Mathematical Programming 48 (1990) 265-284
North-Holland

SENSITIVITY ANALYSIS BASED HEURISTIC ALGORITHMS FOR MATHEMATICAL PROGRAMS WITH VARIATIONAL INEQUALITY CONSTRAINTS

Terry L. FRIESZ
George Mason University, Fairfax, VA 22030, USA

Roger L. TOBIN
GTE Laboratories Incorporated, Waltham, MA 02254, USA

Hsun-Jung CHO and Nihal J. MEHTA
University of Pennsylvania, Philadelphia, PA 19104, USA

Received 17 February 1989
Revised manuscript received 20 November 1989

In this paper we consider heuristic algorithms for a special case of the generalized bilevel mathematical programming problem in which one of the levels is represented as a variational inequality problem. Such problems arise in network design and economic planning. We obtain derivative information needed to implement these algorithms for such bilevel problems from the theory of sensitivity analysis for variational inequalities. We provide computational results for several numerical examples.

Key words: Bi-level programming, variational inequalities, sensitivity analysis, nonsmooth optimization, heuristics.

1. Introduction

In this paper we consider a special case of the generalized bilevel mathematical programming problem studied by Marcotte (1986), Harker and Pang (1990) and Harker and Choi (1987). Namely, if y and x represent the decision vectors associated with the upper and lower levels respectively, then we wish to solve

$$\text{minimize} \quad F(x, y) \tag{1}$$

$$\text{subject to} \quad L \leqslant y \leqslant U, \tag{2}$$

$$Z(x, y)(z - x) \geqslant 0 \quad \forall z \in K(y), \tag{3}$$

$$x \in K(y), \tag{4}$$

where $F : \mathbb{R}^n \times \mathbb{R}^q \to \mathbb{R}^1$ is differentiable in (x, y), $L \in \mathbb{R}^q_+$, $U \in \mathbb{R}^q_+$, and $K(y)$ is the feasible set, possibly dependent on y, of x-variables:

$$K(y) = \{x \in \mathbb{R}^n : g(x, y) \geqslant 0, h(x, y) = 0\}. \tag{5}$$

We will consider a finite dimensional problem with $x = (x_1, \ldots, x_n)$; $y = (y_1, \ldots, y_q)$; $g : \mathbb{R}^n \times \mathbb{R}^q \to \mathbb{R}^m$, differentiable and concave in x; $h : \mathbb{R}^n \times \mathbb{R}^q \to \mathbb{R}^p$, linear in x; and $Z : \mathbb{R}^n \times \mathbb{R}^q \to \mathbb{R}^n$, differentiable and strictly monotone in x. As shown in Section 2, these conditions guarantee that the variational inequality (3)-(4) has a unique solution $\boldsymbol{x}^*$ for each $\boldsymbol{y}$.

Problems of this form are difficult to solve for several reasons:

(i) Evaluation of the objective function $F(\cdot,\cdot)$ at a given point $\bar{\boldsymbol{y}}$ is difficult since it requires finding $\bar{\boldsymbol{x}} \in K(\bar{\boldsymbol{y}})$ solving the variational inequality

$$Z(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})(\boldsymbol{z}-\bar{\boldsymbol{x}}) \geqslant 0 \quad \forall \boldsymbol{z} \in K(\bar{\boldsymbol{y}}). \tag{6}$$

(ii) Evaluation of the gradient vector of the objective function with respect to $\boldsymbol{y}$ at a point $\bar{\boldsymbol{y}}$ (including the variation due to changes in $\boldsymbol{x}$ induced by changes in $\boldsymbol{y}$ through the variational inequality constraints (3)) requires solving the variational inequality (6) and performing a sensitivity analysis of $\boldsymbol{x}$ with respect to $\boldsymbol{y}$.

(iii) The problem (1)-(4) is a nonconvex minimization problem. The implicit function $\boldsymbol{x} = \xi(\boldsymbol{y})$ defined by the variational inequality constraints (3) is in effect a nonlinear equality constraint.

(iv) The problem is a nonsmooth optimization problem. The implicit function $\boldsymbol{x} = \xi(\boldsymbol{y})$ is not differentiable everywhere. Because the problem is nonconvex, the standard subgradient methods do not apply.

Many algorithms for (1)-(4) and related problems have been proposed in the literature. Kolstad (1985) suggests that the algorithms proposed can fit into the following typology: (i) extreme point search techniques, (ii) Kuhn-Tucker methods, and (iii) descent methods. In descent methods, the most difficult task is that of calculating derivatives of functions in the first level problem since these functions depend parametrically on decision variables from the second level problem. Several authors, including De Silva (1978), Magnanti and Wong (1984), Kolstad and Lasdon (1986) and Tobin and Friesz (1988), have proposed calculating such derivatives using results from nonlinear programming and variational inequality sensitivity analysis. In fact, De Silva (1978), Kolstad and Lasdon (1986) and Friesz, Tobin and Miller (1988) implement algorithms for bilevel mathematical programs which employ sensitivity analysis theory.

Because of points (i) and (ii) mentioned above, methods are required that minimize the number of objective function evaluations or the number of objective gradient evaluations. Because of (iii), the best that can be hoped for with most computationally efficient procedures is a local minimum. Because of (iv), the gradient of the objective function does not exist everywhere. Additionally, the gradient vector can change discontinuously, and therefore the length of the gradient vector does not provide any information about appropriate step length.

There is some theoretical basis for optimism, however. For many applications, the implicit function $\boldsymbol{x} = \xi(\boldsymbol{y})$, even though not differentiable everywhere, is relatively smooth, and the objective function strongly convex in $\boldsymbol{y}$. In these cases, a local minimum is likely to be a global minimum. As is shown subsequently, the points of non-differentiability of the implicit function ξ make up a set of measure zero, and therefore, points exist in a small neighborhood of a non-differentiable point at which the implicit function is differentiable. Therefore, rather than worry about finding subgradients at points of non-differentiability, the step length or the step direction can be perturbed slightly, and the gradient calculated at the nearby point.

The aim of this work is to devise algorithms that do not require many objective function or gradient evaluations. Descent algorithms require either objective function evaluations, evaluation of the objective gradient vector, or a similar evaluation to determine a descent direction. Therefore, in any primal approach, such information will be required at each step of the algorithm, at least. The more difficult aspect of a descent algorithm for such problems is to determine step lengths. To guarantee improvement in the objective, some sort of search or line minimization is required along the descent direction to determine how far one can move and still improve. This generally requires many objective function evaluations. The step length determination is further complicated by the fact that the derivatives of the implicit function are discontinuous, and a descent direction may discontinuously become an ascent direction.

To avoid the aforementioned complexities involved in determining step lengths that lead to an improvement at every step, it seems natural to employ heuristic methods that use a predetermined step length sequence. At this point, little can be said theoretically about the convergence of such heuristics for the problem studied here, although empirical tests may provide evidence of the usefulness of such an approach.

In this paper, we analyze three heuristic sensitivity analysis based algorithms for problem (1)-(4). The first heuristic is a descent algorithm which employs an Armijo-like step size rule designed to deal with the inherent nonconvexity of problem (1)-(4); variational inequality sensitivity analysis is used to calculate derivatives of the objective function $F(x, y)$ with respect to y-variables, recognizing that the x-variables are implicit functions of y. This algorithm is included to demonstrate the difficulties in step length determination. The second heuristic, for which several variants are included, is similar to the first except that step sizes are determined a priori and decrease monotonically with the number of iterations. The third heuristic presented in this paper is a generalization of the "equilibrium decomposed optimization" algorithm proposed by Suwansirikul et al. (1987). In particular, this generalization finds the descent direction along each coordinate axis through the use of derivatives calculated with variational inequality sensitivity analysis, and the step length is determined by a bisection rule as in a Bolzano search.

After briefly presenting the necessary results from the theory of variational inequality sensitivity analysis, we give formal statements of each of the three algorithms considered and present extensive numerical tests.

2. Summary of key results from variational inequality sensitivity analysis

This section contains results summarized from Tobin (1986) and Kyparisis (1987) which are presented without proof. More general results can be found in Pang

(1988) and Kyparisis (1989). However, the generality provided there is not required for our purposes.

Let

$\boldsymbol{Z}(\boldsymbol{x}, \boldsymbol{y})$ be once differentiable in $(\boldsymbol{x}, \boldsymbol{y})$,

$\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y})$ be concave in $\boldsymbol{x}$ and twice continuously differentiable in $(\boldsymbol{x}, \boldsymbol{y})$,

$\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y})$ be linear affine in $\boldsymbol{x}$ and once continuously differentiable in $\boldsymbol{y}$.

Now consider the following perturbed variational inequality, denoted as VI($\boldsymbol{y}$): find $\boldsymbol{x}^* \in K(\boldsymbol{y})$ such that

$$\boldsymbol{Z}(\boldsymbol{x}^*, \boldsymbol{y})^{\mathrm{T}}(\boldsymbol{x} - \boldsymbol{x}^*) \geqslant 0 \quad \text{for all } \boldsymbol{x} \in K(\boldsymbol{y}) \tag{7}$$

where

$$K(\boldsymbol{y}) = \{\boldsymbol{x} \mid \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}) \geqslant 0, \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) = 0\}. \tag{8}$$

Theorem 1. *Consider the following conditions on the perturbed variational inequality* VI($\bar{\boldsymbol{y}}$):

(a) *The constraints $g_i(\boldsymbol{x}, \bar{\boldsymbol{y}})$ are concave in $\boldsymbol{x}$, and $\boldsymbol{x}^* \in K$, $\lambda^* \in \mathbb{R}^m$ and $\mu^* \in \mathbb{R}^p$ satisfy*

$$\boldsymbol{Z}(\boldsymbol{x}^*, \bar{\boldsymbol{y}}) - \nabla \boldsymbol{g}(\boldsymbol{x}^*, \bar{\boldsymbol{y}})^{\mathrm{T}} \lambda^* - \nabla \boldsymbol{h}(\boldsymbol{x}^*, \bar{\boldsymbol{y}})^{\mathrm{T}} \mu^* = 0, \tag{9}$$

$$\lambda^{*\mathrm{T}} \boldsymbol{g}(\boldsymbol{x}^*, \bar{\boldsymbol{y}}) = 0, \tag{10}$$

$$\lambda^* \geqslant 0. \tag{11}$$

(b) *$\boldsymbol{Z}(\boldsymbol{x}^*, \bar{\boldsymbol{y}})$ is such that*

$$\boldsymbol{w}^{\mathrm{T}} \nabla \boldsymbol{Z}(\boldsymbol{x}^*, \bar{\boldsymbol{y}}) \boldsymbol{w} > 0 \tag{12}$$

for all $\boldsymbol{w} \neq 0$ such that

$$\nabla g_i(\boldsymbol{x}^*, \bar{\boldsymbol{y}}) \boldsymbol{w} \geqslant 0 \quad \textit{for } i \textit{ such that } g_i(\boldsymbol{x}^*, \boldsymbol{y}) = 0, \tag{13}$$

$$\nabla g_i(\boldsymbol{x}^*, \bar{\boldsymbol{y}}) \boldsymbol{w} = 0 \quad \textit{for } i \textit{ such that } \lambda_i^* > 0, \tag{14}$$

$$\nabla h_i(\boldsymbol{x}^*, \bar{\boldsymbol{y}}) \boldsymbol{w} = 0 \quad \textit{for } i = 1, \ldots, p. \tag{15}$$

(b′) *$\nabla \boldsymbol{Z}(\boldsymbol{x}^*, \bar{\boldsymbol{y}})$ is positive definite.*

(c) *The gradients, $\nabla g_i(\boldsymbol{x}^*, \bar{\boldsymbol{y}})$ for i such that $g_i(\boldsymbol{x}^*, \bar{\boldsymbol{y}}) = 0$ and $\nabla h_i(\boldsymbol{x}^*, \bar{\boldsymbol{y}})$ for $i = 1, \ldots, p$, are linearly independent.*

(d) *The strict complementary slackness conditions*

$$\lambda_i^* > 0 \quad \textit{when } g_i(\boldsymbol{x}^*, \bar{\boldsymbol{y}}) = 0$$

are satisfied.

We then have the following:

(i) *If* (a) *is satisfied, then $\boldsymbol{x}^*$ is a solution to* VI($\bar{\boldsymbol{y}}$).

(ii) *If, in addition,* (b) *or* (b′) *is satisfied, then $\boldsymbol{x}^*$ is a locally unique solution to* VI($\bar{\boldsymbol{y}}$).

(iii) *If, in addition,* (c) *is satisfied, then λ^* and μ^* are unique and for $\boldsymbol{y}$ in a neighborhood of $\bar{\boldsymbol{y}}$, there exists a unique, directionally differentiable, Lipschitz continuous function $[\boldsymbol{x}(\boldsymbol{y})^{\mathrm{T}}, \lambda(\boldsymbol{y})^{\mathrm{T}}, \mu(\boldsymbol{y})^{\mathrm{T}}]^{\mathrm{T}}$ where $\boldsymbol{x}(\boldsymbol{y})$ is a locally unique solution to* VI($\boldsymbol{y}$), $\lambda(\boldsymbol{y})$

and $\mu(y)$ are unique associated multipliers satisfying (a) *and* (b) *above for* VI(y), *and with*

$$[x(\bar{y})^{\mathrm{T}}, \lambda(\bar{y})^{\mathrm{T}}, \mu(\bar{y})^{\mathrm{T}}]^{\mathrm{T}} = [x^{*\mathrm{T}}, \lambda^{*\mathrm{T}}, \mu^{*\mathrm{T}}]^{\mathrm{T}}.$$

Additionally, in a neighborhood of $\bar{y}$, the set of binding constraints is unchanged and the binding constraint gradients are linearly independent at $x(y)$.

(iv) *If, in addition,* (d) *is satisfied, then the function $[x(y)^{\mathrm{T}}, \lambda(y)^{\mathrm{T}}, \mu(y)^{\mathrm{T}}]^{\mathrm{T}}$ is differentiable in a neighborhood of $\bar{y}$.* □

Theorem 2. *If the vector x^* is a solution to the variational inequality* VI($\bar{y}$) *and the gradients $\nabla g_i(x^*, \bar{y})$ for i such that $g_i(x^*, \bar{y}) = 0$ and $\nabla h_i(x^*, \bar{y})$ for $i = 1, \ldots, p$ are linearly independent, then there exists $\lambda^* \in \mathbb{R}^m$, $\mu^* \in \mathbb{R}^p$ such that* (9), (10) *and* (11) *are satisfied.* □

Let x^* be a solution to VI($\bar{y}$) satisfying Theorem 2. Then, for $y = \bar{y}$ and $(x, \lambda, \mu) = (x^*, \lambda^*, \mu^*)$ we have

$$Z(x, y) - \sum_{i=1}^{m} \lambda_i \nabla g_i(x, y)^{\mathrm{T}} - \sum_{i=1}^{p} \mu_i \nabla h_i(x, y)^{\mathrm{T}} = 0, \tag{16}$$

$$\lambda_i g_i(x, y) = 0 \quad \text{for } i = 1, 2, \ldots, m, \tag{17}$$

$$h_i(x, y) = 0 \quad \text{for } i = 1, 2, \ldots, p. \tag{18}$$

Let the Jacobian matrix of the system (16), (17) and (18) with respect to $w = (x, \lambda, \mu)$ be denoted as J_w^* and with respect to y as J_y^*. Then we may state the following results:

Corollary 1. *If the constraints $g_i(x, \bar{y})$ are concave and the conditions* (b) (*or* (b′)), (c) *and* (d) *in Theorem* 2.1 *are satisfied, then the inverse of J_w^* exists and the derivatives of (x^*, λ^*, μ^*) with respect to y are given by*

$$\nabla_y w(\bar{y}) = J_w^{-1}(\bar{y})[-J_y(\bar{y})]. \quad \square \tag{19}$$

Furthermore, we may show that the points of nondifferentiability of the implicit function $x = \xi(y)$ defined by the variational inequality constraints (3) form a set of measure zero, and, therefore, points exist in a small neighborhood of a nondifferential point at which this implicit function is differentiable. In particular, the following result holds:

Theorem 3. *Suppose x^* solves the variational inequality* (7) *and* (8) *for $y = \bar{y}$ and the conditions of* (a), (b) (*or* (b′)) *and* (c) *of Theorem* 1 *are satisfied. Then in any open neighborhood of $\bar{y}$ there exists points $\hat{y}$ such that the implicit function $w(y) = [x(y)^{\mathrm{T}}, \lambda(y)^{\mathrm{T}}, \mu(y)^{\mathrm{T}})]^{\mathrm{T}}$ is differentiable at $\hat{y}$.*

Proof. By Theorem 1, $w(y)$ is Lipschitz continuous in an open neighborhood $U(\bar{y})$ of $\bar{y}$. By the theorem of Rademacher (e.g., see Rockafellar, 1981), $w(y)$ is differentiable almost everywhere on $U(\bar{y})$. □

Theorem 3 allows the development of algorithms for (1)–(4) using the gradient to determine a descent direction. If at step k, y^k is such that $w(y^k)$ is not differentiable, then by a systematic search in an ε-neighborhood of y^k, a point $\hat{y}^k$ can be found where $w(\hat{y}^k)$ is differentiable. This may be accomplished by returning to y^{k-1} and perturbing the step length and/or direction. In practice, it is not likely that a step will land on a point at which the implicit function is non-differentiable. If it does, it is likely that a single change in the step length will be sufficient.

3. Variations of a gradient projection descent algorithm

In this section we propose several variations of a gradient projection descent algorithm for solving problems like (1)–(4). These algorithms are heuristics because the problem is nonconvex. Also, we cannot guarantee that certain derivatives needed for their implementation will always exist, although by Theorem 2.3 we do know that there will be a point in the neighborhood of the current iterate for which the required derivatives exist. The two descent algorithms proposed in this section differ from one another in the manner in which stepsizes are determined. Specifically, in the first algorithm considered, the stepsize is determined at each iteration according to the Armijo-like rule, which has been analyzed in great detail by Bertsekas (1982a, 1982b). In the others, step sizes are determined a priori and decrease monotonically with the number of iterations.

If the conditions (a) and (b) in Theorem 2.1 hold, we know that for each y such that $L \leqslant y \leqslant U$, the variational inequality (3)–(4) has a unique solution, $x(y)$. Hence, we may rewrite the objective function $F(\cdot,\cdot)$ of (1) as

$$M(y) = F(x(y), y). \tag{20}$$

Then, (1)–(4) becomes

$$\begin{aligned} &\text{minimize} \quad M(y) = F(x(y), y) \\ &\text{subject to} \quad L \leqslant y \leqslant U. \end{aligned} \tag{21}$$

Let $[\cdot]_L^U$ denote projection on the feasible region: i.e., for every $y = [y_1, \ldots, y_n]^{\mathrm{T}}$,

$$[y]_L^U = \begin{bmatrix} \min[\max[L_1, y_1], U_1] \\ \vdots \\ \min[\max[L_n, y_n], U_n] \end{bmatrix}.$$

The gradient projection algorithm uses iterations of the form

$$y^{k+1} = [y^k - \alpha_k \nabla M(y^k)^{\mathrm{T}}]_L^U.$$

In other words, the algorithm proceeds as follows. First, an initial feasible vecto y_0 is chosen. Then at each successive iteration (indexed by k), we determine the vector y^k by moving in descent direction the distance $\alpha_k \nabla M(y^k)$. Thus, there are two major aspects of the algorithm: determining the descent direction and determining the stepsize.

The algorithm using the Armijo-like type step length determination for solving the bilevel problem (1)–(4) is given as follows:

Algorithm 1.

```
Step 0. Determine an initial value $y^0$; set $k = 0$.
Step 1. Solve the variational inequality problem, (3)–(4), for given $y^k$ and get $x^k$.
Step 2. Calculate the sensitivity information $\nabla_y x^k$.
Step 3. Calculate $\nabla_y M$ using sensitivity analysis information and the chain rule.
Step 4. Calculate the stepsize using the Armijo rule (Bertsekas, 1982b).
Step 5. Calculate $y^{k+1} = [y^k - \alpha_k \nabla M(y^k)^{\mathrm{T}}]_L^U$.
Step 6. If $|y_i^{k+1} - y_i^k| < \varepsilon$ for all $i$, then stop, otherwise let $k = k + 1$ and go to Step 1.
```

We shall refer to Algorithm 1 as the sensitivity descent algorithm with Armijo-like steps, or SDAA for short. In Step 3, $\nabla_y M = (\ldots, \partial M/\partial y_i, \ldots)$,

$$\frac{\partial M}{\partial y_i} = \sum_j \frac{\partial F}{\partial x_j}\frac{\partial x_j}{\partial y_i} + \frac{\partial F}{\partial y_i},$$

and $\partial x_j/\partial y_i$ are calculated using eq. (19).

Step 4 of the algorithm may be replaced with a step size rule with a predetermined sequence of monotonically decreasing step sizes. We will consider three different step size sequences α_k. For each of these, we will consider two different direction vectors d^k: the gradient vector and the normalized gradient vector. This leads to the following general statement of a gradient projection algorithm with predetermined step size sequences:

Algorithm 2.

```
Step 0. Determine an initial value $y^0$; set $k = 0$.
Step 1. Solve the variational inequality problem, (3)–(4), for given $y^k$ and get $x^k$.
Step 2. Calculate the sensitivity information $\nabla_y x^k$.
Step 3. Calculate $\nabla_y M$ using sensitivity analysis information and the chain rule.
        Determine $d^k$ according to either (i) $d^k = \nabla_y M$ or (ii) $d^k = (1/\|\nabla_y M\|)\nabla_y M$.
Step 4. Calculate the stepsize $\alpha_k$.
Step 5. Calculate $y^{k+1} = [y^k - \alpha_k \nabla M(y^k)^{\mathrm{T}}]_L^U$.
Step 6. If $|y_i^{k+1} - y_i^k| < \varepsilon$ for all $i$, then stop, otherwise let $k = k + 1$ and go to Step 1.
```

In the case that the direction vector $d^k = \nabla_y M$, the Algorithm 2 is termed SDAP (sensitivity descent algorithm — predetermined). In the case that $d^k = (1/\|\nabla_y M\|)\nabla_y M$, it is termed NSDAP (normalized SDAP). For each of these, we will investigate three step size sequences:

1. $\alpha_k = \beta/(k+1)$.
2. $\alpha_k = \beta/2^k$.
3. $\alpha_k = \beta/(k+2)^2$.

The sum of the first is a non-convergent series, and in general will require the most iterations. Because, given enough iterations, it can move the solution to any point, the value of β and the starting solution y_0 are in one sense not critical. However, a poor choice of β could cause a prohibitively large number of iterations. The sums of the second and third sequences converge, and therefore, if the values of β and the starting solution $\boldsymbol{y}^0$ are not carefully chosen, some feasible point (perhaps optimal) may not be reachable. Both the second and third sequences are such that if the current solution is very near the minimum, and the next step is relatively large and takes the next solution to a point far from the minimum, the remaining steps can take the solution back to the minimum.

Algorithms based on the non-normalized gradient direction and predetermined step sequence j $(j=1,2,3)$ will be referred to as SDAP*j*. Those based on the normalized gradient sequence and predetermined step sequence j will be referred to as NSDAP*j*.

Unless otherwise noted, in all of the computational tests, the starting solution $\boldsymbol{y}^0$ will be the center of the region defined by $\boldsymbol{L} \leqslant \boldsymbol{y} \leqslant \boldsymbol{U}$. This allows for the smallest initial step.

For NSDAP1, β is chosen to allow the algorithm to travel to the furthest extreme point in approximately 11 iterations; that value of β is given as

$$\beta_{N1} = \tfrac{1}{6}\sqrt{\sum_i (U_i - L_i)^2}$$

since $\sum_{k=0}^{10} \alpha_k \approx 3\beta$ and the starting point is the center of the feasible region. For SDAP1, we let

$$\beta_1 = \sqrt{\sum_i (U_i - L_i)^2} / (3\|\nabla_y M(\boldsymbol{y}^0)\|).$$

This is a heuristic and is based on the assumption that the norm of the gradient gets smaller as the algorithm progresses and is, on average, half of its value at $\boldsymbol{y}^0$. If this assumption holds, the first step will be twice as big as that in NSDAP1, and will reduce more quickly. In general, the non-normalized algorithms are unpredictable since the gradient vector $\nabla_y M(\boldsymbol{y}^k)$ can vary greatly, even discontinuously, over the feasible region.

In the case of the second step length sequence, the series

$$\sum_{k=0}^{\infty} \alpha_k = 2\beta$$

and so for the normalized case NSDAP2, β is chosen to be

$$\beta_{N2} = \tfrac{1}{4}\sqrt{\sum_i (U_i - L_i)^2}.$$

Using this value, if the algorithm always travels toward the most distant extreme point, it can reach it in the limit. Using the same reasoning as before about the gradient, for SDAP2 we let β be

$$\beta_2 = \sqrt{\sum_i (U_i - L_i)^2}/(2\|\nabla_y M(y^0)\|).$$

Finally, the third step length is such that

$$\sum_{k=0}^{\infty} \alpha_k = (\tfrac{1}{6}\pi^2 - 1)\beta \approx 0.645\beta.$$

In the third step sequence, as is the case with the other two,

$$\alpha_k \leqslant \sum_{i=k+1}^{\infty} \alpha_i \quad \text{for all } k,$$

allowing for the recovery from too large a step. Thus, for NSDAP3 we take β to be

$$\beta_{N3} = \frac{1}{1.14}\sqrt{\sum_i (U_i - L_i)^2},$$

which is based on the desire to be able to reach the vicinity of the furthest point in not too many (say 12) iterations and the fact that $\sum_{k=0}^{11} \alpha_k \approx 0.57\beta$. Then, similarly to the previous case, for SDAP3 we take β to be

$$\beta_3 = \sqrt{\sum_i (U_i - L_i)^2}/(0.57\|\nabla_y M(y^0)\|).$$

In the case of the first algorithm, SDAA with the Armijo-like steps, we employ as a value of β,

$$\beta_A = \sqrt{\sum_i (U_i - L_i)^2}/(10\|\nabla_y M(y^0)\|).$$

Here, each step is smaller than those in previous cases, since it is not systematically reduced, and we desire to limit the number of times the step must be reduced by the Armijo rule.

4. An interior point, bisection heuristic

The third heuristic for problem (1)-(4) is a generalization of the decomposition scheme suggested by Suwansirikul et al. (1987) for the equilibrium network design problem. It is similar to that scheme in its strategy of applying simultaneous one dimensional searches along each coordinate axis. It is different in that actual derivatives are calculated in conjunction with a Bolzano search method. In the decomposition scheme of Suwansirikul et al. (1987), when the Bolzano search was employed, approximate derivatives were used that assume the contribution from the lower level problem is small. Otherwise, Suwansirikul et al. (1987) employed a simultaneous Golden Section search which does not require derivatives. In the decomposition heuristic proposed here, the feasible region is reduced at each iteration and the current solution is at the center of the region. The descent direction is in the direction of one of the extreme points of the region and the step length is half the distance to the extreme point. In particular, the step length is determined by using simultaneous Bolzano line searches along each coordinate axis.

The algorithm is as follows:

Algorithm 3.

Step 0. Set $y^0 = [y_i^0] = [\frac{1}{2}(L_i + U_i)]$, $L_i^0 = L_i$, $U_i^0 = U_i$ and set $k = 0$.

Step 1. Solve the variational inequality problem (3)-(4) for given $\boldsymbol{y}^k$ and get $\boldsymbol{x}^k$.

Step 2. Calculate the sensitivity information $\nabla_y \boldsymbol{x}^k$.

Step 3. Calculate $\nabla_y M$ using sensitivity analysis information and the chain rule.

Step 4. Calculate $\boldsymbol{L}^{k+1}$ and $\boldsymbol{U}^{k+1}$ according to

$$\text{if } \partial M/\partial y_i \leq 0, \quad \text{set } L_i^{k+1} = y_i^k, \;\; U_i^{k+1} = U_i^k, \tag{22}$$

$$\text{if } \partial M/\partial y_i > 0, \quad \text{set } U_i^{k+1} = y_i^k, \;\; L_i^{k+1} = L_i^k. \tag{23}$$

Step 5. Calculate $\boldsymbol{y}^{k+1}$ as

$$y_i^{k+1} = \tfrac{1}{2}(U_i^{k+1} + L_k^{k+1}) \quad \forall i. \tag{24}$$

Step 6. If $|y_i^{k+1} - y_i^k| < \varepsilon$ for all i, then stop, otherwise let $k = k + 1$ and go to Step 1.

Since Suwansirikul et al. (1987) refer to their heuristic as "equilibrium decomposed optimization", we refer to this generalization as generalized equilibrium decomposed optimization, or GEDO for short.

5. Numerical examples

In this section we present several numerical examples drawn from equilibrium network design, hierarchical mathematical programming and game theory, which are solved using the algorithms presented here.

In these algorithms, most of the computational effort is in solving the variational inequality at each iteration. Therefore, the number of variational inequalities solved is used as the measure of comparison. This also allows us to compare our results to the method using Hooke–Jeeves pattern searches (Abdulaal and LeBlanc, 1979) (referred to as H–J), and to the equilibrium decomposed optimization method (EDO) (Suwansirikul, Friesz and Tobin, 1987).

Examples 1, 2 and 3 are specific cases of the well-known equilibrium network design problem, wherein network-wide congestion costs are minimized subject to flow conservation, budget and nonnegativity constraints, as well as variational inequality constraints which force the flow pattern to be a user equilibrium. As such the equilibrium network design problem is a leader–follower game on a network. The equilibrium network design problem employs the following notation:

- f = vector of arc flows,
- c = vector of arc unit travel costs,
- y = vector of arc capacity improvements,
- G = vector of arc improvement costs,
- Ω = set of feasible flow patterns satisfying conservation and nonnegativity constraints,
- I = an identity matrix.

This notation leads to the following problem statement:

$$\begin{aligned} &\text{minimize} && c(f, y)f + \theta G I \\ &\text{subject to} && c(f, y)(f - \bar{f}) \geq 0 \;\; \forall \bar{f} \in \Omega(y), \\ &&& f \in \Omega(y), \\ &&& y \geq 0, \end{aligned}$$

where θ is an approximate dual variable for the budget constraint. Note that successful application of the heuristic algorithms introduced in this paper to the equilibrium network design problem requires that the nonuniqueness of path variables be dealt with in a manner which preserves the regularity conditions stated in Section 2, as in Tobin and Friesz (1988).

Examples 1, 2 and 3 are taken from Suwansirikul, Friesz and Tobin (1987) and appear there in complete detail. Example 1 corresponds to their test network 1, Example 2 corresponds to their test network 3 and Example 3 corresponds to their test network 4.

Example 1. In this example, which has 5 arcs and 4 nodes, the traffic demand between the single origin–destination pair was varied from 100 to 300, creating four different levels of traffic congestion. Table 1 shows the results of applying the algorithms to these four problems.

Example 2. In this example, which has 16 arcs and 6 nodes, the traffic demand between one O–D pair (node 1 to node 6) was varied from 2.5 to 10.0 and the demand between the other (node 6 to node 1) was simultaneously varied from 5.0 to 20.0, creating three different levels of traffic congestion. Tables 2 shows the results for these problems.

Example 3. This example is based on the aggregate network of Sioux Falls, South Dakota, has 76 arcs and 24 nodes. Only 10 of the 76 arcs are candidates for improvements. The traffic demands on the 552 O–D pairs represent peak hour flow. The results for the algorithms are shown in Tables 3a and 3b. Table 3b shows the effect of a different starting point for the NSDAP algorithms. Because of implementation difficulties with the special methods required to satisfy the uniqueness requirments, the sensitivity analysis information required for this example was generated

Table 1
Results for small network design problem

Demand		MINOS[a]	H-J	EDO	SDAA	GEDO	SDAP1	SDAP2	SDAP3	NSDAP1	NSDAP2	NSDAP3
100	y_1	1.34	1.25	1.31	1.34	1.33	1.33	1.38	1.33	1.34	1.34	1.36
	y_2	1.21	1.20	1.19	1.21	1.20	1.20	1.25	1.19	1.21	1.21	1.21
	y_3	0.00	0.00	0.00	0.10	0.02	0.17	0.00	0.07	0.00	0.00	0.00
	y_4	0.97	0.95	0.94	0.97	0.95	0.95	1.00	0.94	0.96	0.97	0.95
	y_5	1.10	1.10	1.10	1.10	1.11	1.09	1.14	1.08	1.10	1.10	1.11
	Z	1 200.58	1 200.61	1 200.64	1 200.60	1 200.58	1 200.62	1 200.61	1 200.59	1 200.58	1 200.58	1 200.59
	#VI[b]	—	11	8	13	7	8	6	5	26	7	12
150	y_1	6.05	5.95	5.98	6.05	5.99	6.05	5.93	6.01	6.16	6.04	6.01
	y_2	5.47	5.65	5.52	5.47	5.47	5.47	5.40	5.49	5.57	5.46	5.43
	y_3	0.00	0.00	0.00	0.11	0.01	0.34	0.21	0.35	0.00	0.03	0.00
	y_4	4.64	4.60	4.61	4.64	4.65	4.64	4.56	4.63	4.77	4.64	4.63
	y_5	5.27	5.20	5.27	5.27	5.24	5.27	5.20	5.28	5.42	5.27	5.26
	Z	3 156.21	3 156.38	3 156.24	3 156.23	3 156.23	3 156.38	3 156.42	3 156.40	3 156.50	3 156.21	3 156.23
	#VI	—	14	11	19	9	21	8	11	6	8	10
200	y_1	12.78	13.00	12.86	12.98	12.87	12.97	12.82	12.91	13.01	12.98	12.95
	y_2	11.73	11.75	12.02	11.73	12.00	11.74	12.08	12.06	11.76	11.73	11.70
	y_3	0.00	0.00	0.00	0.16	0.00	1.27	2.57	2.47	0.00	0.31	0.00
	y_4	10.34	10.25	10.33	10.34	10.33	10.34	10.47	10.46	10.34	10.34	10.34
	y_5	11.74	11.75	11.77	11.74	11.75	11.73	11.87	11.84	11.74	11.74	11.75
	Z	7 068.12	7 068.21	7 068.45	7 068.15	7 068.42	7 088.55	7 096.70	7 095.86	7 086.13	7 086.26	7 086.12
	#VI	—	20	12	40	11	152	12	60	134	11	26
300	y_1	28.45	28.44	28.11	28.46	28.12	28.46	32.70	28.40	28.43	28.45	28.52
	y_2	25.73	25.75	26.03	25.72	26.02	25.72	32.24	25.88	25.70	25.73	25.78
	y_3	0.00	0.00	0.00	0.16	0.00	0.72	0.88	1.39	0.00	2.90	0.01
	y_4	23.40	24.44	23.39	23.40	23.40	23.39	26.26	23.38	23.39	23.40	23.40
	y_5	26.57	26.56	25.58	26.57	26.57	26.57	29.92	26.63	26.57	26.56	26.57
	Z	21 209.9	21 209.9	21 210.5	21 209.9	21 210.7	21 210.7	21 848.5	21 212.9	21 209.9	21 222.5	21 209.9
	#VI	—	28	14	27	14	92	21	45	447	16	22

[a]See Tan et al. (1979).
[b]Number of variational inequality problems solved.

Table 2
Results for 16 arcs network design problem

Demand		MINOS[a]	H-J	EDO	SDAA	GEDO	SDAP1	SDAP2	SDAP3	NSDAP1	NSDAP2	NSDAP3
(1)→(6)	y_1	0.00	0.00	0.00	0.00	0.00	0.36	0.22	0.36	0.49	0.85	0.87
=2.5	y_2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.54	0.54
(6)→(1)	y_3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.00	0.00	0.00
=5.0	y_4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.21
	y_5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_6	5.00	0.30	1.84	2.32	0.00	1.12	1.13	1.17	2.01	1.31	1.30
	y_7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.53	0.54
	y_9	0.00	0.00	0.00	0.00	0.00	0.36	0.22	0.36	0.49	0.85	0.86
	y_{10}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_{11}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_{12}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_{13}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_{14}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.53	0.54
	y_{15}	1.33	0.10	0.02	1.33	0.00	0.00	0.00	0.00	1.40	0.05	0.02
	y_{16}	0.00	0.30	1.84	0.78	0.75	1.19	1.13	1.16	1.07	1.26	1.27
	Z	92.10	90.10	92.41	96.52	90.58	92.92	92.30	92.96	99.30	102.05	102.13
	#VI[b]	—	39	11	14	10	7	5	5	45	7	11
(1)→(6)	y_1	0.00	0.00	0.00	0.00	0.00	0.00	0.60	0.51	2.48	2.78	2.73
=5.0	y_2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.42	1.82	1.75
(6)→(1)	y_3	0.00	1.20	0.13	0.00	0.00	0.00	0.00	0.00	0.06	0.35	0.34
=10.0	y_4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.56	0.46
	y_5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_6	6.58	3.00	6.26	4.97	5.00	4.85	4.75	4.76	4.62	4.61	4.60
	y_7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.56	0.46
	y_8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.26	1.70	1.62
	y_9	0.00	0.00	0.00	0.00	0.00	0.00	0.59	0.51	2.48	2.78	2.73
	y_{10}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_{11}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_{12}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

	y_{13}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	y_{14}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.30	1.73	1.66
	y_{15}	7.01	3.00	0.13	0.00	0.00	0.00	0.00	0.00	0.09	0.66	0.55
	y_{16}	0.22	2.80	6.26	6.69	5.00	6.48	6.24	6.20	5.25	4.83	4.95
	Z	211.25	215.08	201.84	199.84	202.30	199.94	202.50	202.21	224.43	235.51	233.56
	#VI	—	54	10	14	10	17	7	13	16	8	14
(1)→(6)	y_1	0.00	0.00	0.00	0.00	0.01	0.00	0.04	0.00	3.27	4.99	4.89
= 10.0	y_2	4.61	5.40	4.88	4.61	4.62	4.63	4.66	4.62	5.06	5.43	5.48
(6)→(1)	y_3	9.86	8.18	8.59	9.79	6.24	7.80	8.46	7.42	7.09	6.66	6.72
= 20.0	y_4	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
	y_5	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
	y_6	7.71	8.10	7.48	8.38	15.63	12.72	14.24	13.75	13.42	13.49	13.39
	y_7	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.01	0.00
	y_8	0.59	0.90	0.85	0.59	0.60	0.65	0.59	0.64	1.89	3.29	3.25
	y_9	0.00	0.00	0.00	0.00	0.01	0.00	0.20	0.11	3.37	5.03	4.95
	y_{10}	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
	y_{11}	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
	y_{12}	0.00	0.00	0.00	0.00	3.76	1.81	4.18	2.19	2.60	2.90	2.96
	y_{13}	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
	y_{14}	1.32	3.90	1.54	1.43	1.62	1.85	1.69	1.84	3.51	4.56	4.56
	y_{15}	19.14	8.10	0.26	0.00	14.99	16.96	15.88	16.70	16.20	16.13	15.87
	y_{16}	0.85	8.40	12.52	18.38	5.01	6.21	6.85	6.66	7.52	8.45	8.34
	Z	557.14	557.22	540.74	523.45	566.61	563.64	570.46	565.02	581.23	593.28	592.71
	#VI	—	134	12	14	10	13	9	13	28	9	13

[a]See Tan et al. (1979).

[b]Number of variational inequality problems solved.

Table 3a

Results for the Sioux Falls network design problem

	H-J	H-J	EDO	GEDO	SDAP1	SDAP2	SDAP3	NSDAP1	NSDAP2	NSDAP3
Initial value of y	2.0	1.0	12.5	12.5	12.5	12.5	12.5	12.5	12.5	12.5
y_{16}	4.8	3.8	4.59	5.0	5.55	6.17	6.16	5.14	7.49	6.28
y_{17}	1.2	3.6	1.52	1.5	2.26	3.06	3.14	1.66	5.08	3.45
y_{19}	4.8	3.8	5.45	5.0	5.55	6.17	6.18	5.12	7.50	6.31
y_{20}	0.8	2.4	2.33	1.5	2.25	3.08	3.14	1.61	5.07	3.45
y_{25}	2.0	2.8	1.27	2.5	4.53	5.81	5.72	2.86	7.45	5.89
y_{26}	2.6	1.4	2.33	2.5	4.56	5.82	5.74	2.86	7.46	5.90
y_{29}	4.8	3.2	0.41	4.5	4.74	4.59	4.82	4.72	5.73	5.08
y_{39}	4.4	4.0	4.59	3.5	4.52	4.86	4.93	4.40	6.31	5.15
y_{48}	4.8	4.0	2.71	4.5	4.82	4.66	4.86	4.74	5.77	5.10
y_{74}	4.4	4.0	2.71	4.0	4.51	4.86	4.90	4.37	6.23	5.14
Z	80.78	81.21	83.08	84.14	83.79	84.49	84.06	83.08	87.08	86.21
#VI[a]	58	108	12	8	15	7	10	59	8	17

[a]Number of variational inequality problems solved.

Table 3b

Results for the Sioux Falls network design problem

Initial value of y	NSDAP1	NSDAP2	NSDAP3
	9.0	9.0	9.0
y_{16}	5.14	5.19	5.25
y_{17}	1.60	1.65	1.73
y_{19}	5.17	5.19	5.28
y_{20}	1.59	1.55	1.60
y_{25}	2.76	2.81	2.79
y_{26}	2.76	2.86	2.86
y_{29}	4.67	4.68	4.58
y_{39}	4.43	4.32	4.40
y_{48}	4.75	4.72	4.59
y_{74}	4.41	4.35	4.30
Z	83.33	82.97	82.91
#VI[a]	66	8	19

[a]Number of variational inequality problems solved.

using finite differences rather than eq. (19). This, of course, defeats the intention of developing algorithms with minimal number of solutions of the variational inequality, but indicates how the algorithm would perform in terms of number of iterations if eq. (19) had been used; the solutions of the variational inequality used for the finite differences (ten per iteration) are not reported in the tables.

Example 4. This example is the following bilevel nonlinear programming problem studied by De Silva (1978):

$$\begin{aligned}&\text{minimize} \quad y_1^2-2y_1+y_2^2-2y_2+x_1(y)^2+x_2(y)^2\\&\text{subject to} \quad 0\leqslant y_1, y_2\leqslant 2,\end{aligned}$$

where $\boldsymbol{x}(\boldsymbol{y})=[x_1(y)x_2(y)]^{\mathrm{T}}$ solves $E(\boldsymbol{y})$ which is defined below.

$$\begin{aligned}E(\boldsymbol{y}):\quad &\text{minimize} \quad (x_1-y_1)^2+(x_2-y_2)^2\\&\text{subject to} \quad 0.25-(x_i-1)^2\geqslant 0, \ i=1,2.\end{aligned}$$

This program has a point of nondifferentiability at the optimal solution $(y_1^*, y_2^*)=(0.5, 0.5)$. The results are presented in Table 4.

Example 5. This example is a Stackelberg leader-follower game studied by Henderson and Quandt (1980); it is stated as

$$\max_{q_1\geqslant 0} \ \pi_1=q_1F(q_1+q_2)-C_1(q_1)$$

subject to

$$\max_{q_2\geqslant 0} \ \pi_2=q_2F(q_1+q_2)-C_2(q_2),$$

Table 4
Results for De Silva problem[a]

	SDAA	GEDO	SDAP1	SDAP2	SDAP3	NSDAP1	NSDAP2	NSDAP3	SUMT
y_1	0.5004	0.50	0.5026	0.5012	0.5099	0.50	0.4995	0.50	0.505
y_2	0.5004	0.5	0.5026	0.5012	0.5099	0.50	0.4995	0.50	0.505
Z	−1.0	−1.0	−1.0	−1.0	−0.9996	−1.0	−0.999	−1.0	
No. of subproblems	25	16	49	6	31	31	16	21	30

[a]See De Silva (1978).

Table 5
Results for Stackelberg problem[a]

	SDAA	GEDO	SDAP1	SDAP2	SDAP3	NSDAP1	NSDAP2	NSDAP3	EXACT
y	93.16	93.33	92.61	94.82	93.51	93.33	93.26	93.18	93.33
Z	3266.66	3266.67	3266.47	3265.84	3266.65	3266.67	3266.67	3266.66	3266.67
No. of subproblems	22	15	35	10	5	102	10	23	

[a]See Henderson and Quandt (1980).

where π_1 denotes the profit of the leader and π_2 denotes the profit of the follower. The following functions are employed:

inverse market demand: $p = F(q_1 + q_2) = 100 - 0.5(q_1 + q_2)$;

leader's total cost: $C_1(q_1) = 5q_1$;

follower's total cost: $C_2(q_2) = 0.5q_2^2$.

The exact solution is known to be $(q_1, \pi_1) = (93\frac{1}{3}, 3266\frac{2}{3})$. The results are presented in Table 5.

6. Discussion of results

One important result that is not presented in the tables should be discussed first. As discussed in point (iv) of Section 1, nondifferentiable points were, in fact, encountered in the 16 arc example. A nondifferentiable point is easily detected, since the Jacobian matrix J_w in eq. (19) is not invertible. As implemented, failure of the inversion routine triggered an adjustment of the step size (as per Theorem 2.3). As would be expected, these nondifferentiable points were only encountered for some starting points.

Another observation is that for the larger problems, local optima clearly exist, and the choice of starting point has an effect on the solution found.

Rather than discuss the results for each example, it is more instructive to look at patterns over all examples. The gradient projection method using the Armijo step size rule was expected to require, on average, solving more than one variational inequality problem to determine the step length for one iteration. However, for the value of β chosen for these numerical examples, except for a few cases (e.g., Example 1, demand 200), re-evaluation of step size was not required, and the algorithm performed better than expected.

The first of the predetermined step length algorithms, based on the sequence $\beta/(k+1)$, as expected did not perform well in terms of number of iterations, but did well in terms of objective values. The algorithm SDAP1 requires fewer iterations than NSDAP1, since in general the norm of the gradient decreases as the solution progresses, and the change in the solution satisfies the termination criterion sooner. The algorithm NSDAP1 is predictable, however, since the magnitude of the change in the solution vector is known a priori. Because of the slow decrease in the step size, many iterations are required to obtain stopping accuracy.

The gradient projection algorithms based on the second step size sequence $(\beta/2^k)$ and the third step size sequence $(\beta/(k+2)^2)$ worked well. The second sequence requires fewer iterations than the third, and the additional iterations of the third sequence do not provide significantly improved solutions.

As is the case for the first step size sequence, the non-normalized step algorithms converged more rapidly than the normalized in many cases, but for the network design examples, increased congestion eliminated or reversed this. This occurs because the congestion level has little impact on the normalized step, but has a large impact on the non-normalized step. In Examples 2 and 3, the normalized steps converged to different local solutions than the non-normalized steps, and had poorer objective values. When the normalized steps were run from a different starting point (Table 3b), they found the same local optima as the non-normalized steps.

Another interesting comparison is between GEDO and NSDAP2. These algorithms have the same step length sequence, but different step direction. One would expect NSDAP2 to perform better than GEDO since it uses a steepest descent direction and the descent direction in GEDO is restricted. This is the case when they both converge to the same local solution, but in Examples 2 and 3, NSDAP2 converged to poorer solutions. The performance of GEDO is comparable to that of EDO.

It appears that these algorithms can be useful for solving mathematical programs with large scale variational inequalities as constraints. A suggested approach, which will be tested in future work, is based on the observation that quite good solutions can be obtained in very few iterations. The approach is to use SDAP2 and NSDAP2 at several widely separated starting points, and then select the best solution obtained. This solution is then used as a starting point for the algorithm using the Armijo step length rule. Since the Armijo rule guarantees improvement at each step, and the predetermined step algorithm has put the starting point close to the local optimum, the local optimum should be found in few Armijo iterations. Careful selection of β, however, will be required for the Armijo steps. This approach is easy to implement since the same algorithmic structure, except for step length rule, is used throughout.

References

M. Abdulaal and L.J. LeBlanc, "Continuous equilibrium network design models," *Transportation Research* 13B (1979) 19-32.

D.P. Bertsekas, "Projected Newton methods for optimization problems with simple constraints," *SIAM J. Control and Optimization* 20(2) (1982a) 221-246.

D.P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods* (Academic Press, New York, 1982b).

A.H. De Silva, "Sensitivity formulas for nonlinear factorable programming and their application to the solution of an implicitly defined optimization model of US crude oil production," Dissertation, George Washington University (Washington, D.C., 1978).

T.L. Friesz, T. Miller and R.L. Tobin, "Algorithms for spatially competitive network facility-location," *Environment and Planning B* 15 (1988) 191-203.

P.T. Harker and J.-S. Pang, "Finite-dimensional variational inequality and nonlinear complementarity problems: A survey of the theory, algorithms and applications," *Mathematical Programming (Series B)* 48 (1990) 161-220, this issue.

P.T. Harker and S.-C. Choi, "A penalty function approach for mathematical programs with variational inequality constraints," Decision Science Working Paper 87-09-08, University of Pennsylvania (Philadelphia, PA, 1987).

J.M. Henderson and R.E. Quandt, *Microeconomic Theory* (McGraw-Hill, New York, 1980, 3rd ed.).

C.D. Kolstad, "A review of the literature on bi-level mathematical programming," Los Alamos National Laboratory Report, LA-10284-MS (Los Alamos, 1985).

C.D. Kolstad and L.S. Lasdon, "Derivative evaluation and computational experience with large bi-level mathematical programs," BEBR Faculty Working Paper No. 1266, University of Illinois (Urbana-Champaign, IL, 1986).

J. Kyparisis, "Sensitivity analysis framework for variational inequalities," *Mathematical Programming* 38 (1987) 203-213.

J. Kyparisis, "Solution differentiability for variational inequalities and nonlinear programming problems," Department of Decision Sciences and Information Systems, College of Business Administration, Florida International University (Miami, FL, 1989).

T.L. Magnanti and R.T. Wong, "Network design and transportation planning: models and algorithms," *Transportation Science* 18(1) (1984) 1-55.

P. Marcotte, "Network design problem with congestion effects: a case of bilevel programming," *Mathematical Programming* 34 (1986) 142-162.

J.-S. Pang, "Solution differentiability and continuation of Newton's method for variational inequality problems over polyhedral sets," Department of Mathematical Sciences, The Johns Hopkins University (Baltimore, MD, 1988).

R.T. Rockafellar, *The Theory of Subgradients and its Applications to Problems of Optimization: Convex and Nonconvex Functions* (Heldermann, Berlin, 1981).

C. Suwansirikul, T.L. Friesz and R.L. Tobin, "Equilibrium decomposed optimization: a heuristic for the continuous equilibrium network design problem," *Transportation Science* 21(4) (1987) 254-263.

H.N. Tan, S.B. Gershwin and M. Athaus, "Hybrid optimization in urban traffic networks," Report No. DOT-TSC-RSPA-79-7, Laboratory for Information and Decision System, M.I.T. (Cambridge, MA, 1979).

R.L. Tobin, "Sensitivity analysis for variational inequalities," *Journal of Optimization Theory and Applications* 48 (1986) 191-204.

R.L. Tobin and T.L. Friesz, "Sensitivity analysis for equilibrium network flow," *Transportation Science* 22(4) (1988) 242-250.