Les Cahiers du GERAD ISSN: 0711–2440

Computing a sparse projection into a box

D. Orban

G–2022–12

April 2022

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Computing a sparse projection into a box

Dominique Orban [a, b]

[a] *GERAD, Montréal (Qc), Canada, H3T 1J4*

[b] *Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal (Qc), Canada, H3C 3A7*

dominique.orban@gerad.ca

April 2022
Les Cahiers du GERAD
G–2022–12

Abstract : We describe a procedure to compute a projection of $w \in \mathbb{R}^n$ into the intersection of the so-called *zero-norm* ball $k\mathbb{B}_0$ of radius k, i.e., the set of k-sparse vectors, with a box centered at a point of $k\mathbb{B}_0$. The need for such projection arises in the context of certain trust-region methods for nonsmooth regularized optimization. Although the set into which we wish to project is nonconvex, we show that a solution may be found in $O(n \log(n))$ operations. We describe our Julia implementation and illustrate our procedure in the context of two trust-region methods for nonsmooth regularized optimization.

Résumé : Nous proposons une procédure pour calculer une projection de $w \in \mathbb{R}^n$ dans l'intersection de la soi-disant boule en *norme zéro* $k\mathbb{B}_0$ de rayon k, c'est-à-dire l'ensemble des vecteurs ayant au plus k composantes non nulles, et d'une boîte centrée en point de $k\mathbb{B}_0$. Cette projection est nécessaire dans le contexte de certaines méthodes de région de confiance pour l'optimisation non lisse régularisée. Bien que l'ensemble dans lequel on projette est non convexe, il est possible d'obtenir une solution en $O(n \log(n))$ opérations. Nous décrivons notre implémentation dans le langage Julia et illustrons la procédure dans le context de deux méthodes de région de confiance pour l'optimisation non lisse régularisée.

Acknowledgements: The author wishes to thank Aleksandr Aravkin and Robert Baraldi for fruitful discussions that made this research possible.

1 Introduction

We describe a procedure to compute a projection of a point in $\mathbb{R}^n$ into the intersection of the set of k-sparse vectors with a box centered at a k-sparse vector.

Specifically, let $\Delta\mathbb{B}_\infty$ be the ℓ_∞-norm ball of radius $\Delta \geq 0$ and centered at the origin, and $x+\Delta\mathbb{B}_\infty$ be the same ball centered at $x \in \mathbb{R}^n$. The set of k-sparse vectors in $\mathbb{R}^n$, otherwise known as the ℓ_0-pseudonorm "ball" of radius $k \in \{0, 1, \ldots, n\}$, is denoted $k\mathbb{B}_0$ and is the set of vectors with at most k nonzero components. Assume that $x \in k\mathbb{B}_0$. For given $w \in \mathbb{R}^n$, we seek to compute

$$p(w) \in P(w) := \operatorname{argmin} \{\|w - y\|_2 \mid y \in C\} \qquad C := k\mathbb{B}_0 \cap (x + \Delta\mathbb{B}_\infty). \tag{1}$$

Because C is closed, $P(w) \neq \varnothing$, but because C is nonconvex, $P(w)$ may contain several elements. In (1), we seek a global minimum—local nonglobal minima sometimes exist, but are of no particular interest here. Although it may appear as though the problem has exponential complexity due to the combinatorial nature of k-sparsity, we show that a solution may be found in $O(n \log(n))$ operations. We describe our Julia implementation and illustrate our procedure in the context of two trust-region methods for nonsmooth regularized optimization.

Context

The computation of (1) occurs in the evaluation of proximal operators encountered during the iterations of the trust-region method of Aravkin et al. [1] for nonsmooth regularized optimization. Their method is designed for problems of the form

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; f(x) + h(x), \tag{2}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ has Lipschitz-continuous gradient and $h : \mathbb{R}^n \to \mathbb{R} \cup \{\pm\infty\}$ is lower semi-continuous and proper. In large-scale data fitting and signal reconstruction problems, $h(x) = \chi(x \mid k\mathbb{B}_0)$ encodes sparsity constraints and is of interest if one is to recover a solution with at most k nonzero elements, where $\chi(\cdot \mid A)$ is the indicator of $A \subseteq \mathbb{R}^n$, i.e.,

$$\chi(x \mid A) = \begin{cases} 0 & \text{if } x \in A, \\ \infty & \text{otherwise.} \end{cases}$$

All iterates x_j generated are feasible in the sense that $x_j \in k\mathbb{B}_0$. At iteration j, a step s is computed in

$$\underset{u}{\operatorname{argmin}} \; \tfrac{1}{2}\|u - v\|_2^2 + h(x_j + u) + \chi(u \mid \Delta_j\mathbb{B}_\infty),$$

where $v \in \mathbb{R}^n$ is given and $\Delta_j\mathbb{B}_\infty$ is the trust region centered at the origin of radius $\Delta_j > 0$. With the change of variables $z := x_j + u$, we may rewrite the above as

$$\underset{z}{\operatorname{argmin}} \; \tfrac{1}{2}\|z - w\|_2^2 + \chi(z \mid k\mathbb{B}_0) + \chi(z \mid x_j + \Delta_j B_\infty) - \{x_j\},$$

where $w := x_j + v$, which precisely amounts to (1) with x_j in the role of x and Δ_j in the role of Δ because the two indicators may be combined into the indicator of the intersection.

Because nonsmooth regularized problems often involve a nonlinear least squares smooth term, Aravkin et al. [2] develop a Levenberg-Marquardt variant of their trust region method. The latter requires the same projections as just described.

Notation

Let $\operatorname{supp}(x) := \{i = 1, \ldots, n \mid x_i \neq 0\}$ be the *support* of x. If $A \subseteq \mathbb{R}^n$ is closed and $A \neq \varnothing$, we denote

$$\operatorname{proj}(w \mid A) := \operatorname{argmin} \{\|w - y\|_2 \mid y \in A\},$$

the projection of w into A, which is a set with at least one element.

When the projection of w into A is unique, such as happens when A is convex, we slightly abuse notation and write $y = \text{proj}(w \mid A)$ instead of $\{y\} = \text{proj}(w \mid A)$.

If $B \subseteq \mathbb{R}^n$, the notation $\text{proj}(\text{proj}(w \mid A) \mid B)$ refers to the set $\{z \in \text{proj}(y \mid B) \text{ for some } y \in \text{proj}(w \mid A)\}$.

If $S \subseteq \{1, \ldots, n\}$, the cardinality of S is denoted $|S|$, and its complement is S^c. For such S and for $x \in \mathbb{R}^n$, we denote x_S the subvector of x indexed by S and $A_S := \{x \in \mathbb{R}^n \mid x_{S^c} = 0\}$. Clearly, $0 \in A_S$ for any such S.

Because

$$k\mathbb{B}_0 = \bigcup \{A_S \mid S \subseteq \{1, \ldots, n\},\ |S| = k\},$$

[5, p. 175], we refer to A_S as a *piece* of $k\mathbb{B}_0$.

Related research

Duchi et al. [11] describe how to project efficiently into the ℓ_1-norm ball. The ℓ_1-norm is probably the most widely used convex approximation of the ℓ_0 norm as minimizing $\|x\|_1$ promotes sparsity under certain conditions—see, e.g., [10] and the vast ensuing compressed sensing literature.

Gupta et al. [12] describe how to project into the intersection of an ℓ_1-norm ball with a box, which may be seen as a relaxation of (1). Thom and Palm [17] and Thom et al. [18] propose a linear-time and constant space algorithm to compute a projection into a hypersphere with a prescribed sparsity, where sparsity is measured by the ratio of the ℓ_1 to the ℓ_2 norm.

Beck and Eldar [6] provide optimality conditions for the minimization of a smooth function over $k\mathbb{B}_0$. Beck and Hallak [7] provide optimality conditions for problems of the form (1) where the box is replaced with a symmetric set satisfying certain conditions. Unfortunately, (1) does not satisfy those conditions unless $x = 0$, at which point it is easy to see that a solution simply consists in chaining the projection into $k\mathbb{B}_0$ with that into $\Delta\mathbb{B}_\infty$. That is what Luss and Teboulle [15, Proposition 4.3] do with $1\mathbb{B}_2$ instead of $\Delta\mathbb{B}_\infty$.

Bolte et al. [9, Proposition 4] show how to project into the intersection of $k\mathbb{B}_0$ with the nonnegative orthant.

Kyrillidis et al. [13] explain how to compute a sparse projection into the simplex, which is probably the most closely related research to our objectives. The simplex necessarily intersects all pieces of $k\mathbb{B}_0$, which need not be the case for (1).

2 Geometric intuition

Naively chaining the projection into one set with that into the other, in either order, does not necessarily yield a point into the intersection of the two sets, even if the latter are convex. Figures 1 and 2 illustrates two situations that we may encounter when $k = 1$ and $n = 2$.

A few simple observations about Figures 1 and 2 reveal some difficulties associated with the computation of $p(w)$:

1. because both components of w_1 are equal in absolute value, as indicated by the thin diagonal in Figure 2, $\text{proj}(w_1 \mid k\mathbb{B}_0)$ is a set with two elements, and projecting those into $x + \Delta\mathbb{B}_\infty$ yields $p(w_1)$ (the correct global minimum) and p_2 (a spurious local minimum);
2. moving w_1 up slightly would preserve $p(w_1)$, but projecting into $1\mathbb{B}_0$ first would lead to p_2;
3. moving w_1 slightly to the right would result in a projection that is slightly to the right of $p(w_1)$ on the figure, but projecting into $1\mathbb{B}_0$ first would lead to p_2;
4. moving w_1 further to the right would result in $P(w_1) = \{p(w_1), p_2\}$ and moving it further still would result in $P(w_1) = \{p_2\}$;
5. in the rightmost plot, chaining the projections either way leads to a point that does not even lie in the intersection.

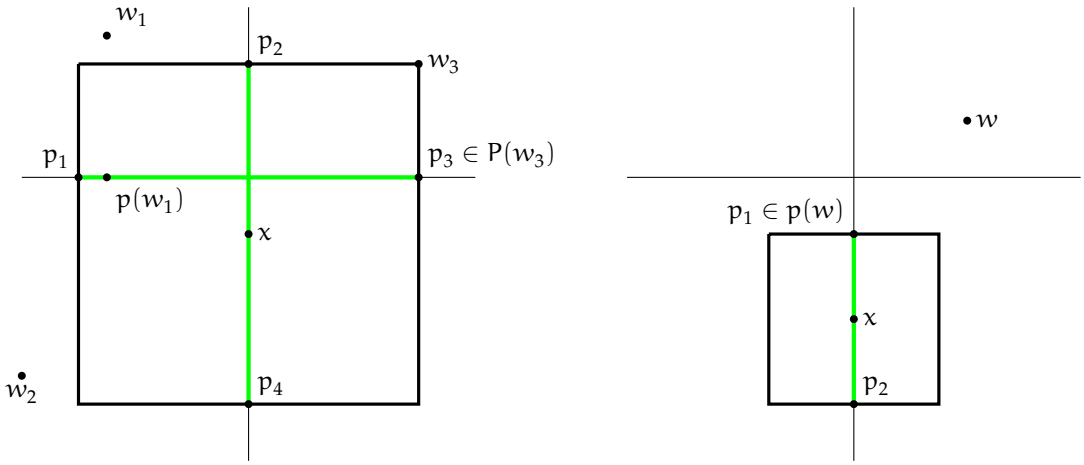

Figure 1: The set composed of the two axes is $1\mathbb{B}_0$ in $\mathbb{R}^2$, the box is $x + \Delta\mathbb{B}_\infty$ and the green set is their intersection. Left: $P(w_1) = \{p(w_1)\}$, and $P(w_2) = \{p_1\}$. With respect to w_1, the other cardinal points are p_2, a local minimum, p_3, a local maximum, and p_4, a global maximum. Right: the intersection of $1\mathbb{B}_0$ with $x + \Delta\mathbb{B}_\infty$ is entirely determined by supp(x), $P(w) = \{p_1\}$ while p_2 is a global maximum.

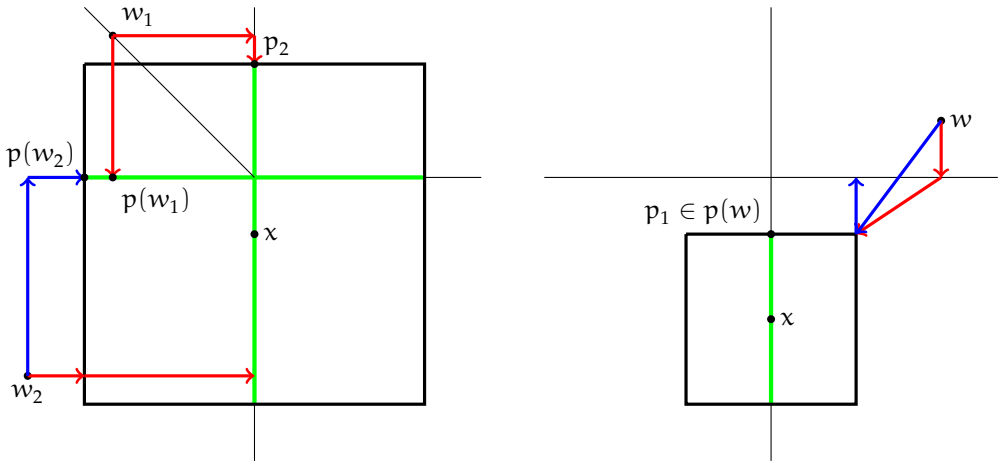

Figure 2: Simply composing the projection into $1\mathbb{B}_0$ with that into $x + \Delta\mathbb{B}_\infty$, in either order, may lead to an erroneous projection.

Note that $1\mathbb{B}_0$ is a special case for any value of n: its intersection with $x + \Delta\mathbb{B}_\infty$ consists in either a single line segment, or n segments. Indeed, the first possibility is that the nonzero component of x is $|x_i| > \Delta$. In that case, any $y \in 1\mathbb{B}_0$ with $y_j \neq 0$ and $i \neq j$ satisfies $\|y - x\|_\infty \geq |x_i| > \Delta$, and therefore $y \notin x + \Delta\mathbb{B}_\infty$. The only other possibility is that $|x_i| \leq \Delta$, in which case $0 \in x + \Delta\mathbb{B}_\infty$, and therefore, all pieces of $1\mathbb{B}_0$ intersect the box.

For $1 < k < n$, however, the intersection may consist in any number of pieces between 1 and $\binom{n}{k}$.

Figure 3 illustrates situations that may arise for $k = 1$ or 2 and $n = 3$.

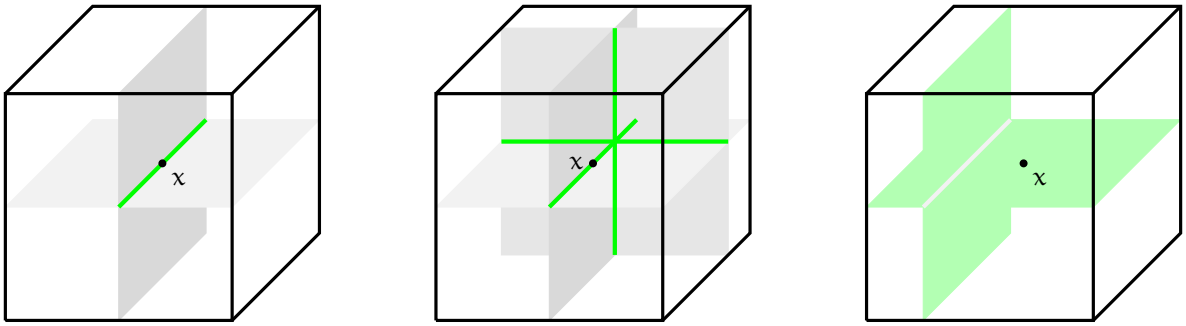

Figure 3: Left: the green segment represents a possible intersection of $1\mathbb{B}_0$ with a box in $\mathbb{R}^3$. The gray plane sections only serve to position the segment visually in three dimensions. Center: another possible intersection of $1\mathbb{B}_0$ with a box in $\mathbb{R}^3$. The box either intersects a single axis, or all of them. Right: the green region is a possible intersection of $2\mathbb{B}_0$ with a box in $\mathbb{R}^3$. The gray segment only serves as a visual aid and is part of the intersection.

3 Background and preliminary results

The unique projection y of any w into $x + \Delta\mathbb{B}_\infty$ has components

$$y_i = \max(x_i - \Delta, \min(w_i, x_i + \Delta)), \quad i = 1, \ldots, n.$$

Given $S \subseteq \{1, \ldots, n\}$, we obtain the unique projection of any w into A_S by setting $w_i = 0$ for all $i \in S^c$.

A projection y of any w into $k\mathbb{B}_0$ is a vector that has the same k largest components in absolute value as w, and the rest of its components set to zero [5, Lemma 6.71].

In the vein of Beck and Eldar [6], it is possible to state necessary optimality conditions for the more general problem

$$\underset{y \in \mathbb{R}^n}{\text{minimize}} \; f(y) \quad \text{subject to } y \in C, \tag{3}$$

of which (1) is a special case. Despite the fact that our algorithm is not based on such necessary conditions, they are relevant in their own right, and we now review and specialize them to (1).

Lemma 1. *Let $y^\star$ be a solution of (3) where f is continuously differentiable.*

1. *If $\|y^\star\|_0 < k$, then for all $i = 1, \ldots, n$,*

$$\frac{\partial f(y^\star)}{\partial y_i} \begin{cases} \leq 0 & \text{if } y_i^\star = x_i + \Delta \\ \geq 0 & \text{if } y_i^\star = x_i - \Delta \\ = 0 & \text{otherwise;} \end{cases}$$

2. *if $\|y^\star\|_0 = k$, the same conditions hold for all $i \in \operatorname{supp}(y^\star)$.*

Proof. The proof follows that of [6, Theorem 2.1]. If $\|y^\star\|_0 < k$, then for all $i = 1, \ldots, n$,

$$0 \in \underset{t \in \mathbb{R}}{\operatorname{argmin}} \{g(t) \mid \|y^\star + te_i - x\|_\infty \leq \Delta\},$$

where e_i is the i-th column of the identity, and $g(t) := f(y^\star + te_i)$.

Because $y^\star \in x + \Delta\mathbb{B}_\infty$, the constraint above reduces to $|y_i^\star + t - x_i| \leq \Delta$. The conclusion follows directly from the standard KKT conditions by noting that $g'(0) = \partial f(y^\star)/\partial y_i$.

If $\|y^\star\|_0 = k$, the same reasoning goes for all $i \in \operatorname{supp}(y^\star)$. □

By analogy with [6, Theorem 2.1], a candidate satisfying the conditions of Lemma 1 is called a *basic feasible* point. The following corollary follows directly from Lemma 1 with $f(y) := \frac{1}{2}\|w - y\|_2^2$.

Corollary 1. *Let $y^\star$ be a solution of (1).*

1. *If $\|y^\star\|_0 < k$, then for all $i = 1, \ldots, n$,*

$$y_i^\star \begin{cases} \leq w_i & \text{if } y_i^\star = x_i + \Delta \\ \geq w_i & \text{if } y_i^\star = x_i - \Delta \\ = w_i & \text{otherwise;} \end{cases}$$

2. *if $\|y^\star\|_0 = k$, the same conditions hold for all $i \in \operatorname{supp}(y^\star)$.*

Lemma 1 and Corollary 1 are only necessary conditions, and they are rather weak; there often exist vectors satisfying the conditions stated that are not solutions of (3) or (1). Consider for example $k = 1$ in $\mathbb{R}^2$, $x = (0, -1)$, $\Delta = 2$, and $w = (2, 3)$. Then, $y = (2, 0)$ satisfies the conditions of Corollary 1: $\|y\|_0 = 1$, $\operatorname{supp}(y) = \{1\}$ and $y_1 = x_1 + \Delta \leq w_1$. However, $P(w) = \{(0, 1)\}$. Indeed, $\|w - (0, 1)\| = 2\sqrt{2} < 3 = \|w - y\|$.

Observe that thanks to [6, Lemma 2.1], the number of basic feasible points of (1) is finite. Therefore, so is the cardinality of $P(w)$.

For a constant $L > 0$, Beck and Eldar define $y \in C$ to be L-stationary for (3) if it satisfies $y \in \operatorname{proj}(y - L^{-1}\nabla f(y) \mid C)$, a condition insipired by optimality conditions for convex problems. They state the following result, whose proof remains valid for (3).

Lemma 2 (6, Lemma 2.2)**.** *For any $L > 0$, $y \in \mathbb{R}^n$ is L-stationary for (3) if and only if $y \in C$ and*

$$\frac{\partial f(y)}{\partial y_i} = 0 \quad (i \in \operatorname{supp}(y)) \qquad \text{and} \qquad \left|\frac{\partial f(y)}{\partial y_i}\right| \leq L M_k(y) \quad (i \notin \operatorname{supp}(y)),$$

where $M_k(y)$ is the kth largest component of y in absolute value.

With $f(y) := \frac{1}{2}\|w - y\|_2^2$, L-stationarity reads $y \in \operatorname{proj}(y - L^{-1}(y - w) \mid C)$. Due to the simple form of $\nabla f(y) = y - w$, Lemma 2 specializes as follows.

Corollary 2. *For any $L > 0$, $y \in \mathbb{R}^n$ is L-stationary for (1) if and only if $y \in C$ and*

$$w_i = y_i \quad (i \in \operatorname{supp}(y)), \qquad \text{and} \qquad |w_i| \leq L M_k(y) \quad (i \notin \operatorname{supp}(y)).$$

As a special case of Corollary 2, if $\|y\|_0 < k$, then $M_k(y) = 0$ and we obtain $w_i = 0$ for $i \notin \operatorname{supp}(y)$. In that case, L-stationarity turns out to be independent of L and requires that $y = w$, i.e., there is a unique L-stationary point if $w \in C$, and there are no L-stationary points if $w \notin C$.

L-stationarity is stronger than basic feasibility in the sense that if y is L-stationary for (3) for any $L > 0$, then y is also a basic feasible point [6, Corollary2.1].

Under a Lipschitz assumption, solutions of (3) are L-stationary, as stated in the following result.

Proposition 1 (6, Theorem 2.2)**.** *Assume ∇f is Lipschitz continuous with constant L_f and y solves (3). Then, for any $L > L_f$,*

1. *y is L-stationary;*
2. *$\operatorname{proj}(y - L^{-1}\nabla f(y) \mid C)$ is a singleton.*

Proof. The proof follows by verifying that [6, Lemma 2.4] continues to hold for (3) and the proof of [6, Theorem 2.2] holds unchanged. □

Proposition 1 clearly applies to (1) as the gradient of $f(y) := \frac{1}{2}\|w - y\|_2^2$ is Lipschitz continuous with constant $L_f = 1$. Thus, solutions of (1) are L-stationary for $L > 1$.

Based on L-stationarity, Beck and Eldar [6] study the iteration $y^+ \in \text{proj}(y - L^{-1}\nabla f(y) \mid C)$ and show convergence to an L-stationary point for (3) under the assumption that ∇f is Lipschitz continuous. Unfortunately, in the case of (1), solving the subproblem is as difficult as solving (1) directly.

Finally, Beck and Eldar define the concept of componentwise (CW) optimality as follows: $y \in C$ is CW-minimum for (3) if

1. $\|y\|_0 < k$ and $f(y) = \min_t f(y + te_i)$ for $i = 1, \ldots, n$, or
2. $\|y\|_0 = k$ and $f(y) \leq \min_t f(y - y_i e_i + te_j)$ for $i \in \text{supp}(y)$ and $j = 1, \ldots, n$.

They observe that any solution is a CW-minimum [6, Theorem 2.3] and that any CW-minimum is a basic feasible point [6, Lemma 2.5]. The concept of CW-minimum allows them to show that any solution of (3) is L-stationary for a value L that can be significantly smaller than L_f. Based on those observations, they propose two coordinate descent-type methods that converge to a CW-minimum.

In the next section, we present a number of properties of (1) and an algorithm that identifies a solution directly, without resort to the above stationarity conditions.

4 Computing the projection

We begin with a few simple observations.

Lemma 3. *Let $S \subseteq \{1, \ldots, n\}$ such that $|S| = k$. If $y \in x + \Delta\mathbb{B}_\infty$ and $z = \text{proj}(y \mid A_S)$, then $\|z - x\|_\infty \leq \|y - x\|_\infty$ and, in particular, $z \in x + \Delta\mathbb{B}_\infty$.*

Proof. Without loss of generality, we may write $z = (y_S, 0)$. Observe now that

$$\Delta \geq \|y - x\|_\infty = \max(\|y_S - x_S\|_\infty, \|y_{S^c} - x_{S^c}\|_\infty) = \max(\|y_S - x_S\|_\infty, \|y_{S^c}\|_\infty) \geq \|y_S - x_S\|_\infty = \|z - x\|_\infty,$$

because $x_{S^c} = 0$. □

Lemma 3 holds because of the geometry of $k\mathbb{B}_0$ respective to $\mathbb{B}_\infty$ and is specific to the ℓ_∞-norm. Indeed, consider for example a ball defined in the ℓ_2-norm and set $x = (0, -1) \in 1\mathbb{B}_0$ and $\Delta = 2$. For $y_1 = (\frac{3}{4}, -\frac{1}{4})$, we have $z_1 = \text{proj}(y_1 \mid 1\mathbb{B}_0) = (\frac{3}{4}, 0)$ and $\|z_1 - x\|_2 > \|y - x\|_2$. In this example, $z_1 \in x + \Delta\mathbb{B}_2$, but consider now $y_2 = (2, -1)$. Then, $z_2 = \text{proj}(y_2 \mid 1\mathbb{B}_0) = (2, 0) \notin x + \Delta\mathbb{B}_2$.

Lemma 4. *If $w \in x + \Delta\mathbb{B}_\infty$, then $P(w) = \text{proj}(w \mid k\mathbb{B}_0)$.*

Proof. Any $y \in \text{proj}(w \mid k\mathbb{B}_0)$ has the same k largest components in absolute value as w, and the rest of its components set to zero. Thus, there must exist $S \subseteq \{1, \ldots, n\}$ with $|S| = k$ such that $y = \text{proj}(w \mid A_S)$. By Lemma 3, $\|y - x\|_\infty \leq \|w - x\|_\infty \leq \Delta$ so that $y \in x + \Delta\mathbb{B}_\infty$, and hence, $y \in C$. If there were $z \in C$ such that $\|z - w\|_2 < \|y - w\|_2$, because $z \in k\mathbb{B}_0$, there would be a contradiction with the definition of y. Therefore, y is a closest point to w in C. □

Lemma 5. *$C = A_{\text{supp}(x)} \cap (x + \Delta\mathbb{B}_\infty)$ if and only if $|x_i| > \Delta$ for all $i \in \text{supp}(x)$.*

Proof. The result follows from the observation that for any $i \in \text{supp}(x)$, there is no $y \in x + \Delta\mathbb{B}_\infty$ with $y_i = 0$. Indeed, if $y_i = 0$, $\|y - x\|_\infty \geq |y_i - x_i| = |x_i| > \Delta$. □

Lemma 6. *For any $S \subseteq \{1, \ldots, n\}$ and any $w \in \mathbb{R}^n$,*

$$\text{proj}(w \mid A_S \cap (x + \Delta\mathbb{B}_\infty)) = \text{proj}(\textit{proj}(w \mid x + \Delta\mathbb{B}_\infty) \mid A_S) = \text{proj}(\textit{proj}(w \mid A_S) \mid x + \Delta\mathbb{B}_\infty),$$

whose unique element is the vector y such that $y_S = \text{proj}(w_S \mid x_S + \Delta\mathbb{B}_\infty)$ and $y_{S^c} = 0$.

In particular, if $C = A_{\text{supp}(x)} \cap (x + \Delta\mathbb{B}_\infty)$, then $P(w) = \{\text{proj}(\text{proj}(w \mid x + \Delta\mathbb{B}_\infty) \mid A_{\text{supp}(x)})\}$.

Proof. The projection is unique because $A_S \cap (x+\Delta\mathbb{B}_\infty)$ is convex. If $y := \text{proj}(w \mid x+\Delta\mathbb{B}_\infty)$ observe that $z := \text{proj}(y \mid A_S) \in A_S \cap (x+\Delta\mathbb{B}_\infty)$ by Lemma 3.

In order to show that $z \in \text{proj}(w \mid A_S \cap (x+\Delta\mathbb{B}_\infty))$, pick any other $\bar{z} \in A_S \cap (x+\Delta\mathbb{B}_\infty)$. By construction, $\bar{z} = (\bar{y}_S, 0)$ for some $\bar{y}$. Among the infinitely many possible $\bar{y}$, we may choose the one such that $\bar{y}_{S^c} = y_{S^c}$. Then,

$$\|w-y\|_2^2 = \|w_S - y_S\|_2^2 + \|w_{S^c} - y_{S^c}\|_2^2 = \|w_S - z_S\|_2^2 + \|w_{S^c} - y_{S^c}\|_2^2,$$

and

$$\|w-\bar{y}\|_2^2 = \|w_S - \bar{y}_S\|_2^2 + \|w_{S^c} - \bar{y}_{S^c}\|_2^2 = \|w_S - \bar{z}_S\|_2^2 + \|w_{S^c} - y_{S^c}\|_2^2.$$

By definition of y, $\|w-y\|_2 \le \|w-\bar{y}\|_2$ and the above therefore implies $\|w_S - z_S\|_2^2 \le \|w_S - \bar{z}_S\|_2^2$. Because $z_{S^c} = \bar{z}_{S^c} = 0$, we may add $\|w_{S^c}\|_2^2$ to both sides of the previous inequality to obtain $\|w-z\|_2 \le \|w-\bar{z}\|_2$. □

Lemma 5 provides an easily computable criterion to determine that $C = A_{\text{supp}(x)} \cap (x+\Delta\mathbb{B}_\infty)$, and, thanks to Lemma 6, we find an element of $P(w)$ by setting all components of $\text{proj}(w \mid x+\Delta\mathbb{B}_\infty)$ that are not in $\text{supp}(x)$ to zero. Such situation is represented in the rightmost plot of Figure 1.

By Lemma 5, if there is $|x_i| \le \Delta$, then $x+\Delta\mathbb{B}_\infty$ intersects other pieces of $k\mathbb{B}_0$ than $A_{\text{supp}(x)}$. We now determine which pieces, and their number. Let

$$s(x) := \{i \in \text{supp}(x) \mid |x_i| \le \Delta\} \quad \text{and} \quad \ell(x) := \{i \in \text{supp}(x) \mid |x_i| > \Delta\}$$

be the *small* and *large* nonzero components of x.

In the special case where $s(x) = \text{supp}(x)$, i.e., *all* nonzero components of x are small, $x+\Delta\mathbb{B}_\infty$ intersects *all* pieces of $k\mathbb{B}_0$ because $0 \in C$. Unfortunately, there are

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$$

of them. As it turns out, it is possible to compute $p(w) \in P(w)$ for any $w \in \mathbb{R}^n$ in $O(n\log(n))$ operations. In view of Lemma 4, we assume that $w \notin x+\Delta\mathbb{B}_\infty$.

We may decompose (1) as suggested in [13] and observe that $y_\star \in \text{proj}(w \mid C)$ if and only if $S_\star$ and $y_\star$ are in

$$\underset{\substack{S\subseteq\{1,\dots,n\}\\ |S|=k}}{\text{argmin}} \quad \underset{y\in A_S\cap(x+\Delta\mathbb{B}_\infty)}{\text{argmin}} \quad \|w-y\|_2^2. \tag{4}$$

In the case of $\mathbb{B}_\infty$, we know that $y \in A_S \cap (x+\Delta\mathbb{B}_\infty)$ if and only if $y \in x+\Delta\mathbb{B}_\infty$ and $y_{S^c} = 0$, i.e., if and only if $y_S \in x_S + \Delta\mathbb{B}_\infty$ and $y_{S^c} = 0$. Thus, we may rewrite (4) as

$$\underset{\substack{S\subseteq\{1,\dots,n\}\\ |S|=k}}{\text{argmin}} \quad \underset{\substack{y\in x+\Delta\mathbb{B}_\infty\\ y_{S^c}=0}}{\text{argmin}} \quad \|w_S - y_S\|_2^2 + \|w_{S^c}\|^2 = \underset{\substack{S\subseteq\{1,\dots,n\}\\ |S|=k}}{\text{argmin}} \quad \underset{\substack{y_S\in x_S+\Delta\mathbb{B}_\infty\\ y_{S^c}=0}}{\text{argmin}} \quad \|w_S - y_S\|_2^2 - \|w_S\|^2.$$

For fixed S, the unique solution of the inner problem is $y = y(S)$ such that $y_S = \text{proj}(w_S \mid x_S+\Delta\mathbb{B}_\infty)$ and $y_{S^c} = 0$. Thus, the problem reduces to finding the optimal piece, determined by

$$S_\star \in \underset{\substack{S\subseteq\{1,\dots,n\}\\ |S|=k}}{\text{argmax}} \quad \|w_S\|^2 - \|w_S - y_S\|^2. \tag{5}$$

Because (5) requires examining all pieces of $k\mathbb{B}_0$, it may be solved by noting that

$$\|w_S\|^2 - \|w_S - y_S\|^2 = e^T z, \quad e = (1,1,\dots,1), \quad z_i = w_i^2 - (w_i - y_i)^2,\ i \in S,$$

i.e., the objective is the sum of the components of z with indices in S. Without any further restriction on S, one possibility is to compute $y = \text{proj}(w \mid x + \Delta\mathbb{B}_\infty)$, z_i for all $i = 1, \ldots, n$ and retain the k largest entries, as those will yield the largest sum. Applying the procedure described in Algorithm 4.1 with $L = \varnothing$ corresponds to the steps just outlined. By $\pi^{-1}(1)$, we mean the element of F that is permuted to first position in the ordering. The main cost is the computation of π, which can be obtained in $O(n \log(n))$ operations.

Algorithm 4.1 Compute the projection of w into $C := k\mathbb{B}_0 \cap (x + \Delta\mathbb{B}_\infty)$.

Require: $w \in \mathbb{R}^n$, $w \notin x + \Delta\mathbb{B}_\infty$, $L \subseteq \{1, \ldots, n\}$, $|L| \leq k$ — *supp(proj($w \mid C$)) must contain L*
1: compute $y := \text{proj}(w \mid x + \Delta\mathbb{B}_\infty)$
2: **if** $|L| = k$ **then return** L and proj($y \mid A_L$) — *Lemmas 5 and 6*
3: set $F := L^c$ and form w_F^2, $w_F - y_F$, $(w_F - y_F)^2$, and $z := w_F^2 - (w_F - y_F)^2$ — *componentwise*
4: compute a permutation π that sorts the components of z in decreasing order
5: set $S := L \cup \{\pi^{-1}(1), \ldots, \pi^{-1}(k - |L|)\}$ — *L and the indices of the $k - |L|$ largest elements of z*
6: set $y_{S^c} = 0$
7: **return** S and y.

Consider now the case where $\ell(x) \neq \varnothing$. If $i \in \ell(x)$, $x + \Delta B_\infty$ cannot intersect any A_S such that $i \notin S$. Indeed, any $y \in \mathbb{R}^n$ such that $y_i = 0$ satisfies $\|y - x\|_\infty \geq |y_i - x_i| = |x_i| > \Delta$. If $s(x) = \varnothing$, we are in the context of Lemma 6. Thus, we may focus on the case where both $s(x)$ and $\ell(x)$ are nonempty. Necessarily, $1 < |s(x)| + |\ell(x)| \leq k$ and $|\ell(x)| < k$. Constraining $S \subseteq \{1, \ldots, n\}$ to contain $\ell(x)$ leaves $k - |\ell(x)|$ indices to be chosen among the remaining $n - |\ell(x)|$, for a total of

$$\binom{n - |\ell(x)|}{k - |\ell(x)|} = \frac{(n - |\ell(x)|)!}{(k - |\ell(x)|)!\,(n - k)!}$$

possibilities. Again, it appears as though the complexity of identifying S is exponential in n in the worst case. However, the only difference with (5) is that $S_\star$ is now constrained to contain $\ell(x)$. It follows that we may apply Algorithm 4.1 with $L = \ell(x)$. If $m := n - |\ell(x)|$, the procedure has $O(m \log(m)) = O(n \log(n))$ complexity.

5 Implementation and numerical results

We implemented Algorithm 4.1 in the Julia language [8] version 1.7 as part of the ShiftedProximalOperators package of Baraldi and Orban [4], whose main objective, as the name implies, is to collect proximal operators of nonsmooth terms with one or two shifts, i.e., $h(x_k + s_j + t)$, with and without a trust-region constraint, where x_k and s_j are fixed iterates set during an outer and an inner iteration. ShiftedProximalOperators is used inside the RegularizedOptimization package of Baraldi and Orban [3], which implements, among others, the trust-region methods for nonsmooth regularized problems of Aravkin et al. [1, 2].

We employ Algorithm 4.1 to solve (1) inside two trust-region methods for nonsmooth regularized problems of the form (2). The trust region is defined in the ℓ_∞-norm in both, and provides the box $x + \Delta\mathbb{B}_\infty$, where x is the current iterate and Δ the trust-region radius. At iteration j of the method of Aravkin et al. [1], a step s_j is computed as an approximate solution of the model

$$\underset{s}{\text{minimize}} \;\; q(s) + \psi(s; x_j) + \chi(s \mid \Delta\mathbb{B}_\infty), \qquad q(s) := \nabla f(x_j)^T s + \tfrac{1}{2} s^T B_j s,$$

where $B_j = B_j^T \in \mathbb{R}^{n \times n}$ is a limited-memory LBFGS or LSR1 approximation of the Hessian of f, and $\psi(s; x_j) \approx h(x_j + s)$. Below, we choose $\psi(s; x_j) = h(x_j + s) = \chi(x_j + s \mid k\mathbb{B}_0)$ for an appropriate value of $k \in \mathbb{N}$. s_j is computed using an adaptive stepsize variant of the proximal gradient algorithm named R2 [1] that generates inner iterates $s_{j,l}$, starting with $s_{j,0} := s_j$. At iteration l of R2, we compute a step t_l that solves

$$\underset{t}{\text{minimize}} \;\; \nabla q(s_{j,l-1})^T t + \tfrac{1}{2}\sigma_l \|t\|_2^2 + \psi(s_{j,l-1} + t; x_j) + \chi(s_{j,l-1} + t \mid \Delta\mathbb{B}_\infty),$$

where $\sigma_l > 0$. If we complete the square and perform the change of variables $y = x_j + s_{j,l-1} + t$, we obtain a problem of the form (1). We refer to the method outlined above as TR.

The second trust-region method is a variant specialized to the case $f(x) = \frac{1}{2}\|F(x)\|_2^2$, where $F : \mathbb{R}^n \to \mathbb{R}^m$ inspired from the method of Levenberg [14] and Marquardt [16], where we redefine $q(s) := \frac{1}{2}\|J(x)s + F(x)\|_2^2$. We refer to the latter as LMTR.

In both methods, the decrease in the model achieved by s_j is denoted ξ. Of particular interest is the decrease achieved by $s_{j,1}$—the first step in the inner iterations—which is denoted ξ_1. It is possible to show that $\sqrt{\xi_1}$ may be used as a criticality measure for (2). Each method stops as soon as $\sqrt{\xi_1} \leq \epsilon + \epsilon\sqrt{\xi_{1,0}}$ where $\xi_{1,0}$ is the ξ_1 observed at the first outer iteration and $\epsilon = 10^{-6}$.

We illustrate the behavior of the trust-region methods on the LASSO / basis pursuit denoise problem, in which we fit a linear model to noisy observations $Ax \approx b$, where the rows of $A \in \mathbb{R}^{m\times n}$ are orthonormal. We set $b = Ax_\star + \varepsilon$, where $\|x_\star\|_0 = k$ with its nonzero components set to ± 1 randomly and $\varepsilon \sim \mathcal{N}(0, 0.01)$. In our experiment, we set $m = 200$, $n = 512$, and $k = 10$. We formulate the problem as

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \ \tfrac{1}{2}\|Ax - b\|_2^2 + \chi(x \mid k\mathbb{B}_0). \tag{6}$$

We report results in the form of the solver output in Listings 1 to 3, where *outer* is the outer iteration counter j, *inner* is the number of inner R2 iterations at each outer iteration, $f(x)$ and $h(x)$ are the value of the smooth and nonsmooth part of the objective, respectively, $\sqrt{(}\xi_1)$ is our criticality measure, $sqrt\xi$ is the square root of the decrease achieved the by step s_j, ρ is the ratio of actual versus predicted reduction used to accept or reject the step, Δ is the trust-region radius, $\|x\|$ and $\|s\|$ are the ℓ_∞-norm of the iterate and step, respectively, $\|B_j\|$ is the spectral norm of B_j, and $1/\nu$ is the regularization parameter σ_l in the R2 model. In Listing 1, B_j is a limited-memory SR1 operator with memory 5. In Listing 2, B_j is a limited-memory BFGS operator with memory 5. All methods use the initial guess $x_0 = 0$.

Listing 1: TR iterations with L-SR1 on (6).

```
outer    inner     f(x)      h(x)      √ξ1       √ξ         ρ         Δ        ‖x‖       ‖s‖       ‖
    Bⱼ‖
1        2  1.9e+00  0.0e+00 8.9e-01 8.9e-01  1.5e+00 1.0e+00 0.0e+00 4.7e-01 1.0e+00
2        9  7.4e-01  0.0e+00 4.8e-01 7.2e-01  1.2e+00 1.4e+00 4.7e-01 5.4e-01 1.0e+00
3       12  1.0e-01  0.0e+00 1.8e-01 2.3e-01  1.4e+00 1.6e+00 1.0e+00 3.3e-01 1.0e+00
4       17  3.0e-02  0.0e+00 8.7e-02 1.4e-01  1.0e+00 1.6e+00 1.1e+00 2.9e-01 1.0e+00
5       22  1.0e-02  0.0e+00 1.6e-02 2.6e-02  1.0e+00 1.6e+00 1.0e+00 3.3e-02 1.0e+00
6       18  9.5e-03  0.0e+00 2.7e-03 4.4e-03  1.0e+00 1.6e+00 1.0e+00 6.0e-03 1.0e+00
7        8  9.4e-03  0.0e+00 3.6e-04 3.7e-04  1.5e+00 1.6e+00 1.0e+00 2.6e-04 1.0e+00
8       10  9.4e-03  0.0e+00 2.0e-04 3.0e-04  1.0e+00 1.6e+00 1.0e+00 3.5e-04 1.0e+00
9        6  9.4e-03  0.0e+00 2.0e-05 3.3e-05  1.0e+00 1.6e+00 1.0e+00 4.5e-05 1.0e+00
10       1  9.4e-03  0.0e+00 2.1e-06 2.4e-06  1.2e+00 1.6e+00 1.0e+00 2.7e-06 1.0e+00
TR: terminating with ξ1 = 1.3038641262246793e-6
TR relative error
norm(TR_out.solution - sol) / norm(sol) = 0.014710272483962346
```

Listing 2: TR iterations with L-BFGS on (6).

```
outer    inner      f(x)      h(x)      √ξ1       √ξ         ρ         Δ        ‖x‖       ‖s‖      ‖
     B_j‖
1        2  1.9e+00  0.0e+00 8.9e-01 8.9e-01  1.5e+00 1.0e+00 0.0e+00 4.7e-01 1.0e+00
2       18  7.4e-01  0.0e+00 3.6e-01 7.1e-01  1.0e+00 1.4e+00 4.7e-01 5.4e-01 1.7e+00
3       23  2.1e-01  0.0e+00 7.1e-02 1.0e-01  1.6e+00 1.6e+00 1.0e+00 9.0e-02 1.8e+00
4       14  2.0e-01  0.0e+00 4.3e-02 2.7e-01  1.6e+00 1.6e+00 1.0e+00 3.6e-01 2.1e+00
5       12  8.6e-02  0.0e+00 1.1e-01 2.4e-01  1.2e+00 1.6e+00 1.1e+00 5.0e-01 2.3e+00
6       18  1.6e-02  0.0e+00 2.9e-02 6.6e-02  1.2e+00 1.6e+00 1.1e+00 1.3e-01 2.5e+00
7       23  1.1e-02  0.0e+00 1.4e-02 2.7e-02  1.4e+00 1.6e+00 1.1e+00 3.2e-02 2.6e+00
8       20  9.8e-03  0.0e+00 7.4e-03 1.7e-02  1.1e+00 1.6e+00 1.0e+00 2.4e-02 2.6e+00
9       14  9.5e-03  0.0e+00 1.7e-03 3.7e-03  1.2e+00 1.6e+00 1.0e+00 5.3e-03 2.7e+00
10      14  9.4e-03  0.0e+00 7.7e-04 1.6e-03  1.2e+00 1.6e+00 1.0e+00 1.6e-03 2.5e+00
11      15  9.4e-03  0.0e+00 3.0e-04 6.1e-04  1.2e+00 1.6e+00 1.0e+00 6.1e-04 2.4e+00
12       9  9.4e-03  0.0e+00 8.9e-05 2.0e-04  1.2e+00 1.6e+00 1.0e+00 3.1e-04 2.5e+00
13       8  9.4e-03  0.0e+00 3.1e-05 6.2e-05  1.3e+00 1.6e+00 1.0e+00 7.6e-05 2.5e+00
14       8  9.4e-03  0.0e+00 1.4e-05 3.0e-05  1.2e+00 1.6e+00 1.0e+00 3.0e-05 2.5e+00
15       4  9.4e-03  0.0e+00 4.3e-06 8.6e-06  1.1e+00 1.6e+00 1.0e+00 5.5e-06 2.6e+00
16       3  9.4e-03  0.0e+00 2.5e-06 5.8e-06  1.0e+00 1.6e+00 1.0e+00 4.3e-06 2.6e+00
TR: terminating with ξ1 = 1.0999297328606739e-6
TR relative error
norm(TR_out.solution - sol) / norm(sol) = 0.014709629662551134
```

Listing 3: LMTR iterations on (6).

```
outer    inner      f(x)      h(x)      √ξ1       √ξ         ρ         Δ        ‖x‖       ‖s‖
         1/ν
1        9  1.9e+00  0.0e+00 8.9e-01 1.4e+00  1.0e+00 1.0e+00 0.0e+00 1.0e+00 1.0
    e+00
2       11  1.1e-02  0.0e+00 2.3e-02 4.1e-02  1.0e+00 3.0e+00 1.0e+00 7.0e-02 1.0
    e+00
3       11  9.4e-03  0.0e+00 3.8e-04 6.8e-04  1.0e+00 3.0e+00 1.0e+00 1.2e-03 1.0
    e+00
4        4  9.4e-03  0.0e+00 7.0e-06 1.2e-05  1.0e+00 3.0e+00 1.0e+00 1.8e-05 1.0
    e+00
LMTR: terminating with ξ1 = 2.797637121965124e-12
LMTR relative error
norm(LMTR_out.solution - sol) / norm(sol) = 0.014710437655962767
```

Figure 4 shows the exact solution $x_\star$, and the objective history of each solver. All three solvers find a solution where the amplitude of the peaks are within 10^{-2} of the correct amplitude. It is not surprising that LMTR, which exploits the least-squares structure of (6) performs better than TR; its model is exact at each iteration, which is reflected in the fact that $\rho = 1$ at each iteration in Listing 3. TR also performs well, although, surprisingly, the potentially indefinite L-SR1 Hessian approximations of the positive definite Hessian A^TA yield fewer iterations than the positive-definite L-BFGS approximation.

From a computation cost point of view, each outer TR iteration costs one evaluation of f and, if the step is accepted, one evaluation of ∇f. In Listings 1 and 2, every step is accepted. Each inner R2 iteration in TR costs a product between the limited-memory quasi-Newton approximation and a vector, and an execution of Algorithm 4.1. Each outer LMTR iteration costs one evaluation of $F(x)$. Each inner R2 iteration in LMTR costs a Jacobian-vector product, a transposed-Jacobian-vector product, and an executation of Algorithm 4.1.

In each method, each step is a sum of R2 steps, each of which is a projection of the form (1). Figure 5 shows the first three LMTR steps. At iteration 1 (leftmost plot), the trust-region constraint is active, i.e., the step norm $\|s\|_\infty = \Delta$, which means that at least one of the projections computed during the R2 iterations resulted in a point in $k\mathbb{B}_0$ at the boundary of $x+\Delta\mathbb{B}_\infty$. At subsequent LMTR iterations, $\|s\|_\infty < \Delta$, which is expected in trust-region methods as convergence occurs, and means that at least the final projection computed during the R2 iterations resulted in a point lying strictly inside $x + \Delta\mathbb{B}_\infty$.

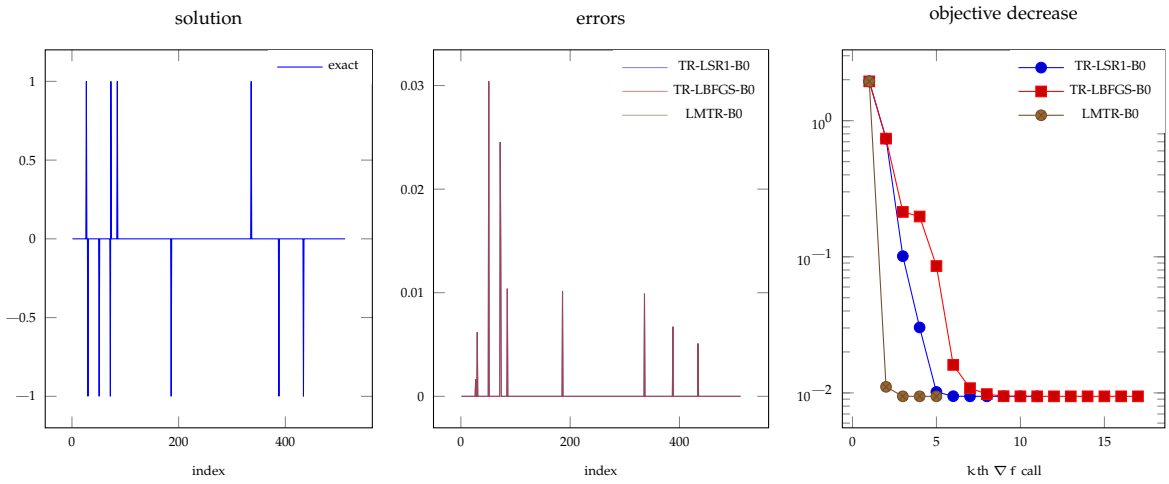

Figure 4: Exact solution of (6) (left), absolute errors (center), and objective decrease history as a function of the number of ∇f evaluations (right).

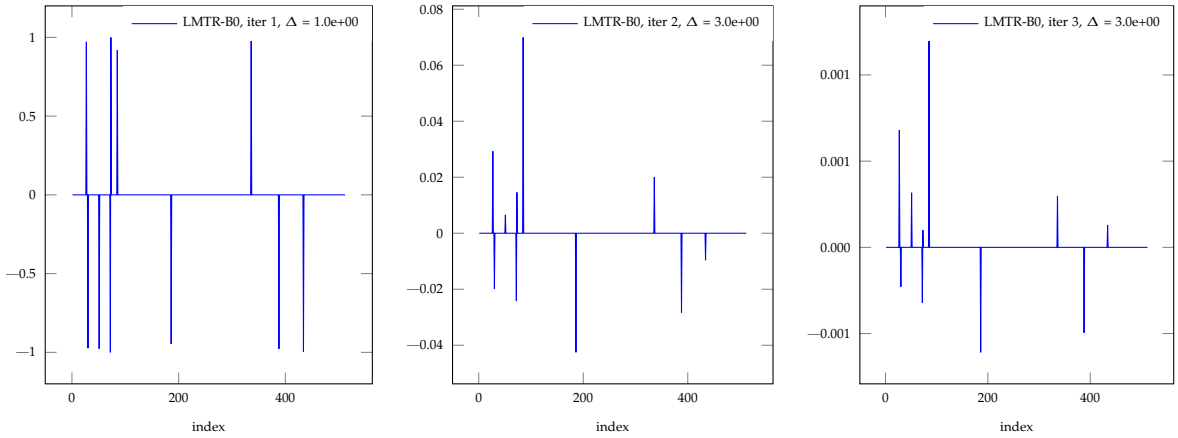

Figure 5: First three steps generated during the iterations of LMTR applied to (6). At iteration 1, the trust-region constraint is active (left). It is inactive at subsequent iterations.

6 Closing remarks

Although C is a nonconvex set, there exists an efficient projection into it, and the latter can be used to design proximal methods for nonsmooth regularized problems [1, 2]. Algorithm 4.1 makes it possible to solve sparsity-constrained problems by way of trust-region methods. It also makes it conceivable to tackle the more general problem (3) by way of one of the algorithms proposed by [6].

Possible extensions of this work include balls defined by other norms, such as other ℓ_p norms or elliptical norms. However, it is not clear that Algorithm 4.1 generalizes in a straightforward way. Indeed, the key is that the projection into $x + \Delta\mathbb{B}_\infty$ is defined componentwise. It is not difficult to sketch an example where the same procedure using the Euclidean norm yields an erroneous projection.

Another possible generalization is to consider $x \notin k\mathbb{B}_0$, as might occur in an infeasible method.

The exploration of such generalizations is the subject of ongoing research.

References

[1] A. Aravkin, R. Baraldi, and D. Orban. A proximal quasi-Newton trust-region method for nonsmooth regularized optimization. Cahier du GERAD G–2021–12, GERAD, Montréal, QC, Canada, 2021. To appear in SIAM J. Optim.

[2] A. Aravkin, R. Baraldi, and D. Orban. A Levenberg-Marquardt method for nonsmooth regularized least squares. Cahier du GERAD G–2021–XX, GERAD, Montréal, QC, Canada, 2022. In preparation.

[3] R. Baraldi and D. Orban. RegularizedOptimization.jl: Algorithms for regularized optimization. https://github.com/JuliaSmoothOptimizers/RegularizedOptimization.jl, February 2022.

[4] R. Baraldi and D. Orban. ShiftedProximalOperators.jl: Proximal operators for regularized optimization. https://github.com/JuliaSmoothOptimizers/ShiftedProximalOperators.jl, February 2022.

[5] A. Beck. First-Order Methods in Optimization. Number 25 in MOS-SIAM Series on Optimization. SIAM, Philadelphia, USA, 2017. DOI: 10.1137/1.9781611974997.

[6] A. Beck and Y. C. Eldar. Sparsity constrained nonlinear optimization: Optimality conditions and algorithms. SIAM J. Optim., 23(3):1480–1509, 2013. DOI: 10.1137/120869778.

[7] A. Beck and N. Hallak. On the minimization over sparse symmetric sets: Projections, optimality conditions, and algorithms. Math. Oper. Res., 41(1):196–223, 2016. DOI: 10.1287/moor.2015.0722.

[8] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: A fresh approach to numerical computing. SIAM Rev., 59(1):65–98, 2017. URL https://doi.org/10.1137/141000671.

[9] J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. Math. Program., (146):459–494, 2014. DOI: 10.1007/s10107-013-0701-9.

[10] E. J. Candès, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. Commun. Pur. Appl. Math., 59(8):1207–1223, 2006. DOI: 10.1002/cpa.20124.

[11] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l_1-ball for learning in high dimensions. In Proceedings of the 25th International Conference on Machine Learning, ICML '08, pages 272–279, New York, NY, USA, 2008. ISBN 9781605582054. DOI: 10.1145/1390156.1390191. URL https://doi.org/10.1145/1390156.1390191.

[12] M. D. Gupta, S. Kumar, and J. Xiao. L1 projections with box constraints. arXiv report, 2010. URL https://arxiv.org/abs/1010.0141. Primary class cs.DS.

[13] A. Kyrillidis, S. Becker, V. Cevher, and C. Koch. Sparse projections onto the simplex. In S. Dasgupta and D. McAllester, editors, Proceedings of the 30th International Conference on Machine Learning, volume 28 of Proceedings of Machine Learning Research, pages 235–243, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/kyrillidis13.html.

[14] K. Levenberg. A method for the solution of certain problems in least squares. Q. Appl. Math., (2): 164–168, 1944. DOI: 10.1090/qam/10666.

[15] R. Luss and M. Teboulle. Conditional gradient algorithms for rank-one matrix approximations with a sparsity constraint. SIAM Rev., 55(1):65–98, 2013. DOI: 10.1137/110839072.

[16] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. Journal of the Society for Industrial and Applied Mathematics, 11(2):431–441, 1963. DOI: 10.1137/0111030.

[17] M. Thom and G. Palm. Efficient sparseness-enforcing projections. arXiv Report 1303.5259v1, Ulm University, 2013. URL https://arxiv.org/abs/1303.5259. Primary class cs.CG.

[18] M. Thom, M. Rapp, and G. Palm. Efficient dictionary learning with sparseness-enforcing projections. Int. J. Comput. Vis., (114):168—194, 2015. DOI: 10.1007/s11263-015-0799-8.