The Lions-Mercier Splitting Algorithm and the Alternating Direction Method are Instances of the Proximal Point Algorithm

by

*Jonathan Eckstein**

Abstract

Suppose we have two maximal monotone operators A and B over a Hilbert space and wish to find a zero of the operator $A+B$. We introduce an auxiliary maximal monotone operator $S_{\lambda,A,B}$ whose set of zeroes is very closely related to that of $A+B$. In the case that B is the normal cone operator of a linear subspace, $S_{\lambda,A,B}$ is virtually identical to the partial inversion operator of Spingarn [18],[19]. Furthermore, when $r=1$, the resolvent $(I + rS_{\lambda,A,B})^{-1}$ of $S_{\lambda,A,B}$ is the operator $G(\lambda)$ of Lions-Mercier [9]. Thus, the algorithm of Lions-Mercier is in fact the proximal point algorithm applied to $S_{\lambda,A,B}$. We also conclude that Spingarn's technique for minimizing a convex function over a linear subspace is essentially a special case of the Lions-Mercier approach. Further, as the Lions-Mercier algorithm generalizes the alternating direction method for convex programming [4]-[8], as shown by Gabay [5], we obtain that the alternating direction method is in fact an instance of the proximal point algorithm. We will rederive this result directly.

Massachusetts Institute of Technology
Operations Research Center
Laboratory for Information and Decision Systems
Center for Intelligent Control Systems
Cambridge, MA 02139

* This research was carried out at the Operations Research Center and the Laboratory for Information and Decision Systems. It was supported in part by the Army Research Office, under grant number DAAL03-86-K-0171.

1. *Introduction*

This report aims to make clear the relationship between a number of similar algorithms involving monotone operators, namely the *proximal point algorithm* first introduced by Martinet [10] and further developed by Rockafellar [16], the Lions-Mercier method for finding the zero of the sum of two maximal monotone operators [9], Spingarn's method of partial inverses for monotone operators [18],[19], the "progressive hedging" stochastic programming algorithm of Rockafellar and Wets [17], and the alternating direction method of multipliers for convex programming of Glowinski-Marocco, Gabay-Mercier, and other authors [4]-[8]. It is already known [5] that the alternating direction method is a special case of Lions-Mercier. Also, the Spingarn and progressive hedging methods have been known since inception to be instances of the proximal point algorithm. We demonstrate two (apparently) new facts: that the Lions-Mercier method is in fact an instance of the proximal point algorithm, and that the Spingarn and Rockafellar-Wets methods are essentially special cases of Lions-Mercier (see Figure 4 at the end of this report for a summary of these relationships). We also give a new development of the Lions-Mercier method.

The basic problem we consider is that of finding a zero of the monotone operator $A+B$, where A and B are maximal monotone operators.

2. *Foundations*

Given any two sets X, Y we will say that a (possibly multivalued) *mapping* $X \rightrightarrows Y$ is simply any subset of $X \times Y$. If A is such a mapping and $x \in X$, we write Ax or $A(x)$ to denote the set $Ax = \{y \in X \mid (x, y) \in A\}$. Essentially, we make no distinction between a mapping and its graph. This approach is not standard, but it simplifies many of the following definitions and proofs. If A is single-valued, that is, the cardinality of Ax is at most 1 for all $x \in X$, we will

write $\diamond Ax$ to denote the unique member y of Ax for all x such that $Ax \neq \emptyset$. The *domain* of a mapping A is its projection onto the first coordinate,

$$\text{dom } A = \{ x \in X \mid \exists\, y \in Y : (x, y) \in A \} = \{ x \in X \mid Ax \neq \emptyset \}.$$

We say that A has *full domain* if $\text{dom } A = X$. The *range* or *image* of A is similarly defined as its projection onto the second coordinate,

$$\text{im } A = \{ y \in Y \mid \exists\, x \in X : (x, y) \in A \} .$$

The inverse $A^{-1}: Y \rightrightarrows X$ of a mapping $A: X \rightrightarrows Y$ is $\{(y, x) \mid (x, y) \in A\}$. Given two mappings $T: X \rightrightarrows Y$ and $R: Y \rightrightarrows Z$, their *composition* RT or $R \circ T$ is given by

$$RT = \{ (x, z) \mid \exists\, y \in Y : (x, y) \in T, (y, z) \in R \},$$

hence

$$(RT)x = \bigcup_{y \in Tx} Ry .$$

If Y is a real vector space, we make the additional definitions

$$cA = \{(x, cy) \mid (x, y) \in A\} \qquad \forall\, c \in \mathbb{R}, A: X \rightrightarrows Y$$

$$A+B = \{(x, y+z) \mid (x, y) \in A, (x, z) \in B\}, \quad \forall\, A, B: X \rightrightarrows Y .$$

A *zero* of A is any $x \in X$ such that $0 \in Ax$. The set of all zeroes of A, $A^{-1}(0)$, will also be denoted $\text{zer}(A)$. Now let $\mathcal{H}$ be a real Hilbert space with inner product $\langle \cdot, \cdot \rangle$ and the corresponding norm topology. An *operator* on $\mathcal{H}$ is any mapping $\mathcal{H} \rightrightarrows \mathcal{H}$. We will always use I to denote the *identity* operator $\{(x, x) \mid x \in \mathcal{H}\}$.

An operator A is *monotone* if

$$\langle x' - x, y' - y \rangle \geq 0 \qquad \forall\, (x, y), (x', y') \in A .$$

A *maximal* monotone operator is a monotone operator that is not strictly contained in any other monotone operator. Perhaps the most common example of a maximal monotone operator is the subgradient map of a convex function [12], [13]. Note that for A, B monotone,

it is a straightforward excercise to prove that $A+B$ is also monotone. However, it is not always true that if A,B are *maximal* monotone, $A+B$ is *maximal* monotone (see [3] and [14]). One must usually require an additional "qualification" condition such as

$$\text{ri}(\text{dom } A) \cap \text{ri}(\text{dom } B) \neq \varnothing$$

in order to have $A+B$ maximal. A proof of the following result may be found in [3]:

Lemma 1. Given a monotone operator A on $\mathcal{H}$, A is maximal if and only if $\text{im}(I+A) = \mathcal{H}$.

We omit the proof of the following lemma, as it is entirely straightforward:

Lemma 2. Let $r > 0$ be a real number, and A an operator on $\mathcal{H}$. Then $\text{zer}(A) = \text{zer}(rA)$, and A is (maximal) monotone if and only if rA is (maximal) monotone.

3. *Firmly Nonexpansive Mappings*

We will now demonstrate the complete symmetry that exists between the collection of maximal monotone operators on $\mathcal{H}$ and the class of so-called *firmly nonexpansive* operators with full domain on $\mathcal{H}$. This correspondence is already known, but is not particularly prominent in the existing literature. An operator K on $\mathcal{H}$ is said to be *nonexpansive* if

$$\|x' - x\| \geq \|y' - y\| \quad \forall\ (x, y), (x', y') \in K \ .$$

Lemma 3. All nonexpansive operators are single-valued.

Proof. Given $(x, y_1), (x, y_2) \in K$, $0 = \|x - x\| \geq \|y_1 - y_2\|$, hence $y_1 = y_2$. ■

K is *firmly nonexpansive* [9] if

$$\langle x' - x, y' - y \rangle \geq \|y' - y\|^2 \quad \forall\ (x, y), (x', y') \in K \ .$$

Lemma 4. All firmly nonexpansive operators are monotone and nonexpansive. Thus, they are also single-valued.

Proof. Let K be firmly nonexpansive. Since

$$\langle x'-x, y'-y\rangle \geq \|y'-y\|^2 \geq 0 \quad \forall\ (x,y),\ (x',y') \in K\ ,$$

K is clearly monotone. An equivalent, but lengthier, statement of the firm nonexpansiveness condition is given in [10],[16], namely

$$\|y'-y\|^2 \leq \|x'-x\|^2 - \|(x'-y')-(x-y)\|^2 \quad \forall\ (x,y),\ (x',y') \in K\ .$$

(The equivalence may be checked by direct calculation.) From this form of the condition, it is clear that K is nonexpansive, and hence by Lemma 3 that it is single-valued. ■

We now consider the problem of locating a *fixed point* of K, that is some $x^* \in \mathcal{H}$ such that $Kx^* = \{x^*\}$. A natural approximation scheme is the iteration $x^{k+1} = \diamond Kx^k$, starting from some arbitrary point $x^0 \in \mathcal{H}$. If w is any fixed point of K, we note that from the proof of Lemma 4 one has

$$\|x^{k+1}-w\|^2 \leq \|x^k-w\|^2 - \|x^{k+1}-x^k\|^2 \quad \forall\ k \geq 0\ .$$

It is then immediate that the distance between x^k and *every* fixed point is nonincreasing, and that $\|x^{k+1}-x^k\| \to 0$. Thus, if any fixed points exist, the sequence $\{x^k\}$ must be bounded. From here, depending on the conditions of the problem at hand, a number of different arguments (dating back to [10]) may be used to deduce the convergence, usually in the weak topology, of $\{x^k\}$ to a fixed point of K. In finite dimension, convergence is guaranteed.

We now demonstrate the connection with the topic of monotone operators. Given any maximal monotone operator T and real number $r > 0$, we define the *resolvent* $J_{r,T}$ of T to be $(I+rT)^{-1}$.

Proposition 1. An operator T on $\mathcal{H}$ is monotone if and only if $J_{1,T} = (I+T)^{-1}$ is firmly nonexpansive. It is maximal monotone if and only if $(I+T)^{-1}$ is firmly nonexpansive and has full domain.

Proof. To establish the first statement we note that

$$\begin{aligned} (x, y) \in T \quad &\Leftrightarrow \quad (x + y, x) \in (I+T)^{-1} \\ \langle x' - x, y' - y \rangle \geq 0 \quad &\Leftrightarrow \quad \langle x' - x + y' - y, x' - x \rangle \geq \|x' - x\|^2 \ . \end{aligned}$$

By Lemma 1, T is maximal if and only if $\mathrm{im}(I+T) = \mathcal{H}$. This is in turn true if and only if the operator $(I+T)^{-1}$ has full domain. This establishes the second statement. ■

Corollary 1. An operator K is firmly nonexpansive if and only if $K^{-1} - I$ is monotone. K is firmly nonexpansive with full domain if and only if $K^{-1} - I$ is maximal monotone.

Corollary 2. The functional taking $T \mapsto (I+T)^{-1}$ is a bijection between the collection $\mathcal{M}(\mathcal{H})$ of maximal monotone operators on $\mathcal{H}$ and the collection $\mathcal{F}(\mathcal{H})$ of firmly nonexpansive operators on $\mathcal{H}$.

Corollary 3. For any maximal monotone operator T and real number $r > 0$, the *resolvent* $J_{r,T} = (I+rT)^{-1}$ is firmly nonexpansive (hence single-valued) and has full domain.

Proof. By Lemma 2, rT is maximal monotone, hence $(I+rT)^{-1}$ is firmly nonexpansive and has full domain. ■

This last result is a seminal one in the field of monotone operators, and was first established by Minty [11]. Although it may appear that we have obtained the result without using the more subtle parts of the development in [11], it appears that similar analysis is still needed to prove Lemma 1.

Lemma 5. Given any maximal monotone operator T, real number $r > 0$, and $x \in \mathcal{H}$, $0 \in Tx$ if and only if $J_{r,T}(x) = \{x\}$.

Proof. By direct calculation, $J_{r,T} = \{(x + ry, x) \mid (x, y) \in T\}$. Hence,

$$0 \in Tx \quad \Leftrightarrow \quad (x, 0) \in T \quad \Leftrightarrow \quad (x, x) \in J_{r,T} \ .$$

Since $J_{r,T}$ is single-valued, the proof is complete. ■

This last observation motivates the following asymptotic procedure for approximating a zero of T [10]. Start with any $r > 0$ and $x^0 \in \mathcal{H}$; then repeatedly compute

$$x^{k+1} = \diamond J_{r,T}(x^k) = \diamond (I + rT)^{-1} x^k \ .$$

Rockafellar [16] proposed and extensively analyzed a generalization of this scheme, which he called the *proximal point algorithm.* Here one takes a *sequence* $\{r_k\}$ of positive real numbers, bounded away from zero, and an arbitrary starting point $x^0 \in \mathcal{H}$. One then performs the iteration

$$x^{k+1} = \diamond J_{r_k,T}(x^k) = \diamond (I + r_k T)^{-1} x^k \ .$$

The arguments of [16] hinge primarily on the firmly nonexpansive properties of the resolvents $(I + r_k T)^{-1}$. A partial statement of a principal result of [16] is:

Proposition 2. Let T be a maximal monotone operator. If zer(T) is nonempty, then the sequence $\{x^k\}$ generated by the proximal point algorithm is bounded and converges weakly to a zero of T. If zer(T) is empty, $\{x^k\}$ is unbounded.

It is important to note that *any* algorithm consisting of the iteration $x^{k+1} = \diamond K x^k$, where K is firmly nonexpansive with full domain, may be considered an application of the proximal point algorithm to the maximal monotone operator $K^{-1} - I$ corresponding to K, with $\{r_k\}$ fixed at 1. This fact is not prominent in the literature of the proximal point algorithm.

4. *The Splitting Operator*

We now consider the problem of finding a zero of $A+B$, where A and B are both maximal monotone operators on $\mathcal{H}$. In the typical case that $A+B$ is maximal, one may consider applying the proximal point algorithm to $A+B$. However, we will primarily be interested in cases where $J_{r,A+B}$ is hard to compute, in which case such an approach may not be practical. We instead present a method which will ultimately require only evaluations of $J_{\lambda,A}$ and $J_{\lambda,B}$, where $\lambda > 0$ is some positive real number.

To motivate the analysis, we may rephrase the problem of locating a zero of $A+B$ as that of finding $(u, b) \in B$, $(v, a) \in A$ such that

$$\begin{aligned} v &= u \\ a &= -b \ . \end{aligned}$$

For any real number $\lambda > 0$, an equivalent system of equations is

$$\begin{aligned} u - v &= 0 \\ v+\lambda a &= u - \lambda b \ . \end{aligned}$$

One way to determine such u, b, v and a is to find a zero of the operator

$$S_{\lambda,A,B} = \{ \ (v+\lambda b, \ u - v) \mid (u, b) \in B, \ (v, a) \in A, \ v+\lambda a = u - \lambda b \} \ ,$$

which may also be written

$$S_{\lambda,A,B} = \{ \ (v+\lambda b, \ u - v) \mid (u, b) \in B, \ (v, \lambda^{-1}(u - v) - b) \in A \ \}.$$

We call $S_{\lambda,A,B}$ the *splitting operator* associated with A and B. The choice of $v+\lambda b$ as the first member of each ordered pair in $S_{\lambda,A,B}$ may seem unmotivated at this time. However, as we shall see in Proposition 4, it leads to $S_{\lambda,A,B}$ being maximal monotone whenever A and B are. We do not exclude the possibilities that other, similarly-constructed operators might also be monotone and/or result in useful algorithms.

The zeroes of $S_{\lambda,A,B}$ and those of $A+B$ are not identical, but are very closely related:

Proposition 3. $\mathrm{zer}(S_{\lambda,A,B}) = \{\, u+\lambda b \mid b \in Bu,\ -b \in Au \,\} \subseteq \{\, u+\lambda b \mid u \in \mathrm{zer}(A+B),\ b \in Bu \}$.

Proof. Let $S = S_{\lambda,A,B}$ and $Z = \{\, u+\lambda b \mid b \in Bu,\ -b \in Au \,\}$. We wish to show that $\mathrm{zer}(S)$ is equal to Z. Let $z \in \mathrm{zer}(S)$. Then there exist some $u, b, v \in \mathcal{H}$ such that $v+\lambda b = z$, $u - v = 0$, $(u, b) \in B$, and $(v, \lambda^{-1}(u - v) - b) \in A$. These conditions simplify to $u+\lambda b = z$, $(u, b) \in B$, and $(u, -b) \in A$, hence $z \in Z$. Conversely, if $z \in Z$, then $z = u+\lambda b$, $b \in Bu$, and $-b \in Au$. Setting $u = v$, we see that $(z, 0) \in S$. Finally, the inclusion $Z \subseteq \{\, u+\lambda b \mid u \in \mathrm{zer}(A+B),\ b \in Bu \}$ follows because $b \in Bu$ and $-b \in Au$ imply that $u \in \mathrm{zer}(A+B)$. ■

In the remainder of this report, we will continue to use the abbreviation $S = S_{\lambda,A,B}$ where convenient.

Proposition 4. If A and B are maximal monotone operators, $S = S_{\lambda,A,B}$ is also a maximal monotone operator.

Proof. First we show that S is monotone. Let $u, b, v, u', b', v' \in \mathcal{H}$ be such that $(u, b) \in B$, $(v, \lambda^{-1}(u - v) - b) \in A$, $(u', b') \in B$, $(v', \lambda^{-1}(u' - v') - b') \in A$. Then

$$
\begin{aligned}
&\langle (v'+\lambda b') - (v+\lambda b),\ (u' - v') - (u - v) \rangle \\
&\quad = \lambda \langle (v'+\lambda b') - (v+\lambda b),\ \lambda^{-1}(u' - v') - b' - \lambda^{-1}(u - v) + b \rangle \\
&\qquad + \lambda \langle (v'+\lambda b') - (v+\lambda b),\ b' - b \rangle \\
&\quad = \lambda \langle v' - v,\ \lambda^{-1}(u' - v') - b' - \lambda^{-1}(u - v) + b \rangle \\
&\qquad + \lambda^2 \langle b' - b,\ \lambda^{-1}(u' - v') - b' - \lambda^{-1}(u - v) + b \rangle \\
&\qquad + \lambda \langle v' - v,\ b' - b \rangle + \lambda^2 \langle b' - b,\ b' - b \rangle \\
&\quad = \lambda \langle v' - v,\ \lambda^{-1}(u' - v') - b' - \lambda^{-1}(u - v) + b \rangle \\
&\qquad + \lambda \langle b' - b,\ u' - u \rangle - \lambda \langle b' - b,\ v' - v \rangle - \lambda^2 \langle b' - b,\ b' - b \rangle \\
&\qquad + \lambda \langle v' - v,\ b' - b \rangle + \lambda^2 \langle b' - b,\ b' - b \rangle \\
&\quad = \lambda \langle v' - v,\ \lambda^{-1}(u' - v') - b' - \lambda^{-1}(u - v) + b \rangle + \lambda \langle b' - b,\ u' - u \rangle .
\end{aligned}
$$

By the monotonicity of A and B, the two terms in the final line are nonnegative, so we obtain that $\langle (v'+\lambda b') - (v+\lambda b), (u'-v') - (u-v) \rangle \geq 0$, and S is monotone.

By Lemma 1, it now suffices to show that $\mathrm{im}(I+S) = \mathcal{H}$. Since A and B are maximal monotone, so are λA and λB, hence $\mathrm{im}(I+\lambda A) = \mathrm{im}(I+\lambda B) = \mathcal{H}$. Take any $w \in \mathcal{H}$. Then there exists some u such that $u+\lambda b = w$ and $(u, b) \in B$. Furthermore, there is some v such that $v+\lambda a = u - \lambda b$ and $(v, a) \in A$. Then $a = \lambda^{-1}(u-v) - b$, so we may substitute u, b, and v into the definition of S to obtain that $(I+S)(v+\lambda b) \ni u - v + v + \lambda b = u + \lambda b = w$. ■

Thus, to approximate a zero u of $A+B$, one may apply the proximal point algorithm (with any positive sequence $\{r_k\}$ bounded away from zero) to $S_{\lambda,A,B}$ to obtain a sequence $\{z^k\}$ converging (weakly) to a zero z of $S_{\lambda,A,B}$. Since the resolvent operator $(I+\lambda B)^{-1} = J_{\lambda,B}$, considered as a point-to-point mapping, is Lipschitz, it is also continuous, so the sequence $\{u^k\} = \{J_{\lambda,B}(z^k)\}$ converges (weakly) to the zero $u = J_{\lambda,B}(z)$ of $A+B$. We will call this procedure the *splitting algorithm*. Interestingly, $S_{\lambda,A,B}$ is always maximal if A and B are, even if $A+B$ is not. Therefore the splitting algorithm is applicable even in cases in which $A+B$ is not maximal.

5. *Implementation Issues*

To implement the splitting algorithm, one must be able to calculate the value of $(I + rS)^{-1}$ at an arbitrary point. A straightforward calculation yields that

$$(I + rS)^{-1} = \{ ((1-r)v + ru + \lambda b, v + \lambda b) \mid (u, b) \in B, (v, \lambda^{-1}(u-v) - b) \in A \} .$$

Thus, to calculate $(I + rS)^{-1}(z)$, one must find $(u, b) \in B$ and $(v, a) \in A$ such that

$$(1-r)v + ru + \lambda b = z \qquad a = \lambda^{-1}(u-v) - b) .$$

Alternatively, we may state the problem as that of finding $u, v \in \mathcal{H}$ such that

$$z - (ru + (1-r)v) \in \lambda B \qquad -z + ((1+r)u - rv) \in \lambda A .$$

This does not appear to be a particularly easy problem. Specifically, it does not appear to be any less difficult than the calculation of $J_{r,\,A+B}$ at an arbitrary point z, which we are expressedly trying to avoid. By comparison, that calculation involves finding $(u, b) \in B$ such that $(u, \lambda^{-1}(z-u)-b) \in A$.

Consider, however, what happens in the splitting algorithm when one fixes r at 1. Then one has only to find

$$\begin{array}{lll} (u, b) \in B & \text{such that} & u+\lambda b = z \\ (v, a) \in A & \text{such that} & v+\lambda a = u-\lambda b \ . \end{array}$$

The conditions $(u, b) \in B$, $u+\lambda b = z$ uniquely determine $u=\diamond J_{\lambda,B}(z)$ and $b=\lambda^{-1}(z-u)$ independently of v. Once u is known, then v is likewise uniquely determined by $u=\diamond J_{\lambda,A}(u-\lambda b)$. Thus, we have achieved a *decomposition* in which the calculation of $J_{1,S} = (I+S)^{-1}$ is replaced by separate, sequential evaluations of $J_{\lambda,A} = (I+\lambda A)^{-1}$ and $J_{\lambda,B} = (I+\lambda B)^{-1}$. It seems that keeping r=1 at all times is essential to the decomposition. Spingarn [19] has already recognized this phenomenon, but in a more restrictive context. From now on, unless indicated otherwise, we assume that the sequence r_k=1 for all k whenever the splitting algorithm is applied. In the next section, we will show that with this restriction, the splitting algorithm is equivalent to the method of Lions-Mercier [9].

Figure 1 illustrates the calculation of $(I+\lambda S)^{-1}(z)$. The calculation of (u, b) may be thought of as finding the point of intersection of B with the diagonal "line" of slope $-1/\lambda$ passing through the point $(z, 0)$. To subsequently calculate v, one moves from (u, b) to the point $(u, -b)$, and then subsequently along another diagonal "line" of slope $-1/\lambda$ until one reaches the point of intersection with A. The final (and usually much easier) computation of $v+\lambda b$ can then be depicted as a vertical movement to the point (v, b), followed by a diagonal movement to the horizontal axis. For compactness, the graph depicts the average $\frac{1}{2}(A+B)$ of A and B, rather than their sum; these two operators have the same set of zeroes.

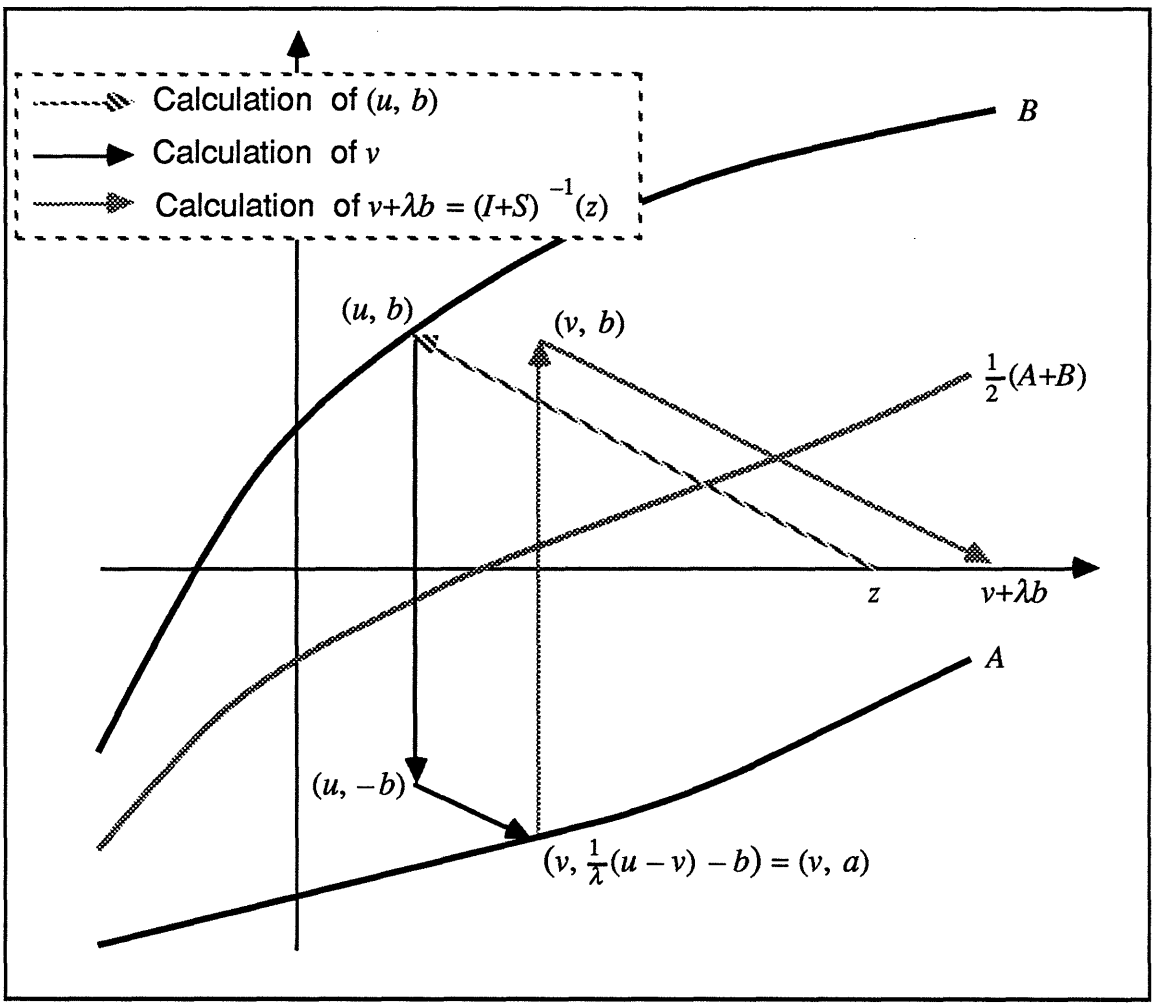

Figure 1: Computing $(I + S)^{-1}(z)$.

The decomposition of the computation of $(I + S)^{-1}(z)$ into sequential computations of $(u, b) \in B$ and $(v, a) \in A$ yields an alternate description of the splitting algorithm. We may rewrite the sequence $\{z^k\}$ it generates, which obeys $z^{k+1} = (I + S)^{-1}(z^k)$, as $\{u^k + \lambda b^k\}$, where $(u^k, b^k) \in B$ is the unique pair in B such that $u^k + \lambda b^k = z^k$. We thus obtain the following description of the algorithm:

(1) Starting from some $(u^k, b^k) \in B$, compute $(v^k, a^k) \in A$ such that $v^k+\lambda a^k = u^k - \lambda b^k$.

(2) Find $(u^{k+1}, b^{k+1}) \in B$ such that $u^{k+1}+\lambda b^{k+1} = v^k+\lambda b^k$.

This representation of the algorithm may be more pertinent algorithmically, since one is presumably most interested in the sequence $\{u^k\}$, which converges to a zero of $A+B$, rather than $\{z^k\}$ itself. Figure 2 depicts the iteration (1)-(2). Unfortunately, the algorithm cannot be described in terms of this sequence alone, for although $\{u^k\}$ obeys the recursion

$$u^{k+1} \in (I + \lambda B)^{-1}[(I + \lambda A)^{-1}(I - \lambda B) + \lambda B]u^k \ ,$$

the operator $(I + \lambda B)^{-1}[(I + \lambda A)^{-1}(I - \lambda B) + \lambda B]$ is not single-valued.[1]

We may think of (1) as a "mapping step" which computes the value of the operator $J_{1,S}$ at the current iterate, and (2) as a "re-representation step" which puts the output of (1) into the more useful form $u^{k+1}+\lambda b^{k+1}$, where $(u^{k+1}, b^{k+1}) \in B$. Note that it is possible to initialize the iteration (1)-(2) at any point $(u^0, b^0) \in \mathcal{H} \times \mathcal{H}$, even one that is not in B. The reason for this is that (u^1, b^1) is automatically in B after the first execution of (2), hence the algorithm quickly gets "on track".

Figure 3 gives another depiction of the splitting algorithm, using the fact that the zeroes of $A+B$ are precisely the points where the operator B and the the (generally non-monotone) operator $-A = (-1)A$ intersect.

---

[1]However, this description of the evolution of $\{u_k\}$, which resembles a certain approximation scheme for differential equations, appears to have been one of the inspirations for the work of Lions-Mercier [7].

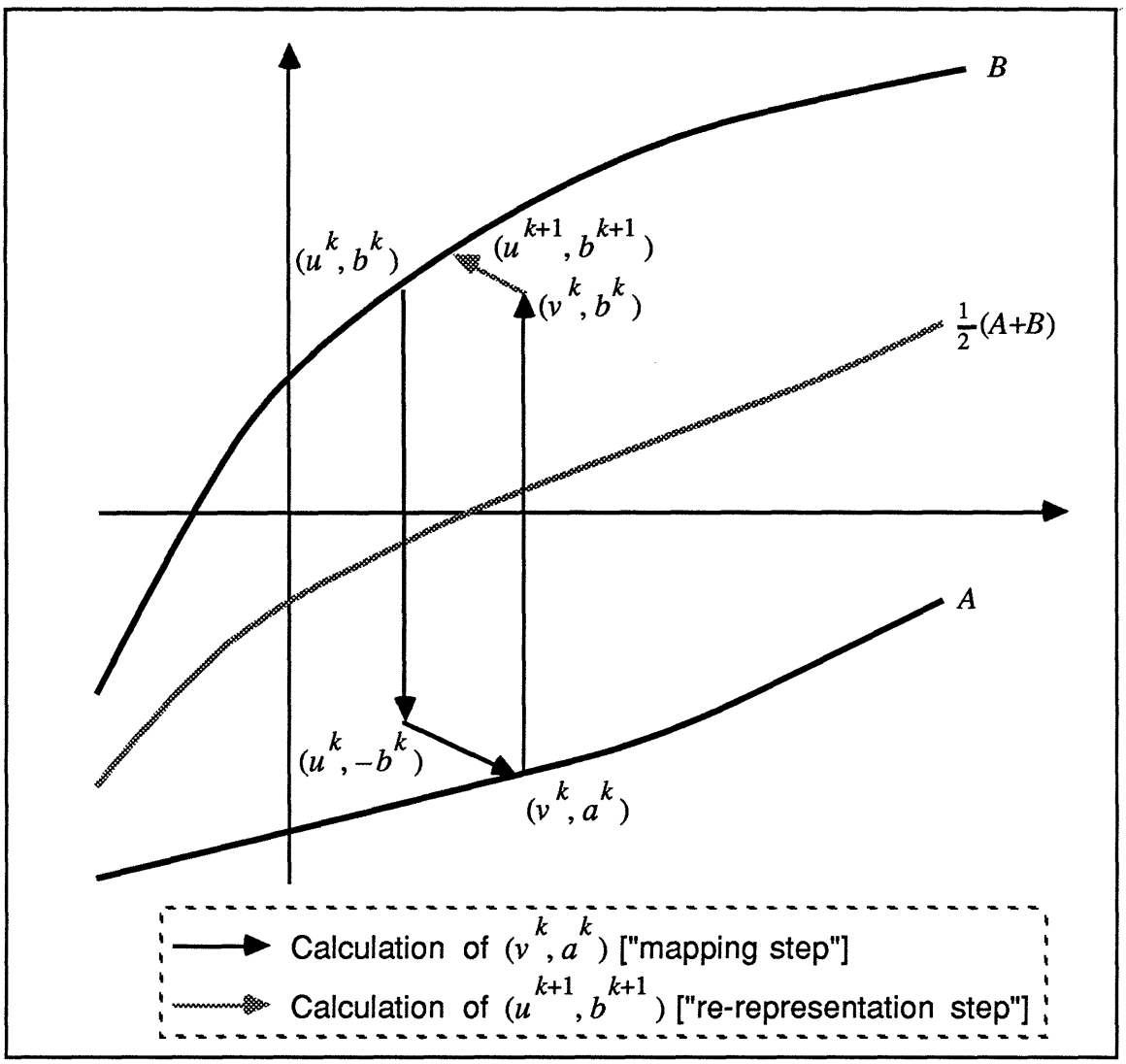

Figure 2: The splitting algorithm.

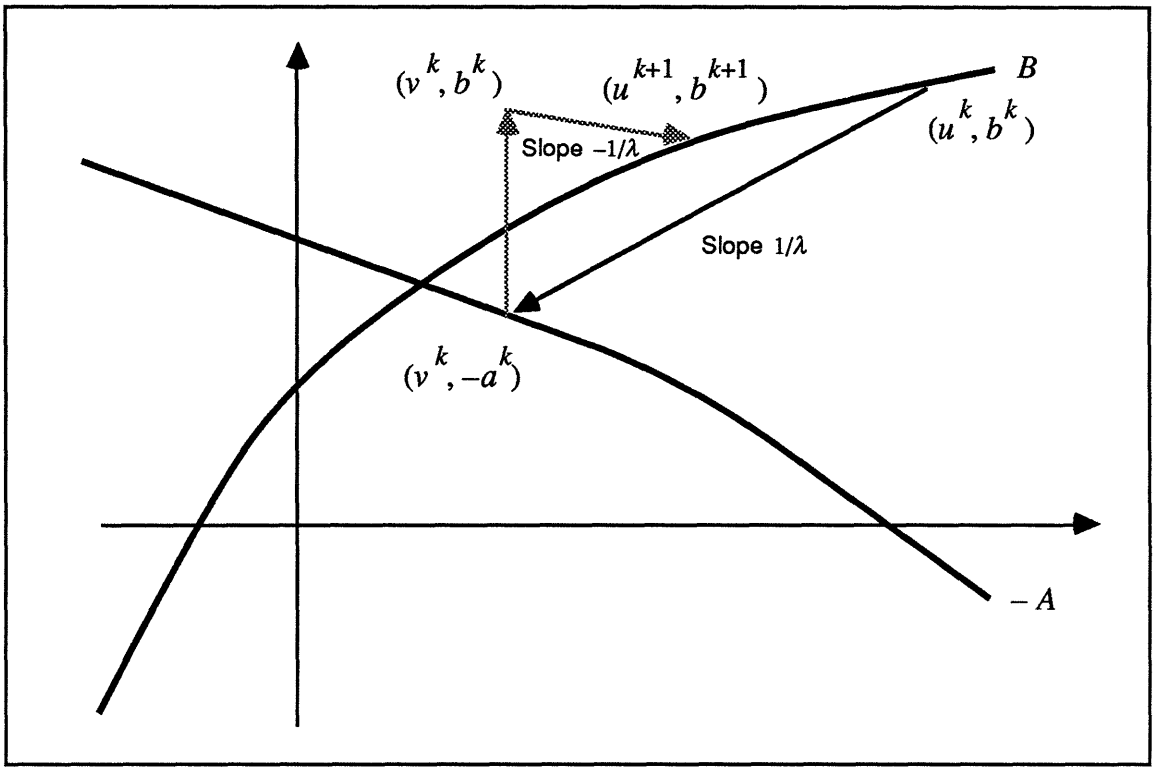

Figure 3: Alternate Depiction of the Splitting Algorithm

6. Relationship to the Method of Partial Inverses

We now consider a special case. Given an convex set $C \subseteq \mathcal{H}$, the *normal cone* operator N_C associated with C is $\{ (x, y) \mid x \in C, \langle x' - x, y\rangle \leq 0 \ \forall\, x' \in C\}$. It is well known that N_C is a maximal monotone operator, and that it is also the subgradient map of the convex indicator function

$$\delta_C(x) = \begin{cases} 0, & x \in C \\ +\infty, & x \notin C. \end{cases}$$

Now suppose $\mathcal{V}$ is a linear subspace of $\mathcal{H}$. Then $N_{\mathcal{V}} = \mathcal{V} \times \mathcal{V}^{\perp}$. Given any $w \in \mathcal{H}$, we denote the projections of w onto $\mathcal{V}$ and $\mathcal{V}^{\perp}$ by $w_{\mathcal{V}}$ and $w_{\mathcal{V}^{\perp}}$, respectively. Consider the case $B = N_{\mathcal{V}}$. In this case the problem of locating a zero of $A+B$ reduces to finding $(u, v) \in A$ such that $u \in \mathcal{V}$ and $-v \in \mathcal{V}^{\perp}$, or, equivalently, $u \in \mathcal{V}$ and $v \in \mathcal{V}^{\perp}$. In this case we have

$$S_{\lambda,A,B} = \{ (v+\lambda b, u-v) \mid u \in \mathcal{V}, b \in \mathcal{V}^{\perp}, (v, \lambda^{-1}(u-v)-b) \in A \}$$

Given any $(v, a) \in A$, the unique solution to

$$\lambda^{-1}(u-v)-b = a, \quad u \in \mathcal{V}, \quad b \in \mathcal{V}^{\perp}$$

is

$$u = (v+\lambda a)_{\mathcal{V}}, \quad b = -(\lambda^{-1}z + a)_{\mathcal{V}^{\perp}}.$$

It follows that

$$S = \{ (z_{\mathcal{V}} - \lambda a_{\mathcal{V}^{\perp}}, \lambda a_{\mathcal{V}} - z_{\mathcal{V}^{\perp}}) \mid (z, a) \in A \} .$$

For any maximal monotone operator T and subspace $\mathcal{V}$, Spingarn [18],[19] defines the *partial inverse*

$$T_{\mathcal{V}} = \{ (x_{\mathcal{V}} + y_{\mathcal{V}^{\perp}}, y_{\mathcal{V}} + x_{\mathcal{V}^{\perp}}) \mid (x, y) \in T \} .$$

Thus, except for a difference in signs, S is the partial inverse $(\lambda A)_{\mathcal{V}}$ of λA. The main application of the partial inverse is to find some $(x, y) \in T$ such that $x \in \mathcal{V}$ and $y \in \mathcal{V}^{\perp}$. Taking $T=A$, this is the problem we are already considering. To solve it, Spingarn suggests applying the proximal point algorithm to approximate a zero z of $T_{\mathcal{V}}$, and then projecting z onto $\mathcal{V}$ and $\mathcal{V}^{\perp}$, respectively, to obtain x and y. Spingarn does not consider the parameter λ, but by the the closure of $\mathcal{V}$ under scalar multiplication, it makes no difference whether we take $T=A$ or $T=\lambda A$; in the latter case, we must simply scale y by $1/\lambda$ to obtain the final answer. For essentially the same reason, the difference is signs between S and $(\lambda A)_{\mathcal{V}}$ is of no consequence. It is easily confirmed that $J_{\lambda,B}$ is the projection operator for $\mathcal{V}$, so our splitting algorithm yields the same procedure as Spingarn suggests, with the exception of the previously noted sign difference. The stochastic programming method of Rockafellar and Wets [17] is in fact identical to the splitting algorithm, λ being present and there being no difference in signs.

7. Relationship to the Lions-Mercier Splitting Algorithm

Lions and Mercier [9] suggested the following algorithm for finding a zero of $A+B$: starting with an arbitrary $v^0 \in \mathcal{H}$, perform the iteration $v^{k+1} = \diamond G_{\lambda}(v^k)$, where

$$G_\lambda = J_{\lambda,A}(2J_{\lambda,B} - I) + I - J_{\lambda,B} .$$

(Note that this operator is called $G(\lambda)$ in [9].) We now give the key result of this report:

Proposition 5. $G_\lambda = (I + S_{\lambda,A,B})^{-1}$.

Proof. By direct calculation. First, we note

$$J_{\lambda,A} = \{ (y+\lambda a, y) \mid (y, a) \in A \}$$
$$J_{\lambda,B} = \{ (x+\lambda b, x) \mid (x, b) \in B \} .$$

Therefore,

$$2J_{\lambda,B} - I = \{ (x+\lambda b, x - \lambda b) \mid (x, b) \in B \}$$
$$J_{\lambda,A}(2J_{\lambda,B} - I) = \{ (x+\lambda b, y) \mid (x, b) \in B, (y, a) \in B, y+\lambda a = x - \lambda b \}$$
$$J_{\lambda,A}(2J_{\lambda,B} - I) = \{ (x+\lambda b, y) \mid (x, b) \in B, (y, \lambda^{-1}(x - y) - b) \in A \}$$
$$G_\lambda = J_{\lambda,A}(2J_{\lambda,B} - I) + I - J_{\lambda,B} = \{ (x+\lambda b, y+\lambda b) \mid (x, b) \in B, (y, \lambda^{-1}(x - y) - b) \in A \} .$$

It then follows that

$$G_\lambda^{-1} - I = \{ (y+\lambda b, x - y) \mid (x, b) \in B, (y, \lambda^{-1}(x - y) - b) \in A \} = S_{\lambda,A,B} ,$$

establishing the claim. ■

Thus, the Lions-Mercier method is simply the splitting algorithm applied to A and B, with r_k=1 for all k. We therefore conclude that the Lions-Mercier algorithm is indeed a special case of the proximal point algorithm. (In fact, we first constructed $S_{\lambda,A,B}$ by combining the observations of section 3 with Lions and Mercier's proof that G_λ is firmly nonexpansive.) Given the analysis of the previous section, we may also conclude that the "progressive hedging" algorithm of Rockafellar and Wets [17] and the partial inverse techniques of Spingarn [18], [19] (apart from minor sign changes) are special cases of Lions-Mercier in which B is the normal cone operator of a linear subspace.

8. *Derivation of the Alternating Direction Method*

We now demonstrate, following [5], that the alternating direction method for convex programming is an instance of the splitting algorithm. We consider convex programs of the general form introduced by Rockafellar in [13],

$$\text{(P)} \qquad \min_{x \in \mathbb{R}^n} \; f(x) + g(Mx) \;,$$

where M is an $m \times n$ real matrix, and $f\colon \mathbb{R}^n \to (-\infty, +\infty]$ and $g\colon \mathbb{R}^m \to (-\infty, +\infty]$ are closed proper convex functions. While the results of this section should extend to general Hilbert space, we will confine ourselves to $\mathbb{R}^n$ and $\mathbb{R}^m$, under the canonical inner products, in order to keep the linear algebra straightforward. One may rewrite the program (P) as

$$\text{(P')} \qquad \begin{array}{ll} \min & f(x) + g(z) \\ \text{s.t.} & Mx = z \\ & x \in \mathbb{R}^n \\ & z \in \mathbb{R}^m \;. \end{array}$$

Thus, a dual program to (P') is

$$\begin{array}{ll} \max & \phi(p) \\ \text{s.t.} & p \in \mathbb{R}^m \;, \end{array}$$

where

$$\begin{aligned} \phi(p) &= \inf_{(x,z) \in \mathbb{R}^{n+m}} \{ f(x) + g(z) + p^{\mathsf{T}}(Mx - z) \} \;, \\ &= \inf_x \{ f(x) + p^{\mathsf{T}} M x \} + \inf_z \{ g(z) - p^{\mathsf{T}} z \} \\ &= -f^*(-M^{\mathsf{T}} p) - g^*(p) \;, \end{aligned}$$

where the * denotes the convex conjugacy operation. The dual to (P) may then be written

$$\text{(D)} \qquad \min_{p \in \mathbb{R}^m} \; f^*(-M^{\mathsf{T}} p) + g^*(p) \;.$$

As has been demonstrated in [5] and [15], the method of multipliers for convex programming (see [1] for a survey) is in fact an implementation of the proximal point algorithm as applied to the maximal monotone operator $\partial[f^* \circ (-M^T) + g^*]$. (When we mix matrices and multivalued mappings, we mean the matrices to be interpreted as the corresponding single-valued linear mapping; *e.g.* M should be interpreted as $\{(x, Mx) \mid x \in \mathbb{R}^n\}$.)

We now demonstrate, following [5], that, subject to certain regularity conditions, the alternating direction method of [4]-[8], and [2] is in fact an implementation of the splitting method as applied to the maximal monotone operators $A = \partial[f^* \circ (-M^T)]$ and $B = \partial g^* = (\partial g)^{-1}$. In other words, it is the proximal point algorithm with $r=1$ applied to the maximal monotone operator $S = S_{\lambda, \partial[f^* \circ (-M^T)], \partial g^*}$. It should be noted that in some exceptional cases this may not be a valid approach to solving (D), the reason being that while it is always true that

$$\partial[f^* \circ (-M^T) + g^*] \supseteq \partial[f^* \circ (-M^T)] + \partial g^* ,$$

equality may not hold unless some kind of additional qualification condition is met. Thus it is conceivable that the set of solutions to (D), $\mathrm{zer}(\partial[f^* \circ (-M^T) + g^*])$, may be nonempty, but that S has no zeroes. We leave the most precise and useful qualification conditions to be refined in future research, and assume for the moment that

$$\partial[f^* \circ (-M^T) + g^*] = \partial[f^* \circ (-M^T)] + \partial g^* = A + B .$$

Then, $\mathrm{zer}(S)$ is nonempty if and only if there are solutions to (D). We further assume that $\mathrm{im}(M^T) \cap \mathrm{ri}(\mathrm{dom}\, f^*) \neq \varnothing$, hence (see [13]) that

$$A = \partial[f^* \circ (-M^T)] = -M \circ \partial f^* \circ (-M^T) = -M \circ (\partial f)^{-1} \circ (-M^T) .$$

Again, other appropriate qualification conditions remain to be worked out.

We now restate the alternating direction method: one starts with arbitrary $z^0, p^0 \in \mathbb{R}^m$ and then performs the iteration

$$x^{k+1} = \arg\min_{x \in \mathbb{R}^n} \left\{ f(x) + (p^k)^T Mx + \frac{\lambda}{2} \|Mx - z^k\|^2 \right\}$$

$$z^{k+1} = \arg\min_{z \in \mathbb{R}^m} \left\{ g(z) - (p^k)^T z + \frac{\lambda}{2} \|Mx^{k+1} - z\|^2 \right\}$$

$$p^{k+1} = p^k + \lambda(Mx^{k+1} - z^{k+1}) \quad .$$

Proposition 6. Let the sequences $\{x^k\}$, $\{z^k\}$, $\{p^k\}$ be defined by the alternating direction method recursion above. Then for every $k \geq 1$,

$$p^{k+1} + \lambda z^{k+1} = \diamond(I + S_{\lambda, \partial[f^* \circ (-M^T)], \partial g^*})^{-1}(p^k + \lambda z^k).$$

Proof. The explicit form of the operator A is

$$A = -M \circ (\partial f)^{-1} \circ (-M^T) = \{ (u, -Mw) \mid (w, -M^T u) \in \partial f \} \quad .$$

The explicit form of B is just $B = \{ (u, b) \mid (b, u) \in \partial g \}$. From the minimization step over x, we know that for all $k \geq 0$,

$$\begin{aligned} & 0 \in \partial f(x^{k+1}) + M^T p^k + \lambda M^T (Mx^{k+1} - z^k) \\ \Leftrightarrow \quad & \left(x^{k+1}, -M^T(p^k + \lambda(Mx^{k+1} - z^k))\right) \in \partial f \\ \Rightarrow \quad & (p^k + \lambda(Mx^{k+1} - z^k), -Mx^{k+1}) \in A \quad . \end{aligned}$$

From the minimization step over z, we likewise know that

$$\begin{aligned} & 0 \in \partial g(z^{k+1}) - p^k + \lambda(z^{k+1} - Mx^{k+1}) \\ \Leftrightarrow \quad & (z^{k+1}, p^k + \lambda(Mx^{k+1} - z^{k+1})) = (z^{k+1}, p^{k+1}) \in \partial g \\ \Leftrightarrow \quad & (p^{k+1}, z^{k+1}) \in B \quad . \end{aligned}$$

Recall that

$$(I+S)^{-1} = \{ (u+\lambda b, v+\lambda b) \mid (u, b) \in B, (v, \lambda^{-1}(u - v) - b) \in A \} \quad .$$

For any $k \geq 1$, make the following substitutions:

$$u = p^k \qquad b = z^k \qquad v = p^k + \lambda(Mx^{k+1} - z^k) \quad .$$

Since $k \geq 1$, we have $(u, b) = (p^k, z^k) \in B$ by our analysis of the z minimization step. Also,

$$\lambda^{-1}(u - v) - b = \lambda^{-1}\big(p^k - (p^k + \lambda(Mx^{k+1} - z^k))\big) - z^k = -Mx^{k+1},$$

hence

$$(v, \lambda^{-1}(u - v) - b) = (p^k + \lambda(Mx^{k+1} - z^k), -Mx^{k+1}) ,$$

which we already know to be a member of A by our analysis of the x minimization step. Thus, the substitutions are legal, and we conclude that $(I+S)^{-1}$ contains

$$\begin{aligned}(u+\lambda b, v+\lambda b) &= (p^k + \lambda z^k, p^k + Mx^{k+1}) \\ &= (p^k + \lambda z^k, p^{k+1} + \lambda z^{k+1}) .\end{aligned}$$

The proof is complete. ■

Thus, at every iteration except possibly the first, the alternating direction method performs essentially the same calculation as the splitting method as applied to the specially defined maximal monotone operators A and B. Note that the minimization over x implements the "mapping step" (1) of section 5, whereas the minimization over z and the update of p perform the "re-representation step" (2). That the conclusion of Proposition 6 may not hold for k=0 is a direct manifestation of the general phenomenon noted at the end of section 5, namely that if the initial iterate does not lie in the operator B, it may take the algorithm an iteration to get "on track" by producing an iterate that is a member of B.

By the theory of the proximal point algorithm, we are now guaranteed (at least in finite dimension, where weak convergence and conventional convergence are equivalent) that the sequence $\{p^k + \lambda z^k\}$ is convergent. Since the operator $J_{\lambda,B} = (I+\lambda B)^{-1}$ is nonexpansive, it is (Lipschitz) continuous, hence the sequence $\{J_{\lambda,B}(p^k + \lambda z^k)\}$ is also convergent. But this sequence is identical with $\{p^k\}$, except possibly in its first element. Therefore, we may conclude that the sequences $\{p^k\}$ and $\{z^k\}$ are bounded and convergent, with $\{p^k\}$ converging to a solution of (D). From here, one may now pursue the usual convergence proofs for the alternating direction method.

9. Conclusion

In conclusion, we have illuminated the connection between a large number of existing algorithms. In particular, we have established that the Lions-Mercier splitting algorithm for maximal monotone operators is an instance of the proximal point algorithm, and hence that the alternating direction method of multipliers, already known to be an instance of Lions-Mercier, is also a member of the proximal point class. Furthermore, the algorithms of Spingarn and Rockafellar-Wets, already known to be proximal point variants, are in fact special cases of Lions-Mercier. Figure 4 illustrates the exact taxonomy of these algorithms.

The important point is that it is the decomposition principle of section 4 that makes possible the algorithm of Lions-Mercier, the decomposition method of Spingarn [19], and the alternating direction method of multipliers. It appears that one must keep the parameter sequence $\{r_k\}$ of the proximal point algorithm fixed at 1 to take advantage of this principle. An interesting question for further research is whether it is still possible to obtain convergence if one varies the parameter λ from iteration to iteration of the splitting algorithm.

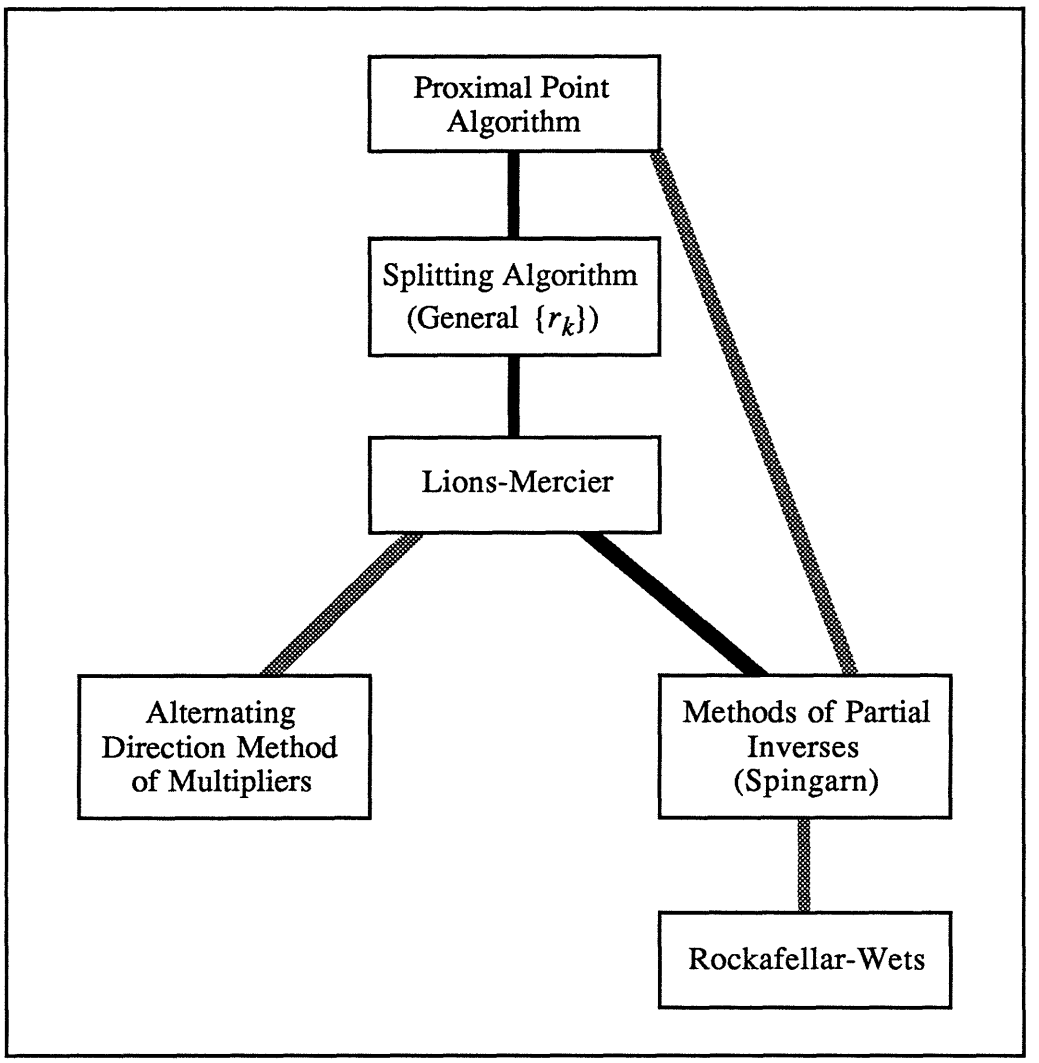

Figure 4: Taxonomy of algorithms discussed. Black lines indicate relationships established in this report; gray lines indicate relationships that were already known.

References

[1] BERTSEKAS, D. (1982), *Constrained Optimization and Lagrange Multiplier Methods* (New York: Academic Press).

[2] BERTSEKAS, D., TSITSIKLIS, J. (1989), *Parallel and Distributed Computation: Numerical Methods* (Englewood Cliffs: Prentice-Hall).

[3] BRÉZIS, H., (1973), *Opérateurs Maximaux Monotones et Semi-Groupes de Contractions dans les Espaces de Hilbert* (Amsterdam: North-Holland).

[4] FORTIN, M., GLOWINSKI, R. (1983), "On Decomposition-Coordination Methods Using an Augmented Lagrangian". In M. FORTIN, R. GLOWINSKI, editors, *Augmented*

Lagrangian Methods: Applications to the Solution of Boundary-Value Problems (Amsterdam: North-Holland).

[5] GABAY, D. (1983), "Applications of the Method of Multipliers to Variational Inequalities". In M. FORTIN, R. GLOWINSKI, editors, *Augmented Lagrangian Methods: Applications to the Solution of Boundary-Value Problems* (Amsterdam: North-Holland).

[6] GABAY, D., MERCIER B. (1976), "A Dual Algorithm for the Solution of Nonlinear Variational Problems via Finite Element Approximations". *Computers and Mathematics with Applications* **2**:17-40.

[7] GLOWINSKI, R., LE TALLEC, P. (1987), *Augmented Lagrangian Methods for the Solution of Variational Problems*. MRC Technical Summary Report #2965, Mathematics Research Center, University of Wisconsin – Madison.

[8] GLOWINSKI, R., MARROCO, A. (1975), "Sur L'Approximation, par Elements Finis d'Ordre Un, et la Resolution, par Penalisation-Dualité, d'une Classe de Problemes de Dirichlet non Lineares". *Revue Française d'Automatique, Informatique et Recherche Opérationelle* **9**(R-2):41-76.

[9] LIONS, P. L., MERCIER, B. (1979), "Splitting Algorithms for the Sum of Two Nonlinear Operators". *SIAM Journal on Numerical Analysis* **16**(6):964-979.

[10] MARTINET, B. (1972), "Determination Approchée d'un Point Fixe d'une Application Pseudo-Contractante. Cas de l'Application prox". *Comptes Rendus de l'Academie des Sciences, Paris, Série A* **274**:163-165.

[11] MINTY, G. J. (1962), "Monotone (Nonlinear) Operators in Hilbert Space". *Duke Mathematics Journal* **29**:341-346.

[12] ROCKAFELLAR, R. T. (1970), "On the Maximal Monotonicity of Subdifferential Mappings". *Pacific Journal of Mathematics* **33**(1):209-216.

[13] ROCKAFELLAR, R. T. (1970), *Convex Analysis* (Princeton: Princeton University Press).

[14] ROCKAFELLAR, R. T. (1970), "On the Maximality of Sums of Nonlinear Monotone Operators". *Transactions of the American Mathematical Society* **149**:75-88.

[15] ROCKAFELLAR, R. T. (1976), "Augmented Lagrangians and Applications of the Proximal Point Algorithm in Convex Programming". *Mathematics of Operations Research* **1**(2):97-116.

[16] ROCKAFELLAR, R. T. (1976), "Monotone Operators and the Proximal Point Algorithm". *SIAM Journal on Control and Optimization* **14**(5):877-898.

[17] ROCKAFELLAR, R. T., WETS R. J.-B. (1987), *Scenarios and Policy Aggregation in Optimization under Uncertainty*. Working paper WP-87-119, International Institute for Applied Systems Analysis, Laxenberg, Austria.

[18] SPINGARN, J. E. (1983), "Partial Inverse of a Monotone Operator". *Applied Mathematics and Optimization* **10**:247-265.

[19] SPINGARN, J. E. (1985), "Application of the Method of Partial Inverses to Convex Programming: Decomposition". *Mathematical Programming* **32**:199-233.