Efficient Projective Methods for the Split Feasibility Problem and its Applications to Compressed Sensing and Image Deblurring

Suparat Kesornprom\textsuperscript{a}, Nattawut Pholasa\textsuperscript{b}, Prasit Cholamjiak\textsuperscript{a}

\textsuperscript{a}School of Science, University of Phayao, Phayao 56000, Thailand

Abstract. In this paper, new projective algorithms using linesearch technique are proposed to solve the split feasibility problem. Weak convergence theorems are established, under suitable conditions, in a real Hilbert space. Some numerical experiments in compressed sensing and image debluring are also provided to show its implementation and efficiency. The main results improve the corresponding results in the literature.

1. Introduction

Let $C$ and $Q$ be nonempty, closed and convex subsets of real Hilbert spaces $H_1$ and $H_2$, respectively. In this work, we aim to study the split feasibility problem (SFP) which is to find

$$x^* \in C \text{ such that } Ax^* \in Q$$

(1)

where $A : H_1 \to H_2$ is a bounded linear operator. This problem was introduced and studied by Censor and Elfving [9] in Euclidean spaces. Censor et al. in Section 2 of [6] (see also [18]) introduced the prototypical Split Inverse Problem (SIP). In this, there are given two vector spaces $X$ and $Y$ and a linear operator $A : X \to Y$. In addition, two inverse problems are involved. The first one, denoted by IP1, is formulated in the space $X$ and the second one, denoted by IP2, is formulated in the space $Y$. Given these data, the Split Inverse Problem is formulated as follows: find a point $x^* \in X$ that solves IP1 and such that the point $y^* = Ax^* \in Y$ solves IP2.

In recent years the Nonlinear Split Feasibility Problem (NLSFP) gained a lot of interest, see e.g., [20, 24]. In addition, the non-convex case is also very attractive from the application point of view, see [22].

In what follows, we denote by $A^*$ the adjoint operator of $A$. Let

$$f(x) = \frac{1}{2}|| (I - P_Q) A(x) ||^2$$

(2)
be an objective function and consider the constrained convex minimization problem:

$$\min_{x \in C} f(x).$$

(3)

The split feasibility problem (SFP) is equivalent to constrained convex minimization problem (3).

Since the function $f$ in (2) is differentiable, it is known that $\nabla f = A'(I - P_Q)A$ and $x$ solves (3) if and only if

$$x = P_C(x - a\nabla f(x)), \quad a > 0.$$  

(4)

This suggests a simple iterative method which is called the projected gradient method for solving (3). It is defined by

$$x_{k+1} = P_C(x_k - \alpha_k \nabla f(x_k)),$$

(5)

where $\{\alpha_k\}$ is a positive real sequence.

Korpelevich [17] and Antipin [1] proposed the following extragradient method for solving (3):

$$y_k = P_C(x_k - \alpha_k \nabla f(x_k)),$$

$$x_{k+1} = P_C(x_k - \alpha_k \nabla f(y_k)),$$

(6)

where $\{\alpha_k\}$ is a real sequence in $(0, \frac{1}{2})$ and $L$ is a Lipschitz constant of $\nabla f$.

In 2000, Tseng [32] introduced the following modified extragradient method:

$$y_k = P_C(x_k - \alpha_k \nabla f(x_k)),$$

$$x_{k+1} = y_k + \alpha_k (\nabla f(x_k) - \nabla f(y_k)),$$

(7)

where $\{\alpha_k\}$ is a real sequence in $(0, \frac{1}{2})$ and $L$ is a Lipschitz constant of $\nabla f$.

The SFP relates to various problems in applied sciences such as signal recovery, image restoration, LASSO problem, linear equations and others. Due to its applications, there have been many algorithms proposed for solving (1). See, for examples, [6, 7, 10, 14–16, 19, 30, 31].

Throughout this paper, we define $F : H_1 \to H_1$ by

$$F(x) = A'(I - P_Q)A(x).$$

(8)

We next recall some well-known algorithms that can be employed for solving (1). Byrne [4, 5] suggested the CQ algorithm which is defined by the following way: $x_1 \in H_1$ and

$$x_{k+1} = P_C(x_k - \alpha_k F(x_k)),$$

(9)

where $\alpha_k \in (0, 2/L)$ and $L$ is the spectral radius of $A^*A$. The notations $P_C$ and $P_Q$ stand for the projections of $H_1$ onto $C$ and $H_2$ onto $Q$, respectively. In practice, the sets $C$ and $Q$ are usually defined by

$$C = \{x \in H_1 : c(x) \leq 0\},$$

(10)

where $c : H_1 \to \mathbb{R}$ is a convex and lower semicontinuous function and

$$Q = \{y \in H_2 : q(y) \leq 0\},$$

(11)

where $q : H_1 \to \mathbb{R}$ is a convex and lower semicontinuous function.

In 2004, Yang [35] established a relaxed CQ algorithm for solving the SFP. The idea of this method is to replace $P_C$ and $P_Q$ by projections onto half spaces $C_k$ and $Q_k$. Here the sets $C_k$ and $Q_k$ are defined by

$$C_k = \{x \in H_1 : c(x_k) + \langle \xi_k, x - x_k \rangle \leq 0\},$$

(12)
where $\xi_k \in \partial c(x_k)$, and
\[
Q_k = \{ y \in H_2 : q(Ax_k) + \langle \eta_k, y - Ax_k \rangle \leq 0 \},
\] (13)
where $\eta_k \in \partial q(Ax_k)$.

Define $F_k : H_1 \to H_1$ by
\[
F_k(x) = A^*(I - P_{Q_k})A(x).
\] (14)

Precisely, Yang [35] introduced the following relaxed CQ algorithm.

**Algorithm 1.** Let $x_1 \in H_1$ and define
\[
x_{k+1} = P_C(x_k - \alpha_k F_k(x_k))
\] (15)
where $\alpha_k \in (0, 2/L)$.

However, the stepsizes in CQ algorithm (9) and relaxed CQ algorithm (15) depend on the spectral radius of $A^*A$. We note that to compute the spectral radius is difficult in general and this usually results in slow convergence.

Recently, Qu and Xiu [28] modified Yang’s relaxed CQ algorithm by using the Armijo-line searches in Euclidean spaces. Later, Gibali et al. [19] proposed the relaxtion CQ algorithm in Hilbert spaces for solving the SFP. It is defined by the following manner:

**Algorithm 2.** For any $\sigma > 0$, $\rho \in (0, 1)$ and $\mu \in (0, 1)$. Let $x_1 \in H_1$ and define
\[
y_k = P_C(x_k - \alpha_k F_k(x_k))
\] (16)
where $\alpha_k = \sigma \rho^m_k$ and $m_k$ is the smallest nonnegative integer such that
\[
\alpha_k \| F_k(x_k) - F_k(y_k) \| \leq \mu \| x_k - y_k \|.
\] (17)
Define
\[
x_{k+1} = P_C(x_k - \alpha_k F_k(y_k)).
\] (18)

Gibali et al. [19] proved that the sequence $\{x_n\}$ generated by Algorithm 2 converges weakly to a solution of SFP.

In 2012, Zhao et al. [37] introduced the modified CQ algorithm to solve the SFP as follows:

**Algorithm 3.** Let $x_1 \in H_1$, $\alpha_0 > 0$, $\rho \in (0, 1)$, $\mu \in (0, 1)$, $\beta \in (0, 1)$ and let
\[
y_k = P_C(x_k - \alpha_k F_k(x_k))
\] (19)
where $\alpha_k = \alpha \rho^m_k$ and $m_k$ is the smallest nonnegative integer such that
\[
\alpha_k \| F(x_k) - F(y_k) \| \leq \mu \| x_k - y_k \|.
\] (20)
Define
\[
x_{k+1} = P_C(y_k - \alpha_k (F(y_k) - F(x_k))).
\] (21)

If
\[
\alpha_k \| F(x_{k+1}) - F(x_k) \| \leq \beta \| x_{k+1} - x_k \|
\] (22)
then set $\alpha_k = \alpha_0$, otherwise, set $\alpha_k = \alpha_k$.

Very recently, Dong et al. [12] proposed the modified projection and contraction methods and the relaxation variants to solve the SFP as follows:
Algorithm 4. For any $\sigma > 0$, $\rho \in (0, 1)$ and $\mu \in (0, 1)$, take arbitrarily $x_1 \in H_1$ and let

$$y_k = P_C(x_k - \alpha_k F(x_k))$$

where $\alpha_k = \sigma \rho^m$ and $m_k$ is the smallest nonnegative integer such that

$$\alpha_k \|F(x_k) - F(y_k)\| \leq \mu \|x_k - y_k\|.$$  

(24)

Define

$$x_{k+1} = x_k - \gamma \delta_k d(x_k, y_k)$$

(25)

where $\gamma \in (0, 2)$

$$d(x_k, y_k) = (x_k - y_k) - \alpha_k (F(x_k) - F(y_k))$$

(26)

and

$$\delta_k = \frac{\langle x_k - y_k, d(x_k, y_k) \rangle + \alpha_k \|I - P_Q A(y_k)\|^2}{\|d(x_k, y_k)\|^2}.$$  

(27)

They also provided some numerical experiments that show the efficiency of the proposed algorithm.

In this paper, inspired by the previous works, we propose a modification of CQ algorithm and relaxed CQ algorithm to solve the split feasibility problem. We then prove the weak convergence of this algorithm in real Hilbert spaces. Our result mainly improves the results of Dong et al. [12] and others. Some preliminary experiments are also given in compressed sensing and image deblurring to show its implementation and efficiency.

2. Preliminaries and lemmas

In this section, we give some definitions and lemmas which are used in the main results. Let $H$ be a real Hilbert space and $C$ be a nonempty subset of $H$.

1. A mapping $T : C \to C$ is said to be firmly nonexpansive if, for all $x, y \in C$,

$$\langle x - y, Tx - Ty \rangle \geq \|Tx - Ty\|^2.$$  

(28)

2. A function $f : H \to \mathbb{R}$ is said to be convex if

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda) f(y)$$

for all $\lambda \in (0, 1)$ and $x, y \in H$.

3. $F$ is said to be monotone on $C$ if

$$\langle F(x) - F(y), x - y \rangle \geq 0, \ \forall x, y \in C.$$  

(30)

4. $F$ is said to be $\tau_n$-inverse strongly monotone (shortly, $\tau_n$-ism) with $\tau_n > 0$ if

$$\langle F(x) - F(y), x - y \rangle \geq \tau_n \|F(x) - F(y)\|^2, \ \forall x, y \in C.$$  

(31)

5. $F$ is said to be Lipschitz continuous on $C$ with constant $\lambda > 0$ if

$$\|F(x) - F(y)\| \leq \lambda \|x - y\|, \ \forall x, y \in C.$$  

(32)

6. A mapping $f : C \to C$ is said to be a contraction if there exists a constant $a \in (0, 1)$ such that

$$\|f(x) - f(y)\| \leq a \|x - y\|, \ \forall x, y \in C.$$  

(33)
(7) A differentiable function $f$ is convex if and only if there holds the inequality:
\[ f(z) \geq f(x) + \langle \nabla f(x), z - x \rangle \]
for all $z \in H$.

(8) An element $g \in H$ is called a subgradient of $f : H \to \mathbb{R}$ at $x$ if
\[ f(z) \geq f(x) + \langle g, z - x \rangle \]
for all $z \in H$, which is called the subdifferentiable inequality.

(9) A function $f : H \to \mathbb{R}$ is said to be subdifferentiable at $x$ if it has at least one subgradient at $x$.

(10) The set of subgradients of $f$ at the point $x$ is called the subdifferentiable of $f$ at $x$, which is denoted by $\partial f(x)$.

(11) A function $f$ is said to be subdifferentiable if it is subdifferentiable at all $x \in H$. If a function $f$ is differentiable and convex, then its gradient and subgradient coincide.

(12) A function $f : H \to \mathbb{R}$ is said to be weakly lower semi-continuous (shortly, w-lsc) at $x$ if $x_n \rightharpoonup x$ implies $f(x) \leq \liminf_{n \to \infty} f(x_n)$.

We know that the orthogonal projection $P_C$ from $H$ onto a nonempty closed convex subset $C \subset H$ is a typical example of a firmly nonexpansive mapping, which is defined by
\[ P_C x := \arg \min_{y \in C} \| x - y \|^2 \]
for all $x \in H$.

**Lemma 1.** [3] Let $C$ be a nonempty closed convex subset of a real Hilbert space $H$. Then, for any $x \in H$, the following assertions hold:

1. $(x - P_C x, z - P_C x) \leq 0$ for all $z \in C$;
2. $\|P_C x - P_C y\|^2 \leq \langle P_C x - P_C y, x - y \rangle$ for all $x, y \in H$;
3. $\|P_C x - z\|^2 \leq \|x - z\|^2 - \|P_C x - x\|^2$ for all $z \in C$.

From Lemma 1, the operator $I - P_C$ is also firmly nonexpansive, where $I$ denotes the identity operator, i.e., for any $x, y \in H$,
\[ \langle (I - P_C) x - (I - P_C) y, x - y \rangle \geq \|(I - P_C) x - (I - P_C) y\|^2. \]

**Lemma 2.** [21] Let $C$ be a nonempty subset of a real Hilbert space $H$ and $\{x_n\}$ be a sequence in $H$ that satisfies the following properties:

1. $\lim_{n \to \infty} \|x_n - x\|$ exists for each $x \in C$;
2. every sequential weak limit point of $\{x_n\}$ is in $C$.

Then $\{x_n\}$ converges weakly to a point in $C$.

### 3. The modified projection and contraction methods

In this section, we introduce a projection algorithm using linesearch and prove the weak convergence. Assume that the SFP (1) is consistent, i.e. its solution set, denoted by $S$, is nonempty.

**Algorithm 5.** Set $\sigma > 0$, $\rho \in (0, 1)$ and $\mu \in (0, \frac{1}{2})$. Choose $x_1 \in H_1$ and define
\[ y_k = P_C(x_k - \alpha_k F(x_k)) \]
where \( \alpha_k = \alpha \rho^m \) and \( m_k \) is the smallest nonnegative integer such that

\[
\alpha_k \|F(x_k) - F(y_k)\| \leq \mu \|x_k - y_k\|. \quad (40)
\]

Define

\[
x_{k+1} = y_k - \gamma \delta_k d(x_k, y_k)
\]

where \( \gamma \in (0, 2) \)

\[
d(x_k, y_k) = (x_k - y_k) - \alpha_k (F(x_k) - F(y_k))
\]

and

\[
\delta_k = \frac{\alpha_k \| (I - P_C) Ay_k \|^2}{\gamma \|d(x_k, y_k)\|^2}.
\]

Remark 1. If \( d(x_k, y_k) = 0 \), then

\[
\langle x_k - y_k - \alpha_k (F(x_k) - F(y_k)), x_k - y_k \rangle = 0.
\]

From (44), it follows that

\[
\|x_k - y_k\|^2 = \alpha_k \langle F(x_k) - F(y_k), y_k - x_k \rangle
\leq \alpha_k \|F(x_k) - F(y_k)\| \|x_k - y_k\|
\leq \mu \|x_k - y_k\|^2,
\]

which gives

\[
x_k = y_k, \forall k \geq 0
\]

From definition of \( y_k \), we see that

\[
x_k = P_C(x_k - \alpha_k F(x_k))
\]

Hence, \( x_k = y_k \) is a solution.

Lemma 3. [36] The line rule (40) is well defined. Besides, \( \alpha' \leq \alpha_k \leq \sigma \), where \( \alpha' = \min\{\sigma, \frac{\mu}{\gamma} \} \).

This lemma shows that the linesearch (40) has a finite number of iteration for \( \alpha_k \).

Theorem 1. The sequence \( \{x_k\} \) generated by Algorithm 5 weakly converges to a solution in \( S \).

Proof. Let \( z \in S \). Then \( z = P_C(z) \) and \( Az = P_C(Az) \). It follows that

\[
\|x_{k+1} - z\|^2 = \|y_k - \gamma \delta_k d(x_k, y_k) - z\|^2
\leq \|y_k - z\|^2 - 2\gamma \delta_k \langle y_k - z, d(x_k, y_k) \rangle + \gamma^2 \delta_k^2 \|d(x_k, y_k)\|^2.
\]

By the definitions of \( y_k \) and \( d(x_k, y_k) \), we get

\[
y_k = P_C(y_k - (\alpha_k F(y_k) - d(x_k, y_k))).
\]

From Lemma 1 (1), it follows that

\[
\langle x - y_k, \alpha_k F(y_k) - d(x_k, y_k) \rangle \geq 0, \forall x \in C.
\]

Setting \( x = z \) in (56), we have

\[
\langle y_k - z, d(x_k, y_k) - \alpha_k F(y_k) \rangle \geq 0
\]

\[
\langle y_k - z, d(x_k, y_k) \rangle \geq 0.
\]
which implies that

\[
\langle y_k - z, d(x_k, y_k) \rangle \geq \alpha_k \langle y_k - z, F(y_k) \rangle.
\] (52)

Since \( F(y_k) = A^*(I - P_Q)y_k \) and \( Az = P_Q(Az) \), it follows that

\[
\alpha_k \langle y_k - z, F(y_k) \rangle = \alpha_k \langle y_k - z, A^*(I - P_Q)y_k \rangle
\]
\[
= \alpha_k \langle Ay_k - Az, (I - P_Q)y_k \rangle
\]
\[
= \alpha_k \langle Ay_k - Az, (I - P_Q)y_k - (I - P_Q)Az \rangle
\]
\[
\geq \alpha_k \| (I - P_Q)y_k \|^2,
\] (53)

where the last inequality follows by the firm nonexpansiveness of \( I - P_Q \). By Lemma 1(3), we have

\[
\| y_k - z \|^2 = \| P_C(x_k - \alpha_k F(x_k)) - z \|^2
\]
\[
\leq \| x_k - \alpha_k F(x_k) - z \|^2 - \| y_k - x_k + \alpha_k F(x_k) \|^2
\]
\[
= \| x_k - z \|^2 - 2\alpha_k \langle x_k - z, F(x_k) \rangle + \alpha_k^2 \| F(x_k) \|^2 - \| y_k - x_k \|^2
\]
\[
-2\alpha_k \langle y_k - x_k, F(x_k) \rangle - \alpha_k^2 \| F(x_k) \|^2
\]
\[
= \| x_k - z \|^2 - 2\alpha_k \langle x_k - z, F(x_k) \rangle - \| y_k - x_k \|^2 - 2\alpha_k \langle y_k - x_k, F(x_k) \rangle.
\] (54)

Since \( F(z) = 0 \) and \( I - P_Q \) is firmly nonexpansive, it also follows that

\[
2\alpha_k \langle x_k - z, F(x_k) \rangle = 2\alpha_k \langle x_k - z, F(x_k) - F(z) \rangle
\]
\[
= 2\alpha_k \langle x_k - z, A^*(I - P_Q)x_k - A^*(I - P_Q)Az \rangle
\]
\[
= 2\alpha_k \langle Ax_k - Az, (I - P_Q)x_k - (I - P_Q)Az \rangle
\]
\[
\geq 2\alpha_k \| (I - P_Q)x_k \|^2.
\] (55)

On the other hand, using (34), we obtain

\[
2\alpha_k \langle y_k - x_k, F(x_k) \rangle = 2\alpha_k \langle y_k - x_k, F(x_k) - F(y_k) + F(y_k) \rangle
\]
\[
= 2\alpha_k \langle y_k - x_k, F(x_k) - F(y_k) \rangle + 2\alpha_k \langle y_k - x_k, F(y_k) \rangle
\]
\[
\geq -2\alpha_k \| y_k - x_k \| \| F(x_k) - F(y_k) \|
\]
\[
+ 2\alpha_k \frac{1}{2} (\| (I - P_Q)y_k \|^2 - \| (I - P_Q)Ax_k \|^2)
\]
\[
= -2\alpha_k \| y_k - x_k \| \| F(x_k) - F(y_k) \|
\]
\[
+ \alpha_k \| (I - P_Q)y_k \|^2 - \alpha_k \| (I - P_Q)Ax_k \|^2.
\] (56)

Combining (54)-(56), we obtain

\[
\| y_k - z \|^2 \leq \| x_k - z \|^2 - 2\alpha_k \| (I - P_Q)x_k \|^2 - \| y_k - x_k \|^2 + 2\alpha_k \| y_k - x_k \| \| F(x_k) - F(y_k) \|
\]
\[
-2\alpha_k \| (I - P_Q)y_k \|^2 + \alpha_k \| (I - P_Q)Ax_k \|^2.
\] (57)
From (40), (48) and Lemma 3, we have
\[
\|x_{k+1} - z\|^2 \leq \|x_k - z\|^2 - 2\alpha_k \|I - P_QAx_i\|^2 - \|y_k - x_i\|^2 + 2\mu \|y_k - x_i\|^2 \\
-\alpha_k \|I - P_QAy_i\|^2 + \alpha_k \|I - P_QAx_i\|^2 - 2\gamma \delta \alpha_k \|I - P_QAy_i\|^2 \\
+\gamma^2 \delta^2 \|d(x_k, y_k)\|^2
\]
\[
= \|x_k - z\|^2 - \alpha_k \|I - P_QAx_i\|^2 - (1 - 2\mu) \|y_k - x_i\|^2 - \alpha_k \|I - P_QAy_i\|^2 \\
-2\gamma \delta \alpha_k \|I - P_QAx_i\|^2 + \gamma \delta \|d(x_k, y_k)\|^2
\]
\[
= \|x_k - z\|^2 - \alpha_k \|I - P_QAx_i\|^2 - (1 - 2\mu) \|y_k - x_i\|^2 \\
-\gamma \delta \|d(x_k, y_k)\|^2 - 2\gamma \delta \alpha_k \|I - P_QAy_i\|^2
\]
\[
\leq \|x_k - z\|^2 - \frac{\mu \ell}{L} \|I - P_QAx_i\|^2 - (1 - 2\mu) \|y_k - x_i\|^2 \\
-2\gamma \delta \frac{\mu \ell}{L} \|I - P_QAy_i\|^2
\]
\[
\leq \|x_k - z\|^2 - \frac{\mu \ell}{L} \|I - P_QAx_i\|^2 - (1 - 2\mu) \|y_k - x_i\|^2. 
\]
(58)

This shows that the sequence \(||x_k - z||\) is decreasing and thus converges to a point in \(H_1\). Hence \(\{x_i\}\) is bounded. From (58), we see that
\[
\lim_{k \to \infty} \|y_k - x_i\| = 0 
\]
(59)
and
\[
\lim_{k \to \infty} \|I - P_QAx_i\| = 0. 
\]
(60)

Since the sequence \(\{x_i\}\) is bounded, there is a cluster point \(x^*\) of \(\{x_i\}\) with a subsequence \(\{x_k\}\) weakly converging to \(x^*\). From (59), it follows that \(\{x_k\}\) also weakly converges to \(x^*\).

Next, we show that \(x^*\) is in \(S\). From (60) and the boundedness of \(\{x_k\}\), it implies that \(Ax^* \in Q\). From (8) and (60), it is easy to see that \(\lim \|F(x_k)\| = 0\). By (39) and (59), we also have
\[
\|x_k - P_C(x_k)\| \leq \|x_k - y_k\| + \|y_k - P_C(x_k)\| \\
\leq \|x_k - y_k\| + \alpha_k \|F(x_k)\| \\
\rightarrow 0, \text{ as } i \to \infty, 
\]
(61)

which implies \(x^* \in C\). So \(x^*\) is in \(S\). Hence, we can conclude that the sequence \(\{x_i\}\) weakly converges to a point in \(S\). This completes the proof. \(\square\)

4. The modified relaxation projection and contraction methods

In this section, we introduce the modified relaxation projection and contraction methods.

To this end, we assume that the sets \(C\) and \(Q\) satisfy the following conditions:

The set \(C\) is given by
\[
C = \{x \in H_1 : c(x) \leq 0\}, 
\]
(62)
where $c : H_1 \to \mathbb{R}$ is a convex and lower semicontinuous function and $C$ is a nonempty set. The set $Q$ is given by

$$Q = \{ y \in H_2 : q(y) \leq 0 \}, \quad \text{(63)}$$

where $q : H_2 \to \mathbb{R}$ is a convex and lower semicontinuous function and $Q$ is a nonempty set. Assume that $c$ and $q$ are subdifferentiable on $C$ and $Q$, respectively, and $c$ and $q$ are bounded on bounded sets. Note that this condition is automatically satisfied in finite dimensional spaces.

For any $x \in H_1$, at least one subgradient $\xi \in \partial c(x)$ can be calculated, where $\partial c(x)$ is defined as follows:

$$\partial c(x) = \{ z \in H_1 : c(u) \geq c(x) + \langle u - x, z \rangle, \forall u \in H_1 \}. \quad \text{(64)}$$

For any $y \in H_2$, at least one subgradient $\eta \in \partial q(y)$ can be calculated, where

$$\partial q(x) = \{ w \in H_2 : q(u) \geq q(y) + \langle v - y, w \rangle, \forall v \in H_2 \}. \quad \text{(65)}$$

Define the sets $C_k$ and $Q_k$ by the following half-spaces:

$$C_k = \{ x \in H_1 : c(x_k) + \langle \xi_k, x - x_k \rangle \leq 0 \}, \quad \text{(66)}$$

where $\xi_k \in \partial c(x_k)$, and

$$Q_k = \{ y \in H_2 : q(Ax_k) + \langle \eta_k, y - Ax_k \rangle \leq 0 \}, \quad \text{(67)}$$

where $\eta_k \in \partial q(Ax_k)$.

By the definition of the subgradient, it is clear that $C \subseteq C_k$ and $Q \subseteq Q_k$. The projections onto $C_k$ and $Q_k$ are easy to compute since $C_k$ and $Q_k$ are two half-spaces.

**Algorithm 6.** For any constants $\sigma > 0$, $\rho \in (0, 1)$ and $\mu \in (0, \frac{1}{2})$, let $x_1$ be arbitrarily in $H_1$ and define

$$y_k = P_{C_k}(x_k - \alpha_k F_k(x_k)) \quad \text{(68)}$$

where $\alpha_k = \sigma \rho^m$ and $m_k$ is the smallest nonnegative integer such that

$$\alpha_k \| F_k(x_k) - F_k(y_k) \| \leq \mu \| x_k - y_k \|. \quad \text{(69)}$$

Define

$$x_{k+1} = y_k - \gamma \delta_k d(x_k, y_k) \quad \text{(70)}$$

where $\gamma \in (0, 2)$,

$$d(x_k, y_k) = \| x_k - y_k - \alpha_k F_k(x_k) - F_k(y_k) \| \quad \text{(71)}$$

and

$$\delta_k = \frac{\alpha_k \| (I - P_{Q_k}) Ay_k \| ^2}{\gamma \| d(x_k, y_k) \| ^2}. \quad \text{(72)}$$

**Theorem 2.** The sequence $\{ x_k \}$ generated by Algorithm 6 weakly converges to a solution in $S$.

**Proof.** Following the lines of the proof of Theorem 1, we can show that

$$\| x_{k+1} - z \|^2 \leq \| x_k - z \|^2 - \frac{\mu \ell}{L} \| (I - P_{Q_k}) Ax_k \|^2 - (1 - 2\mu) \| y_k - x_k \|^2 - 2\gamma \delta_k \frac{\mu \ell}{L} \| (I - P_{Q_k}) Ay_k \|^2 \leq \| x_k - z \|^2 - \frac{\mu \ell}{L} \| (I - P_{Q_k}) Ax_k \|^2 - (1 - 2\mu) \| y_k - x_k \|^2. \quad \text{(73)}$$
Moreover, we also have
\[
\lim_{k \to \infty} \| y_k - x_k \| = 0 \quad (74)
\]
and
\[
\lim_{k \to \infty} \| (I - P_{Q_k}) Ax_k \| = 0. \quad (75)
\]
Let \( x^* \) be a cluster point of \( \{ x_k \} \) with \( \{ x_k \} \) converging to \( x^* \). From (74), it follows that \( \{ x_k \} \) also weakly converges to \( x^* \).

Next, we show that \( x^* \) is in \( S \). In fact, since \( y_{k_i} \in C_{k_i} \), by the definition of \( C_{k_i} \), we have
\[
c(x_{k_i}) + \langle \xi_{k_i}, y_{k_i} - x_{k_i} \rangle \leq 0, \quad (76)
\]
where \( \xi_{k_i} \in \partial c(x_{k_i}) \). By the assumption that \( \xi_{k_i} \) is bounded and (74), we have
\[
c(x_{k_i}) \leq -\langle \xi_{k_i}, y_{k_i} - x_{k_i} \rangle \\
\leq \| \xi_{k_i} \| \| y_{k_i} - x_{k_i} \| \\
\to 0 \text{ as } i \to \infty,
\]
which implies \( c(x^*) \leq 0 \) by the lower semicontinuity of \( C \). Hence \( x^* \in C \). Since \( P_{Q_k}(Ax_{k_i}) \in Q_{k_i} \), we also have
\[
q(Ax_{k_i}) + \langle \eta_{k_i}, P_{Q_k}(Ax_{k_i}) - Ax_{k_i} \rangle \leq 0, \quad (78)
\]
where \( \eta_{k_i} \in \partial q(Ax_{k_i}) \). From the boundedness of \( \{ \eta_{k_i} \} \) and (75), it follows that
\[
q(Ax_{k_i}) \leq \| \eta_{k_i} \| \| P_{Q_k}(Ax_{k_i}) - Ax_{k_i} \| \to 0 \quad (79)
\]
as \( i \to \infty \). So we obtain \( q(Ax^*) \leq 0, \text{ i.e., } Ax^* \in Q \). Thus \( x^* \) is in \( S \). By Lemma 2, we conclude that \( \{ x_k \} \) weakly converges to a point in \( S \). We thus complete the proof. \( \Box \)

5. Application to signal recovery

In this section, we test our algorithm to show the efficiency in compressed sensing in frequency domain.

In signal processing, compressed sensing can be modeled as the following under determined linear equation system:
\[
y = Ax + \varepsilon, \quad (80)
\]
where \( x \in \mathbb{R}^N \) is a vector with \( m \) nonzero components to be recovered, \( y \in \mathbb{R}^M \) is the observed or measured data with noisy \( \varepsilon \), and \( A : \mathbb{R}^N \to \mathbb{R}^M \) \((M < N)\) is a bounded linear observation operator. Finding the solutions of (80) can be seen as solving the LASSO problem [33]
\[
\min_{x \in \mathbb{R}^N} \frac{1}{2} \| y - Ax \|_2^2 \text{ subject to } \| x \|_1 \leq t, \quad (81)
\]
where \( t > 0 \) is a given constant. In particular, if \( C = \{ x \in \mathbb{R}^N : \| x \|_1 \leq t \} \) and \( Q = \{ y \} \), then the LASSO problem can be considered as the SFP.

In this experiment, the sparse vector \( x \in \mathbb{R}^N \) is generated by the uniform distribution in the interval \([-2, 2]\) with \( m \) nonzero elements. The matrix \( A \in \mathbb{R}^{M \times N} \) is generated by the normal distribution with mean zero and variance one. The observation \( y \) is generated by white Gaussian noise with signal-to-noise ratio \( \text{SNR}=40 \). The process is started with \( t = m \) and initial point \( x_1 \) is picked randomly.
The restoration accuracy is measured by the mean squared error as follows:

$$E_k = \frac{1}{N} \| x_k - x \|_2^2 < \varepsilon,$$

(82)

where \( x_k \) is an estimated signal of \( x \) and \( \varepsilon \) is a given error.

We give some numerical results of Algorithms 1, 2, 4 and 6. Let \( \sigma = 3 \), \( \rho = 0.9 \), \( \gamma = 1.8 \) and \( \mu = 0.4 \). In this numerical experiment, we use Matlab R2018b to write all codes.

We test four cases as follow:

Case 1: \( N = 512 \), \( M = 256 \) and \( m = 10 \);
Case 2: \( N = 1024 \), \( M = 512 \) and \( m = 30 \);
Case 3: \( N = 2048 \), \( M = 1024 \) and \( m = 50 \);
Case 4: \( N = 4096 \), \( M = 2048 \) and \( m = 100 \).

The numerical results are reported as follows.

Table 1: Computational results to recover the signal

| Methods   | \( \varepsilon = 10^{-3} \) | \( \varepsilon = 10^{-4} \) |
|-----------|-------------------------------|-------------------------------|
|           | Iter | CPU | Iter | CPU | Iter | CPU |
| Case 1    |      |     |      |     |      |     |
| Algorithm 1 | 28   | 0.3973 | 84   | 1.3127 |
| Algorithm 2 | 30   | 0.5643 | 86   | 1.4148 |
| Algorithm 4 | 27   | 0.5813 | 36   | 0.6383 |
| Algorithm 6 | 17   | 0.3389 | 29   | 0.6335 |
| Case 2    |      |     |      |     |      |     |
| Algorithm 1 | 49   | 3.7598 | 95   | 7.7859 |
| Algorithm 2 | 51   | 7.2222 | 99   | 14.9803 |
| Algorithm 4 | 32   | 4.6847 | 47   | 8.0774 |
| Algorithm 6 | 22   | 3.6419 | 34   | 5.8058 |
| Case 3    |      |     |      |     |      |     |
| Algorithm 1 | 41   | 30.0557 | 94   | 115.8860 |
| Algorithm 2 | 43   | 36.1808 | 94   | 93.0088 |
| Algorithm 4 | 30   | 25.3104 | 37   | 37.1473 |
| Algorithm 6 | 20   | 17.0654 | 29   | 29.5174 |
| Case 4    |      |     |      |     |      |     |
| Algorithm 1 | 39   | 248.8680 | 71   | 465.5332 |
| Algorithm 2 | 41   | 147.2881 | 74   | 264.2121 |
| Algorithm 4 | 32   | 118.6846 | 60   | 215.0726 |
| Algorithm 6 | 19   | 74.8502 | 34   | 122.7263 |

In Table 1, we see that our Algorithm 6 has a less number of iterations and CPU time than Algorithms 1, 2 and 4 do in each cases.

Next, we show the graphs of original signal and recovered signal by Algorithms 1, 2, 4 and 6 when \( N = 512 \), \( M = 256 \), \( m = 10 \) and \( \varepsilon = 10^{-3} \). The number of iterations and CPU time are reported in the figures.
Figure 1: From top to bottom: original signal, observation data, recovered signal by Algorithms 1, 2, 4 and 6 in Case 1, respectively.
We next show the graphs of signal recovery by Algorithms 1, 2, 4 and 6 when $N = 4096$, $M = 2048$, $m = 100$ and $\epsilon = 10^{-4}$.

Figure 2: From top to bottom: original signal, observation data, recovered signal by Algorithms 1, 2, 4 and 6 in Case 4, respectively.
We next show the error plotting of Algorithms 1, 2, 4 and 6 in Case 1 and Case 4.

![Figure 3: $E_k$ versus number of iterations in Case 1](image1)

![Figure 4: $E_k$ versus number of iterations in Case 4](image2)

From Figure 3 and 4, we observe that Algorithms 1, 2, 4 and 6 can be applied to signal recovery problem. Also, we note that Algorithm 6 has a good performance for this problem. It requires a small number of iterations and CPU time in numerical comparison.

6. Application to image restoration

As mentioned earlier that SFP can apply to many real-world problems. In this section, we present an application to image restoration problems using our main result. We provide some comparisons to other algorithms.

For a RGB scale image of $M$ pixels wide by $N$ pixels height, each pixel value is known to range from 0 to 255. Let $D = M \times N$. Then the underlying real Hilbert space is $\mathbb{R}^D$ equipped with the standard Euclidean
norm $\| \cdot \|_2$, and let $C = [0, 255]^D$. In order to estimate an approximation of the vector $x$, which represents the image of the original image scene, we consider the convex minimization model:

$$\min_{x \in C} \| Ax - y \|_2.$$  \hspace{1cm} (83)

By choosing $Q = \{ y \}$, the problem (83) can be seen as SFP (1). Therefore, we can apply our algorithm to solve image restoration problem.

In this numerical experiment, we use Matlab R2018b to write all codes. To determine the efficiency of algorithms, we need an image quality measure of restored images. We define the Peak Signal-to-Noise Ratio (SNR) in decibel (dB) as follows:

$$\text{PSNR} = 20 \log_{10} \frac{\| \bar{x} \|_2}{\| x - \bar{x} \|_2},$$  \hspace{1cm} (84)

where $\bar{x}$ is an original image and $x$ is a restored image. It can be observed that the larger PSNR values, the better restored images. To begin, set the initial point $x_0$ to be $0 \in \mathbb{R}^D$. Set all parameters by $\sigma = 0.1$, $\rho = 0.3$, $\mu = 0.01$ and $\gamma = 0.3$. Each image is degraded by a motion blur with a motion length 15, 30, 45, 60 and an angle 180. Then the numerical results are reported in Tables 2-4.
Table 2: Numerical comparison for Algorithms 3, 4 and 5 of Cat image (size=384 × 512) for each motion length.

| motion length | Iter | Algorithm 3 | Algorithm 4 | Algorithm 5 |
|---------------|------|-------------|-------------|-------------|
| 15 Red        | 500  | 16.3829     | 21.8416     | 28.4803     |
| 15.2780       | 1500 | 18.8747     | 28.0028     | 31.0664     |
|               | 2500 | 20.3751     | 29.4057     | 33.3359     |
| Green         | 500  | 16.6123     | 22.3260     | 30.0840     |
| 15.5079       | 1500 | 19.0836     | 28.5551     | 34.9486     |
|               | 2500 | 20.5847     | 29.8299     | 39.6045     |
| Blue          | 500  | 16.3846     | 22.1783     | 30.0731     |
| 15.2867       | 1500 | 18.8649     | 28.2531     | 32.8363     |
|               | 2500 | 20.3506     | 29.6004     | 36.1965     |
| 30 Red        | 500  | 14.3873     | 18.5316     | 23.2017     |
| 13.5449       | 1500 | 16.2169     | 23.1547     | 25.4769     |
|               | 2500 | 17.2081     | 24.3563     | 27.1030     |
| Green         | 500  | 14.5810     | 18.8749     | 24.1111     |
| 13.7345       | 1500 | 16.4350     | 23.4089     | 27.8878     |
|               | 2500 | 17.4299     | 24.6307     | 31.2110     |
| Blue          | 500  | 14.3615     | 18.7789     | 24.3106     |
| 13.5187       | 1500 | 16.1906     | 23.1145     | 30.3329     |
|               | 2500 | 17.1750     | 24.2668     | 34.9433     |
| 45 Red        | 500  | 13.3905     | 17.0191     | 23.5165     |
| 12.6151       | 1500 | 15.0767     | 22.9588     | 27.3520     |
|               | 2500 | 15.9724     | 24.6974     | 29.6749     |
| Green         | 500  | 13.5494     | 17.4840     | 24.8367     |
| 12.7667       | 1500 | 15.2525     | 23.7401     | 30.4577     |
|               | 2500 | 16.1504     | 25.6345     | 33.1714     |
| Blue          | 500  | 13.3591     | 17.5345     | 26.0984     |
| 12.5863       | 1500 | 15.0876     | 23.8176     | 30.0078     |
|               | 2500 | 15.9923     | 25.6284     | 31.9806     |
| 60 Red        | 500  | 12.7475     | 16.6084     | 20.9304     |
| 11.9738       | 1500 | 14.3126     | 20.6654     | 23.8163     |
|               | 2500 | 15.1075     | 22.0668     | 25.6994     |
| Green         | 500  | 12.8991     | 17.0952     | 21.6524     |
| 12.1172       | 1500 | 14.4934     | 21.1951     | 25.7100     |
|               | 2500 | 15.3059     | 22.6414     | 28.7144     |
| Blue          | 500  | 12.7236     | 16.9615     | 22.7128     |
| 11.9738       | 1500 | 14.2535     | 21.1349     | 27.0384     |
|               | 2500 | 15.0523     | 22.5753     | 29.3498     |
Table 3: Numerical comparison for Algorithms 3, 4 and 5 of Flower image (size=436 × 581) for each motion length.

| motion length | PSNR (dB) | Iter | Algorithm 3 | Algorithm 4 | Algorithm 5 |
|---------------|-----------|------|-------------|-------------|-------------|
| 15 Red 14.5043 | 500       | 15.8800 | 20.9102 | 28.3568 | 2.175908 |
|               | 1500      | 18.8107 | 27.8765 | 31.1837 | 31.7963 |
|               | 2500      | 20.3663 | 29.9398 | 32.7696 | 34.6631 |
| Green 13.5214 | 500       | 14.7885 | 22.3640 | 27.7539 | 25.8711 |
|               | 1500      | 17.6993 | 28.5551 | 30.7161 | 29.6034 |
|               | 2500      | 19.2332 | 29.8299 | 34.6631 | 30.4692 |
| Blue 14.5043  | 500       | 11.5805 | 22.1783 | 25.8711 | 23.9068 |
|               | 1500      | 14.7687 | 28.2531 | 28.7938 | 29.6034 |
|               | 2500      | 16.3425 | 29.6034 | 30.4692 | 30.4692 |
| 30 Red 11.6408| 500       | 12.6945 | 17.5511 | 23.9708 | 1.0205 |
|               | 1500      | 15.0474 | 23.6652 | 27.2366 | 25.7113 |
|               | 2500      | 16.2712 | 25.7113 | 28.5671 | 29.8185 |
| Green 10.8898 | 500       | 11.9130 | 17.2443 | 23.7777 | 22.0678 |
|               | 1500      | 14.2947 | 23.0851 | 27.8288 | 29.1728 |
|               | 2500      | 15.5008 | 25.1728 | 29.8185 | 30.4692 |
| Blue 7.9578  | 500       | 9.0205  | 14.2360 | 21.0311 | 25.3762 |
|               | 1500      | 11.3884 | 20.4351 | 23.9366 | 25.3762 |
|               | 2500      | 12.6548 | 22.6948 | 25.3762 | 25.3762 |
| 45 Red 10.0959| 500       | 11.2232 | 16.2864 | 21.6577 | 10.6243 |
|               | 1500      | 13.8158 | 20.6374 | 24.1833 | 20.4077 |
|               | 2500      | 14.9816 | 22.4024 | 26.0404 | 25.7436 |
| Green 9.6107 | 500       | 10.6243 | 15.9176 | 20.4077 | 14.1141 |
|               | 1500      | 12.9926 | 20.1874 | 25.7436 | 21.5196 |
|               | 2500      | 14.1141 | 21.5196 | 33.2497 | 25.3246 |
| Blue 7.2346  | 500       | 8.0913  | 12.6382 | 18.4416 | 12.6382 |
|               | 1500      | 10.1583 | 18.0640 | 21.2062 | 19.7364 |
|               | 2500      | 11.2730 | 19.7364 | 23.0863 | 23.0863 |
| 60 Red 9.1517 | 500       | 10.1365 | 14.8332 | 21.3515 | 9.1517 |
|               | 1500      | 12.0693 | 20.8865 | 24.0216 | 20.8865 |
|               | 2500      | 13.1579 | 22.6918 | 25.3246 | 22.6918 |
| Green 8.8377 | 500       | 9.6996  | 15.1357 | 20.2725 | 8.8377 |
|               | 1500      | 11.6472 | 20.4646 | 23.5771 | 11.6472 |
|               | 2500      | 12.6518 | 21.9962 | 25.3317 | 21.9962 |
| Blue 6.8093  | 500       | 7.5010  | 11.6169 | 17.3878 | 7.5010 |
|               | 1500      | 9.0997  | 17.1928 | 20.1489 | 9.0997 |
|               | 2500      | 10.0391 | 19.0494 | 21.4323 | 19.0494 |
Table 4: Numerical comparison of PSNR values of Temple image (size=581 × 432) each motion length.

| motion length | Iter | Algorithm 3 | Algorithm 4 | Algorithm 5 |
|---------------|------|--------------|--------------|--------------|
| 15            | Red  | 500          | 15.2004      | 19.0802      | 27.8120      |
|               |      | 1500         | 17.3632      | 26.5238      | 31.0685      |
|               |      | 2500         | 18.7359      | 28.6759      | 32.9181      |
|               | Green| 500          | 16.5184      | 20.3138      | 29.3533      |
|               |      | 1500         | 18.7599      | 27.7870      | 32.3749      |
|               |      | 2500         | 20.1512      | 29.7041      | 34.0459      |
|               | Blue | 500          | 17.8056      | 22.4030      | 30.2959      |
|               |      | 1500         | 20.1113      | 29.3933      | 33.5506      |
|               |      | 2500         | 21.5074      | 31.1807      | 35.2785      |
| 30            | Red  | 500          | 13.4104      | 16.1524      | 22.1375      |
|               |      | 1500         | 15.0103      | 20.9424      | 24.6278      |
|               |      | 2500         | 15.8666      | 22.9244      | 25.7783      |
|               | Green| 500          | 14.5928      | 17.4967      | 23.5112      |
|               |      | 1500         | 16.2930      | 22.4286      | 26.0190      |
|               |      | 2500         | 17.2013      | 24.3866      | 27.2886      |
|               | Blue | 500          | 15.6445      | 18.9835      | 24.8991      |
|               |      | 1500         | 17.4787      | 24.4009      | 27.2539      |
|               |      | 2500         | 18.4356      | 26.0283      | 28.4147      |
| 45            | Red  | 500          | 12.4420      | 15.6032      | 21.9892      |
|               |      | 1500         | 14.0749      | 21.0037      | 25.4849      |
|               |      | 2500         | 14.8852      | 23.2634      | 27.3369      |
|               | Green| 500          | 13.5300      | 16.8465      | 23.2119      |
|               |      | 1500         | 15.3078      | 22.4460      | 25.3298      |
|               |      | 2500         | 16.1652      | 24.8112      | 28.5828      |
|               | Blue | 500          | 14.4353      | 18.1228      | 24.5189      |
|               |      | 1500         | 16.4149      | 24.0860      | 28.0582      |
|               |      | 2500         | 17.3238      | 26.1620      | 29.6786      |
| 60            | Red  | 500          | 11.7898      | 14.5487      | 19.5199      |
|               |      | 1500         | 13.2249      | 18.9355      | 21.7620      |
|               |      | 2500         | 13.9734      | 20.5929      | 22.8371      |
|               | Green| 500          | 12.7634      | 15.6730      | 21.0301      |
|               |      | 1500         | 14.3215      | 20.3644      | 23.2214      |
|               |      | 2500         | 15.1202      | 22.0877      | 24.3413      |
|               | Blue | 500          | 13.5083      | 16.7875      | 22.4595      |
|               |      | 1500         | 15.2758      | 21.8304      | 24.6498      |
|               |      | 2500         | 16.1448      | 23.4691      | 25.7597      |

From Tables 2-4, the reports show that PSNR of Algorithm 5 is higher than Algorithm 3 and Algorithm 4 in each motion lengths. From this point of view, we conclude that our proposed Algorithm 5 has a better convergence behavior than Algorithm 3 defined by Zhao et al. [37] and Algorithm 4 defined by Dong et al. [12].

Next, we show the original images for Cat image (size= 384 × 512), Flower image (size=436 × 581) and Temple image (size=581 × 432).
We next demonstrate the blurred images for each motion length.
Next, we demonstrate the recovered images by using Algorithms 3, 4 and 5 for the motion length 15 and the number of iterations is 2500.

Figure 7: Recovered images with the motion length 15.
We demonstrate the recovered images by using Algorithms 3, 4 and 5 for the motion length 30 and the number of iterations is 2500.

Figure 8: Recovered images with the motion length 30.
We demonstrate the recovered images by using Algorithms 3, 4 and 5 for the motion length 45 and the number of iterations is 2500.

Figure 9: Recovered images with the motion length 45.
We demonstrate the recovered images by using Algorithms 3, 4 and 5 for the motion length 60 and the number of iterations is 2500.

Figure 10: Recovered images with the motion length 60.
We next provide the PSNR plotting of Algorithms 3, 4 and 5.

![PSNR Graphs for Cat Image](image1.png)

(a) PSNR (Red)  (b) PSNR (Green)  (c) PSNR (Blue)

Figure 11: Graphs of PSNR for red, green and blue for Algorithms 3, 4 and 5 of Cat image with motion length 45.

![PSNR Graphs for Flower Image](image2.png)

(a) PSNR (Red)  (b) PSNR (Green)  (c) PSNR (Blue)

Figure 12: Graphs of PSNR for red, green and blue for Algorithms 3, 4 and 5 of Flower image with motion length 45.

![PSNR Graphs for Temple Image](image3.png)

(a) PSNR (Red)  (b) PSNR (Green)  (c) PSNR (Blue)

Figure 13: Graphs of PSNR for red, green and blue for Algorithms 3, 4 and 5 of Temple image with motion length 45.

From Figures 11-13, it is observed that the PSNR of red, green and blue of Algorithm 5 is higher than Algorithms 3 and 4 in comparison. It shows the applicability and efficiency the proposed method for solving the image deblurring problem which is the application of the SFP.

7. Conclusions

In this work, we proposed new and efficient algorithms for the split feasibility problem. We show that the sequence generated by the proposed method converges weakly to a solution of the SFP.
numerical experiments reveal that our algorithms outperform algorithms defined by Zhao et al. [37] and Dong et al. [12].

Acknowledgments

This project is funded by National Research Council of Thailand (NRCT) under grant no. N41A640094. The authors wish to thank University of Phayao and Thailand Science Research and Innovation grant no. FF65-UoE001.

References

[1] A.S. Antipin, On a method for convex programs using a symmetrical modification of the Lagrange function, Ekon. Mat. Metody 12, (1976)116-1173.
[2] H.H. Bauschke, P.L. Combettes, A weak-to-strong convergence principle for Fejér-monotone methods in Hilbert spaces. Math. Oper. Res. 26, (2001)248-264.
[3] H.H. Bauschke, P.L. Combettes, Convex Analysis and Monotone Operator Theory in Hilbert Spaces, Springer, London (2011).
[4] C. Byrne, Iterative oblique projection onto convex sets and the split feasibility problem, Inverse Probl. 18, (2002)441-453.
[5] C. Byrne, A unified treatment of some iterative algorithms in signal processing and image reconstruction, Inverse Probl. 20, (2004)103-120.
[6] Y. Censor, A. Gibali, S. Reich, Algorithms for the split variational inequality problem, Numer Algor. 59(2), (2012)301-323.
[7] Y. Censor, T. Bortfeld, B. Martin, A. Trofimov, A unified approach for inversion problems in intensity-modulated radiation therapy, Phys. Med. Biol. 51, (2006)2353-2365.
[8] Y. Censor, T. Bortfeld, B. Martin, A. Trofimov, The split feasibility model leading to a unified approach for inversion problems in intensity-modulated radiation therapy, Technical Report 20 April: Department of Mathematics, University of Haifa, Israel, 2005.
[9] Y. Censor, T. Elfving, A multiprojection algorithms using Bregman projection in a product space, Numer. Algor. 8, (1994)221-239.
[10] Y. Censor, T. Elfving, N. Kopf, T. Bortfeld, The multiple-sets split feasibility problem and its applications for inverse problem, Inverse Probl. 21, (2005)2071-2084.
[11] Y. Censor, M. Jiang, G. Wang, Iterative algorithms for the multiple-sets split feasibility problem. In editors: Biomedical mathematics: promising directions in imaging therapy planning and inverse problems. Madison (WI): Medical Physics Publishing., (2010)243-279.
[12] Q.L. Dong, Y.C. Tang, Y.J. Cho, T.M. Rassias, Optimal choice of the step length of the projection and contraction methods for solving the split feasibility problem, J. Glob.Optim., 71(2), (2018) 341-360.
[13] B. Eicke, Iteration methods for convexly constrained ill-posed problems in Hilbert spaces, Numer. Funct. Anal. Optim. 13, (1992)413-29.
[14] M. Fukushima, A relaxed projection method for variational inequalities, Math. Program. 35, (1986)58-70.
[15] S. Kesornprom, N. Pholasa, P. Cholamjiak, On the convergence analysis of the gradient-CQ algorithms for the split feasibility problem, Numer. Alg., (2019)1-21
[16] S. Kesornprom, P. Cholamjiak, Proximal type algorithms involving linesearch and inertial technique for split variational inclusion problem in hilbert spaces with applications, Optim. 68(12), (2019)2369-2395
[17] G.M. Korpelevich, The extragradient method for finding saddle points and other problems, Ekon. Mate. Metody 12, (1976)747-756.
[18] A. Gibali, A new split inverse problem and application to least intensity feasible solutions, Pure Appl. Funct. Anal. 2, (2017)243-258.
[19] A. Gibali, L.W. Liu, Y.C. Tang, Note on the modified relaxation CQ algorithm for the split feasibility problem, Optim. Lett. 12, (2017)1-14.
[20] A. Gibali, K.H. Küfer, P. Süss, Successive linear programing approach for solving the nonlinear split feasibility problem, Journal of Nonlinear and Convex Analysis, 15, (2014)345-353.
[21] Z. Opial, Weak convergence of the sequence of successive approximations for nonexpansive mappings, Bull Am Math Soc. 73(4), (1967)591-597.
[22] S. Penfold, R. Zalas, M. Casiraghi, M. Brooke, Y. Censor, R. Schulte, Sparsity constrained split feasibility for dose-volume constraints in inverse planning of intensity-modulated photon or proton therapy, Physics in Medicine and Biology 62, (2017)3599-3618.
[23] L. Landweber, An iterative formula for Fredholm integral equations of the first kind, Am. J.Math. 73(3), (1951)615-624.
[24] Z. Li, D. Han, W. Zhang, A self-adaptive projection-type method for nonlinear multiple-sets split feasibility problem, Inverse Problems in Science and Engineering iFirst,(2012) 1-16.
[25] G. López, V. Martín-Márquez, F.H. Wang, H.K. Xu, Solving the split feasibility problem without prior knowledge of matrix norms, Inverse Prob. 2012. doi:10.1088/0266-5611/28/8/085004.
[26] G. López, V. Martín-Márquez, H.K. Xu, Perturbation techniques for nonexpansive mappings with applications, Nonlinear Anal. RealWorld Appl., 10, (2009)2369-2383.
[27] B. Qu, N. Xiu, A note on the CQ algorithm for the split feasibility problem, Inverse Prob. 21, (2005)1655-1665.
[28] B. Qu, N. Xiu, A new halfspace-relaxation projection method for the split feasibility problem, Linear Algebr. Appl. 428, (2008)1218-1229.
[29] R.T. Rockafellar, Monotone operators and the proximal point algorithm, SIAM J. Control Optim. 14, (1976)877-898.
[30] S. Suantai, S. Kesornprom, P. Cholamjiak, A new hybrid CQ algorithm for the split feasibility problem in Hilbert spaces and its applications to compressed sensing, Math. 7(9), (2019)789.
[31] S. Suantai, S. Kesornprom, P. Cholamjiak, Modified Proximal Algorithms for Finding Solutions of the Split Variational Inclusions, Math. 7(8), (2019)708.
[32] P. Tseng, A modified forward-backward splitting method for maximal monotone mappings, SIAM J. Control Optim. 38, (2000)431-446.
[33] R. Tibshirani, Regression shrinkage and selection via the lasso, J. R. Stat. Soc. Ser. B. Stat. Methodol. 58, (1996)267-288.
[34] H.K. Xu, Iterative methods for the split feasibility problem in infinite-dimensional Hilbert spaces, Inverse Prob. 26, (2010)105018.
[35] Q. Yang, The relaxed CQ algorithm for solving the split feasibility problem, Inverse Prob. 20, (2004)1261-1266.
[36] W. Zhang, D. Han, Z. Li, A self-adaptive projection method for solving the multiple-sets split feasibility problem, Inverse Probl. 25, (2009)115001.
[37] J. Zhao, Y. Zhang, Q. Yang, Modified projection methods for the split feasibility problem and the multiple-sets split feasibility problem, Appl. Math. Comput. 219(4), (2012)1644-1653.
[38] J. Zhao, Q. Yang, Self-adaptive projection methods for the multiple-sets split feasibility problem, Inverse Probl. 27, (2011)035009.