A Unifying Geometric Solution Framework and Complexity Analysis for Variational Inequalities

Thomas L. Magnanti and Georgia Perakis

OR 276-93 December 1993

A unifying geometric solution framework and complexity analysis for variational inequalities *

Thomas L. Magnanti † Georgia Perakis ‡

February 1993

Revised December 1993

Abstract

In this paper, we propose a concept of polynomiality for variational inequality problems and show how to find a near optimal solution of variational inequality problems in a polynomial number of iterations. To establish this result we build upon insights from several algorithms for linear and nonlinear programs (the ellipsoid algorithm, the method of centers of gravity, the method of inscribed ellipsoids, and Vaidya's algorithm) to develop a unifying geometric framework for solving variational inequality problems. The analysis rests upon the assumption of strong-f-monotonicity, which is weaker than strict and strong monotonicity. Since linear programs satisfy this assumption, the general framework applies to linear programs.

*Preparation of this paper was supported, in part, by NSF Grant 9312971-DDM from the National Science Foundation.

†Sloan School of Management and Operations Research Center, MIT, Cambridge, MA 02139.

‡Operations Research Center, MIT, Cambridge, MA 02139.

1 Introduction

In this paper, in the context of a general geometric framework, we develop complexity analysis for solving a variational inequality problem

$$VI(f,K): \quad Find \;\; x^{opt} \in K \subseteq R^n: \; f(x^{opt})^t(x - x^{opt}) \geq 0, \; \forall x \in K, \tag{1}$$

defined over a compact, convex (constraint) set K in R^n. In this formulation $f: K \subseteq R^n \to R^n$ is a given function and x^{opt} denotes an (optimal) solution of the problem. Variational inequality theory provides a natural framework for unifying the treatment of equilibrium problems encountered in problem areas as diverse as economics, game theory, transportation science, and regional science. Variational inequality problems also encompass a wide range of generic problem areas including mathematical optimization problems, complementarity problems, and fixed point problems. Any linear programming problem

$$Min_{x \in P} \; c^t x,$$

defined over a polyhedron P can be cast as a variational inequality problem as follows:

$$VI(c,P): \quad Find \;\; x^{opt} \in P \subseteq R^n \; : \; c^t(x - x^{opt}) \geq 0, \; \forall x \in P. \tag{2}$$

Therefore, linear programming is a special case of a variational inequality problem (1), with $f(x) = c$ and with a polyhedron P as a constraint set.

Let $MIN(F,K)$ denote an optimization problem with an objective function $F(x)$ and constraints $x \in K$. The function $f(x)$ of the variational inequality problem $VI(f,K)$ associated with the minimization problem $MIN(F,K)$ is defined by $f(x) = \nabla F(x)$. As is well-known, a variational inequality problem $VI(f,K)$ is equivalent to a (strictly) convex nonlinear programming problem $MIN(F,K)$ (that is they have the same set of solutions) if and only if the Jacobian matrix of the problem function f is a symmetric and positive semidefinite (definite) matrix.

Our goal in this paper is to introduce and study a concept of complexity for variational inequality problems. We wish to address several questions:

• *What does polynomial efficiency mean for the general asymmetric variational inequality problem?*

• *How can we measure polynomial efficiency for this problem class?*

• *Using this notion, can we develop efficient (polynomial-time) algorithms for solving general asymmetric variational inequality problems?*

In particular, we establish the following results:

1. We introduce the concept of polynomial efficiency for variational inequality problems. We propose a notion of an input size and a "near optimal" solution for the general $VI(f, K)$ problem, extending results from linear and nonlinear programming.

2. Using these notions, we show how to find a "near optimal" solution of the $VI(f, K)$ problem in a polynomial number of iterations. We achieve this result by providing a general geometric framework for finding a "near optimal" solution, assuming a condition of strong-f-monotonicity on the problem function f. We also analyze some properties of this condition which is weaker than strict or strong monotonicity, assumptions that researchers typically impose in order to prove convergence of algorithms for $VI(f, K)$. Linear programs, when cast as variational inequality problems, satisfy neither of these conditions but trivially satisfy strong-f-monotonicity.

3. This general framework unifies some algorithms known for "efficiently" solving linear programs. As a result, we also show that these algorithms extend to solving variational inequality problems since they become special cases of this framework. The unification of these algorithms also provides a better understanding of their common properties and a common proof technique for their convergence.

4. As a subproblem of our general geometric framework, we need to be able to optimize a linear objective function over a general convex set. Our analysis also provides a polynomial-time algorithm for finding a near optimal solution for this problem. (Lovasz [15] develops a different analysis for essentially the same algorithm for this problem.)

Our approach is motivated by Khatchiyan's [11] ellipsoid algorithm for linear programming. The ellipsoid algorithm, introduced by Shor [26] and Yudin and Nemirovski [23] for solving convex optimization problems, has certain geometric properties, which we further exploit in this paper to solve variational inequality problems efficiently. (For a complexity analysis of the ellipsoid algorithm applied to convex optimization over the unit cube, see Vavasis [30]). Several geometric algorithms proposed in the literature, primarily for linear programming problems, and applied in a generalized form to convex optimization problems, have similar geometric properties. The following algorithms are of particular interest in this paper:

1. The Method of Centers of Gravity (Levin [14], 1968).
2. The Ellipsoid Algorithm (Khatchiyan [11], 1979).
3. The Method of Inscribed Ellipsoids (Khatchiyan, Tarasov, Erlikh [12], 1988).
4. Vaidya's algorithm (Vaidya [29], 1989).

Motivated by the celebrity of these geometric algorithms for solving linear programming problems, in this paper we examine the question of whether similar geometric algorithms will efficiently solve variational inequality problems. The general geometric framework, which we describe in detail in Section 3, stems from some common geometric characteristics that are shared by all these algorithms: they all generate a sequence of "nice" sets of the same type, and use the notion of a center of a "nice" set. At each iteration, the general framework maintains a convex set that is known to contain all solutions of the variational inequality problem and "cuts" the set with a hyperplane, reducing its volume and generating a new nice set. After a polynomial number of iterations, one of the previously generated centers of the nice sets will be a near optimal solution to the variational inequality problem.

Our analysis in this paper rests upon the condition of strong-f-monotonicity. Gabay [5], implicitly introduced the concept of strong-f-monotonicity and Tseng, [28], using the name co-coercivity, explicitly stated this condition. Magnanti and Perakis ([18], [17] and [25]) have used the term strong-f-monotonicity for this condition, a choice of terminology that highlights the similarity between this concept and the terminology strong monotonicity, which has become so popular in the literature. We use the following notation and definitions throughout this paper.

The *argmin* of a real-valued function F over a set K is defined as

$$argmin_{x \in K} F(x) = y \in \{x^* : \quad F(x^*) = min_{x \in K} F(x)\}.$$

Definition 1 .

1. *f is <u>monotone</u> on K if $(f(x) - f(y))^t(x - y) \geq 0 \quad \forall x, y \in K$.*

2. *f is <u>strictly monotone</u> on K if $(f(x) - f(y))^t(x - y) > 0 \quad \forall x, y \in K, \quad x \neq y$.*

3. *f is <u>strongly monotone</u> on K if for some positive constant a, $(f(x) - f(y))^t(x - y) \geq a\|x - y\|_2^2 \quad \forall x, y \in K$. (In this expression $\|.\|_2$ denotes the Euclidean distance.)*

4. *f is <u>monotone</u> (strictly or strongly) at a point x^* if $(f(x) - f(x^*))^t(x - x^*) \geq 0 \quad \forall x \in K$ ($> 0 \quad \forall x \neq x^*$ for strict monotonicity, while for strong monotonicity $\exists a > 0$ such that $\geq a\|x - x^*\|_2^2$).*

If f is differentiable, the following conditions imply monotonicity:

1. f is monotone if the Jacobian matrix ∇f is positive semidefinite on the set K.

2. f is strictly monotone if the Jacobian matrix ∇f is positive definite on the set K.

3. f is strongly monotone if the Jacobian matrix ∇f is uniformly positive definite on the set K.

We now review some facts concerning matrices.

Definition 2 . *A positive definite and symmetric matrix S defines an inner product $(x, y)_S = x^t S y$. This inner product induces a norm with respect to the matrix S via*

$$\|x\|_S^2 = x^t S x.$$

This inner product is related to the Euclidean distance since

$$\|x\|_S = (x,x)_S^{1/2} = (x^t S x)^{1/2} = \|S^{1/2}x\|_2.$$

In this expression, $S^{1/2}$ is the matrix satisfying $S^{1/2}S^{1/2} = S$. This inner product, in turn, induces an operator norm on any operator B. Namely:

$$\|B\|_S = \sup_{\|x\|_S=1} \|Bx\|_S.$$

The operator norms $\|B\|_S$ and $\|B\| \equiv \|B\|_I$ are related since

$$\|B\|_S = \sup_{\|x\|_S=1} \|Bx\|_S = \sup_{\|S^{1/2}x\|_2=1} \|S^{1/2}Bx\|_2 =$$

$$= \sup_{\|S^{1/2}x\|_2=1} \|S^{1/2}BS^{-1/2}S^{1/2}x\|_2 = \|S^{1/2}BS^{-1/2}\|.$$

So

$$\|B\|_S = \|S^{1/2}BS^{-1/2}\|$$

and similarly

$$\|B\| = \|S^{-1/2}BS^{1/2}\|_S.$$

Finally, if λ is a positive scalar and K is any set, we let $K(\lambda) = \frac{1}{\lambda}K$ be a scaled set defined as

$$x \in K \text{ if and only if } \frac{1}{\lambda}x \in K(\lambda).$$

In this paper we often use the following estimate for the volume of a ball $S(z,R)$ of radius R about the point z:

$$R^n n^{-n} \leq vol(S(z,R)) = R^n vol(S(0,1)) = R^n \frac{\pi^{n/2}}{\Gamma(n/2+1)} \leq R^n 2^n.$$

In this expression, $\Gamma(x) := \int_0^\infty e^{-t}t^{x-1}dt$ for $x > 0$ is the gamma-function with $\Gamma(n) = (n-1)!$. See Groetschel, Lovasz and Schrijver [6].

This paper is organized as follows. In Section 2 we define what we mean by the input size and a near optimal solution for a variational inequality problem. In Section 3 we develop a unifying geometric framework for solving the general asymmetric variational inequality problem. In Section 4 we examine the assumptions we need to impose on the problem in order to develop our results. We discuss some properties of strong-f-monotonicity and its relationship with the traditional conditions of strict and strong monotonicity. In Section 5 we develop complexity results for the general geometric framework and the four special cases we consider. We show that the geometric framework encompasses the four algorithms known for solving linear and nonlinear programs. Therefore, these four algorithms solve the variational inequality problem as well. Our complexity results show that the volume of an underlying set, known to contain all optimal solutions, becomes "small" in a polynomial number of iterations (polynomial in the number n of variables and the input size L_1 of the problem). Moreover, we prove that whenever the problem function satisfies the strong-f-monotonicity condition, some point in the "small" underlying set is a near optimal solution to the variational inequality problem. Finally, in Section 6 we summarize our conclusions and address some open questions.

2 The input size of a VIP

In linear programming we measure the complexity of an algorithm as a function of the size of the input data given by the problem. In this section we give a definition of the input size of a variational inequality problem VIP that corresponds to the definition of the input size of an LP.

In a linear programming problem

$$min_{x \in P}\ c^t x$$

the feasible set is a polyhedron

$$P = \{x \in R^n : \quad Ax = b, \quad x \geq 0\},$$

defined by an $m \times n$ integer matrix $A = (a_{i,j})_{m \times n}$ and a vector $b = (b_i)$ in Z^m.

Definition 3 . *The input size of a linear program is the length of the input* (c, A, b) *when represented as a sequence of binary digits.*

When we are interested in solving the feasibility problem (i.e., find a point $x \in P$) the input size L is defined as

$$L = mn + \lceil log|\pi| \rceil = mn + \lceil \sum_j \sum_i (log|a_{ij}|) \rceil + \lceil \sum_i (log|b_i|) \rceil.$$

In this expression π is the product of nonzero terms in A and b and $\lceil x \rceil$ is the integer round-up of x. Therefore, the input size is bounded by

$$L \leq mn(\lceil log|data_{max}| \rceil + 1) + m\lceil log|data_{max}| \rceil = O(mn\lceil log|data_{max}| \rceil),$$

with $data_{max} = max_{\{1 \leq i \leq m, 1 \leq j \leq n\}}(a_{i,j}, b_i)$. For problems that satisfy the similarity assumption, i.e., $data_{max} = O(p(n))$ for some polynomial $p(n)$,

$$L \leq O(mnlog(n)).$$

The natural question that arises at this point is,

what is the corresponding definition of L for VIPs defined over general convex sets?

Before answering this question, we state some assumptions we impose and set some notation and definitions.

ASSUMPTIONS

A1. The feasible set K is a convex, compact subset of R^n, with a nonempty interior.

A2. The function $f : K \subseteq R^n \rightarrow R^n$ of $VI(f, K)$ is a nonzero, continuous function, that satisfies the condition of strong-f-monotonicity.

Definition 4 . *A function $f : K \subseteq R^n \rightarrow R^n$ is strongly-f-monotone if there exists a positive constant $a > 0$ for which*

$$[f(x) - f(y)]^t[x - y] \geq a\|f(x) - f(y)\|_2^2 \text{ for all } x, y \in K.$$

The constant a is the strong-f-monotonicity constant. In Section 4 we will show that for linear programs any constant a satisfies the strong-f-monotonicity condition.

We will show that assumptions A1 and the continuity of f imply that the problem function f is a uniformly bounded function for some constant M (i.e., $\|f(x)\| \leq M \ \forall x \in K$).

Our complexity results will involve this constant.

Section 4 contain a further analysis of these assumptions.

We now define an input size for the feasibility problem.

Definition 5 . *L and l are positive constants given as part of the input of the problem that satisfy the following conditions:*

1. *The feasible set K is contained in a ball of radius 2^L, centered at the origin.*

2. *The feasible set K contains a ball of radius 2^{-l}.*

3. *$L \geq logn + 3$.*

Observe that if we scale K by a factor $\lambda \in R^+$, then the radii 2^L and 2^{-l} scale by a factor λ as well; that is, if $K(\lambda) = \frac{1}{\lambda}K$, then the radii $2^{L(\lambda)}$ and $2^{-l(\lambda)}$ for the inscribing and inscribed balls for the set $K(\lambda)$ become $2^{L(\lambda)} = \frac{2^L}{\lambda}$ and $2^{-l(\lambda)} = \frac{2^{-l}}{\lambda}$.

Without loss of generality, we assume that the feasible set K contains the origin. We can relax this assumption by replacing Definition 5 with the statement that the feasible set K is contained in a ball of radius 2^L centered at some point a_0 in K.

The constant L we have defined for a linear program defined over a bounded polyhedron satisfies conditions 1-3 of Definition 5 because the basic feasible solutions of a linear programming problem (2) are contained in a ball of radius 2^L (for a proof of this result, see Papadimitriou and Steiglitz [24], page 181). Moreover, if the polyhedron P has a nonempty interior, it contains a ball of radius 2^{-l} which is related to the input size L by $l = (L-1)(2n^2 + 2n)$ (see Nemhauser and Wolsey [22]). In analyzing linear programs, researchers typically assume that the underlying data are all integer. Therefore, in this setting scaling (see the comment after Definition 5) is not an issue.

For the general variational inequality problem, there is no prevailing notion of input size as defined in linear programming. Definition 5 provides a definition for the feasibility part of the problem. Clearly, L and l depend on the feasible set K, i.e., $L = L(K)$ and $l(K)$, and are part of our input data.

At this point we might observe that even for variational inequality problems that have an equivalent nonlinear programming formulation, with a convex, Lipschitz continuous objective function, there is no finite algorithm for finding the global optimum exactly (see Vavasis [30]). Nonlinear programming algorithms find near optimal solutions. Similarly, our goal in solving a variational inequality problem is to find a near optimal solution defined in the following sense, which generalizes Vavasis notion of an "ϵ-approximate" global optimum for nonlinear programming problems in [30].

Definition 6 . *Let $\epsilon \geq 0$ be a given constant; an ϵ-near optimal solution is a point $x^* \in K$ satisfying the following system*

$$f(x^*)^t(x - x^*) \geq -\epsilon 2^{-l} M \quad \forall x \in K.$$

We will refer to ϵ as the nearness constant. Observe that if ϵ equals zero then x^* is an optimal solution x^{opt} of $VI(f, K)$.

When a $VI(f, K)$ has an equivalent convex programming formulation, an ϵ-near optimal solution in the sense of Definition (6) is also an "ϵ-approximate" solution as defined in [30], since

$$F(x) - F(x^*) \geq f(x^*)^t(x - x^*),$$

and so x^* is a near optimal solution, in objective value, to the problem $min_{x \in K} F(x)$.

So far we have defined the constants L and l solely in terms of the feasible set K: these definitions do not depend on the problem function f. We now define a positive constant L_1 to account not only for the feasible set, but also for the problem function f.

Definition 7 . *The size of a variational inequality (describing both its feasible set as well as its problem function f) is a positive constant L_1 with the value*

$$L_1 = log(\frac{9.2^{4L+3l}}{\epsilon^2 M a}).$$

Note that the constant L_1 depends on the nearness constant ϵ, the strong-f-monotonicity constant a, the uniform boundedness constant M, and the constants L and l defined previously.

Proposition 2.1

The nearness constant ϵ as defined in Definition 6, and the constant L_1 as defined in Definition 7 are scale invariant.

Proof:

Suppose that we scaled the feasible set K by a scalar λ. Definition 5 implies that $2^{L(\lambda)} = \frac{2^L}{\lambda}$ and $2^{-l(\lambda)} = \frac{2^{-l}}{\lambda}$. The strong-f-monotonicity constant $a(\lambda) = \frac{a}{\lambda}$. The uniform boundedness constant is scale invariant since

$$M(\lambda) = sup_{y \in K(\lambda)} \|f(\lambda y)\| = sup_{x \in K} \|f(x)\| = M.$$

Moreover, we can easily see that the nearness constant ϵ of Definition 6 is scale invariant (i.e., $\epsilon(\lambda) = \epsilon$), since

$$x^* \in K \text{ satisfies } f(x^*)^t(x - x^*) \geq -\epsilon 2^{-l} M \text{ for all } x \in K \Leftrightarrow$$

$$y^* \in K(\lambda) \text{ satisfies } f(\lambda y^*)^t(y - y^*) \geq -\epsilon \frac{2^{-l}}{\lambda} M \text{ for all } y \in K(\lambda).$$

Then $L_1 = log(\frac{9.2^{4L+3l}}{\epsilon^2 Ma}) = log(\frac{9.\frac{2^{4L}}{\lambda^4}\lambda^3 2^{3l}}{\epsilon^2 M \frac{a}{\lambda}}) = log(\frac{9.2^{4L(\lambda)+3l(\lambda)}}{\epsilon(\lambda)^2 M(\lambda) a(\lambda)})$. Finally, if we scaled the problem function f, then ϵ would still be scale invariant, since Definition 6 involves f in both sides of the inequality (through the uniform boundedness constant M on the righthand side). Q.E.D.

In closing this section, we might note that we could define an ϵ-near optimal solution as follows

Definition 8 . *An ϵ-near optimal solution is a point $x^* \in K$ satisfying the condition*

$$f(x^*)^t(x - x^*) \geq -\epsilon \ \ \forall x \in K.$$

In this case the nearness constant ϵ is not scale invariant, since if we scale the feasible set K by λ, the constant becomes $\frac{\epsilon}{\lambda}$. However, in this case if we define

$$L_1 = 4L + l + log(\frac{9M}{\epsilon^2 a}) = log(\frac{9M2^{4L+l}}{\epsilon^2 a}),$$

then it is easy to see, as in the analysis of Proposition 2.1, that L_1 is scale invariant.

The algorithms that we present in this paper compute an ϵ-near optimal solution in a polynomial number of iterations (in terms of n and L_1). For notational convenience, in our analysis we use Definition 8. However, if we replace ϵ by $\epsilon 2^{-l} M$ and the constant L_1 by $L_1 = log(\frac{9.2^{4L+3l}}{\epsilon^2 Ma})$ throughout, the analysis remains the same, so the results apply to the scale invariant version of the model (that is, Definition 6).

Remarks:

1. Our definition of L and l are geometric. We could also provide algebraic definitions. Namely, for some fixed point $a_0 \in K$ (which we have assumed as the origin)

$$L = log\ (argmax_{y \in K} \|y - a_0\|).$$

To avoid a specific choice of a_0 we could define

$$L = log\ (argmin_{x \in K}[max_{y \in K} \|y - x\|]).$$

2. Suppose that the constraints of the feasible set K are given explicitly by convex functions g_j for $j = 1, ..., m$ in the sense that $K = \{x \in R^n : \ \ g_j(x) \leq 0 \ \ j = 1, ..., m\}$. Suppose that we are given points x_j^* for $j = 1, .., m$ satisfying the conditions $g_j(x_j^*) = 0$ and $g_i(x_j^*) \leq 0$. The polyhedron $P = \{x \in R^n : \ \ \nabla g_j(x_j^*)(x - x_j^*) \leq 0, \ \ j = 1, ..., m\}$ contains the feasible set K and its faces are tangent to the constraints of the feasible set K. We could define the constant L for the variational inequality problem to be the input size L (as defined in Definition 3) of this polyhedron P. When the feasible set K is a polyhedron, the problem is linearly constrained and therefore its input size (as defined here) for the set K coincides with Definition 3 of the input size in the linear case.

3. Observe that in the previous definitions we did not specify any restrictions on the choice of the nearness constant ϵ. For linear programs we can set ϵ in Definition (8) equal to 2^{-L} in order to be able to round an ϵ-near optimal solution to an exact solution. For general variational inequality problems rounding to an exact solution is not possible in finite time.

Nevertheless, we could still express the input size L_1 in terms of only L, M, and a (and as $O(L)$) if we select $\epsilon = 2^{-L}$ in Definition (8) and let $l = (2n^2 + n)(L - 1)$.

3 A general geometric framework for solving the VIP efficiently

In the introduction we mentioned four known algorithms for solving linear programming problems in a polynomial number of iterations. For linear programs, when the algorithms have converged (in the sense of the shrinking volume of the underlying set), the algorithms' iterate is a "near" optimal solution to the problem. We would like to establish a similar result for variational inequalities. In this section we describe a general class of polynomial-time algorithms for solving variational inequalities that includes these four algorithms as special cases. For any algorithm in this class, we show that if the problem function satisfies the strong-f-monotonicity condition, assumption A2, then the final point encountered is a near optimal solution. The next (well-known) lemma describes the key property that underlies these four algorithms for the VIP.

LEMMA 3.1: If the problem function f is monotone, then for any $y \in K$ the set $K_y = \{x \in K : \quad f(y)^t(y - x) \geq 0\}$ contains all optimal solutions of the variational inequality problem.

Proof:

Let $y \in K$. An optimal solution x^{opt} of $VI(f, K)$ belongs in the feasible set K and satisfies (1), that is, $f(x^{opt})^t(x - x^{opt}) \geq 0 \;\; \forall x \in K$. Since $y \in K$, the monotonicity of f shows that $f(y)^t(y - x^{opt}) \geq f(x^{opt})^t(y - x^{opt}) \geq 0$, and therefore, $x^{opt} \in K_y$. Q.E.D.

For any $y \in K$, we refer to the inequality $f(y)^t(y - x) \geq 0$ as an optimality cut because it retains all (optimal) solutions of the variational inequality problem. As we noted in the introduction, all four algorithms generate a "nice" set, with a "center". They perform one of the following two computations:

A feasibility cut. Given the "center" of a "nice" set, if the "center" does not lie in the feasible set, we "cut" the "nice" set at the center with a separating hyperplane, therefore separating the center from the feasible set and moving towards feasibility.

An optimality cut. If the center is feasible but not a (near) optimal solution, we "cut" the given "nice" set at its center with an optimality cut therefore separating the center from the set of the optimal solutions and moving closer to an optimal solution.

Let $OC \subseteq \{1, 2, ..., \}$ denote the set of iterations for which we make an optimality cut and for any k, let $OC(k)$ denote the set of iterations, within the first k, for which we make an optimality cut, i.e., $OC(k) = OC \cap \{1, 2, ..., k\}$.

Table I shows the definitions of the "nice" sets and "centers" (that we define for a general framework Section 3.1) for these four algorithms. In Section 3.2 we provide more details concerning the "nice"sets and "centers" of each algorithm.

Table I: "Nice" sets and "centers".

Algorithm	**"Nice" Set P_k**	**"Center" x^k**
General framework	General "nice" set	General "Center"
Method of centers of gravity	Convex, compact set	Center of gravity
Ellipsoid Method	Ellipsoid	Center of the ellipsoid
Method of inscribed ellipsoids	Convex, compact set	Center of maximum inscribed ellipsoid
Vaidya's algorithm	Polytope	Volumetric center

3.1 The General Geometric Framework

In this section we describe a general geometric framework that incorporates an entire class of geometric algorithms that share the common characteristics mentioned in the last section.

COMMON CHARACTERISTICS - DEFINITIONS

Definition 9 . *"Nice" sets $\{P_k\}_{k=0}^{\infty}$ are a sequence of subsets of R^n satisfying the following conditions:*

1. *The sets P_k are compact, convex subsets of R^n, with a nonempty interior.*

2. *The sets P_k belong in the same class of sets (ellipsoids, polytopes, simplices, convex sets).*

3. *These sets satisfy the condition $P_{k+1} \supseteq P_k \cap H_k^c$, for a half space H_k^c defined by a hyperplane H_k.*

4. *We choose the set P_0 so that it contains the set K and has a volume no more than $2^{n(2L+1)}$.*

Definition 10 . *A sequence of points $\{x^k\}_{k=0}^{\infty} \subseteq R^n$ is a sequence of centers of the "nice" sets $\{P_k\}_{k=0}^{\infty}$ if*

1. *$x^k \in interior(P_k)$.*

2. *$P_k \supseteq P_{k-1} \cap H_{k-1}^c = P_{k-1} \cap \{x \in R^n : c_{k-1}^t x \geq c_{k-1}^t x^{k-1}\}$ for a given vector c_{k-1}.*

3. *$vol(P_k) \leq b(n) vol(P_{k-1})$, or $vol(P_k) \leq [b(n)]^k 2^{n(2L+1)}$ for a function $b(n)$, $0 < b(n) < 1$, of the number n of the problem variables.*

These conditions ensure that the algorithm makes sufficiently deep cuts of the "nice" sets at each iteration, and so produces a substantial reduction in their volumes.

To initiate this algorithm, one possibility is to choose P_0 as a simplex $P_0 = \{x \in R^n : \sum_{i=1}^{n} x_i \leq n2^L, \; x_i \geq -2^L, i = 1, ..., n\}$. In this case, $P_0 \subseteq S(0, n2^L)$ whose volume is no more than $n^n 2^{nL} 2^n = 2^{nL+n+nlogn} \leq 2^{2nL+n}$.

We are now ready to describe the general geometric framework for solving the VIP in polynomial-time.

THE GENERAL GEOMETRIC FRAMEWORK

ITERATION 0:

Start by constructing an initial "nice" set $P_0 \supseteq K_0 = K$. Find its center x^0. Set $k = 1$.

ITERATION k:

Part a: (Feasibility cut: cut toward a feasible solution.)

If $x^{k-1} \in K$, then go to part b.

If $x^{k-1} \notin K$, then choose a hyperplane through x^{k-1} that supports K (that is, one halfspace defined by the hyperplane contains K) .

An oracle returns a vector c for which: $c^t x \geq c^t x^{k-1}$ for all $x \in K$.

Update the "nice" set P_{k-1} by constructing a new "nice" set:

$P_k \supseteq P_{k-1} \cap \{x \in R^n : c^t x \geq c^t x^{k-1}\}$ and the set $K_k := K_{k-1}$. Find the center x^k of P_k.

Set $k \leftarrow k+1$ and go to iteration k.

Part b: (Optimality cut: cut towards an optimal solution).

Let $c = -f(x^{k-1})$ and set $K_k = K_{k-1} \cap \{x \in R^n : c^t x \geq c^t x^{k-1}\}$. Construct the new "nice" set $P_k \supseteq P_{k-1} \cap \{x \in R^n : c^t x \geq c^t x^{k-1}\}$ and find its center x^k. Set $k \leftarrow k+1$ and repeat iteration k.

STOP in $k^* = O(-\frac{nL_1}{logb(n)})$ iterations of the algorithm. Find an $\frac{\epsilon}{2}$-near optimal solution $y_{\frac{\epsilon}{2}}^{k(i)}$ to the problem $min_{y \in K} f(x^i)^t(y - x^i)$. The proposed solution to the VIP is $x^* = x^j$ with $j = argmax_{i \in OC(k^*)}[f(x^i)^t(y_{\frac{\epsilon}{2}}^{k(i)} - x^i)]$.

As we will see in Section 5, within k^* iterations, this geometric algorithm will determine that one of the points x^i for $i \in OC(k^*)$ is an ϵ-near optimal solution. To ascertain which of these ponts is a near optimal solution, we need to solve a series of optimization problems, namely, for each $i \in OC(k^*)$, find an $\frac{\epsilon}{2}$-near optimal solution to $min_{y \in K} f(x^i)^t(y - x^i)$. That is, we need to be able to (approximately) optimize a linear objective function over the convex set K. Since this problem is also a variational inequality problem with a constant $f(x^i)$ as a problem function, we can apply the geometric algorithm to it as well. In this special case, as we will show, it is easy to determine an $\frac{\epsilon}{2}$-near optimal solution: it is the point $y_{\frac{\epsilon}{2}}^{k(i)}$ from the points y^j for $j \in OC(k(i))$ that has the minimum value of $f(x^i)^t y^j$.

Alternatively, we can apply a stopping criterion at each iteration and terminate the algorithm earlier. We check whether

$$max_{i \in OC(k)}[(f(x^i)^t(y_{\frac{\epsilon}{2}}^k(i) - x^i)]) \geq -\frac{\epsilon}{2}.$$

Then the general framework terminates with the knowledge that one of the points x^i is an ϵ-near optimal solution, in *at most* $k^* = O(-\frac{nL_1}{logb(n)})$ iterations. We will establish this result in Theorem 5.1 and Theorem 5.2.

In Section 5 we show that if the problem function f satisfies the strong-f-monotonicity condition, then the point x^* is indeed a near optimal solution.

To help in understanding this general geometric framework, in Figure 1 we illustrate one of its iterations.

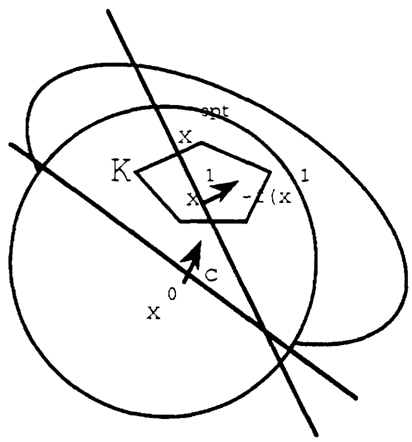

Figure 1: The general geometric framework

4 Assumptions for convergence of the general framework to a near optimal solution

In this section we analyze assumptions A1 and A2 that we introduced in Section 2 and that we impose on the variational inequality problem in order to prove convergence of the general framework (and its four special cases) and establish complexity results. We intend to show that these assumptions are quite weak. For convenience, we repeat the statement of these assumptions.

A1. The feasible set K is a convex, compact subset of R^n, with a nonempty interior. (Therefore for some positive constants L and l, $K \subseteq S(0, 2^L)$ and K contains a ball of radius 2^{-l}. These constants are given as an input with the feasible set. We assume $L \geq logn + 3$).

A2. The function $f : K \subseteq R^n \rightarrow R^n$ of $VI(f, K)$ is a nonzero continuous function, satisfying the strong-f-monotonicity condition: There exists a positive constant $a > 0$ for which

$$[f(x) - f(y)]^t[x - y] \geq a\|f(x) - f(y)\|_2^2 \;\; \forall x, y \in K.$$

Assumptions A1 and A2 guarantee that the general geometric framework is (globally) convergent to a VIP optimal solution. Moreover, these assumptions enable us to prove complexity results for the VIP. Namely, the geometric framework that we introduced in Section 3 (including, as special cases, variational inequality versions of the four algorithms we have mentioned) computes an ϵ-near optimal VIP solution in a polynomial number of iterations.

We now examine the implications of these assumptions.

1. *Compactness and Continuity*

   If the problem function f is continuous on the compact feasible set K, then Weierstrass' Theorem implies that f is also a uniformly bounded function on K, i.e., $\|f(x)\| = \|f(x)\|_2 \leq M$. We need an estimate of the constant M for the complexity results we develop. Let $\|.\|$ denote the operator norm defined as $\|f\| = sup_{x \neq 0} \frac{\|f(x)\|}{\|x\|}$. Since by assumption, K is a bounded set (included in a ball of radius 2^L centered at the origin), the operator norm inequality implies that $\|f(x)\|$ for $x \in K$ is, in general, uniformly boundedby the constant $M = \|f\|2^L$. When $f(x) = c$, the VIP becomes an LP problem with $M = \|c\|$. For affine $VIPs$ with $f(x) = Qx - c$, $M = \|Q\|2^L + \|c\|$.

2. *Nonempty Interior and Boundedness of the set K*

   Let $z \in K$ and let R be a given positive constant. For the convergence results we introduce next, we need to estimate the volume of the set $S(x^{opt}, R) \cap K$. The lemma we present next gives us the estimate

$$vol[S(x^{opt}, R) \cap K] \geq (\frac{R}{n2^{L+l+1}})^n.$$

LEMMA 4.1:

Let K be a convex set that (i) is contained in a ball of radius 2^L about the origin, and (ii) contains a ball S' of radius 2^{-l}. Then if B is a ball of radius $R \leq 2^{L+1}$ about any point P in K (i.e., $B = S(P, R)$),

$$vol[B \cap K] \geq (\frac{R}{n2^{L+l+1}})^n.$$

Proof:

Since K is contained in a ball of radius 2^L about the origin, the ball $S(P, 2^{L+1})$ must contain the set S'. Let $\lambda = \frac{R}{2^{L+1}}$ and consider the one-to-one transformation T of S' defined by

$$y = T(y') = \lambda y' + (1 - \lambda)P \quad \text{for any} \quad y' \in S'.$$

Let z' be the center of the ball S' and $z = T(z')$. Note that for any $y \in S = T(S')$,

$$\|y - z\| = \|\lambda(y' - z')\| = |\lambda| \|y' - z'\|.$$

Therefore, $\|y - z\| \leq \lambda 2^{-l}$ if and only if $\|y' - z'\| \leq 2^{-l}$ and so $S = T(S')$ is a ball with radius $\lambda 2^{-l}$ about $z = T(z')$. Note further that if $y \in S$, then

$$\|y - P\| = \|\lambda(y' - P)\| \leq \lambda(2^{L+1}) = R.$$

So $S \subseteq B$. Moreover, by convexity, $S \subseteq K$. Consequently,

$$vol[B \cap K] = vol(S(P, R) \cap K) \geq vol(S) \geq [\lambda 2^{-l}]^n n^{-n} \geq (\frac{R}{n2^{L+l+1}})^n.$$

Q.E.D.

3. *Monotonicity*

When the function f is a monotone, continuous function (which assumption A2 implies) and the feasible set K is a convex, closed set (assumption A1), then the following problem formulations are equivalent:

a. Find $x^{opt} \in K \subseteq R^n: \ f(x^{opt})^t(x - x^{opt}) \geq 0, \ \forall x \in K \quad VI(f, K).$

b. Find $x^{opt} \in K \subseteq R^n: \ f(x)^t(x - x^{opt}) \geq 0, \ \forall x \in K$

(See Auslender [1]).

4. *Asymmetry*

   The general framework applies to asymmetric $VIPs$. We do not assume that the Jacobian matrix ∇f exists, or that if it does, that it is symmetric.

5. *Strong-f-monotonicity*

   Let us now examine the condition of strong-f-monotonicity, introduced in assumption A2.

Definition 11 . *The generalized inverse f^{-1} of a problem function f is the point to set map*

$$f^{-1} : f(K) \subseteq R^n \to 2^K,$$

defined by $f^{-1}(X) = \{x \in K : \ \ f(x) = X\}$.

Definition 12 . *A point to set map $g : R^n \to 2^{R^n}$ is strongly monotone if for every $x \in g^{-1}(X)$ and $y \in g^{-1}(Y)$ there is a constant $a > 0$ satisfying,*

$$(x - y)^t(X - Y) \geq a\|X - Y\|_2^2.$$

Proposition 4.1:

The problem function f is strongly-f-monotone, if and only if its generalized inverse f^{-1} is strongly monotone in $f(K)$.

Proof:

If $x \in f^{-1}(X)$ and $y \in f^{-1}(Y)$ then $f(x) = X, \ \ f(y) = Y$, and the strong-f-monotonicity condition

$$[f(x) - f(y)]^t[x - y] \geq a\|f(x) - f(y)\|_2^2, \quad \forall x, y \in K$$

becomes

$$[x-y]^t[X-Y] \geq a\|X-Y\|_2^2, \quad \forall X, Y \in f(K),$$

which coincides with the strong monotonicity condition on the generalized inverse function f^{-1}. Q.E.D.

Gabay [5] used strong monotonicity of f^{-1} and Tseng [28] referred to this concept as co-coercivity.

We now examine the relationship between strong-f-monotonicity and strict, strong and usual monotonicity.

Proposition 4.2:

a. If f is a strongly monotone and Lipschitz continuous function, then f is strongly-f-monotone.

b. If f is a one-to-one and strongly-f-monotone function, then f is also strictly monotone.

c. If f is a strongly-f-monotone function, then f is a monotone function.

Proof:

a. The function f is strongly monotone when

$$\exists b > 0 \text{ such that} : [f(x) - f(y)]^t[x-y] \geq b\|x-y\|_2^2 \;\; \forall x, y \in K.$$

The function f is Lipschitz continuous when

$$\exists C > 0 \text{ such that} : \|f(x) - f(y)\|_2 \leq C\|x-y\|_2 \; \forall x, y \in K.$$

Combining these conditions, we see that

$$[f(x) - f(y)]^t[x-y] \geq b\|x-y\|_2^2 = \frac{bC^2}{C^2}\|x-y\|_2^2 \geq \frac{b}{C^2}\|f(x)-f(y)\|_2^2.$$

So we conclude that f is strongly-f-monotone with a defining constant $a = \frac{b}{C^2} > 0$.
b. When f is a one-to-one function $f(x) \neq f(y)$ whenever $x \neq y$. Then the strong-f-monotonicity property implies that

$$[f(x) - f(y)]^t[x - y] \geq a\|f(x) - f(y)\|_2^2 > 0 \ \forall x, y \in K, \ with \ x \neq y.$$

So $[f(x) - f(y)]^t[x - y] > 0 \ \forall x, y \in K, \ with \ x \neq y$ which implies that f is strictly monotone.
c. Strong-f-monotonicity property implies that

$$[f(x) - f(y)]^t[x - y] \geq a\|f(x) - f(y)\|_2^2 \geq 0 \quad \forall x, y \in K.$$

So $[f(x) - f(y)]^t[x - y] \geq 0 \quad \forall x, y \in K$ and, therefore, f is monotone. Q.E.D.

Next we show how we can determine whether function f satisfies the strong-f-monotonicity property. We then illustrate this result on some specific examples.

Proposition 4.3:

When the Jacobian matrix of f satisfies the property that
$\nabla f(x')^t - a\nabla f(x')^t \nabla f(y')$ is a positive semidefinite matrix $\forall x', y' \in K$ and for some constant $a > 0$, then f is a strongly-f-monotone function.
The converse is also true in the following sense, on an open, convex subset $D_0 \subseteq K$ of the feasible set K, for each point $x' \in D_0$, for some point $y' \in D_0$ and constant $a > 0$, $\nabla f(x')^t - a\nabla f(x')^t \nabla f(y')$ is a positive semidefinite matrix.
Proof:
"$\Rightarrow$"
Let $\phi : [0, 1] \rightarrow R$ be defined as:

$$\phi(t) = [f(tx + (1 - t)y)]^t[(x - y) - a(f(x) - f(y))].$$

When applied to the function ϕ, the mean value theorem, that is,

$$\exists t' \in [0, 1] \quad such \quad that: \ \phi(1) - \phi(0) = \frac{d\phi(t)}{dt}|_{t=t'}$$

implies that

$$[f(x)-f(y)]^t[x-y]-a\|f(x)-f(y)\|_2^2 = (x-y)^t[\nabla f(x')]^t[(x-y)-a(f(x)-f(y))] \quad (3)$$

with $x' = t'x + (1-t')y$.

Now define $\phi_1 : [0,1] \to R$ as

$$\phi_1(t) = (x-y)^t[\nabla f(x')]^t[(tx+(1-t)y) - a(f(tx+(1-t)y))].$$

When applied to the function ϕ_1, the mean value theorem states that

$$\exists t'' \in [0,1] \text{ such that} : \phi_1(1) - \phi_1(0) = \frac{d\phi_1(t)}{dt}|_{t=t''}.$$

and so, when combined with (3),

$$[f(x)-f(y)]^t[x-y] - a\|f(x)-f(y)\|_2^2 =$$

$$= (x-y)^t[\nabla f(x')]^t[(x-y)-a(f(x)-f(y))] = (x-y)^t[\nabla f(x')]^t[I - a\nabla f(y')](x-y),$$

with $y' = t''x + (1-t'')y$.

The last expression implies that if $[\nabla f(x')]^t[I - a\nabla f(y')]$ is a positive semidefinite matrix, for some $a > 0$, then f is strongly-f-monotone.

"$\Leftarrow$"

If f is a strongly-f-monotone function then, as shown above, for some $a > 0$ and for all $x, y \in D_0 \subseteq K$

$$[f(x)-f(y)]^t[x-y] - a\|f(x)-f(y)\|_2^2 =$$

$$(x-y)^t[\nabla f(x')]^t[(x-y)-a(f(x)-f(y))] =$$

(with $x' = t'x + (1-t')y$ for some $t' \in [0,1]$)

$$(x-y)^t[\nabla f(x')]^t[I - a\nabla f(y')](x-y) \geq 0,$$

with $y' = t''x + (1-t'')y$ for some $t'' \in [0,1]$.

Since $x, y \in D_0$, which is an open, convex set, we conclude that for each point $x' \in D_0$, some point $y' \in D_0$ and some constant $a > 0$ satisfies the condition that $[\nabla f(x')]^t[I - a\nabla f(y')]$ is a positive semidefinite matrix. Q.E.D.

Remark:

Note that if $[\nabla f(x')]^t[I - a\nabla f(y')]$ is positive definite, then f is "strictly" strongly-f-monotone, that is, whenever $f(x) \neq f(y)$

$$[f(x) - f(y)]^t[x - y] > a\|f(x) - f(y)\|_2^2.$$

EXAMPLES:

(a) *The linear programming problem:*

$min_{x \in K}\, c^t x$, defined over a polytope K, trivially satisfies the strong-f-monotonicity condition. As we have noted, LP problems can be viewed as variational inequality problems with $f(x) = c$. The problem satisfies the strong-f-monotonicity condition since

$$0 = (c - c)^t(x - y) \geq a\|c - c\|^2 = 0,$$

for all constants $a > 0$ (therefore we could choose $a = 1$). LP problems do not satisfy strong or strict monotonicity properties.

(b) *The affine variational inequality problem:*

In this case

$$f(x) = Mx - c$$

the condition of Proposition 4.3 becomes:

check whether for some $a > 0$, the matrix

$$\nabla f(x')^t - a\nabla f(x')^t \nabla f(y') = M^t - aM^tM,$$

is positive semidefinite.

(c) Let $f(x) = \begin{bmatrix} g_1 & 0 & \dots & 0 \\ 0 & g_2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & g_n \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix}$ for some constants g_i and c_i.

Then $\nabla f = \begin{bmatrix} g_1 & 0 & \dots & 0 \\ 0 & g_2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & g_n \end{bmatrix}$,

and $\nabla f^t(I - a\nabla f) = \begin{bmatrix} g_1 - ag_1^2 & 0 & \dots & 0 \\ 0 & g_2 - ag_2^2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & g_n - ag_n^2 \end{bmatrix}$.

If $g_i \geq 0$, for $i = 1, 2, ...n$ and $a \leq \frac{1}{max_{1 \leq i \leq n} g_i}$, then ∇f is a positive semidefinite matrix.

If $g_i = 0$ for every $i = 1, ..., n$, then $\nabla f^t(I - a\nabla f) = 0$, which is (trivially) a positive semidefinite matrix.

(d) Let $f(x) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 3 \end{bmatrix}$.

This function f does not satisfy strict or strong monotonicity assumptions. This result is easy to check since the Jacobian matrix $\nabla f = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}$ is not positive definite (the first row is all zeros). Nevertheless the matrix

$\nabla f^t(I - a\nabla f) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2-4a & -2a \\ 0 & 1-2a & 2-5a \end{bmatrix}$ is positive semidefinite for all $0 < a \leq 1/4$. Therefore, the problem function f satisfies strong-f-monotonicity.

Remark: Further characterizations of the strong-f-monotonicity condition can be found in [17], [18] and [19].

5 Complexity and convergence analysis of the general geometric framework

In this section we show that the general geometric framework we introduced in Section 3, and the four special cases that we examine in Section 5.3, converge to an ϵ-near optimal solution (see Definitions 8 and 6) in $O(-\frac{nL_1}{logb(n)})$ iterations. Furthermore, when the volumes of the "nice" sets decrease at a rate $b(n) = O(2^{-\frac{n}{pol(n)}})$, the framework converges in a polynomial number of iterations, in terms of L_1 and n.

5.1 Complexity analysis for the general geometric framework

We begin our analysis by establishing a complexity result for the general framework.

THEOREM 5.1.:

Let $L_1 = 4L + l + log(\frac{9M}{\epsilon^2 a})$. After $O(-\frac{nL_1}{logb(n)})$ iterations, the volume of the "nice" sets P_k is no more than 2^{-nL_1}.

Proof:

Since $volP_k \leq b(n)volP_{k-1}$, $volP_k \leq (b(n))^k volP_0$, where $0 < b(n) < 1$.

Moreover, $volP_0 \leq 2^{n(2L+1)}$.

Consequently, $volP_k \leq [b(n)]^k 2^{n(2L+1)} = 2^{klogb(n)+2nL+n}$.

Therefore in $k^* = O(-\frac{nL_1}{\log b(n)})$ iterations, with $L_1 = 4L + l + log(\frac{9M}{\epsilon^2 a})$,

$volP_{k^*} \leq 2^{-nL_1}$.

Q.E.D.

In Theorem 5.2 we will show that in at most $k^* = O(-\frac{nL_1}{logb(n)})$ iterations when the volume of the set P_k is no more than 2^{-nL_1} at least one of the points x^j, for $j \in OC(k^*)$ is an $\frac{\epsilon}{2}$-near optimal solution. Using this fact, in Theorem 5.3, we show that if $y_{\frac{\epsilon}{2}}^{k(i)}$ is an $\frac{\epsilon}{2}$-near optimal solution of $min_{x\in K} f(x^i)^t(x - x^i)$, then the point x^i satisfying $j = argmax_{i\in OC(k^*)}[f(x^i)^t(y_{\frac{\epsilon}{2}}^{k(i)} - x^i)]$ is an ϵ-near optimal solution.

Remarks:

1. Since $0 < b(n) < 1$ implies that $logb(n) < log1 = 0$, $-\frac{n}{logb(n)} > 0$, so it makes sense to

write $k = O(-\frac{nL_1}{\log b(n)})$ since this quantity is positive.

2. In order for any realization of the general geometric framework to be a polynomial algorithm, we need the volumes of the nice sets to decrease at a rate $b(n) = O(2^{-\frac{n}{pol(n)}})$ for some polynomial function $pol(n)$ of n. Then $O(-\frac{nL_1}{\log b(n)}) = O(pol(n)L_1)$, with $L_1 = 4L + l + log(\frac{9M}{\epsilon^2 a})$. Within $O(pol(n)L_1)$ iterations the volume of P_k becomes no more than 2^{-nL_1} (this is true because when $b(n) = O(2^{-\frac{n}{pol(n)}})$, $\log\ b(n) = -O(\frac{n}{pol(n)})$ implying that $pol(n) = -O(\frac{n}{\log\ b(n)})$ so $k^* = O(pol(n)L_1)$).

5.2 The general convergence theorem

We prove the general convergence theorem for problems that satisfy assumptions A1 and A2. We first show that within at most $k^* = O(-\frac{nL_1}{logb(n)})$ iterations at least one point $x^* = x^j,\ \ j \in OC(k^*)$, solves the VIP. This point also satisfies the property that

$$0 \leq f(x^*)^t(x^* - x^{opt}) \leq \gamma. \tag{4}$$

We define $\gamma^{1/2}$ as $\frac{-\beta+\sqrt{\beta^2+2\epsilon}}{2}$, with $\beta = (\frac{2^{2L+2}}{a})^{1/2}$. Therefore γ depends on the nearness constant ϵ, the strong-f-monotonicity constant a, and the constant L. We first establish an underlying lemma and proposition.

LEMMA 5.1:

Suppose the variational inequality $VI(f,K)$ satisfies assumptions A1 and A2. If

$$f(x^k)^t(x^k - x^{opt}) > \gamma,$$

$\gamma > 0$, then

$$S(x^{opt}, \frac{\gamma}{M}) \cap K \subseteq \{x \in K : f(x^k)^t(x^k - x) \geq 0\}.$$

Proof:

If $x \in S(x^{opt}, \frac{\gamma}{M}) \cap K$, then

$$x \in K \ \text{ and } \ \|x - x^{opt}\| \leq \frac{\gamma}{M}.$$

Since $x^k \in K$ and $f(x^k)^t(x^k - x^{opt}) > \gamma$ by assumption, the uniform boundedness of f and Cauchy's inequality imply that

$$f(x^k)^t(x^k - x) = f(x^k)^t(x^k - x^{opt}) + f(x^k)^t(x^{opt} - x) \geq$$

$$\gamma - \|f(x^k)\| \|x - x^{opt}\| \geq \gamma - M\frac{\gamma}{M} = \gamma - \gamma = 0.$$

Consequently, $x \in \{x \in K : f(x^k)^t(x^k - x) \geq 0\}$. Therefore, any point in $S(x^{opt}, \frac{\gamma}{M}) \cap K$ also belongs to $\{x \in K : f(x^k)^t(x^k - x) \geq 0\}$. Q.E.D.

Proposition 5.1:

Suppose the variational inequality problem $VI(f, K)$ satisfies assumptions A1 and A2. Let x^{opt} be an optimal solution of $VI(f, K)$. The general framework computes at least one point $x^* \in K$ (encountered within at most $k^* = O(-\frac{nL_1}{log b(n)})$ iterations of the general framework) that satisfies the following inequality

$$0 \leq f(x^*)^t(x^* - x^{opt}) \leq \gamma. \tag{5}$$

In this expression $\gamma^{1/2} = \frac{-\beta+\sqrt{\beta^2+2\epsilon}}{2}$, ϵ is a small positive constant, and $\beta = (\frac{2^{2L+2}}{a})^{1/2}$.

Remark:

Note that if $\beta^2 > 2\epsilon$, then $\frac{\epsilon}{3\beta} \leq \gamma^{1/2} \leq \frac{\epsilon}{2\beta}$. To establish this result, we multiply the numerator and denominator of the expression defining $\gamma^{1/2}$ by $\beta + \sqrt{\beta^2 + 2\epsilon}$, giving

$$\gamma^{1/2} = \frac{\epsilon}{\beta + \sqrt{\beta^2 + 2\epsilon}}.$$

Since $2\epsilon \geq 0$, $2\epsilon \leq \beta^2$ and $1 + \sqrt{2} < 3$, this expression implies the asserted bounds.

Proof:

In Theorem 5.1 we showed that within $k^* = O(-\frac{nL_1}{log b(n)})$ iterations the volume of P_{k^*} is no more than 2^{-nL_1}. Therefore, the general framework encounters at least one point x^j that lies in K (we have performed at least one optimality cut) since the VIP has a feasible solution (see assumption A1).

Assume that within $k^* = O(-\frac{nL_1}{log b(n)})$ iterations of the general framework, each point x^j, $j \in OC(k^*)$ satisfies the property

$$f(x^j)^t(x^j - x^{opt}) > \gamma \quad \text{for all} \quad j \in OC(k^*).$$

The construction of the nice sets P_k, implies that $\cap_{j\in OC(k)}\{x \in K : f(x^j)^t(x^j - x) \geq 0\} \subseteq P_k$. In the final iteration of the algorithm, the set P_{k^*} has volume no more than 2^{-nL_1} (see Theorem 5.1), so the set $\cap_{j\in OC(k^*)}\{x \in K : f(x^j)^t(x^j - x) \geq 0\}$ has volume at most 2^{-nL_1}. This result and Lemma 5.1 implies that:
$S(x^{opt}, \frac{\gamma}{M}) \cap K \subseteq \cap_{j\in OC(k^*)}\{x \in K : f(x^j)^t(x^j - x) \geq 0\} \subseteq P_{k^*}$.
Note that $\cap_{j\in OC(k^*)}\{x \in K : f(x^j)^t(x^j - x) \geq 0\} \subseteq P_{k^*}$ because each P_k is defined by both optimality and feasibility cuts and each feasibility cut contains K. Thus

$$S(x^{opt}, R) \cap K \subseteq K_{k^*} \subseteq P_{k^*},$$

with $R = \frac{\gamma}{M} \geq \frac{\epsilon^2}{9M\beta^2} = \frac{\epsilon^2 a}{9M2^{2L+2}}$. So the set $S(x^{opt}, \frac{\epsilon^2 a}{9M2^{2L+2}}) \cap K$ is contained in P_{k^*}. Moreover, since $L_1 = 4L + l + log(\frac{9M}{\epsilon^2 a})$, the final set P_{k^*} has volume at most $2^{-nL_1} = 2^{-n[4L+l+log(\frac{9M}{\epsilon^2 a})]}$. But as shown in Lemma 4.1, since $R \leq 2^{L+1}$,

$$vol[S(x^{opt}, R) \cap K] \geq (\frac{R}{n2^{L+l+1}})^n.$$

Therefore, the volume of $S(x^{opt}, R) \cap K$ is at least

$$2^{-nlogn+nlogR-n(L+l+1)} \geq 2^{-nlogn-n(1+L+l)-2n-2nL-nlog(\frac{9M}{\epsilon^2 a})}.$$

But since the volume of P_k is at most $2^{-n[4L+l+log(\frac{9M}{\epsilon^2 a})]}$ and

$$2^{-n(4L+l)-nlog(\frac{9M}{\epsilon^2 a})} < 2^{-nlogn-n(L+l+1)-2nL-2n-nlog(\frac{9M}{\epsilon^2 a})},$$

the set $S(x^{opt}, \frac{\gamma}{M}) \cap K$ is contained in P_{k^*}, which, as shown above, has a smaller volume. This *contradiction* shows that our initial assumption, that $f(x^j)^t(x^j - x^{opt}) > \gamma$ for all $j \in OC(k^*)$ is untenable. So we conclude that in at most $k = O(-\frac{nL_1}{logb(n)})$ iterations, at least one point $x^j = x^*$ satisfies the condition

$$f(x^*)^t(x^* - x^{opt}) \leq \gamma.$$

Since $x^* \in K$, the equivalent VIP formulation implies that $f(x^*)^t(x^* - x^{opt}) \geq 0$. Therefore, $0 \leq f(x^*)^t(x^* - x^{opt}) \leq \gamma$. Q.E.D.

Now we are ready to prove the main convergence theorem.

THEOREM 5.2:

For variational inequality problems that satisfy assumptions A1 and A2, within at most $k^* = O(-\frac{nL_1}{logb(n)})$ iterations of the general framework, at least one of the points x^j for $j \in OC(k^*)$ is an $\frac{\epsilon}{2}$-near optimal solution of $VI(f, K)$. In the special case in which $f(x) = c$, a constant, the point x^i with $i = argmin_{j \in OC(k^*)} c^t x^j$ is an $\frac{\epsilon}{2}$-near optimal solution.

In general, $k^* = O(pol(n)L_1)$ when $b(n) = O(2^{-\frac{n}{pol(n)}})$ and so, in this case, the algorithm is polynomial.

Remark: As we show in Tables II and III each of the algorithms that we consider in Section 5.3 satisfies the final condition of this theorem and, therefore, is polynomial.

Proof:

In Proposition 5.1, we have shown that the general framework determines that for some $0 \leq j \leq k$, the point $x^* = x^j$ satisfies $f(x^*)^t(x^* - x^{opt}) \leq \gamma$ with $\gamma^{1/2} = \frac{-\beta + \sqrt{\beta^2 + 2\epsilon}}{2}$ and $\beta = (\frac{2^{2L+2}}{a})^{1/2}$. Moreover, since x^{opt} is a $VI(f, K)$ solution, $x^* \in K$, and the problem function f satisfies the strong-f-monotonicity condition (assumption A2), we see that

$$\gamma \geq f(x^*)^t(x^* - x^{opt}) \geq [f(x^*) - f(x^{opt})]^t[x^* - x^{opt}] \geq a\|f(x^*) - f(x^{opt})\|_2^2 \geq 0.$$

So

$$\|f(x^*) - f(x^{opt})\|_2 \leq (\frac{\gamma}{a})^{1/2}. \tag{6}$$

Now observe that by adding and subtracting several terms, for all $x \in K$

$$f(x^*)^t(x - x^*) =$$

$$= [f(x^*) - f(x^{opt})]^t[x - x^*] + f(x^{opt})^t(x - x^{opt}) + [f(x^{opt}) - f(x^*)]^t[x^{opt} - x^*] + f(x^*)^t(x^{opt} - x^*).$$

The last summation has four terms:

(i) The first term

$$[f(x^*) - f(x^{opt})]^t[x - x^*] \geq -\|f(x^*) - f(x^{opt})\|.\|x - x^*\|,$$

by Cauchy's inequality.

(ii) The second term $f(x^{opt})^t(x - x^{opt}) \geq 0 \;\; \forall x \in K$ since x^{opt} solves $VI(f, K)$.

(iii) The third term $[f(x^{opt}) - f(x^*)]^t[x^{opt} - x^*] \geq 0$ since strong-f-monotonicity implies monotonicity (see Proposition 4.2c.).

(iv) The fourth term $f(x^*)^t(x^{opt} - x^*) \geq -\gamma$ from Proposition 5.1.

Putting these results together, we obtain:

$$\forall x \in K : f(x^*)^t(x - x^*) \geq -\|f(x^*) - f(x^{opt})\|.\|x - x^*\| - \gamma \qquad (7)$$

Definition 5 of L shows that $K \subseteq S(0, 2^L)$, which in turn implies that

$$\|x - x^*\| \leq 2^{L+1}. \qquad (8)$$

Combining inequalities (6), (7) and (8) shows that

$$\forall x \in K : \; f(x^*)^t(x - x^*) \geq -\gamma - (\frac{\gamma 2^{2L+2}}{a})^{1/2}.$$

Setting $\beta = (\frac{2^{2L+2}}{a})^{1/2}$ and choosing γ so that $0 \leq \gamma^{1/2} = \frac{-\beta + \sqrt{\beta^2 + 2\epsilon}}{2}$, we see that $-\gamma - (\frac{\gamma 2^{2L+2}}{a})^{1/2} = -\frac{\epsilon}{2}$. Q.E.D.

Note that we have shown that any point x^j, $j \in OC(k^*)$, that satisfies the condition $f(x^j)^t(x^j - x^{opt}) \leq \gamma$ is an $\frac{\epsilon}{2}$-near optimal solution. In the special case that $f(x)$ is a constant, i.e., $f(x) = c$, this condition becomes $c^t x^j \leq \gamma + c^t x^{opt}$. Therefore, in this case if we select x^* as a point minimizing $\{c^t x^j : \;\; j \in OC(k^*)\}$, then x^* is an $\epsilon/2$-near optimal solution.

Lovasz [15], using a somewhat different analysis, has considered the optimization of a linear function over a convex set. Assuming that the separation problem could be solved only approximately, he showed how to find an ϵ-near optimal, near feasible solution: that is, the solution he generates is within ϵ of the feasible solution and is within ϵ of the optimal objective value.

Theorem 5.2 shows that one of the points x^i, $i \in OC(k^*)$, is an $\frac{\epsilon}{2}$-near optimal solution. Except for the case in which $f(x)$ is a constant, it does not, however, tell us which of these points is the solution. How might we make this determination? The next result provides an answer to this question.

THEOREM 5.3:

Let x^i denote the points generated by the general framework in $k = O(-\frac{nL_1}{logb(n)})$ steps, and let $y_{\frac{\epsilon}{2}}^{k(i)}$ be an $\frac{\epsilon}{2}$-near optimal solution to the problem $min_{y \in K} f(x^i)^t y$. Then the point $x^* = x^j$ with

$$j = argmax_{\{i \in OC(k^*)\}} f(x^i)^t (y_{\frac{\epsilon}{2}}^{k(i)} - x^i),$$

is an ϵ-near optimal solution to VIP.

Proof: In Theorem 5.2 we showed that if the general framework generates the points x^i in $k = O(-\frac{nL_1}{logb(n)})$ steps, then at least one point x^i satisfies the condition $f(x^i)^t(x - x^i) \geq -\frac{\epsilon}{2}$ for all $x \in K$. Therefore, the point $x^j = x^*$ with

$$j = argmax_{\{i \in OC(k^*)\}} f(x^i)^t (y_{\frac{\epsilon}{2}}^{k(i)} - x^i),$$

satisfies

$$f(x^j)^t (y_{\frac{\epsilon}{2}}^{k(j)} - x^j) \geq -\frac{\epsilon}{2}. \tag{9}$$

Furthermore, since $y_{\frac{\epsilon}{2}}^{k(j)}$ is an $\frac{\epsilon}{2}$-near optimal solution to the problem $min_{y \in K} f(x^j)^t y$ then

$$f(x^j)^t (x - y_{\frac{\epsilon}{2}}^{k(j)}) \geq -\frac{\epsilon}{2} \quad \forall x \in K. \tag{10}$$

Adding (9) and (10), we conclude that $x^j = x^*$ with $j = argmax_{\{i \in OC(k^*)\}} f(x^i)^t (y_{\frac{\epsilon}{2}}^{k(i)} - x^i)$ is indeed an ϵ-near optimal solution, that is

$$f(x^j)^t (x - x^j) \geq -\epsilon \quad \forall x \in K.$$

Q.E.D.

Remark:

When the Jacobian matrix of the variational inequality problem is symmetric, then some corresponding objective function F satisfies $\nabla F = f$. The VIP is now equivalent to the nonlinear program $min_{x \in K} F(x)$. Proposition 5.1 implies that for some $0 \leq j \leq k$, the point x^j satisfies the inequality

$$\nabla F(x^j)^t(x^j - x^{opt}) = f(x^j)^t(x^j - x^{opt}) \leq \gamma.$$

The convexity of the objective function F implies that

$$F(x^j) - F(x^{opt}) \leq \nabla F(x^j)^t(x^j - x^{opt}) \leq \gamma.$$

Therefore,

$$min_{i \in OC(k)} F(x^i) \leq F(x^{opt}) + \gamma.$$

The point $x^j = argmin_{i \in OC(k)} F(x^i)$ is within γ of the optimal solution, i.e.,

$$F(x^j) \leq F(x) + \gamma \ \ \forall x \in K.$$

Note that in this case, the near optimal solution x^j need not be the last iterate that the algorithm generates.

5.3 Four geometric algorithms for the VIP

In closing our analysis, in this section we show how four algorithms for solving linear programming and convex programming problems can also efficiently (i.e., in a polynomial number of iterations) solve variational inequality problems and are special cases of the general framework. In particular, we examine the method of centers of gravity, the ellipsoid algorithm, the method of inscribed ellipsoids, and Vaidya's algorithm, as applied to the VIP.

The method of centers of gravity for the VIP

The method of centers of gravity was first proposed for convex programming problems, in 1965, by A. Ju. Levin (see [14]). In order to extend this algorithm to a VIP we use the following definition:

Definition 13 . *If K is a compact, convex set, with a nonempty interior, then its center of gravity is the point:*

$$x = \frac{\int_K y dy}{\int_K dy}.$$

From a physical point of view, x represents the center of gravity of a mass uniformly distributed over a body K.

For a general convex, compact set K, the center of gravity is difficult to compute: even computing $\int_K dy = vol(K)$ is difficult. Recently Dyer, Frieze and Kannan [4] proposed a polynomial randomized algorithm for computing $vol(K)$. If K is a simplex in R^n with vertices $x^0, x^1, ..., x^n$, the center of gravity x is given by

$$x = \frac{x^0 + x^1 + ... + x^n}{n+1}.$$

We now extend the method of centers of gravity for the VIP.

Algorithm 1:

ITERATION 0:

Find the center of gravity x^0 of $P_0 = K$. Set $k = 1$.

ITERATION k:

Let $c_k \equiv -f(x^{k-1})$. "Cut" through x^{k-1} with a hyperplane $H_k = \{x \in R^n : c_k^t x = c_k^t x^{k-1}\}$ defining the halfspace $H_k^c = \{x \in R^n : c_k^t x \geq c_k^t x^{k-1}\}$. Set $P_k = P_{k-1} \cap H_k^c$ and find the center of gravity x^k of P_k. Set $K_k = P_k$. Set $k \leftarrow k+1$ and continue.

STOP in

$$O(nL_1) = O(n(4L + l) + nlog(\frac{9M}{\epsilon^2 a}))$$

iterations of the algorithm.

If $y_{\frac{\epsilon}{2}}^{k(i)}$ is an $\frac{\epsilon}{2}$-near optimal solution of $min_{x \in K} f(x^i)^t (x - x^i)$, the proposed solution is $x^* = x^j$, with $j = argmax_{i \in OC(k)} [f(x^i)^t (y_{\frac{\epsilon}{2}}^{k(i)} - x^i)]$.

The ellipsoid algorithm for the VIP

We now extend the ellipsoid algorithm for the VIP. This algorithm was originally developed for linear programming problems by Leonid Khatchiyan in 1979. H.-J. Luthi has studied the ellipsoid algorithm for solving strongly monotone variational inequality problems (see [16]). He showed that the ellipsoid algorithm, when applied to strongly monotone $VIPs$, induces some subsequence that converges to the optimal solution (which in this case is unique).

Algorithm 2:

ITERATION 0:

Start with $x^0 = 0$, $B_0 = n^2 2^{2L} I$, $K_0 = K$, and an ellipsoid $P_0 = \{x \in R^n : (x - x^0)^t (B_0)^{-1}(x - x^0) \leq 1\}$. Set $k = 1$.

ITERATION k:

Part a: (Feasibility cut: cut toward a feasible solution.)

If $x^{k-1} \in K$ then go to part b.

If $x^{k-1} \notin K$, then choose a hyperplane through x^{k-1} that supports K (that is, one halfspace defined by the hyperplane contains K).

An oracle returns a vector c satisfying the condition $c^t x \geq c^t x^{k-1}$, $\forall x \in K$.

Set $K_k = K_{k-1}$. We update $x^k = x^{k-1} - \frac{1}{n+1}\frac{B_{k-1}c}{\sqrt{c^t B_{k-1} c}}$, $B_k = \frac{n^2}{n^2-1}[B_{k-1} - \frac{2}{n+1}\frac{(B_{k-1}c)(B_{k-1}c)^t}{c^t B_{k-1} c}]$,
$P_k = \{x \in R^n : (x - x^k)^t (B_k)^{-1}(x - x^k) \leq 1\}$,

this new ellipsoid is centered at x^k.

Set $k \leftarrow k + 1$ and repeat iteration k.

Part b : (Optimality cut: cut towards an optimal solution)

$K_k = K_{k-1} \cap \{x \in R^n : (-f(x^{k-1}))^t x \geq (-f(x^{k-1}))^t x^{k-1}\}$.

(We cut with the vector $c = -f(x^{k-1})$).

Then set $x^k \equiv x^{k-1} - \frac{1}{n+1}\frac{B_{k-1}c}{\sqrt{c^t B_{k-1} c}}$

$B_k \equiv \frac{n^2}{n^2-1}[B_{k-1} - \frac{2}{n+1}\frac{(B_{k-1}c)(B_{k-1}c)^t}{c^t B_{k-1} c}]$

$P_k \equiv \{x \in R^n : (x - x^k)^t (B_k)^{-1}(x - x^k) \leq 1\}$.

Set $k \leftarrow k + 1$ and repeat iteration k.

STOP in

$$O(n^2 L_1) = O(n^2(4L + l) + n^2 log(\frac{9M}{\epsilon^2 a}))$$

iterations of the algorithm.

If $y_{\frac{\epsilon}{2}}^{k(i)}$ is an $\frac{\epsilon}{2}$-near optimal solution of $min_{x \in K} f(x^i)^t(x - x^i)$, the proposed solution is $x^* = x^j$, with $j = argmax_{i \in OC(k)}[f(x^i)^t(y_{\frac{\epsilon}{2}}^{k(i)} - x^i)]$.

The method of inscribed ellipsoids for the VIP

We now extend the method of inscribed ellipsoids for the VIP. This algorithm was originally developed for convex programming problems, in 1988, by Khatchiyan, Tarasov, and Erlikh. In order to develop this algorithm for the VIP, we first state the following definition:

Definition 14 . *Let K be a convex body in R^n. Among the ellipsoids inscribed in K, there exists a unique one of maximal volume (see Khatchiyan, Tarasov, and Erlikh, [12]). We will call such an ellipsoid $E^* = E_K^*$* **maximal** *for K and let $vol(K) = max\{volE$: E is an ellipsoid $E \subseteq K\}$ denote its volume as a function of K. We will refer to the center x^* of the maximal ellipsoid as the* **center** *of K.*

Algorithm 3:

ITERATION 0:

Find the center x^0 of $P_0 = K$ (see definition above). Set $k = 1$.

ITERATION k:

Set $c_{k-1} \equiv -f(x^{k-1})$. "Cut" through x^{k-1} with a hyperplane $H_k = \{x \in R^n : c_{k-1}^t x = c_{k-1}^t x^{k-1}\}$, defining a halfspace $H_k^c = \{x \in R^n : c_{k-1}^t x \geq c_{k-1}^t x^{k-1}\}$. Set $P_k = P_{k-1} \cap H_k^c$ and find the center x^k of P_k. Set $K_k = P_k$.

Set $k \leftarrow k + 1$ and repeat iteration k.

STOP in

$$O(nL_1) = O(n(4L + l) + nlog(\frac{9M}{\epsilon^2 a}))$$

iterations of the algorithm.

If $y_{\frac{\epsilon}{2}}^{k(i)}$ is an $\frac{\epsilon}{2}$-near optimal solution of $min_{x \in K} f(x^i)^t(x - x^i)$, the proposed solution is $x^* = x^j$, with $j = argmax_{i \in OC(k)}[f(x^i)^t(y_{\frac{\epsilon}{2}}^{k(i)} - x^i)]$.

Vaidya's algorithm for the VIP

Let $P = \{x \in R^n : Ax \geq b\}$ be a polyhedron defined by a matrix $A \in R^{mn}$ and a column vector $b \in R^m$. Let a_i^t be the ith row of matrix A. For any point $x \in R^n$, the quantity $s_i = a_i^t x - b_i$ denotes the "surplus" in the ith constraint. Let $H(x) = \sum_{i=1}^{m} \frac{(a_i a_i^t)}{(a_i^t x - b_i)^2}$ denote the Hessian matrix of the log-barrier function, which is the sum of minus the logarithm of the surplus; that is, $\sum_{i=1}^{m}(-\log(a_i^t x - b_i))$.

Definition 15 . (**Volumetric center**):*(Vaidya [29]) Assume $P = \{x \in R^n : Ax \geq b\}$ is a bounded polyhedron with a nonempty interior. Let $F(x) \equiv \frac{1}{2}\log(det\ H(x))$. The volumetric center w of the polytope P is a solution of the optimization problem $min_{x \in P} F(x)$.*

To further understand this definition and motivate Vaidya's algorithm for the VIP, we offer the following remarks.

Remarks:

1. Let us see why we call w, as defined above, the "volumetric" center of polytope P. If $x \in P$, the ellipsoid $E(H(x), x, 1) = \{y \in R^n : (y-x)^t H(x)(y-x) \leq 1\}$ lies within P. To establish this result, we argue as follows. If $y \in E(H(x), x, 1)$, then $(y-x)^t H(x)(y-x) \leq 1$. For all $i = 1, 2, ..., m$, $\frac{(a_i a_i^t)}{(a_i^t x - b_i)^2}$ is a positive definite matrix. Substituting $\sum_{i=1}^{m} \frac{(a_i a_i^t)}{(a_i^t x - b_i)^2}$ for $H(x)$, we obtain

$$(y-x)^t \frac{(a_i a_i^t)}{(a_i^t x - b_i)^2}(y-x) \leq \sum_{i=1}^{m}(y-x)^t \frac{(a_i a_i^t)}{(a_i^t x - b_i)^2}(y-x) =$$

$$\sum_{i=1}^{m} \frac{[a_i^t(y-x)]^2}{(a_i^t x - b_i)^2} \leq 1, \ for\ i = 1, 2, ...m.$$

Then $[a_i^t(y-x)]^2 \leq (a_i^t x - b_i)^2$ for $i = 1, 2, ..., m$, which implies that $(a_i^t y)^2 - 2a_i^t y a_i^t x - (b_i)^2 + 2b_i a_i^t x \leq 0$ for $i = 1, 2, ...m$ and so $(a_i^t y - b_i)(a_i^t y + b_i - 2a_i^t x) \leq 0$, $i = 1, 2, ..., m$. Now suppose that $y \notin P$; then $a_i^t y < b_i$ for some $i = 1, 2, ..., m$. The previous inequality implies that $a_i^t y + b_i \geq 2a_i^t x$. Since $x \in P$ (so $a_i^t x \geq b_i$), we can conclude that $a_i^t y \geq b_i$, contradicting the fact that $a_i^t y < b_i$. So $y \in P$.

We might view the ellipsoid $E(H(x), x, 1)$, which is inscribed in P and is centered at a point $x \in P$, as a local quadratic approximation of P. Among all the ellipsoids $E(H(x), x, 1)$, $\forall x \in P$, we select the one that is the best approximation to P. Therefore, we select the ellipsoid $E(H(w), w, 1)$, $w \in P$, that has the maximum volume. If V_n denotes the volume of the unit ball in R^n,

$$vol(E(H(w), w, 1)) = \sqrt{(det\ H(x))^{-1}}V_n,$$

(see Groetschel, Lovasz and Schrijver [6]). So the algorithm computes w by solving the optimization problem

$$max_{x \in P}\ volE(H(x), x, 1) = max_{x \in P}\ [\log(det\ H(x))^{-1/2}V_n],$$

or equivalently, $min_{x \in P}\ 1/2 \log(det\ H(x)) = min_{x \in P}\ F(x)$.
Since a hyperplane through w divides $E(H(w), w, 1)$ into two parts of equal volume ($E(H(w), w, 1)$ is a good inner approximation of polyhedron P), a hyperplane through w also "approximately" divides P into two equal parts. Therefore, we obtain a good rate of decrease of the volume.

2. Note that the definitions of the volumetric center of a polytope, Definition 15, and of the center as defined in the method of inscribed ellipsoids (i.e. the center of its maximal inscribed ellipsoid), Definition 14, are closely related. In the definition of the volumetric center of a polytope, we consider <u>a class</u> of inscribed ellipsoids (the local quadratic approximations to P), and select the one with the maximal volume, its center is the volumetric center. On the other hand, in the method of inscribed ellipsoids, we consider <u>all</u> the inscribed ellipsoids and select the one with the maximal volume; its center is what we define then as "center".

3. The volumetric center is, in general, difficult to compute. Nevertheless, once we manage to compute the volumetric center of an "easy" polytope, we can then find an "approximate" volumetric center of a subpolytope (of the original simplex), through a sequence of Newton-type steps. Thus, although it is difficult to compute the exact

volumetric center of a polytope, it is much easier to compute its "approximate" volumetric center. A similar idea can be applied in the method of inscribed ellipsoids (see L.G.Khatchiyan, S.P.Tarasov, I.I. Erlikh, [12]).

We now describe Vaidya's algorithm for the VIP.

Algorithm 4:

ITERATION 0:

Start with the simplex $P_0 = \{x \in R^n : x_j \geq -2^L, j = 1, 2, ..., n, \sum_{j=1}^{n} x_j \leq n2^L\}$.

Set $K_0 = K \subseteq P_0$. Compute the volumetric center w^0 of P_0 and set it equal to x^0. Set $k = 1$.

(REMARK: Any polytope, whose volumetric center is easy to compute, would be a good choice here.)

ITERATION k:

Part a: (Feasibility cut: cut towards a feasible solution)

Let δ, ϵ be small positive constants satisfying $\delta \leq 10^{-4}, \epsilon \leq 10^{-3}\delta$. Let $Q(x)$ be an approximation of $\nabla^2 F(x)$, namely $Q(x) = \sum_{i=1}^{m} \sigma_i \frac{(a_i a_i^t)}{(a_i^t x - b_i)^2}$ with $\sigma_i = \frac{(a_i^t H(x)^{-1} a_i)}{(a_i^t x - b_i)^2}$. Let $m(x)$ be the largest λ for which $\forall \xi \in R^n : \xi^t Q(x) \xi \geq \lambda \; \xi^t H(x) \xi$. At the beginning of each iteration, we have a point x^{k-1} satisfying the condition that

$$F(x^{k-1}) - F(w^{k-1}) \leq \epsilon^4 m(w^{k-1}).$$

Case 1: If $min_{1 \leq i \leq m} \; \sigma_i(x^{k-1}) \geq \epsilon$, add a plane to the previous polytope P_{k-1}.

Call an oracle with the current point x^{k-1} as its input.

If $x^{k-1} \in K$, then go to part b.

If $x^{k-1} \notin K$, then the oracle returns a vector c for which, for all $x \in K$, $c^t x \geq c^t x^{k-1}$ (The hyperplane through x^{k-1}, defined by c, supports K).

Choose β such that $c^t x^{k-1} \geq \beta$ and $\frac{c^t (H(x^{k-1})^{-1} c}{(c^t x^{k-1} - \beta)^2} = \frac{(\delta\epsilon)^{1/2}}{2}$.

Let $\tilde{A} = \begin{bmatrix} A \\ c \end{bmatrix}$ and $\tilde{b} = \begin{bmatrix} b \\ \beta \end{bmatrix}$ and set $A \equiv \tilde{A}$, $b \equiv \tilde{b}$, $P_{k-1} \equiv \{x \in R^n : Ax \geq b\}$.
We have now added a plane to the polytope. The volumetric center of the new polytope will shift and so will its "approximate" volumetric center (we are computing "approximate" volumetric centers throughout this algorithm). We now compute the new "approximate" volumetric center via a sequence of Newton-type steps as follows:

Let $x^{j-1} = x^{k-1}$. For $j = 1$ to $J = [30 \log(2\epsilon^{-4.5})]$ (where [.] denotes the integer part of a number), do $x^{j-1} \leftarrow x^{j-1} - 0.18Q(x^{j-1})^{-1}\nabla F^{new}(x^{j-1})$. Let $x^{J-1} = x^{k-1}$.
Then set $x^k \equiv x^{k-1}$, $K_k = K_{k-1}$ and $P_k \equiv \{x \in R^n : Ax \geq b\}$ as found above, set $k \leftarrow k+1$ and repeat iteration k.

Case 2: If $min_{1 \leq i \leq m}\ \sigma_i(x^{k-1}) < \epsilon$, then remove a plane from P_{k-1}.
Without loss of generality, suppose that $min_{1 \leq i \leq m}\ \sigma_i(x^{k-1}) = \sigma_m(x^{k-1})$
Let $a_m = c, b_m = \beta$, $A = \begin{bmatrix} \tilde{A} \\ c \end{bmatrix}$ and $b = \begin{bmatrix} \tilde{b} \\ \beta \end{bmatrix}$ and set $A \equiv \tilde{A}$, $b \equiv \tilde{b}$, $P_{k-1} \equiv \{x \in R^n : Ax \geq b\}$. Since the volumetric center shifts, due to the removal of a plane, we will perform a sequence of Newton-type steps to move closer to the new "approximate" volumetric center as follows:
Let $x^{j-1} = x^{k-1}$. For $j = 1$ to $J = [30 \log(4\epsilon^{-3})]$, do $x^{j-1} \leftarrow x^{j-1} - 0.18Q(x^{j-1})^{-1}\nabla F^{new}(x^{j-1})$.
Let $x^{k-1} = x^{J-1}$.
If $x^{k-1} \in K$, then set $x^k \equiv x^{k-1}$, $K_k = K_{k-1}$ and $P_k \equiv \{x \in R^n : Ax \geq b\}$ as found above, set $k \leftarrow k+1$ and repeat iteration k.
If $x^{k-1} \notin K$, then set $x^k \equiv x^{k-1}$, $K_k = K_{k-1}$ and $P_k \equiv \{x \in R^n : Ax \geq b\}$ as found above, set $k \leftarrow k+1$ and repeat iteration k.

Part b: (Optimality cut: cut towards an optimal solution)
Let $c = -f(x^{k-1})$, define β_{k-1} so that $c^t x^{k-1} \geq \beta_{k-1}$ and $\frac{c^t(H(x^{k-1})^{-1}c}{(c^t x^{k-1} - \beta_{k-1})^2} = \frac{(\delta\epsilon)^{1/2}}{2}$. Set $K_k = K_{k-1} \cap \{x \in R^n : (-f(x^{k-1})^t x \geq \beta_{k-1}\}$.
Let $P_k = \{x \in R^n : c^t x \geq \beta_{k-1}\} \cap P_{k-1}$.

In order to find the new "approximate" volumetric center of P_k, we execute the following steps

Let $x^k = x^j$ then for $j = 1$ to $J = \lceil 30 \log(2\epsilon^{-4.5}) \rceil$, do $x^j \leftarrow x^j - 0.18Q(x^j)^{-1}\nabla F^{new}(x^j)$.

Set $x^k = x^J$.

STOP in

$$O(nL_1) = O(n(4L + l) + nlog(\frac{9M}{\epsilon^2 a}))$$

iterations of the algorithm.

If $y_{\frac{\epsilon}{2}}^{k(i)}$ is an $\frac{\epsilon}{2}$-near optimal solution of $min_{x \in K} f(x^i)^t(x - x^i)$, the proposed solution is $x^* = x^j$, with $j = argmax_{i \in OC(k)}[f(x^i)^t(y_{\frac{\epsilon}{2}}^{k(i)} - x^i)]$. Next we attempt to obtain a better understanding of the four algorithms.

Remarks:

1. To select the proposed near optimal solution $x^* = x^j$, in all four algorithms we find an $\frac{\epsilon}{2}$-near optimal solution of a linear objective optimization problem (which is solvable in polynomial time as shown in the end of Subsection 5.2).

   Alternatively, we can apply a stopping criterion at each iteration and terminate the algorithms earlier. We check whether

   $$max_{i \in OC(k)}[(f(x^i)^t(y_{\frac{\epsilon}{2}}^{k(i)} - x^i)]) \geq -\frac{\epsilon}{2}.$$

   Then the algorithm terminates in *at most* $k^* = O(-\frac{nL_1}{logb(n)})$ iterations (the complexity for each of the four algorithms is given explicitely in Tables II and III).

   In Section 5.2 we have shown that if the problem function f satisfies the strong-f-monotonicity condition, then the point x^* is indeed a near optimal solution.

2. All the points x^k generated by algorithms 1 and 3 are feasible, i.e., $x^k \in K$. This is true because $P_k = P_{k-1} \cap H_k^c = (K \cap H_0^c \cap H_1^c \cap \ldots \cap H_{k-1}^c) \cap H_k^c \subseteq K$, so the center of gravity and the center of of the maximum inscribed ellipsoid x^k of P_k respectively, always lie in K. A drawback to algorithm 1 and 3 is that computing the center of gravity and the center of of the maximum inscribed ellipsoid x^k of an arbitrary convex, compact body is relatively hard. As we mentioned, a randomized polynomial

algorithm (see [4]) will find the center of gravity.

Algorithms 1 and 3 use no feasibility cut because the sequence of points x^k (which are centers of P_k) always lies in the feasible set K. Alternatively, we could have chosen an initial set as $P_0 \supseteq K$, the center, or an "approximate" center of which, would be easy to compute. This center, would not necessarily lie in the feasible set K. Step k of algorithms 1 and 3 would include "a feasibility cut" (as in algorithm 2, the ellipsoid). We would "cut" P_0 with the help of an oracle (as in part a of the ellipsoid algorithm) so that the feasible set K belongs in the new set P_1, and continue the algorithm by moving to the new approximate center of this set P_1. When the center lies in K, we would continue with Part b (the optimality cut), as described in step k of algorithms 1 and 3.

3. In a linear programming problem

$$Min_{x \in P} \; c^t x$$

the coefficient $c_k = -f(x^{k-1}) = -c$ is a constant. Then the hyperplanes $H_k = \{x \in R^n : (-c)^t x = (-c)^t x^{k-1}\}$ that cut P_k always remain parallel. This way algorithms 1 always preserves the "shape" of $P_k = P_{k-1} \cap H_k^c$. In other words if after the first cut P_1 is a simplex then so is $P_k, k = 2, 3, \ldots$ As mentioned before, the center of gravity of a simplex P_k, with vertices $x_0^k, x_1^k, \ldots, x_n^k$, is explicitly given by $x^k = \frac{x_0^k + x_1^k + \ldots + x_n^k}{n+1}$. This idea gives rise to the "method of simplices" introduced in 1982 by Levin and Yamnitsky for solving the problem of finding a feasible point of a polytope. Using the previous remark, we can use an implementation of this method to solve linear programs (see Levin and Yamnitsky [31]).

The general geometric framework for the VIP and its four special cases described in this section, choose at the kth iteration either a feasibility or an optimality cut so that the volume of the "nice" set becomes smaller by a factor $b(n)$. In Table II and III summarizes the complexity results for the four algorithms and the general framework.

Table II: Reduction Lemmas.

Algorithm	**Reduction Inequality**
General framework	$vol(P_{k+1}) \leq b(n) vol(P_k)$
Method of centers of gravity	$vol(P_{k+1}) \leq \frac{e-1}{e} vol(P_k)$
Ellipsoid Method	$vol(P_{k+1}) \leq 2^{-\frac{1}{2(n+1)}} vol(P_k)$
Method of inscribed ellipsoids	$vol(P_{k+1}) \leq 0.843 vol(P_k)$
Vaidya's algorithm	$vol(P_{k+1}) \leq const\ vol(P_k)$

Table III: Complexity results and reduction constants.

Algorithm	**Reduction Constant**	**Complexity**
General framework	$b(n)$	$O(-\frac{nL_1}{log b(n)})$
Method of centers of gravity	constant= $\frac{e-1}{e}$	$O(nL_1)$
Ellipsoid Method	$2^{O(\frac{1}{n})} = 2^{-\frac{1}{2(n+1)}}$	$O(n^2 L_1)$
Method of inscribed ellipsoids	constant= 0.843	$O(nL_1)$
Vaidya's algorithm	constant	$O(nL_1)$

Proofs of the reduction inequalities can be found in B.S. Mityagin [21], Yudin and Nemirovski [23], and Levin [14] for algorithm 1, Khatchiyan [11], Papadimitriou and Steiglitz [24], Nemhauser and Wolsey [22], and Vavasis [30] for algorithm 2, L.G.Khatchiyan, S.P.Tarasov and I.I. Erlikh, [12] for algorithm 3 and, finally, in Vaidya [29] for algorithm 4. More details on the complexity results for the four algorithms when applied to linear and convex programs can be found in these references and, when applied to variational inequalities, in [25].

Remarks:

1. Each iteration of algorithm 2 (the Ellipsoid Method for $VIPs$) requires $O(n^3 + T)$ arithmetic operations (or $O(n^2 + T)$ operations using rank-one updates) because of

the matrix inversion, performed at each iteration and the cost T required to call an oracle (that is, to solve the separation problem). Overall, the ellipsoid algorithm for the variational inequality problem requires a total of $O(n^5L_1 + n^2L_1T)$ operations (or $O(n^4L_1 + n^2L_1T)$ using rank-one updates).

2. Each iteration of algorithm 4 (Vaidya's algorithm) requires $O(n^3 + T)$ arithmetic operations.

   Consider the kth iteration:

   (1) Since $\sigma_i = \frac{(a_i^t H(x)^{-1} a_i)}{(a_i^t x - b_i)^2}$ for $1 \leq i \leq m_k$, and the number m_k of constraints of P_k is $m_k = O(n)$, we can evaluate σ_i in $O(n^3)$ arithmetic operations.

   (2) Each iteration executes $O(1)$ Newton steps.

   (3) At each Newton step, we need to evaluate $\nabla F(x) = -\sum_{i=1}^m \sigma_i(x) \frac{a_i}{a_i^t x - b_i}$, $Q(x) = \sum_{i=1}^m \sigma_i(x) \frac{a_i a_i^t}{(a_i^t x - b_i)^2}$. Consequently, if $\sigma_i(x)$ for $i = 1, 2, ..., m$ are available, then we can evaluate $\nabla F(x)$, $Q(x)$, $Q(x)^{-1}$ in $O(n^3)$ operations.

   (4) The oracle is called once at each iteration and costs T operations.

   (5) Computing β so that $\frac{c^t(H(x^{k-1})^{-1}c}{(c^t x^{k-1} - \beta)^2} = \frac{(\delta\epsilon)^{1/2}}{2}$ requires $O(n^3)$ operations.

   As a result of (1)-(5), each iteration requires $O(n^3 + T)$ operations (or as in algorithm 2, $O(n^2 + T)$ using rank-one updates). Overall, Vaidya's algorithm for the variational inequality problem requires a total of $O(nL_1(n^3 + T)) = O(n^4L_1 + TnL_1)$ arithmetic operations (or $O(n^3L_1 + TnL_1)$ using rank-one updates) .

3. The complexity of the algorithms we have considered in this paper depends on the number of variables. Thus these algorithms are most efficient for problems with many constraints. but few variables. Furthermore, the use of an oracle (which establishes whether a given point lies in the feasible set or not) permits us to deal with problems whose constraints are not explicitly given a priori.

4. The method of centers of gravity, the method of inscribed ellipsoids, and Vaidya's algorithm have a better complexity (require fewer iterations) than the ellipsoid algorithm.

One intuitive explanation for this behavior is because when it adds a hyperplane (characterized by a vector c, which is given either by an oracle or by $c = -f(x^k)$), the ellipsoid algorithm generates a half ellipsoid which it immediately encloses in a new ellipsoid, smaller than the original one. The subsequent iterations do not use the vector c. Consequently, we lose a considerable amount of information, since the ellipsoids involved at each iteration are only used in that iteration. On the other hand, the other three algorithms use the sets involved at each iteration (polytopes, convex sets) in subsequent iterations.

5. Vaidya's algorithm not only adds, but also eliminates hyperplanes from time to time. This tactic provides a better complexity (i.e., $O(nL_1)$ number of iterations). If we altered the algorithm so that it only added hyperplanes (i.e., we discard case 2), the algorithm would require $O(n^2L_1^2)$ iterations to decrease the volume of the polytopes P_k to 2^{-nL_1}.

6. From Theorem 5.1 and Tables II and III we conclude that the best complexity result we can achieve with a geometric algorithm that fits in the general framework is $O(nL_1)$, since the reduction constant $b(n)$ can become at best a constant.

The analysis in this paper shows that the general geometric framework we introduced for solving the $VI(f, K)$ problem finds an ϵ-near optimal solution in polynomial number of iterations. If we were not interested in examining whether the algorithm is polynomial, then a similar analysis (without the assumption $b(n) = O(2^{-\frac{n}{pol(n)}})$ would lead us to conclude that some subsequence of the points $\{x^k\}_{k=1}^{\infty}$ generated by the general geometric framework converges to a $VI(f, K)$ solution x^{opt}. The analysis would work under assumptions A1 and A2. The steps we would conduct for establishing this convergence analysis would be similar to the ones we used, namely,

a. We would show that the volume of the nice sets P_k converges to zero (due to the rate of decrease of volume: $volP_k \leq b(n)volP_{k-1}$) using an analysis similar to Theorem 5.1.

b. We would show that the subsequence of $\{x^k\}_{k=1}^{\infty}$ converges to a point x^* satisfying the condition

$$f(x^*)^t(x^* - x^{opt}) = 0$$

(using an analysis similar to Lemma 5.1 and Proposition 5.1)

c. We would show that the point x^* is indeed an optimal solution (using an analysis similar to the one used in Theorem 5.2).

6 Conclusions and open questions

In this paper we considered the variational inequality problem $VI(f, K)$:

$$\text{Find } x^{opt} \in K \subseteq R^n : f(x^{opt})^t(x - x^{opt}) \geq 0, \ \forall x \in K$$

defined by a problem function $f : K \subseteq R^n \to R^n$.

We viewed linear programming problems as variational inequalities $VI(c, P)$:

$$x^{opt} \in P \subseteq R^n : c^t(x - x^{opt}) \geq 0, \ \forall x \in P,$$

defined by a constant problem function $f(x) = c$ and by a polyhedral ground set P. We developed theoretically efficient algorithms for solving the $VI(f, K)$ problem by extending theoretically efficient algorithms for linear programming problems. Moreover, we unified these algorithms in a general geometric framework by exploiting the common geometric characteristics of these algorithms. We introduced complexity results and the notion of polynomiality for the $VI(f, K)$ problem. The question we addressed here was: can we find a "near" optimal solution in "polynomial-time". The convergence analysis used the condition of strong-f-monotonicity, which is weaker than strict and strong monotonicity. Linear programs satisfy this new assumption, so the general geometric framework introduced in this paper also applies to linear programs.

We now address some open questions:

1. In the four geometric algorithms, as well as in the general framework,

can we relax the stopping criterion?

Currently, the algorithm would terminate in *at most* $k = O(-\frac{nL_1}{logb(n)})$ iterations when

$$max_{i \in OC(k)}[f(x^i)^t(y_{\frac{\epsilon}{2}}^{k(i)} - x^i)] \geq -\frac{\epsilon}{2}.$$

In this expression, $y_{\frac{\epsilon}{2}}^{k(i)}$ is an $\frac{\epsilon}{2}$-near optimal solution of the linear optimization problem $min_{x \in K}[f(x^i)^t(x - x^i)]$. Verifying this criterion requires that we find an $\frac{\epsilon}{2}$-near optimal solution of a linear optimization problem at the end of each iteration which, as we have shown, is solvable in polynomial-time.

Can we use a stopping criterion that is easier to check ?

2. *Can we relax the definition of the center of a nice set further?*
Can we replace condition 3, $volP_k \leq b(n)volP_{k-1}$, by a condition that would imply the conclusion of Theorem 5.2.

3. *Can we define new types of "nice" sets and their corresponding centers so that we would design new geometric algorithms for solving VIP as well as LP problems?* These new algorithms would also be special cases of the general geometric framework.

Acknowledgments: We are grateful to Robert Freund for several suggestions that improved the technical content of this paper and to a referee whose probing questions helped us to improve and clarify several points. The second author also wishes to acknowledge the Applied Mathematics Department at Brown University. Part of this work was completed while she was a student there.

References

[1] A. Auslender (1976), *Optimisation: Methodes Numeriques*, Mason, Paris.

[2] C. Baiocchi and A. Capelo (1984), *Variational and Quasivariational Inequalities: Application to Free Boundary Problems*, Wiley, New York.

[3] R.G. Bland, D. Goldfarb and M. Todd, (1981), "The ellipsoid method: A survey", *Operations Research*, 29, 1039-1091.

[4] M. Dyer, A. Friesz and R. Kannan (1989), "A random polynomial time algorithm for approximating the volume of convex bodies", *Proc. 21st Annual ACM Symposium on Theory of Computing*, 375-381 (full paper will appear in *J. ACM*).

[5] D. Gabay (1983), "Applications of the method of multipliers to variational inequalities", in M. Fortin, R. Glowinski, eds., *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*, North-Holland, Amsterdam.

[6] M. Groetschel, L. Lovasz and A. Schrijver (1988), *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, Berlin.

[7] J.H. Hammond (1984), *Solving Asymmetric Variational Inequality Problems and Systems of Equations with Generalized Nonlinear Programming Algorithms*, Ph.D. dissertation, Department of Mathematics, M.I.T., Cambridge M.A.

[8] J.H. Hammond and T.L. Magnanti (1987), "Generalized descent methods for asymmetric systems of equations", *Mathematics of Operations Research*, 12, 678-699.

[9] J.H. Hammond and T.L. Magnanti (1987), "A contracting ellipsoid method for variational inequality problems", Working paper OR, 160-87, Operations Research Center, M.I.T.

[10] P.T. Harker and J-S Pang (1990), "Finite-dimensional variational inequality and nonlinear complementarity problems: a survey of theory, algorithms and applications", *Mathematical Programming*, 48, 161-220.

[11] L.G. Khatchyian (1979), "A polynomial algorithm in linear programming", *Doklady Akad. Nauk USSR*, 244, 1093-1096. Translated as *Soviet Math. Doklady*, 20, 191-194.

[12] L.G. Khatchyian, S.P. Tarasov, and I.I. Erlikh (1988), "The method of Inscribed Ellipsoids", *Soviet Math. Doklady*, vol 37, no 1, 226-230.

[13] D. Kinderlehrer and G. Stampacchia, (1980), *An introduction to variational inequalities and their application*, Academic Press, New York.

[14] A.L. Levin (1965), "On an algorithm for the minimization of convex functions", *Doklady Akad. Nauk USSR*, 160, 1244-1247. Translated as *Soviet Math. Doklady*, 6, 286-290.

[15] L. Lovasz (1986) *An algorithmic theory of numbers, graphs and convexity*, CBMS-NSF Regional Conference Series in Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania.

[16] H.-J. Luthi (1985), "On the solution of variational inequalities by the ellipsoid method", *Mathematics of Operations Research*, 10, 515-522.

[17] T.L. Magnanti and G. Perakis (1993), "On the convergence of classical variational inequality algorithms", Working paper OR 280-93, Operations Research Center, M.I.T.

[18] T.L. Magnanti and G. Perakis (1993), "Convergence conditions for variational inequality algorithms", Working paper OR 282-93, Operations Research Center, M.I.T.

[19] P. Marcotte and D. Zhu (1993), "Co-coercivity and its role in the convergence of iterative schemes for solving variational inequalities", Centre de Recherche sur les Transports, Universite de Montreal, preprint.

[20] M. Minoux (1986), *Mathematical programming: theory and algorithms*, Wiley, New York.

[21] B.S. Mityagin (1969), "Two inequalities for the volume of convex bodies", *Mathematical notes of the Academy of Sciences USSR*, 5, No 1, 61-64, Plenum, New York.

[22] G.L. Nemhauser and L. Wolsey (1988), *Integer and combinatorial optimization*, Wiley, New York.

[23] A.S. Nemirovski and D.B. Yudin (1983), *Problem complexity and method efficiency in optimization*, Wiley.

[24] C.H. Papadimitriou and K. Steiglitz, (1982), *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, New Jersey.

[25] G. Perakis (1992), *Geometric, interior point, and classical methods for solving finite dimensional variational inequality problems*, Ph.D. dissertation, Department of Applied Mathematics, Brown University, Providence R.I.

[26] N.Z. Shor (1977), "Cut off methods with space extension in convex programming problems", *Cybernetics*, 13, 94-96.

[27] N.Z. Shor (1977), "New development trends in nondifferential optimization", *Cybernetics*, 13, 881-886.

[28] P. Tseng (1990), "Further applications of a matrix splitting algorithm to decomposition in variational inequalities and convex programming", *Mathematical Programming*, 48, 249-264.

[29] P.M. Vaidya (1990), "A new algorithm for minimizing convex functions over convex sets", *Preprint*, AT&T Bell Labs., Murray Hill, New Jersey.

[30] S. Vavasis (1992) *Nonlinear optimization; Complexity issues*, Oxford Science Publication, New York.

[31] B. Yamnitsky and L.A. Levin (1982), "An old linear programming algorithm runs in polynomial time", *Proceedings of IEEE*, 327-328.