Approximate Douglas–Rachford algorithm for two-sets convex feasibility problems

R. Díaz Millán 1 · O. P. Ferreira 2 · J. Ugon 1

Received: 14 June 2021 / Accepted: 13 December 2022 / Published online: 6 January 2023 © The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract

In this paper, we propose a new algorithm combining the Douglas–Rachford (DR) algorithm and the Frank–Wolfe algorithm, also known as the conditional gradient (CondG) method, for solving the classic convex feasibility problem. Within the algorithm, which will be named Approximate Douglas–Rachford (ApDR) algorithm, the CondG method is used as a subroutine to compute feasible inexact projections on the sets under consideration, and the ApDR iteration is defined based on the DR iteration. The ApDR algorithm generates two sequences, the main sequence, based on the DR iteration, and its corresponding shadow sequence. When the intersection of the feasible sets is nonempty, the main sequence converges to a fixed point of the usual DR operator, and the shadow sequence converges to the solution set. We provide some numerical experiments to illustrate the behaviour of the sequences produced by the proposed algorithm.

Keywords

Convex feasibility problem · Douglas–Rachford algorithm · Frank–Wolfe algorithm · Conditional gradient method · Inexact projections

Mathematics Subject Classification

65K05 · 90C30 · 90C25

R. Díaz Millán: This author was supported by the Australian Research Council (ARC), Solving hard Chebyshev approximation problems through nonsmooth analysis (Discovery Project DP180100602).

O. P. Ferreira: This author was supported in part by CNPq grant 304666/2021-1.

J. Ugon: This author was supported by the Australian Research Council (ARC), Solving hard Chebyshev approximation problems through nonsmooth analysis (Discovery Project DP180100602).

✉ R. Díaz Millán
r.diazmillan@deakin.edu.au

O. P. Ferreira
orizon@ufg.br

J. Ugon
j.ugon@deakin.edu.au

1 School of Information Technology, Deakin University, Melbourne, Australia

2 IME, Universidade Federal de Goiás, Goiânia, Brazil
1 Introduction

This paper addresses the classic two-sets convex feasibility problem in finite-dimensional Euclidean space. This problem is formally stated as follows:

$$\text{find } x^* \in A \cap B,$$

where the feasible sets $A, B \subset \mathbb{R}^n$ are convex, closed, and nonempty sets. A huge variation of practical applications in different areas of mathematics and physical sciences can be modelled in this format, and so Problem (1) has attracted the attention of many researchers, for example, see [1, 6, 7, 20] and references therein. Among the various algorithms to solve Problem (1), the Douglas–Rachford algorithm is one of the most interesting with a long history dating back to the 1950s, see [11]. When applied to the Problem (1), we can say that this method belongs to the class of the projection algorithm since, at each iteration, a projection on each of the feasible sets is computed, for example, see [2] and the references therein.

This paper aims to present a new algorithm to solve Problem (1). The proposed algorithm, called Approximate Douglas–Rachford algorithm (ApDR algorithm), combines the Douglas-Rachford (DR) algorithm with the conditional gradient method (CondG method) also known as Frank–Wolfe algorithm, see [14, 23]. The CondG method is used as the subroutine of the ApDR to compute feasible inexact projections onto the sets under consideration, defining the iteration based on the DR iteration. Specifically, we use the CondG method to minimize the distance between the point to project and the convex set under consideration. The CondG method is a feasible directions method that minimizes the objective function in each iteration, and returns a feasible approximate solution, that is, an inexact projection onto the set in consideration, see [4, 21]. We introduce a relative error criterion to lower the cost of each projection while preserving the convergence of the ApDR algorithm. The ApDR algorithm generates two sequences, one $(x^k)_{k \in \mathbb{N}}$ based on the DR iteration and the other $(y^k_A)_{k \in \mathbb{N}} \subset A$ corresponding to the shadow sequence of approximate projections. The main results of this paper are as follows. If $A \cap B \neq \emptyset$, then $(x^k)_{k \in \mathbb{N}}$ converge to a point $x^*$, which is a fixed point of the usual Douglas–Rachford operator, $(y^k_A)_{k \in \mathbb{N}}$ converges to $\mathcal{P}_A(x^*)$ the projection of $x^*$ onto $A$ and $\mathcal{P}_A(x^*), A \cap B$.

The idea of using the CondG method to compute inexact projections onto the sets $A$ and $B$, instead of the exact ones, is particularly interesting from a computational point of view. It may not be necessary to compute the exact projections when the iterates are far from $A \cap B$, since it adds to the computational cost that the DR algorithm spends computing the projections. It is noteworthy that the CondG method is easy to implement, has a low computational cost per iteration, and readily exploits separability and sparsity, resulting in high computational performance in different classes of compact sets, see [12, 15, 17, 21, 23]. All these features added to the ApDR algorithm make it very attractive from a computational point of view. In addition, for feasible sets that can only be accessed efficiently through a linear programming oracle, the ApDR algorithm is of considerable interest. It is worth mentioning that the idea of using the CondG method to compute approximate projections has already been used in several papers. Indeed, the most direct precursor is [22] which uses the CondG method to compute inexact projections in an accelerated first-order method for solving the constrained optimisation problem. For more applications of the CondG method as a subroutine to compute inexact projections, we refer the reader to [5, 8, 9, 18, 19]. Another method for solving Problem (1), which do not use exact projection onto the feasible sets $A$ and $B$, can be found in [10]. In the literature, other inexact methods have used Douglas–Rachford ideas to solve the optimisation problem of the sum of two convex functions, and to find a zero in the sum of two maximal monotone operators. These studies focus on algorithms
where an approximate value of the proximal operator replaces the exact computation; papers dealing with these ideas include [13, 25, 26]. In this paper, the projection operator is replaced by another operator, efficiently computed via CondA, which is not assumed to be a stochastic approximation of the original operator. To the best of our knowledge, the combination of the CondG method with the DR algorithm for solving Problem (1) has not yet been considered.

The organization of the paper is as follows. In Sect. 2, we present some notation and basic results used throughout the paper. In Sect. 3 we describe the CondG method and present some results related to inexact version of projection, reflection and Douglas–Rachford operator. In Sect. 4 we present ApDR algorithm and its convergence analysis. Some numerical experiments are provided in Sect. 5. We conclude the paper with some remarks in Sect. 6.

2 Preliminaries

In this section, we present some preliminary results used throughout the paper. We denote:

\[ \mathbb{N} = \{0, 1, 2, 3, \ldots \} \]

\[ \langle \cdot, \cdot \rangle \] is the usual inner product and \( \| \cdot \| \) is the Euclidean norm. Let \( C \subset \mathbb{R}^n \) be closed, convex and nonempty set, the projection and the reflection are, respectively, the maps \( P_C : \mathbb{R}^n \to C \) and \( R_C : \mathbb{R}^n \to \mathbb{R}^n \) defined by

\[
P_C(v) := \arg \min_{z \in C} \| v - z \|, \quad R_C(v) := 2P_C(v) - v. \tag{2}
\]

In the next lemma we present some important properties of the projection mapping.

**Lemma 1** Given a convex and closed set \( C \subset \mathbb{R}^n \) and for all \( v \in \mathbb{R}^n \), the following properties hold:

(i) \( \langle v - P_C(v), z - P_C(v) \rangle \leq 0 \), for all \( z \in C \);

(ii) \( \|P_C(v) - z\|^2 \leq \|v - z\|^2 - \|P_C(v) - v\|^2 \), for all \( z \in C \);

(iii) the projection mapping \( P_C \) is continuous.

**Proof** The items (i) and (iii) are proved in [3, Proposition 3.10, Theorem 3.14]. For item (ii), combine \( \|v - z\|^2 = \|P_C(v) - v\|^2 + \|P_C(v) - z\|^2 - 2\langle P_C(v) - v, P_C(v) - z \rangle \) with item (i).

Let \( C, D \subset \mathbb{R}^n \) be convex, closed, and nonempty sets. Let \( T : C \rightrightarrows D \) denotes a set-valued operator that maps a point in \( C \) to a subset of \( D \) (i.e. \( T(x) \subset D \) for all \( x \in C \)). In the case when \( T(x) \) is a singleton for all \( x \in C \), \( T \) is said to be a single-valued mapping, which is denoted as \( T : C \to D \), and write \( T(x) = z \) whenever \( T(x) = \{z\} \). For a given set-valued operator \( T : C \rightrightarrows D \) and \( y \in C \), we set \( T(x) + y := \{z + y : z \in T(x)\} \). The set of fixed points of an operator \( T \), denoted by \( \text{Fix} T \), is defined by \( \text{Fix} T := \{x \in C : x \in T(x)\} \). The **identity operator** is the mapping \( \text{Id} : \mathbb{R}^n \to \mathbb{R}^n \) that maps every point to itself. Let us recall the **Douglas–Rachford operator** associated to closed and convex sets \( A \) and \( B \), defined by

\[
T_{A,B}(u) = \frac{1}{2} (R_B \circ R_A + \text{Id}) (u). \tag{3}
\]

The next proposition gives a relationship between the fixed point of the Douglas–Rachford operator and the solution set of the feasibility problem, its proof can be found in [2].
Let $A, B \subset \mathbb{R}^n$ be convex closed sets. Then, $A \cap B \neq \emptyset$ if and only if $\text{Fix } T_{A,B} \neq \emptyset$. Furthermore, $P_A(u) \in A \cap B \neq \emptyset$ for all $u \in \text{Fix } T_{A,B}$.

3 Conditional gradient (CondG) method

In this section we recall the classical conditional gradient method (CondG), see for example [4], and the concept of feasible inexact projection operator, see [8]. Associated to the feasible inexact projection operator, we introduce the concept of inexact reflection operator and the inexact Douglas–Rachford operator. We also present some important properties of these operators.

For presenting CondG method, we assume the existence of a linear optimisation oracle (or simply LO oracle) capable of minimising linear functions over the constraint set $C$. We formally state the CondG method in Algorithm 3 to calculate an inexact projection of the point $u \in \mathbb{R}^n$ onto a compact convex set $C$ relative to $y \in C$ and $v \in \mathbb{R}^n$ with forcing parameter $\epsilon \geq 0$.

Algorithm 1 CondG$_C$ method $y_C^+ \in P_C^e(y, v, u)$

1: Take $\epsilon > 0$, $y \in C$ and $v, u \in \mathbb{R}^n$. Set $w_0 = y$ and $\ell = 0$.
2: Use a LO oracle to compute an optimal solution $z_\ell$ and the optimal value $s^*_\ell$ as

$$z_\ell := \arg \min_{z \in C} \langle w_\ell - u, z - w_\ell \rangle, \quad s^*_\ell := \langle w_\ell - u, z_\ell - w_\ell \rangle.$$  (4)

3: If $-s^*_\ell \leq \epsilon \|v - y\|^2$, then stop, and set $y_C^+ := w_\ell$. Otherwise, set

$$w_{\ell+1} := w_\ell + \alpha_\ell (z_\ell - w_\ell), \quad \alpha_\ell := \min \left\{ 1, \frac{-s^*_\ell}{\|z_\ell - w_\ell\|^2} \right\}.$$  (5)

4: Set $\ell \leftarrow \ell + 1$, and go to Step 2.

Let us describe the main features of CondG method; for further details, see, for example, [4, 21, 22]. Let $u \in \mathbb{R}^n$, $\psi_u : \mathbb{R}^n \to \mathbb{R}$ be defined by $\psi_u(z) := \|z - u\|^2/2$, and $C \subset \mathbb{R}^n$ a convex compact set. The CondG method is a specialized version of the classical conditional gradient method applied to the problem $\min_{z \in C} \psi_u(z)$. In this case, (4) is equivalent to the stepsizes $s^*_\ell := \min_{z \in C} \langle \psi_u'(w_\ell), z - w_\ell \rangle$. Since the function $\psi_u$ is convex we have $\psi_u(z) \geq \psi_u(w_\ell) + \langle \psi_u'(w_\ell), z - w_\ell \rangle \geq \psi_u(w_\ell) + s^*_\ell$, for all $z \in C$. Set $w_s := \arg \min_{z \in C} \psi_u(z)$ and $s^*_s := \psi_u(w_s)$. Letting $z = w_s$ in the last inequality we have $\psi_u(w_\ell) \geq \psi_u(w_s) + s^*_s$, which implies that $s^*_s \leq 0$. Thus, $-s^*_s = \langle u - w_\ell, z_\ell - w_\ell \rangle \geq 0 \geq \langle u - w_s, z - w_s \rangle$, for all $z \in C$. Therefore, we can set the stopping criterion to $-s^*_s \leq \epsilon \|y - v\|^2$, which ensures that CondG will terminate after finitely many iterations. When the CondG method computes $w_\ell \in C$ satisfying $-s^*_s \leq \epsilon \|y - v\|^2$, then the method terminates. Otherwise, it computes the stepsize $\alpha_\ell = \arg \min_{\alpha \in [0, 1]} \psi_u(w_\ell + \alpha (z_\ell - w_\ell))$ using exact line search. Since $z_\ell, w_\ell \in C$ and $C$ is convex, we conclude from (5) that $w_{\ell+1} \in C$, thus the CondG method generates a sequence in $C$. Finally, (4) implies that $\langle u - w_\ell, z - w_\ell \rangle \leq -s^*_s$, for all $z \in C$. Hence, considering the stopping criterion $-s^*_s \leq \epsilon \|y - v\|^2$, we conclude that any output of CondG method $y_C^+ \in C$ is a feasible inexact projection onto $C$ of the point $u \in \mathbb{R}^n$ with respect to
\[ y \in C, \ v \in \mathbb{R}^n \] and relative error tolerance function \( \epsilon \|y - v\|^2 \), i.e.,

\[ \langle u - y^+_C, z - y^+_C \rangle \leq \epsilon \|y - v\|^2 \quad \forall \ z \in C. \] (6)

Inspired by (6), in the following we present the feasible inexact projection operator associated to Algorithm 3, see [8].

**Definition 1** The feasible inexact projection operator onto \( C \) relative to \( y \in C \) and \( v \in \mathbb{R}^n \) with forcing parameter \( \epsilon \geq 0 \), denoted by \( \mathcal{P}_\epsilon^C(y, v, \cdot) : \mathbb{R}^n \rightarrow C \), is defined as follows

\[ \mathcal{P}_\epsilon^C(y, v, u) := \left\{ y^+ \in C : \langle u - y^+, z - y^+ \rangle \leq \epsilon \|y - v\|^2, \forall \ z \in C \right\}. \] (7)

Each point \( y^+_C \in \mathcal{P}_\epsilon^C(x, y, u) \) is called a feasible inexact projection of \( u \) onto \( C \) relative to \( x \in C \) and \( y \in \mathbb{R}^n \) with forcing parameter \( \epsilon \geq 0 \).

**Remark 1** Let \( C \) be a closed and convex set, \( v \in \mathbb{R}^n \), \( y \in C \), \( v \in \mathbb{R}^n \) and \( \epsilon \geq 0 \). It follows from Definition 1, item (i) of Lemma 1 and \( \epsilon \geq 0 \) that, \( \mathcal{P}_\epsilon C(u) \in \mathcal{P}_\epsilon^C(y, v, u) \). Hence, \( \mathcal{P}_\epsilon^C(y, v, u) \neq \emptyset \). For \( \epsilon = 0 \), Lemma 1 (i) together with (6) implies that \( \mathcal{P}_\epsilon C(u) = \mathcal{P}_0^C(y, v, u) \), for all \( u \in \mathbb{R}^n \), \( y \in C \) and \( v \in \mathbb{R}^n \). It is worth noting that in order to use Algorithm 3 for projecting exact and inexact onto \( C \) we need to assume that the LO oracle can be used effectively. In particular, we need to assume that the set \( C \subset \mathbb{R}^n \) is compact. For an example where LO oracle cannot be used effectively, see [28].

Below we present a particular counterpart of the firm non-expansiveness of the projection operator to feasible inexact projection operator.

**Proposition 3** Let \( u, v \in \mathbb{R}^n \), \( y_C \in C \) and \( \epsilon \geq 0 \). If \( y^+_C \in \mathcal{P}_\epsilon^C(y_C, v, u) \) and \( x_C = \mathcal{P}_\epsilon C(\tilde{w}) \), then

\[ \|y^+_C - x_C\|^2 \leq \|u - \tilde{w}\|^2 - \|(u - y^+_C) - (\tilde{w} - x_C)\|^2 + 2\epsilon\|y_C - v\|^2. \]

**Proof** Since \( y^+_C \in \mathcal{P}_\epsilon^C(y_C, v, u) \) and \( x_C = \mathcal{P}_\epsilon C(\tilde{w}) \), it follows from (7) and Lemma 1 that

\[ \langle y^+_C - u, y^n_C - x_C \rangle \leq \epsilon\|y_C - v\|^2, \quad \langle \tilde{w} - x_C, y^+_C - x_C \rangle \leq 0. \]

By adding the last two inequalities, some algebraic manipulations yield

\[ -\langle u - \tilde{w}, y^+_C - x_C \rangle \leq -\|y^+_C - x_C\|^2 + \epsilon\|y_C - v\|^2. \]

Since \( \|(u - y^+_C) - (\tilde{w} - x_C)\|^2 = \|u - \tilde{w}\|^2 - 2\langle u - \tilde{w}, y^+_C - x_C \rangle + \|y^+_C - x_C\|^2 \), the desired inequality follows by combination with the last inequality.

**Proposition 4** Let \( u, v \in \mathbb{R}^n \), \( y_C \in C \) and \( \epsilon \geq 0 \). If \( y^+_C \in \mathcal{P}_\epsilon^C(y_C, v, u) \), then \( \|y^+_C - \mathcal{P}_\epsilon^C(u)\| \leq \sqrt{2\epsilon}\|y_C - v\| \).

**Proof** Since \( y^+_C \in \mathcal{P}_\epsilon^C(y_C, v, u) \), using Definition 1 we have \( \langle u - y^+_C, z - y^+_C \rangle \leq \epsilon\|y_C - v\|^2 \), for all \( z \in C \). Thus, some algebraic manipulations give \( \|y^+_C - z\|^2 + \langle u - z, z - y^+_C \rangle \leq \epsilon\|y_C - v\|^2 \), for all \( z \in C \). Since \( \|u - y^+_C\|^2 = \|u - z\|^2 + 2\langle u - z, z - y^+_C \rangle + \|z - y^+_C\|^2 \), for all \( z \in C \), we have

\[ \|y^+_C - z\|^2 + \frac{1}{2}\|y^+_C - u\|^2 - \|u - z\|^2 - \|z - y^+_C\|^2 \leq \epsilon\|y_C - v\|^2, \quad \forall z \in C, \]
which is equivalent to \( \|y_C^+ - z\|^2 + \|y_C^+ - u\|^2 - \|z - u\|^2 \leq 2\varepsilon \|y_C - v\|^2 \), for all \( z \in C \). Substituting \( z = P_C(u) \) into the last inequality and using \( \|P_C(u) - u\| \leq \|y_C^+ - u\| \) the desired result follows.

Next we introduce the concept of inexact reflection operator associated to a convex set \( C \).

**Definition 2** The inexact reflection operator associated to \( C \) relative to \( y_C \in C \) and \( v \in \mathbb{R}^n \) with forcing parameter \( \varepsilon \geq 0 \), denoted by \( R_C^\varepsilon(y_C, v, \cdot) : \mathbb{R}^n \rightrightarrows \mathbb{R}^n \), is a set-valued mapping defined as follows \( R_C^\varepsilon(y_C, v, u) := \{ 2y_C^+ - u : y_C^+ \in \mathcal{P}_C^\varepsilon(y_C, v, u) \} \), or equivalently,

\[
R_C^\varepsilon(y_C, v, u) := 2\mathcal{P}_C^\varepsilon(y_C, v, u) - u. \tag{8}
\]

Each point belonging to \( R_C^\varepsilon(y_C, v, u) \) is called an inexact reflection of \( u \) with respect \( C \) relative to \( y_C \in C \) and \( v \in \mathbb{R}^n \) with forcing parameter \( \varepsilon \geq 0 \).

In the following we present a particular counterpart to the non-expansiveness of the reflection operator to inexact reflection operator.

**Proposition 5** Let \( u, v \in \mathbb{R}^n \), \( y_C, \bar{x}_C \in C \), \( v \in \mathbb{R}^n \), \( \varepsilon \geq 0 \) and \( \delta \geq 0 \). If \( y_C^+ \in \mathcal{P}_C(y_C, break v, u) \) and \( \bar{x}_C = P_C(\bar{w}) \), then \( 2y_C^+ - u \in R_C^\varepsilon(y_C, v, u) \), \( 2\bar{x}_C - \bar{w} \in R_C^\varepsilon(\bar{w}) \), and

\[
\|(2y_C^+ - u) - (2\bar{x}_C - \bar{w})\|^2 \leq \|u - \bar{w}\|^2 + 4\varepsilon \|y_C - v\|^2.
\]

**Proof** The first inclusion follows from Definition 2 and the second from (2). To prove the inequality, first note that direct computation yields

\[
\|(2y_C^+ - u) - (2\bar{x}_C - \bar{w})\|^2 = \|u - \bar{w}\|^2 + 2\left(\|y_C^+ - \bar{x}_C\|^2 - \|u - \bar{w}\|^2 + \|(u - y_C^+) - (\bar{w} - \bar{x}_C)\|^2\right).
\]

Since \( y_C^+ \in \mathcal{P}_C^\varepsilon(y_C, v, u) \) and \( \bar{x}_C = P_C(\bar{w}) \), the desired inequality follows from the last equality together with Proposition 3.

In order to introduce the approximate Douglas–Rachford algorithm in the next section, it is necessary to introduce first the approximate Douglas–Rachford (ApDR) operator.

**Definition 3** Let \( A, B \subset \mathbb{R}^n \) be closed convex sets. The inexact Douglas–Rachford (IDR) operator associated to the sets \( A \) and \( B \) relative to \( y_A \in A \) and \( y_B \in B \) with forcing parameters \( \varepsilon \geq 0 \) and \( \delta \geq 0 \), denoted by \( T_{A,B}^{\varepsilon,\delta}(y_A, y_B, \cdot) : \mathbb{R}^n \rightrightarrows \mathbb{R}^n \), is a set-valued mapping defined as follows

\[
T_{A,B}^{\varepsilon,\delta}(y_A, y_B, u) := \left\{ u + y_B^+ - y_A^+ : y_A^+ \in \mathcal{P}_A(y_A, y_B, u), \ y_B^+ \in \mathcal{P}_B(y_B, y_A, 2y_A^+ - u) \right\}. \tag{9}
\]

or equivalently,

\[
T_{A,B}^{\varepsilon,\delta}(y_A, y_B, u) := \frac{1}{2} \left( R_B^\delta(y_B, y_A, y_B^+, y_A^+, y_B) + u \right). \tag{10}
\]

**Remark 2** For \( \varepsilon = 0 \), it follows from Definition 1 that \( \mathcal{P}_C(u) = \mathcal{P}_C^0(y_A, y_B, u) \), for all \( u \in \mathbb{R}^n \), for all \( y_A \in C \) and \( y_B \in \mathbb{R}^n \). Thus, the second equality in (2) and (8) imply that that \( R_C(u) = R_C^0(y_A, y_B, u) \), for all \( u \in \mathbb{R}^n \), \( y_A \in C \) and \( y_B \in \mathbb{R}^n \). Furthermore, it follows from (3) and (10) that \( T_{A,B}(u) = T_{A,B}^{0,0}(y_A, y_B, u) \), for all \( u \in \mathbb{R}^n \), \( y_A \in A \) and \( y_B \in B \).

We end this section with an important result that we need to prove our main result.
Proposition 6 Let \( A, B \subset \mathbb{R}^n \) be convex closed sets such that \( A \cap B \neq \emptyset \), \( y_A \in A \) and \( y_B \in B \), \( \epsilon \geq 0 \) and \( \delta \geq 0 \). Take \( x \in \mathbb{R}^n \), \( y_A^+ \in \mathcal{P}^e_A(y_A, y_B, x) \), \( y_B^+ \in \mathcal{P}^e_B(y_B, y_A, 2y_A^+ - x) \), and set \( x^+ := x + y_B^+ - y_A^+ \). If \( \tilde{x} \in \text{Fix } T_{A,B} \), then

\[
\|x^+ - \tilde{x}\|^2 \leq \|x - \tilde{x}\|^2 - \|x - x^+\|^2 + 2(\epsilon + \delta)\|y_A - y_B\|^2.
\]

Proof Since \( \tilde{x} \in \text{Fix } T_{A,B} \) we have \( \mathcal{P}_A(\tilde{x}) = \mathcal{P}_B(2\mathcal{P}_A(\tilde{x}) - \tilde{x}) \). Set \( \tilde{x}_A = \mathcal{P}_A(\tilde{x}) \) and \( \tilde{x}_B = \mathcal{P}_B(2\tilde{x}_A - \tilde{x}) \). In this case, \( \tilde{x}_A = \tilde{x}_B \). Due to \( y_B^+ \in \mathcal{P}^e_B(y_B, y_A, 2y_A^+ - x) \) and \( \tilde{x}_B = \mathcal{P}_B(2\tilde{x}_A - \tilde{x}) \), Proposition 5 with \( C = B \), \( \epsilon = \delta, u = 2y_A^+ - x, v = y_A, \tilde{x}_C = \tilde{x}_B \) and \( w = 2\tilde{x}_A - \tilde{x} \) yields

\[
\|(2y_B^+ - (2y_A^+ - x)) - (2\tilde{x}_B - (2\tilde{x}_A - \tilde{x}))\|^2 \leq \|(2y_A^+ - x) - (2\tilde{x}_A - \tilde{x})\|^2 + 4\delta\|y_B - y_A\|^2.
\]

Thus, taking into account that \( \tilde{x}_A = \tilde{x}_B \), we conclude that

\[
\|(2y_B^+ - (2y_A^+ - x)) - \tilde{x}\|^2 \leq \|(2y_A^+ - x) - (2\tilde{x}_A - \tilde{x})\|^2 + 4\delta\|y_B - y_A\|^2. \tag{11}
\]

Considering that \( y_A^+ \in \mathcal{P}^e_A(y_A, y_B, x) \) and \( \tilde{x}_A = \mathcal{P}_A(\tilde{x}) \) we apply again Proposition 5 with \( C = A, u = x, v = y_B, \tilde{x}_C = \tilde{x}_A \) and \( w = \tilde{x} \) to obtain

\[
\|(2y_A^+ - x) - (2\tilde{x}_A - \tilde{x})\|^2 \leq \|x - \tilde{x}\|^2 + 4\epsilon\|y_A - y_B\|^2. \tag{12}
\]

Hence, combining (11) and (12) we conclude that

\[
\|(2y_B^+ - (2y_A^+ - x)) - \tilde{x}\|^2 \leq \|x - \tilde{x}\|^2 + 4(\epsilon + \delta)\|y_B - y_A\|^2. \tag{13}
\]

On the other hand, \( x^+ - \tilde{x} = \frac{1}{2}(x - \tilde{x}) + \frac{1}{2}(2y_B^+ - (2y_A^+ - x) - \tilde{x}) \). Thus, direct computation shows that

\[
\|x^+ - \tilde{x}\|^2 = \frac{1}{4}\|x - \tilde{x}\|^2 + \frac{1}{2}(x - \tilde{x}, 2y_B^+ - (2y_A^+ - x) - \tilde{x}) + \frac{1}{4}\|2y_B^+ - (2y_A^+ - x) - \tilde{x}\|^2. \tag{14}
\]

Since \( x - x^+ = \frac{1}{2}(x - \tilde{x}) - \frac{1}{2}(2y_B^+ - (2y_A^+ - x) - \tilde{x}) \), we have

\[
\|x - x^+\|^2 = \frac{1}{4}\|x - \tilde{x}\|^2 - \frac{1}{2}(x - \tilde{x}, 2y_B^+ - (2y_A^+ - x) - \tilde{x}) + \frac{1}{4}\|2y_B^+ - (2y_A^+ - x) - \tilde{x}\|^2. \tag{15}
\]

Therefore, summing (14) and (15) we conclude that

\[
\|x^+ - \tilde{x}\|^2 + \|x - x^+\|^2 = \frac{1}{2}\|x - \tilde{x}\|^2 + \frac{1}{2}\|2y_B^+ - (2y_A^+ - x) - \tilde{x}\|^2,
\]

Therefore, using (13) we have \( \|x^+ - \tilde{x}\|^2 + \|x - x^+\|^2 \leq \|x - \tilde{x}\|^2 + 2(\epsilon + \delta)\|y_B - y_A\|^2 \), which is equivalent to the desired inequality, and the proof is concluded.

4 Approximate Douglas–Rachford algorithm

The aim of this section is to present a new algorithm for solving the classic feasibility problem, which we name Approximate Douglas–Rachford algorithm. Before presenting the approximate Douglas–Rachford algorithm, let us first recall the classical Douglas–Rachford algorithm.

At each step of the classic Douglas–Rachford algorithm the previous iterate is first reflected through \( A \), then reflected through \( B \), and finally the resulting point is averaged with the previous iterate. In this case the iterative sequence \( \{x^k\}_{k \in \mathbb{N}} \) is defined as

\[
x^{k+1} := \frac{1}{2} \left( \mathcal{R}_B(\mathcal{R}_A(x^k)) + x^k \right) = \frac{1}{2} (\mathcal{R}_B \circ \mathcal{R}_A + \text{Id})(x^k), \quad \forall k \in \mathbb{N}. \tag{16}
\]
**Algorithm 2** Approximate Douglas–Rachford (ApDR) algorithm

1: Let \((\epsilon_k)_{k\in\mathbb{N}}\) and \((\delta_k)_{k\in\mathbb{N}}\) be sequences of non-negative real numbers, \(y^0_A \in A\), \(y^0_B \in B\) and \(x^1 \in \mathbb{R}^n\). Set \(k = 1\).

2: Compute

\[
y^k_A \in \mathcal{P}^\epsilon_k A (y^{k-1}_A, y^{k-1}_B, x^k), \quad y^k_B \in \mathcal{P}^{\delta_k} B (y^{k-1}_B, y^{k-1}_A, 2y^k_A - x^k).
\]

If \(y^k_B = y^k_A\), then stop. Otherwise, set the next iterate \(x^{k+1}\) as follows

\[
x^{k+1} = x^k + y^k_B - y^k_A.
\]

3: Set \(k \leftarrow k + 1\), and go to Step 2.

Then conceptual approximate Douglas–Rachford (ApDR) algorithm is stated in Algorithm 4. Let us describe the main features of Algorithm 4. In Step 1 to compute the next iterate \(x^{k+1}\) in (18) we need first to compute \(y^k_A\) and \(y^k_B\) satisfying (17). Then, it follows from (9) that the next iterate \(x^{k+1}\) satisfies

\[
x^{k+1} \in \mathcal{T}^{\epsilon_k, \delta_k}_{A, B} (y^{k-1}_A, y^{k-1}_B, x^k).
\]

If Algorithm 4 stops, then \(y^k_B = y^k_A\), and (18) implies that \(x^{k+1} = x^k\). Hence, it follows from (19) that \(x^k \in \mathcal{T}^{\epsilon_k, \delta_k}_{A, B} (y^{k-1}_A, y^{k-1}_B, x^k)\). Therefore, Proposition 2 implies that \(y^k_A = y^k_B \in A \cap B\) and we have a solution.

**Remark 3** If \(\epsilon_k = 0\) and \(\delta_k = 0\) then Remark 1 and (17) imply that \(y^k_A = \mathcal{P}_A (x^k)\) and \(y^k_B = \mathcal{P}_B (2y^k_A - x^k)\). Then, it follows from (18) that

\[
x^{k+1} = x^k + \mathcal{P}_B (2\mathcal{P}_A (x^k) - x^k) - \mathcal{P}_A (x^k).
\]

We proceed to show that last equality is equivalent to (16). First note that some algebraic manipulations show that

\[
x^k + \mathcal{P}_B (2\mathcal{P}_A (x^k) - x^k) - \mathcal{P}_A (x^k) = \frac{1}{2} \left( 2\mathcal{P}_B (2\mathcal{P}_A (x^k) - x^k) - (2\mathcal{P}_A (x^k) - x^k) + x^k \right).
\]

Hence, using the last equality and the definition of the reflection \(\mathcal{R}_A (x^k)\) we conclude that

\[
x^{k+1} = \frac{1}{2} \left( 2\mathcal{P}_B (\mathcal{R}_A (x^k)) - \mathcal{R}_A (x^k) + x^k \right),
\]

and by using the definition of \(\mathcal{R}_B (x^k)\), the last inequality becomes (16). Therefore, if \(\epsilon_k = 0\) and \(\delta_k = 0\) then Conditional Douglas–Rachford algorithm retrieve the classical Douglas–Rachford algorithm.

**Remark 4** It follows from Definition 1, the item (i) of Lemma 1, \(\epsilon_k \geq 0\) and \(\delta_k \geq 0\) that \(\mathcal{P}_A (x^k) \in \mathcal{P}^{\epsilon_k} A (y^{k-1}_A, y^{k-1}_B, x^k)\) and \(\mathcal{P}_B (2y^k_A - x^k) \in \mathcal{P}^{\delta_k} B (y^{k-1}_B, y^{k-1}_A, 2y^k_A - x^k)\), i.e., the inexact projection also accepts an exact one. Therefore, if the exact projection onto \(A\) or onto \(B\) is easy to compute then we do not need to use Algorithm 3 to do that task in obtaining (17) in the respective set. For instance, the exact projections onto a hyperplane, box constraint or Lorentz cone is very easy to obtain. For example, see [24, p. 520] and [16, Proposition 3.3].

In the following we present our main result related the convergence of Algorithm 4.
Theorem 7 Let \( \{x^k\}_{k \in \mathbb{N}^*} \) be a sequence generated by Algorithm 4. Assume that \( A \cap B \neq \emptyset \), \( A \subseteq \mathbb{R}^n \) is compact set, \( 0 \leq 2(\epsilon_k + \delta_k) \leq \bar{\epsilon} < 1 \) and \( \bar{\epsilon} \neq 0 \). Then, \( \{x^k\}_{k \in \mathbb{N}^*} \) converges to a point \( x^* \in \text{Fix } T_{A,B} \). As a consequence, \( \{y^k_A\}_{k \in \mathbb{N}} \) converges to \( P_A(x^*) \in A \cap B \).

Proof Let \( \bar{x} \in \text{Fix } T_{A,B} \). Applying Proposition 6 with \( y_A = y_A^{k-1}, y_B = y_B^{k-1}, \epsilon = \epsilon_k \), \( x = x^k, y_A^+ = y_A^{k-1}, y_B^+ = y_B^{k-1}, x^* = x^{k+1} \) and taking into account (18) we obtain that

\[
\|x^{k+1} - \bar{x}\|^2 \leq \|x^k - \bar{x}\|^2 - \|y_A^k - y_B^k\|^2 + 2(\epsilon_k + \delta_k)\|y_A^{k-1} - y_B^{k-1}\|^2, \quad \forall k \in \mathbb{N}.
\]

Since \( 0 \leq 2(\epsilon_k + \delta_k) \leq \bar{\epsilon} < 1 \) we have \( \|x^{k+1} - \bar{x}\|^2 \leq \|x^k - \bar{x}\|^2 - \|y_A^k - y_B^k\|^2 + \bar{\epsilon}\|y_A^{k-1} - y_B^{k-1}\|^2 \), \( \forall k \in \mathbb{N}^* \), which is equivalent to

\[
\|x^{k+1} - \bar{x}\|^2 + \bar{\epsilon}\|y_A^{k-1} - y_B^{k-1}\|^2 \leq \|x^k - \bar{x}\|^2 + \bar{\epsilon}\|y_A^{k-1} - y_B^{k-1}\|^2 - (1 - \bar{\epsilon})\|y_A^{k-1} - y_B^{k-1}\|^2, \quad \forall k \in \mathbb{N}^*.
\]

(20)

It follows from (20) that, for any \( \bar{x} \in \text{Fix } T_{A,B} \), the sequence \( \{\|x^{k+1} - \bar{x}\|^2 + \bar{\epsilon}\|y_A^{k-1} - y_B^{k-1}\|^2\}_{k \in \mathbb{N}^*} \) is monotonous, non-increasing and bounded from below by zero, which implies it must converge. Thus, taking into account that the inequality in (20) also implies that

\[
(1 - \bar{\epsilon})\|y_A^k - y_B^k\|^2 \leq (\|x^k - \bar{x}\|^2 + \bar{\epsilon}\|y_A^{k-1} - y_B^{k-1}\|^2) - (\|x^{k+1} - \bar{x}\|^2 + \bar{\epsilon}k\|y_A^{k-1} - y_B^{k-1}\|^2), \quad \forall k \in \mathbb{N}^*,
\]

and \( 0 < \bar{\epsilon} < 1 \), we have \( \lim_{k \to +\infty} \|y_A^k - y_B^k\|^2 = 0 \). Due to \( \{x^k\}_{k \in \mathbb{N}^*} \) being convergent, we also conclude that \( \{y^k\}_{k \in \mathbb{N}^*} \) is bounded. Let \( \{y^{k_j}\}_{j \in \mathbb{N}} \) be a converging subsequence of \( \{x^k\} \), and \( x^* = \lim_{j \to +\infty} x^{k_j} \). We prove that \( x^* = \text{Fix } T_{A,B} \). For that, we first note that from the fact that \( A \) is a compact set, \( \{y^k_A\}_{k \in \mathbb{N}} \subseteq A \), and \( \lim_{k \to +\infty} \|y^k_A - y_A^*\| = 0 \), we conclude that both sequences \( \{y^k_A\}_{k \in \mathbb{N}} \) and \( \{y^k_B\}_{k \in \mathbb{N}} \) are also bounded. Let \( \bar{y} \) be an cluster point of \( \{y^{k_j}\}_{j \in \mathbb{N}} \) and \( \{y^k_A\}_{k \in \mathbb{N}} \) be a subsequence such that \( \lim_{j \to +\infty} y^{k_j} = \bar{y} \). Since \( \lim_{j \to +\infty} \|y^k_A - y_B^*\| = 0 \) we obtain that \( \lim_{j \to +\infty} \|y^k_B - y_B^*\| = 0 \). Thus, \( \lim_{j \to +\infty} y^k_A = y_A^* \). Hence, both sequences \( \{y^k_A\}_{k \in \mathbb{N}} \) and \( \{y^k_B\}_{k \in \mathbb{N}} \) converge to a same point \( \bar{y} \in A \cap B \). Furthermore, (17) and Definition 1 imply that

\[
\begin{align*}
(x^{k_j} - y^{k_j}_A, z - y^{k_j}_A) & \leq \epsilon_k\|y^{k_j-1}_A - y^{k_j-1}_B\|^2, \quad \forall z \in A, \\
(2y^{k_j}_B - x^{k_j}) - y^{k_j}_A, z - y^{k_j}_A) & \leq \delta_k\|y^{k_j-1}_A - y^{k_j-1}_B\|^2, \quad \forall z \in B.
\end{align*}
\]

Taking the limit as \( \ell \) goes to \( +\infty \) in these inequalities, we conclude that \( (x^* - \bar{y}, z - \bar{y}) \leq 0 \), for all \( z \in A \), and \( (2\bar{y} - x^* - \bar{y}, z - \bar{y}) \leq 0 \) for all \( z \in B \). Hence, \( \bar{y} = P_A(x^*) \) and \( \bar{y} = P_B(2\bar{y} - x^*) \), or equivalently, \( P_A(x^*) = P_B(2P_A(x^*) - x^*) \), which after some algebraic manipulations yields

\[
x^* = \frac{1}{2} \left( 2P_B \left( R_A(x^*) \right) - R_A(x^*) + x^* \right).
\]

Therefore, using (3) we have \( x^* = T_{A,B}(x^*) \). Moreover, \( (\|x^{k+1} - x^*\|^2 + \bar{\epsilon}\|y_A^{k-1} - y_B^{k-1}\|^2)_{k \in \mathbb{N}^*} \) is also monotonous and non-increasing. Considering that \( \lim_{j \to +\infty} \|y^{k_j}_A - y^{k_j}_B\| = 0 \), (18) this implies that \( \lim_{j \to +\infty} \|x^{k_j+1} - x^*\| = x^*, \) i.e., \( \lim_{j \to +\infty} \|x^{k+1} - x^*\| = x^* \). Thus, \( \lim_{j \to +\infty} (\|x^{k+1} - x^*\|^2 + \bar{\epsilon}\|y_A^{k-1} - y_B^{k-1}\|^2) = 0 \). Hence, the sequence \( (\|x^{k+1} - x^*\|^2 + \bar{\epsilon}\|y_A^{k-1} - y_B^{k-1}\|^2)_{k \in \mathbb{N}} \) must converge to zero. Therefore, thanks to \( \lim_{k \to +\infty} \|y^{k-1}_A - y^{k-1}_B\| = 0 \) we have

\[
\lim_{k \to +\infty} \|x^k - x^*\|^2 = \lim_{k \to +\infty} (\|x^k - x^*\|^2 + \bar{\epsilon}\|y^{k-1}_A - y^{k-1}_B\|^2) = 0,
\]
which implies that $\lim_{j \to +\infty} x^k = x^*$, and the proof of the first statement follows. We proceed to prove the second one. Since the sequence $\{x^k\}_{k \in \mathbb{N}}$ converges to $x^* \in Fix(T_{A,B})$, it follows from (18) that $\lim_{k \to \infty} \| y^k_A - y^k_B \| = 0$. Furthermore, by continuity of the projection, the sequence $\{ P_A(x^k) \}_{k \in \mathbb{N}^+}$ converges to $P_A(x^*)$. Using Proposition 4 with $C = A$, $u = x^k$, $v = y_B^k$, $e = \epsilon_k$, $y_C = y_A^{k-1}$ and $y_C^+ = y_A^k$, we have that $\| y_A^k - P_A(x^k) \| \leq \sqrt{2\epsilon_k} \| y_A^{k-1} - y_B^k \|$. Hence, due to $\lim_{k \to \infty} \| y_A^{k-1} - y_B^k \| = 0$, we obtain $\lim_{k \to \infty} \| y_A^k - P_A(x^k) \| = 0$. Since $\{ P_A(x^k) \}_{k \in \mathbb{N}}$ converges to $P_A(x^*)$, we have $\lim_{k \to \infty} y_A^k = P_A(x^*)$, and using Proposition 2 the second statement follows, and the proof is completed.

We end this section with a special case of Algorithm 4, namely, when the exact projection onto $B$ is easy to compute. In this case, we use Algorithm 3 only to compute inexact projection onto the set $A$. We recall that the inexact projection also accepts an exact one, and then the statement of the result is as follows.

**Corollary 8** Let $\{x^k\}_{k \in \mathbb{N}^+}$ be a sequence generated by Algorithm 4. Assume that $A \cap B \neq \emptyset$, $A \subset \mathbb{R}^d$ is a compact set, $0 \leq 2\epsilon_k \leq \tilde{\epsilon} < 1$ and $\tilde{\epsilon} \neq 0$. Furthermore, assume that $y_A^k = P_B(2y_A^k - x^k)$, for all $k \in \mathbb{N}^+$. Then, $\{x^k\}_{k \in \mathbb{N}^+}$ converges to a point $x^* \in Fix T_{A,B}$. As a consequence, $\{y_A^k\}_{k \in \mathbb{N}^+}$ converges to $P_A(x^*) \in A \cap B$.

**Proof** It follows from Remark 1 that $P_B(2y_A^k - x^k) \in P_B(\delta, y_A^{k-1}, 2y_A^k - x^k)$, for all $\delta \geq 0$. Thus, the proof is an immediate consequence of Theorem 7 by taking $\delta_k = 0$, for all $k \in \mathbb{N}^+$.

### 5 Numerical experiments

We applied the approximate Douglas–Rachford algorithm to find the intersection of two ellipses, and of an ellipse and a hyperplane. In each case, we consider the cases when the interior of the intersection is empty or not. In the latter case, the iterates $x_k$ are predicted to converge to a point in the intersection, whereas in the former case, the approximate projections $y_A^k$ are predicted to converge to a point in the intersection. The algorithms were implemented in Julia and are available at https://github.com/ugonj/approximateDR.

We perform four sets of experiments. In each experiment, we apply the approximate Douglas–Rachford algorithm to find the intersection of an ellipse with either an ellipse or a half-space, in both cases testing the scenarios when the interior of the intersection is either empty or nonempty.

Define the matrices $R$ and $D$ as follows

$$R(\theta) := \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}, \quad D(\alpha, \beta) := \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix}.$$  

The first ellipse is given by:

$$E_1 := \{ x \in \mathbb{R}^2 : \langle x - z_1, M_1(x - z_1) \rangle \leq 1 \}$$

where $z_1 = (0, 0)$ and

$$M_1 := R\left(\frac{\pi}{3}\right)^T D(2, 0.2) R\left(\frac{\pi}{3}\right), \quad M_2 := \lambda R\left(\frac{\pi}{4}\right)^T D(2, 0.2) R\left(\frac{\pi}{4}\right).$$

We then find the intersection between the ellipse $E_1$ and the following sets:

$$E_2 := \{ x \in \mathbb{R}^2 : \langle x - z_2, M_2(x - z_2) \rangle \leq 1 \},$$
Table 1 Number of iterations taken by the approximate Douglas Rachford algorithm before convergence, for various values of $\epsilon$

| $\epsilon$ | $E_2$ | $E_3$ | $H_1$ | $H_2$ |
|---|---|---|---|---|
| 0.245 | 52 (2) | 14 | 10 (4) | 5 |
| 0.120 | 51 (2) | 13 | 12 (3) | 5 |
| Exact | 24 (6) | 11 | 15 (4) | 6 |

In brackets the number of iterations until the shadow sequence $\{y_A^k\}$ or $\{y_B^k\}$ reaches the intersection.

Table 2 Relative computational time taken by the approximate Douglas Rachford algorithm compared to the exact Douglas Rachford algorithm, for various values of $\epsilon$

| $\epsilon$ | $E_2$ | $E_3$ | $H_1$ | $H_2$ |
|---|---|---|---|---|
| 0.245 | 0.34 | 0.48 | 0.17 | 0.15 |
| 0.120 | 0.36 | 0.48 | 0.23 | 0.18 |

$E_3 := \{x \in \mathbb{R}^2 : \langle (x - z_3), M_3(x - z_3) \rangle \leq 1\}$,
$H_1 := \{x \in \mathbb{R}^2 : \langle (1, 0)^T, x \rangle \geq 1.3\}$,
$H_2 := \{x \in \mathbb{R}^2 : \langle (1, 0)^T, x \rangle \geq \max_{x \in E_1} \}$

We set $z_2 = (2.3, 1.5)$ and

$$M_2 := \lambda R(\frac{\pi}{4})^T D(2, 0.2) R\left(\frac{\pi}{4}\right),$$

and $z_1 \approx (1.788, -0.664)$ and $M_3 := \lambda M_2$, with $\lambda \approx 0.145$, so that the intersection of the two sets is a single point ($x \approx (1.393, 1.310)$).

All sets intersect the ellipse $E_1$, and $\text{int}(E_1 \cap E_2) \neq \emptyset$, $\text{int}(E_1 \cap E_3) = \emptyset$, $\text{int}(E_1 \cap H_1) \neq \emptyset$, and $\text{int}(E_1 \cap H_2) = \emptyset$.

In all cases, we set the initial point to $x_0 = (-1, 1.5)$. We stop the algorithm when $\|y_A^k - y_B^k\|^2 < 10^{-6}$ (note that $x^k \in \text{Fix} T_{A,B}$ iff $\|y_A^k - y_B^k\|^2 = 0$). In Tables 1 and 2, we summarise the result of these experiments. We report the number of iterations taken by the algorithm in Table 1, namely the number of iterations taken by the sequence $\{x^k\}$ to converge. When the intersection is nonempty, we also report in brackets the number of iterations until the shadow sequence $\{y_A^k\}$ or $\{y_B^k\}$ reaches the intersection (so we have an alternative stopping criterion reached whenever we find a point in the intersection). The shadow sequences do not reach the intersection when the interior is empty (although they converge to it). We present the relative CPU time taken by the Approximate Douglas Rachford method in Table 2. This number is the ratio of the CPU time taken by the appDR method over the CPU time taken by the exact DR, both averaged over 10000 runs, as measured using BenchmarkTools.jl [27] (numbers less than 1 mean that the approximate DR method was faster than the exact DR method). To simulate the case when a closed form for the projection isn’t known, we use the Conditional Gradient algorithm to find the exact projections over the ellipses, and stop when $-s^k \leq 10^{-6}$. In all experiments we project onto half-spaces using a closed form formula for an exact projection. By Corollary 8, the results from this paper apply.

The results presented in these tables indicate that the approximate Douglas Rachford method is competitive with the exact Douglas Rachford method. In all cases the approximate version of the method is faster than the exact version, even when it requires more iterations. In all cases both algorithms require few iterations to reach a point in the intersection, even when it takes more iterations for the sequences to converge to a fix point of the operator.
We illustrate the iterations of the algorithms on our experiments in Figs. 1, 2, 3, 4, 5, 6, 7 and 8. It can be seen that the behaviour of the algorithms are comparable for all values of $\epsilon$.

### 5.1 Higher dimensions

We conduct the same experiment as above. For this section we consider the sets $E \cap A$ where $E$ is an ellipse and $A$ is another ellipse or a half-space, now in large dimensional spaces. Table 3 show that the approximate algorithm outperforms the exact algorithm in dimensions 10 and 50, respectively.
Fig. 5 Iterates of the approximate Douglas Rachford algorithm to find the intersection with nonempty interior of ellipse and a half-plane for $\epsilon = 0.245$, $\epsilon = 0.120$ and exact projections.

Fig. 6 Distance $\|y_A^k - y_B^k\|$, when $\text{int}(E_1 \cap H) \neq \emptyset$ for $\epsilon = 0.245$, $\epsilon = 0.120$ and exact projections.

Fig. 7 Iterates of the approximate Douglas Rachford algorithm to find the intersection with empty interior of ellipse and a half-plane for $\epsilon = 0.245$, $\epsilon = 0.120$ and exact projections.

Fig. 8 Distance $\|y_A^k - y_B^k\|$, when $\text{int}(E_1 \cap H) = \emptyset$ for $\epsilon = 0.245$, $\epsilon = 0.120$ and exact projections.

Remark 5 Since the $ApDR$ algorithm is an inexact version of Douglas–Rachford ($DR$), it only makes sense to compare with $DR$ exact on problems where at least one of the orthogonal projections is "hard" to compute. Otherwise, if the projections are easy to compute, it is better to use the $DR$ directly. Our algorithm combines $DR$ with the $CondG$ method to avoid exact projections. For classes of problems where $CondG$ is not competitive compared to algorithms that compute exact projection, we can expect that $ApDR$ will also not be competitive over the exact algorithms. When $CondG$ is competitive (or the only option), then the $apDR$ will be competitive over other approximate alternating projection methods on similar problems where the exact $DR$ method is more competitive.
| Set A  | interior | $\epsilon$ | ratio $n = 10$ | ratio $n = 50$ |
|--------|----------|------------|----------------|----------------|
| Ellipse $\neq \emptyset$ | 0.0 | 1.0 | 1.0 |
| Ellipse $\neq \emptyset$ | 0.25 | 0.100521 | 0.168136 |
| Ellipse $\neq \emptyset$ | 0.49 | 0.0979546 | 0.16746 |
| Ellipse $\neq \emptyset$ | 0.125 | 0.0898331 | 0.144305 |
| Ellipse $\emptyset$ | 0.0 | 1.0 | 1.0 |
| Ellipse $\emptyset$ | 0.25 | 0.222831 | 0.414691 |
| Ellipse $\emptyset$ | 0.49 | 0.19753 | 0.431551 |
| Ellipse $\emptyset$ | 0.125 | 0.250097 | 0.394936 |
| Half-space $\neq \emptyset$ | 0.0 | 1.0 | 1.0 |
| Half-space $\neq \emptyset$ | 0.25 | 0.0990263 | 0.200922 |
| Half-space $\neq \emptyset$ | 0.49 | 0.0780704 | 0.150503 |
| Half-space $\neq \emptyset$ | 0.125 | 0.135472 | 0.257405 |
| Half-space $\emptyset$ | 0.0 | 1.0 | 1.0 |
| Half-space $\emptyset$ | 0.25 | 0.0990003 | 0.200922 |
| Half-space $\emptyset$ | 0.49 | 0.0783989 | 0.147699 |
| Half-space $\emptyset$ | 0.125 | 0.135597 | 0.236583 |

6 Conclusions

In this paper, we proposed the ApDR algorithm for finding a point in the intersection of a convex, compact set with a convex set. This method is based on the DR algorithm, where exact projections are replaced with inexact projections, which can be found using the CondG method. The algorithm produces a sequence of iterates that converges towards a fix point of the exact Douglas Rachford operator, and the sequence of approximate projections converges towards a point in the intersection between the two sets. Numerical results demonstrate that the proposed algorithm works well in practice, and competitively with the exact DR algorithm. When the interior of the intersection is non-empty both algorithms converged after finitely many iterations. In future work, we plan to extend our results to the case when neither set is bounded, and to the infinite dimensional case. We will also investigate the complexity of the proposed algorithm, and verify the finite convergence of the algorithm when the interior is nonempty.

It is worth mentioning that just as the Alternating Projection (AP) method and DR algorithm share similarities, their approximate versions also do so when the CondG method is used to compute the approximate projection. In [9] an approximate version of the AP method was introduced using the CondG method to compute the approximate projection; as a continuation of this idea, we present here the ApDR algorithm as an alternative for users of the DR algorithm. As a logical and natural consequence, we expect these approximate algorithms to inherit the advantages that exact versions have over each other. We also plan to investigate the relationship between these approximate methods in the near future.
References

1. Aragón Artacho, F.J., Borwein, J.M., Tam, M.K.: Douglas-Rachford feasibility methods for matrix completion problems. ANZIAM J. 55(4), 299–326 (2014)
2. Artacho, F.J.A., Campoy, R., Tam, M.K.: The Douglas-Rachford algorithm for convex and nonconvex feasibility problems. Math. Methods Oper. Res. 91(2), 201–240 (2019)
3. Bauschke, H.H., Combettes, P.L.: Convex analysis and monotone operator theory in Hilbert spaces. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. Springer, New York, 2011. With a foreword by Hédy Attouch
4. Beck, A., Teboulle, M.: A conditional gradient method with linear rate of convergence for solving convex linear systems. Math. Methods Oper. Res. 59(2), 235–247 (2004)
5. Carderera, A., Pokutta, S.: Second-order Conditional Gradient Sliding. arXiv e-prints, page arXiv:2002.08907 (2020)
6. Combettes, P.L.: The convex feasibility problem in image recovery. In: Hawkes, P. (ed.) Advances in imaging and electron physics, vol. 95, pp. 155–270. Academic Press, New York (1996)
7. Combettes, P.L.: Hard-constrained inconsistent signal feasibility problems. EEE Trans. Signal Process. 47, 2460–2468 (1999)
8. de Oliveira, F.R., Ferreira, O.P., Silva, G.N.: Newton’s method with feasible inexact projections for solving constrained generalized equations. Comput. Optim. Appl. 72(1), 159–177 (2019)
9. Díaz Millán, R., Ferreira, O.P., Prudente, L.F.: Alternating conditional gradient method for convex feasibility problems. Comput. Optim. Appl. 80, 245–269 (2021)
10. Díaz Millán, R., Lindstrom, S.B., Roshchina, V.: Comparing averaged relaxed cutters and projection methods: theory and examples. In From analysis to visualization, vol. 313 of Springer Proc. Math. Stat., pages 75–98. Springer, Cham (2020)
11. Douglas, J., Jr, Rachford, H.H., Jr.: On the numerical solution of heat conduction problems in two and three space variables. Trans. Am. Math. Soc. 82, 421–439 (1956)
12. Dunn, J.C.: Convergence rates for conditional gradient sequences generated by implicit step length rules. SIAM J. Control Optim. 18(5), 473–487 (1980)
13. Eckstein, J., Yao, W.: Relative-error approximate versions of Douglas-Rachford splitting and special cases of the ADMM, Math. Program. Ser. A 170, 417–444 (2018)
14. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Nav. Res. Log. 3, 95–110 (1956)
15. Freund, R.M., Grigas, P.: New analysis and results for the Frank-Wolfe method. Math. Program. 155(1–2, Ser. A), 199–230 (2016)
16. Fukushima, M., Luo, Z.-Q., Tseng, P.: Smoothing functions for second-order-cone complementarity problems. SIAM J. Optim. 12(2), 436–460 (2002)
17. Garber, D., Hazan, E.: Faster rates for the frank-wolfe method over strongly-convex sets. In: Proceedings of the 32Nd international conference on international conference on machine learning - Vol. 37, pp. 541–549 (2015)
18. Gonçalves, M.L.N., Melo, J.G.: A Newton conditional gradient method for constrained nonlinear systems. J. Comput. Appl. Math. 311, 473–483 (2017)
19. Gonçalves, M.L.N., Melo, J.G., Monteiro, R.D.C.: Projection-free accelerated method for convex optimization. Optim. Methods Softw. 37(1), 214–240 (2022). https://doi.org/10.1080/10556788.2020.1734806
20. Hesse, R., Luke, D.R., Neumann, P.: Alternating projections and Douglas-Rachford for sparse affine feasibility. IEEE Trans. Signal Process. 62(18), 4868–4881 (2014)
21. Jaggi, M.: Revisiting frank-wolfe: Projection-free sparse convex optimization. Proceedings of the 30th international conference on machine learning. Vol. 28, ICML’13:1–427–1–435 (2013)
22. Lan, G., Zhou, Y.: Conditional gradient sliding for convex optimization. SIAM J. Optim. 26(2), 1379–1409 (2016)
23. Levitin, E.S., Poljak, B.T.: Minimization methods in the presence of constraints. USSR Comput. Math. Math. Phys. 6, 1–50 (1966)
24. Nocedal, J., Wright, S.J.: Numerical optimization. Springer Series in Operations Research and Financial Engineering, 2nd edn. Springer, New York (2006)
25. Marques Alves, M., Geremia, Marina: Iteration complexity of an inexact Douglas-Rachford method and of a Douglas-Rachford-Tseng’s F-B four-operator splitting method for solving monotone inclusions. Numer. Algorithms 82, 263–295 (2019)
26. Marques Alves, M., Eckstein, Jonathan, Geremia, Marina, Melo, Jefferson G.: Relative-error inertial-relaxed inexact versions of Douglas-Rachford and ADMM splitting algorithms. Comput. Optim. Appl. 75, 389–422 (2020)
27. jarrett Revels https://github.com/JuliaCI/BenchmarkTools.jl
28. Rothvoss, T.: The matching polytope has exponential extension complexity. J. ACM 64(6), Art. 41, 19 (2017)

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.