n : c(x) + \langle \xi, y - x \rangle \leq 0\}, \quad (4.4)$$

where $\xi \in \partial c(x)$ is the subgradient of $c$ at a point $x$. In this case the subgradient projection onto $\tilde{C}$, which is an outer approximation to the set $C$, since $C \subseteq \tilde{C}$ is...
calculated as follows. Let $\xi \in \partial c(x)$

$$P_{\tilde{C}}(y) := \begin{cases} y - \frac{c(x)}{\|\xi\|^2} \xi & \text{if } \xi \neq 0, \\ y & \text{otherwise.} \end{cases} \quad (4.5)$$

Observe that if $\xi \neq 0$ then $\tilde{C}$ is a half-space and the whole space otherwise.

So, if all the sets of the CFP, Problem 4.1, have a sublevel sets representation, that is for all $i$

$$C_i = \{x \in \mathbb{R}^n : c_i(x) \leq 0\} \quad (4.6)$$

the Cyclic Subgradient Projection (CSP) method [CL82] can be used successfully. The CSP iterative step is formulated as follows.

$$x^{k+1} = \begin{cases} x^{(k)} - \lambda_k \frac{c_i(x^{(k)})}{\|\xi^{(k)}\|^2} \xi^{(k)} & c_i(x^{(k)}) > 0 \\ x^{(k)} & c_i(x^{(k)}) \leq 0 \end{cases} \quad (4.7)$$

where $\xi^{(k)} \in \partial c_i(x^{(k)})$, $\lambda_k \in [\epsilon_1, 2 - \epsilon_2]$ are called relaxation parameters for arbitrary $\epsilon_1, \epsilon_2 > 0$ and $\{i(\nu)\}$ is a sequence of indices according to which individual sets $C_i$ are chosen, in this case cyclic: $i(\nu) = \nu \mod m + 1$. In the linear case CFP, this method reduces to the well known method of Kaczmarz [Kac37] or Algebraic Reconstruction Technique (ART) and POCS in the field of image reconstruction from projection, see [BGH70].

Observe that our interest here is the binary reconstruction problem (2.26) which is phrased as (2.27), this means that we use projection methods to find a feasible point within the bounded polyhedron $\mathcal{P}$.

### 4.3 Descent and Nonascent Directions for $\ell_0$

Further we apply the superiorization methodology to the objective $\phi(x) = \|x\|_0$. Since $\phi$ is not differentiable in this case we need to introduce non-ascending directions for $\phi$. This is a special case of an idea called superiorization “without gradients”, which was suggested to us based on his work in [CHS16], by Censor in a private communication. Let $x \in \mathbb{R}^n$, set $I = \text{supp}(x)$. Choose one $j \in I$ and define the vector $w = (w_i) \in \mathbb{R}^n$ by

$$w_i = \begin{cases} x_i & \text{if } i \neq j \\ 0 & \text{else} \end{cases} \quad (4.8)$$

Clearly $\|w\|_0 \leq \|x\|_0$. If $\|x\|_0 = s$ for some $0 < s \leq n$, then (4.8) reduces to the projection of the vector $x$ onto the non-convex set

$$Q := \{y \in \mathbb{R}^n : \|y\|_0 \leq s - 1\} \quad (4.9)$$
see [PZC⁺17].

An alternative for choosing a non-ascend direction for \( \phi(x) = \|x\|_0 \) is based on (2.24) from Prop. 2.3. We assume that \( 0 \leq x \leq 1 \) as motivated in Section 2.3 and Section 2.4, and choose for simplicity \( D = I \). First observe that for any \( p \in (0,1) \), the gradient of \( x \mapsto \|x^p\| \) is \( px^{p-1} := (px_1^{p-1}, \ldots, px_n^{p-1}) \), where \( x^p := (x^p_1, \ldots, x^p_n) \) (element-wise). Further choose \( \alpha \in (0,1) \) and define \( w = x - \alpha px^{p-1} \). Then \( \|w\|_0 \leq \|x\|_0 \). Indeed, if for some \( i \in [n] \) we have \( x_i = 0 \) then clearly \( w_i = 0 \). Otherwise it might happen that \( x_i - \alpha px_i^{p-1} = 0 \).

Further work will include superiorization for \( \phi(x) = \|x\|_1 - \|x\|_2 \). In this case we have a DC function which is non-convex but Lipschitz continuous. For such functions we wish to present a general technique (see [BKS08]) to compute non-ascend/descent directions for nonsmooth but Lipschitz continuous objective functions.

**Algorithm 6:** Computation of descent directions for nonsmooth but Lipschitz continuous functions.

**Input:** Choose any \( g^1 \in S_1 \) and \( e \in G \). Given \( z \in P, \lambda > 0, \alpha \in (0,1], c \in (0,1) \) and a tolerance \( \delta > 0 \). Compute

\[
\begin{align*}
v^1 &= \Gamma^i (x, g^1, e, z, \lambda, \alpha) = (\Gamma^1_1, \ldots, \Gamma^1_n), \\
\Gamma^1_i &= \left\{ \begin{array}{ll}
[z(\lambda) \alpha^j e_j]^{-1} \left[ \phi(x^j) - \phi(x^{j-1}) \right] & \text{if } i \neq j \\
(\lambda g_i)^{-1} \left[ \phi(x + \lambda g) - \phi(x) - \lambda \sum_{j=1, j \neq i} \Gamma^1_{j} e_j g_j \right] & i = j \\
\end{array} \right.
\end{align*}
\]

and set \( \mathcal{D}_1(x) = \{v^1\} \).

**Output:** A descent direction \( g \) of \( \phi \) at a point \( x \in \mathbb{R}^n \).

**begin**

\[
\begin{align*}
\text{Given a current iteration vector } g^k \text{ and the set } \mathcal{D}_k(x) \\
\text{repeat} \\
&\text{Compute } i = \arg \max \{ |g^k_j| \mid j = 1, \ldots, n \} \text{ and } \text{set } v^k = \Gamma^i (x, g^k, e, z, \lambda, \alpha). \\
&\text{Find the vector } \|w^k\|^2 = \min \{ \|w\|^2 \mid w \in \mathcal{D}_k(x) \}; \\
&\text{repeat} \\
&\text{Compute } g^{k+1} = -\|w^k\|^{-1} w^k \text{ and } \text{set } v^{k+1} = \Gamma^i (x, g^{k+1}, e, z, \lambda, \alpha) \text{ and } \mathcal{D}_{k+1}(x) = \text{conv} \{ \mathcal{D}_k(x) \cup \{v^{k+1}\} \}; \\
&\text{until } \|\phi(x + \lambda g^{k+1}) - \phi(x)\| > -c \|w^k\| \\
&\text{Exit the loop with the vector } g^{k+1}. \\
&\text{Calculate } g^{k+1} \text{ and set } k \leftarrow k + 1.
\end{align*}
\]
5 Experiments, Discussion

In this section we validate the local minimization approach based on DC-programming on the two image reconstruction models from Sect. 2.3 and Sect. 2.4 but also compare DC minimization to $\ell_0$-superiorization.

5.1 Experimental Set-Up

We consider the reconstruction of three test images from few tomographic projections (2.25). The first and second image are the binary test images from [Bat07] and [HSH12] representing a vascular system containing larger and smaller vessels, see Fig. 5.1 (left), (middle left). The third image is the Shepp-Logan MATLAB phantom, as shown in Fig. 5.1 (middle right).

The measurements are described by $q = Mu$, where

$$M = (M_{\theta_1}^T \ M_{\theta_2}^T \ \cdots \ M_{\theta_n}^T)^T$$

and each block matrix $M_{\theta_i}$ corresponds to a different projecting angle, see Fig. 5.1 (right) for a parallel ray geometry that corresponds to one such angle. We use the MATLAB routine parallel_tomo.m from the AIR Tools package [HSH12] that implements such a tomographic matrix for a given vector of angles. We set $n \in \{32, 64, 128\}$ the image size and use the default value of $p$, i.e. the number of parallel rays for each angle $p = \text{round}(\sqrt{2} \cdot n)$.

For the description of the superiorization results we define some of the used parameters. We denote by $N$ the dimension of the image (that is $N \times N$ pixels image), $nA$ is the number of projections which are used to sample the image, OLI represent the number of iterations in the outer loop, ILI is the number of iterations of the basic algorithm in the inner loop, SI is the total number of superiorizations per
DC-PROGRAMMING VERSUS $\ell_0$-SUPERIORIZATION FOR DISCRETE TOMOGRAPHY

outer iteration. In addition, $\textbf{SR}$ when the stopping rule $\|Ax - b\| < 10^{-5}$ achieved, $\textbf{Res}$ stands for the residual, that is for the reconstructed solution $x$ $\textbf{Res}$ is $\|Ax - b\|$, closely related, $\textbf{Diff}$ is the norm difference between the original image (IOrg) and the reconstructed (IRec) one, that is $\|\text{IOrg-IRec}\|_2$.

5.2 DC-PROGRAMMING RESULTS

5.2.1 Sparse Binary Tomography

We aim to reconstruct exactly the two binary test images $u \in \{0, 1\}^n$ via the three DC programs from Sect. 3.2, 3.3 and 3.4 from a minimal number of equidistant projection angles $\theta_i$ within the interval $[0^\circ, 180^\circ]$. To find the minimal number of projections we vary the number of projecting directions and certify a successful reconstruction if the (relative) distance of the reconstruction, further denoted by $u_{\text{rec}}$ to the original signal $u$ is below a numerical “zero”, i.e. $\frac{\|u_{\text{rec}} - u\|_2}{N} \leq 10^{-6}$. The starting point for each DC algorithm is the default starting point for the LP solver [ApS15].

We also determine the minimal number of projecting angles $n_A$ such that the matrix $M$ is overdetermined or recovery of $u$ via least-squares

$$\min \| Mu - q \|^2_2$$

(5.1)

is successful. Results are summarized in the first six rows of Table 1.

<table>
<thead>
<tr>
<th>dim ($N$)</th>
<th>$n_A$ (overdet)</th>
<th>$n_A$ (LS)</th>
<th>$n_A$ ($\ell_1$)</th>
<th>$n_A$ ($\ell_p$)</th>
<th>$n_A$ ($\ell_{1-2}$)</th>
<th>$n_A$ ($\ell_{1-\alpha \ell_2}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N = 32$</td>
<td>23</td>
<td>29</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>$N = 64$</td>
<td>46</td>
<td>56</td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>$N = 128$</td>
<td>91</td>
<td>110</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>$N = 32$</td>
<td>23</td>
<td>30</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>$N = 64$</td>
<td>46</td>
<td>56</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>$N = 128$</td>
<td>91</td>
<td>110</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>19 ($\alpha = 0.1$)</td>
</tr>
<tr>
<td>$N = 32$</td>
<td>23</td>
<td>29</td>
<td>11</td>
<td>11</td>
<td>10</td>
<td>9 ($\alpha = 0.1$)</td>
</tr>
<tr>
<td>$N = 64$</td>
<td>46</td>
<td>56</td>
<td>14</td>
<td>14</td>
<td>13</td>
<td>12 ($\alpha = 0.3$)</td>
</tr>
<tr>
<td>$N = 128$</td>
<td>91</td>
<td>110</td>
<td>14</td>
<td>13</td>
<td>11</td>
<td>11 ($\alpha = 0.1$)</td>
</tr>
</tbody>
</table>

Table 1: Results for DC-programming. The 1st column shows the test image used and the 2nd column its dimension. The 3rd column indicates the numbers of projecting angles that are necessary for obtaining an overdetermined system. Column 'nA (LS)' gives the minimal number of projecting angles such that each test images is recovered via (5.1). Column 'nA ($\ell_1$)' gives the minimal nA such that $\ell_1$-regularization is exact. Several values of $p \in \{1/2, 1/5, 1/10, 1/15\}$ yield similar results. Column 'nA ($\ell_p$)' gives the minimal nA such that $\ell_p$-regularization is exact. Column 'nA ($\ell_{1-2}$)' gives the minimal nA such that $\ell_{1-2}$-regularization is exact, while the last column gives the minimal nA such that $\ell_{1-\alpha \ell_2}$-regularization is exact. The used $\alpha$ values are also shown. We observe that $\ell_{1-\alpha \ell_2}$-regularization always improves on the $\ell_1$ results. For the gradient-sparse Shepp-Logan test image DC-programming always improves on the convex model.
5.2.2 Gradient-Sparse Discrete Tomography

Here we approximate the model presented in 2.4 using the three proxies for $\ell_0$-minimization: $\ell_p$, with $p \in \{1/2, 1/5, 1/10, 1/15\}$, $\ell_{1-2}$ and $\ell_1 - \alpha \ell_2^2$, with $\{\alpha \in \{0.1, 0.3\}\}$. The feasible set is described by both projecting and box constraints. Results for the gradient-sparse Shepp-Logan phantom are summarized in the last three rows of Table 1. We note that DC programming always reduced the number of projecting angles $n_A$ when compared to $\|\nabla u\|_1$-minimization subject to projecting and box constraints.

5.3 Superiorization Results

In this subsection we illustrate the $\ell_0$-superiorization approach on the image reconstruction models from Sect. 2.3. We consider only the Batenburg vessel image. We choose two variants for the non-ascent directions as explained in Subsection 4.3.

Our experiments are divided into two parts. In the first part, in Figure 5.3, we choose the following parameters N=128, nA=15, OLI=30 and ILI=10m, where $m$ is the number of rows of the matrix $A$. As explained above, one of natural question when applying the superiorization methodology, is the so-called Balancing Question: How to divide the efforts that a superiorization algorithm invests in target function reduction steps (perturbations) versus the efforts invested in feasibility-seeking. Trying to answer this question our experiments in Figure 5.3 are divided into four. In the first, that is in (a) we don’t use any superiorization, just feasibility-seeking algorithm (here Kaczmarz), see Figure 4.1. In (b) one superiorization (=one perturbation)
per outer iteration is used. In this case we noticed that the computational time/effort of the total procedure is higher but the reconstructed image is better with respect to step Res and Diff. In (c) we push that even further and tried to used a minimum of 10 superiorization steps for each outer iteration. In this case, the stopping criteria is achieved only after 13 outer iterations, compared to 3 in the previous scenarios but again we witness that the result is slightly better with respect to Res. Finally, in (d), similarly to (c), we used a minimum of 10 superiorization steps for each outer iteration but also included the addition of box constraints for the feasibility-seeking algorithm. Again there are improvements in Res and Diff and even the stopping criteria is reached after 11 compared to 13 in (c).

Figure 5.3: (a) SI=0, SR=3 and Res=8.2449 \times 10^{-6}, Diff=54.9293. (b) SI=1, SR=3 and Res=8.0233 \times 10^{-6}, Diff=49.2490. (c) SI=10, SR=13 and Res=9.656 \times 10^{-6}. (d) SI=10, SR=11 and Res=9.656 \times 10^{-7}, Diff=47.9286.

In the second part of our experiments the second non-ascend direction (2.24) is used with different choices of the image’s size N. In the first line, (a) and (b), we show how for a relatively small image (N=16) using the even one perturbation per outer iteration improves significantly the reconstructed image and yield a sparser image. In (b) we choose p = 2. In the second line, (c) and (d) we again compared the performances with and without perturbations for an image of size 32 and see in (d) that Diff is slightly better, but remember that only one ”nonexpansive” perturbation is used. Finally, in (e) we decided to test our scheme for a larger image of size 128 but with a fewer number of projections (nA=6) and also decrease the total number of the outer and inner iterations. Although the reconstructed image contains many artifacts, it is still very impressive to see that the measurements Res and Diff are quite good which suggests that this approach can be used an alternative for the DC-programming.

6 Conclusion

Motivated by sparse and gradient-sparse image recovery from few tomographic projections we compare two recent strategies for solving the related $\ell_0$-minimization
DC-programming versus $\ell_0$-superiorization for discrete tomography

Figure 5.4: (a) SI=0, $N = 16$, $nA = 15$, $OLI = 10$, $ILI = 5m$, Diff=7.5427. (b) SI=1, $N=16$, $nA=15$, $OLI=10$, $ILI=5m$, $p = 2$, Diff=6.3964. (c) SI=0, $N=32$, $nA=15$, $OLI=10$, $ILI=5m$, Diff=14.1781 (d) SI=1, $N=32$, $nA=15$, $OLI=10$, $ILI=5m$, $p = 2$, threshold $\epsilon = 10^{-5}$, Diff=13.1946. (e) SI=1, $N=128$, $nA=6$, $OLI=3$, $ILI=2m$, $p = 0.01$ threshold $\epsilon = 10^{-5}$, Res=$5.6340 \times 10^{-9}$, Diff=53.4470.

problem (1.1): the DC-programming approach and the $\ell_0$-superiorization. DC-programming for non-convex $\ell_0$-proxies not always improves on the convex $\ell_1$-relaxation of $\ell_0$, but often gets trapped in local minima. On the other hand $\ell_0$-superiorization yields comparable results with significantly less computational effort. For $\ell_0$-superiorization, we suggest two new non-ascend directions and test their performances for different parameter. Hence, our preliminary results shows that the $\ell_0$-superiorization can be used as an alternative approach.

Acknowledgment

AG is supported by the EU FP7 IRSES program STREVCOMS, grant no. PIRSES-GA-2013-612669.

References


DC-PROGRAMMING VERSUS $\ell_0$-SUPERIORIZATION FOR DISCRETE TOMOGRAPHY


Aviv Gibali  
Department of Mathematics,  
ORT Braude College, P.O. Box 78,  
Karmiel 2161002, Israel.  
Email: avivg@braude.ac.il

Stefania Petra  
Mathematical Imaging Group,  
Heidelberg University, Germany.  
Email: petra@math.uni-heidelberg.de