INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Enhancing Lagrangian Dual Optimization for Linear Programs by Obviating Nondifferentiability

Hanif D. Sherali, Churlzu Lim,

To cite this article:
Hanif D. Sherali, Churlzu Lim, (2007) Enhancing Lagrangian Dual Optimization for Linear Programs by Obviating Nondifferentiability. INFORMS Journal on Computing 19(1):3-13. https://doi.org/10.1287/ijoc.1050.0158

INFORMS Journal on Computing
Vol. 19, No. 1, Winter 2007, pp. 3–13
ISSN 1091-9856 | EISSN 1526-5528 | 07 | 1901 | 0003

DOI 10.1287/ijoc.1050.0158

Enhancing Lagrangian Dual Optimization for Linear Programs by Obviating Nondifferentiability

Hanif D. Sherali, Churlzu Lim
Grado Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061, USA {hanifs@vt.edu, clim2@uncc.edu}

We consider nondifferentiable optimization problems that arise when solving Lagrangian duals of large-scale linear programs. Different from traditional subgradient-based approaches, we design two new methods that attempt to circumvent or obviate the nondifferentiability of the objective function, so that standard differentiable optimization techniques could be used. These methods, called the *perturbation technique* and the *barrier-Lagrangian reformulation*, are implemented as initialization procedures to provide a warm start to a theoretically convergent nondifferentiable optimization algorithm. Our computational study reveals that this two-phase strategy produces much better solutions with less computation in comparison with both the stand-alone nondifferentiable optimization procedure employed, and the popular Held-Wolfe-Crowder subgradient heuristic. Furthermore, the best version of this composite algorithm is shown to consume only about 3.19% of the CPU time required by the commercial linear programming solver CPLEX 8.1 (using the dual simplex option) to produce the same quality solutions. We also demonstrate that this initialization technique greatly facilitates quick convergence in the primal space when used as a warm start for ergodic-type primal recovery schemes.

Key words: nondifferentiable optimization; Lagrangian relaxation; Lagrangian dual; perturbation technique; barrier-Lagrangian reformulation
History: Accepted by William J. Cook, Area Editor for Design and Analysis of Algorithms; received April 2004; revised April 2005; accepted July 2005.

1. Introduction

Consider the linear-programming problem

$$\text{LP:}\quad \text{Minimize}\ c^T x \tag{1a}$$
$$\text{subject to}\ Ax = b \tag{1b}$$
$$l \le x \le u, \tag{1c}$$

where A is an $m \times n$ matrix, and l and u are vectors of (finite) lower and upper bounds, respectively, on the x variables. If LP is a large, ill-conditioned problem, simplex or interior-point solvers can get prohibitively expensive (Adams and Sherali 1993, Sherali and Tuncbilek 1997, for example). Instead, Lagrangian relaxation, or Lagrangian dual optimization, along with a primal recovery scheme, could be more effective for obtaining lower bounds and selecting branching variables (Fisher 1981, Sherali and Myers 1988, Sherali and Choi 1996, Larsson et al. 1999, Barahona and Anbil 2000). The Lagrangian dual of LP is

$$\text{LD:}\quad \text{Maximize}\ \theta(\pi), \tag{2}$$

where $\theta(\pi) \equiv b^T\pi + \min\{c^T x - \pi^T Ax\colon l \le x \le u\}$.

Although the Lagrangian dual optimization problem is a convex program, the nondifferentiability of θ obviates the use of conventional differentiable optimization methods. Hence, various nondifferentiable optimization (NDO) procedures are applied to solve LD, such as subgradient and conjugate subgradient algorithms (Polyak 1967, 1969; Held et al. 1974; Camerini et al. 1975; Sherali and Ulular 1989), bundle methods (Mifflin 1977, Lemarechal 1977, Makela 2002), and space-dilation algorithms (Shor 1970a, b; Sherali et al. 2001). Bundle algorithms require solving quadratic subproblems, and space-dilation procedures need to store matrices and perform matrix operations, each of which can be prohibitive for large problems. On the other hand, conjugate subgradient methods can be effective approaches for solving large problems due to their relatively milder computational and memory requirements. However, this type of approach usually requires a suitable estimate of the optimal objective value to compute step lengths. For example, Sherali et al. (2000) proposed a variable target value method (VTVM) that assures convergence to an optimum when used with a conjugate subgradient direction strategy. Although the VTVM algorithm has demonstrated good performance in solving several standard test problems, we have observed that if the initial target value is far from the optimal objective value, the algorithm might stall for a significant number of iterations while the target value is adjusted to an appropriate level, before a progression

of improvements results. In this vein, Lim and Sherali (2006) recently proposed a modified VTVM that employs a variable acceleration factor when updating target values and adopts a projected quadratic-fit line-search whenever an improvement is attained. Among various direction-finding and step-length strategies investigated in concert with the modified VTVM procedure in their extensive computational study, Lim and Sherali prescribed a generalized Polyak-Kelley cutting-plane (GPKC) technique as a highly effective alternative. This approach simultaneously determines a direction and a step length by projecting an iterate onto the most recent pair of cuts, and then additionally utilizing sequential projections onto several other past cuts to obtain the next iterate.

Besides optimizing the Lagrangian dual, the recovery of primal solutions is another important consideration. A computationally simple idea for deriving primal solutions in Lagrangian dual optimization is to construct an ergodic sequence based on the solutions to the corresponding Lagrangian subproblems during the normal course of the dual search algorithm, in a manner that would induce convergence to an optimal pair of dual and primal solutions. Shor (1985) proved the primal convergence of sequences of this type when using specific weighting schemes in concert with pure subgradient optimization. Sherali and Choi (1996) proposed an extension of such weighting strategies and established primal convergence when employing both pure and deflected subgradient algorithms. Larsson et al. (1999) further extended these weighting schemes to yield primal convergence for general convex programs.

In this paper, we develop two new approaches for solving LD in (2), which attempt to circumvent or obviate the nondifferentiability of θ in providing a warm start for theoretically convergent NDO procedures. One of these methods involves perturbing iterates to create a differentiable trajectory of solutions, while the second approach examines the Lagrangian dual of a barrier reformulation of the problem, which yields a continuously twice differentiable dual function. In doing so, these approaches admit the use of a rich array of effective existing solution technologies for differentiable optimization problems. Note that with the same motivation, Nesterov (2005) has recently independently proposed a method for constructing a smooth ε-approximation having a Lipschitz continuous gradient for a nondifferentiable function, such that an ε-optimal solution can be derived using conventional differentiable optimization procedures. In contrast, although the second of our aforementioned two approaches can be run to produce an ε-optimal solution on its own, we utilize these methods only to provide a good initial starting solution.

In §2, we present a technique that utilizes perturbations at nondifferentiable points to create a differentiable pathway so that conventional differentiable optimization methods can be brought to bear for solving LD. Next, we introduce a barrier-function approach in §3 (see Bazaraa et al. 2006 for a general discussion on barrier-function methods), which provides a continuously twice differentiable Lagrangian dual function, thereby permitting application of standard differentiable optimization techniques to furnish a warm start for both the dual optimization and ergodic-primal-recovery schemes. Some differentiable optimization strategies that we implement for our numerical study are briefly described in §4. Subsequently, in §5, we design algorithmic procedures in which these proposed techniques are applied to initialize the theoretically convergent variable target value method that incorporates generalized Polyak-Kelley cuts (VTVM-GPKC), as proposed by Lim and Sherali (2006), in the hope of enhancing its numerical performance. Computational test results are provided in §6 to demonstrate the efficacy of the proposed dual optimization and ergodic-primal-recovery strategies. The results reveal that the best proposed option improves the quality of solutions produced by the stand-alone VTVM-GPKC algorithm by reducing the optimality gap by 72.5% while being run for only about half as many iterations, and consumes only 3.19% of the CPU effort required by the dual-simplex option of the commercial software CPLEX 8.1 to obtain the same quality near-optimal solutions.

2. Perturbation Technique

The method developed in this section for inducing a differentiable trajectory is called the *perturbation technique* (PT), and is motivated by the fact that the Lagrangian dual function θ is differentiable almost everywhere. Notice that θ is nondifferentiable only at π^k such that there exists a $j \in \{1, \ldots, n\}$ for which the reduced cost $c_j - (\pi^k)^T A^j = 0$, where c_j is the jth element of c, and A^j is the jth column of A. Although the probability that we randomly hit a nondifferentiable point is theoretically zero, this occurs routinely in practice because of the nature of the ascent process over the piecewise-linear objective surface, and due to the characterization of (optimal) basic feasible solutions whereby the basic variables have zero reduced costs. However, as described below, we can use a perturbation technique to evade nondifferentiable points en route toward an optimum as possible, and contend with nondifferentiability whenever this is unavoidable, hopefully only in the vicinity of an optimum.

To implement this idea, suppose that we encounter a nondifferentiable point via the update $\hat{\pi}^k = \pi^{k-1} +$

$\lambda_{k-1}d^{k-1}$, where π^{k-1}, λ_{k-1}, and d^{k-1} are respectively the previous iterate, step length, and search direction. From the condition of nondifferentiability, we have a nonempty index set $E = \{j \in \{1, \ldots, n\}: c_j - (\hat{\pi}^k)^T A^j = 0\}$. To find a suitable differentiable point π^k in the neighborhood of $\hat{\pi}^k$, we want to perturb $\hat{\pi}^k$ to $\pi^k = \hat{\pi}^k + \varepsilon\tilde{\pi}$ such that $c_j - (\pi^k)^T A^j = c_j - (\hat{\pi}^k)^T A^j - \varepsilon\tilde{\pi}^T A^j \neq 0$, $\forall j = 1, \ldots, n$. We can pick $\varepsilon > 0$ small enough for this to occur so long as we have

$$\tilde{\pi}^T A^j \neq 0, \quad \forall j \in E, \tag{3a}$$

and

$$0 < \varepsilon < \bar{\varepsilon} \equiv \min\{\varepsilon' > 0: c_j - (\hat{\pi}^k + \varepsilon'\tilde{\pi})^T A^j = 0 \text{ for some } j \in E^C\}, \tag{3b}$$

where E^C denotes the complement of E with respect to $\{1, \ldots, n\}$.

Once we have $\tilde{\pi}$ and ε satisfying (3a) and (3b), we uniquely obtain

$$x^k = \arg\min\{(c^T - (\hat{\pi}^k + \varepsilon\tilde{\pi})^T A)x: l \leq x \leq u\}. \tag{4}$$

Letting $\theta_k = \theta(\pi^k)$, $\forall k$, and noting that x^k given by (4) also evaluates $\theta(\hat{\pi}^k)$ by construction, we have that $\theta_k - \theta(\hat{\pi}^k) = (g^k)^T(\pi^k - \hat{\pi}^k) = \varepsilon(g^k)^T\tilde{\pi}$, where $g^k \equiv b - Ax^k$ is a subgradient at π^k (or also at $\hat{\pi}^k$). Hence, we will obtain an improvement $\theta_k > \theta_{k-1}$ in the objective function whenever $\theta(\hat{\pi}^k) - \theta_{k-1} > \theta(\hat{\pi}^k) - \theta_k = -\varepsilon(g^k)^T\tilde{\pi}$. Thus, in case $(g^k)^T\tilde{\pi} < 0$, and given that $\theta(\hat{\pi}^k) > \theta_{k-1}$, we could further restrict ε to satisfy

$$\varepsilon < \frac{\theta_{k-1} - \theta(\hat{\pi}^k)}{(g^k)^T\tilde{\pi}}.$$

Consequently, given $\tilde{\pi}$ satisfying (3a), and computing $\bar{\varepsilon}$ via (3b), we select ε according to

$$\varepsilon = \begin{cases} t\min\{\bar{\varepsilon}, [\theta_{k-1} - \theta(\hat{\pi}^k)]/(g^k)^T\tilde{\pi}\} \\ \qquad \text{if } (g^k)^T\tilde{\pi} < 0 \text{ and } \theta(\hat{\pi}^k) > \theta_{k-1}, \\ t\bar{\varepsilon} \quad \text{otherwise,} \end{cases} \tag{5}$$

where $0 < t < 1$ (we used $t = 0.5$).

Toward designing such a perturbation scheme, we commence with $\tilde{\pi} \equiv \hat{\pi}^k$. If (3a) holds true for $\tilde{\pi} = \hat{\pi}^k$, we pick ε according to (5), so that we then have a new differentiable point of the form $\pi^k = \hat{\pi}^k(1 + \varepsilon)$, a simple extension of $\hat{\pi}^k$ itself. Failing this, we could select $\tilde{\pi} = d^{k-1}$, the previous search direction, or $\tilde{\pi} = -\hat{d}^{k-1}$, so that we would effectively test if slightly overstepping, or understepping, in the previous iteration would yield a nondifferentiable point. Otherwise, a revised perturbation vector $\tilde{\pi}_{\text{new}}$ that comes closer to satisfying (3a) is obtained as follows, yielding a finite iterative scheme for achieving (3a).

Given a current $\tilde{\pi}$, consider a perturbation of the form $\tilde{\pi}_{\text{new}} = \tilde{\pi} + \varepsilon_0 A^r$ for an arbitrary $r \in E_0 \equiv \{j \in E: \tilde{\pi}^T A^j = 0\}$, where $\varepsilon_0 > 0$. Then, we have

$$\tilde{\pi}_{\text{new}}^T A^j = \tilde{\pi}^T A^j + \varepsilon_0 (A^r)^T A^j, \quad \forall j \in E, \tag{6}$$

where $\tilde{\pi}^T A^j \neq 0$, $\forall j \in E - E_0$, and

$$\tilde{\pi}_{\text{new}}^T A^r = \varepsilon_0 \|A^r\|^2 > 0. \tag{7}$$

From (6) and (7), if we can ensure that

$$\tilde{\pi}_{\text{new}}^T A^j = \tilde{\pi}^T A^j + \varepsilon_0 (A^r)^T A^j \neq 0, \quad \forall j \in E - E_0, \tag{8}$$

then we can obtain a new index set $E_0^{\text{new}} \equiv \{j \in E: \tilde{\pi}_{\text{new}}^T A^j = 0\}$, which is a proper subset of E_0. Notice that (8) can be zero for some $j \in E - E_0$ only when $\tilde{\pi}^T A^j$ and $(A^r)^T A^j$ are of opposite sign. Hence, let $Z \equiv \{j \in E - E_0: (\tilde{\pi}^T A^j)((A^r)^T A^j) < 0\}$. If $Z = \varnothing$, $\varepsilon_0 > 0$ can be arbitrarily chosen (e.g., we can take $\varepsilon_0 = 1$). Else, in order to make the condition (8) hold true, we select

$$\varepsilon_0 = t_0 \min\{-\tilde{\pi}^T A^j/(A^r)^T A^j: j \in Z\}, \tag{9}$$

where $0 < t_0 < 1$ (we used $t_0 = 0.5$). Reiterating this process until we have $E_0^{\text{new}} = \varnothing$, we would obtain a $\tilde{\pi}$ satisfying (3a). The resulting perturbation routine is summarized below.

Perturbation Routine

Step 1. Given a nondifferentiable point $\hat{\pi}^k$, define the index set $E \equiv \{j \in \{1, \ldots, n\}: c_j - (\hat{\pi}^k)^T A^j = 0\}$. If $E = \varnothing$, return $\pi^k = \hat{\pi}^k$. Otherwise, let $\tilde{\pi} \equiv \hat{\pi}^k$ and $E_0 \equiv \{j \in E: \tilde{\pi}^T A^j = 0\}$, and proceed to Step 2.

Step 2. If $E_0 = \varnothing$, return $\pi^k = \hat{\pi}^k + \varepsilon\tilde{\pi}$, where $\varepsilon > 0$ is determined by (5). Otherwise, proceed to Step 3.

Step 3. Pick any $r \in E_0$. Let $Z \equiv \{j \in E - E_0: (\tilde{\pi}^T A^j)((A^r)^T A^j) < 0\}$. If $Z = \varnothing$, put $\varepsilon_0 = 1$. Otherwise, select ε_0 according to (9). Let $\tilde{\pi} \leftarrow \tilde{\pi} + \varepsilon_0 A^r$ and $E_0 \leftarrow E_0 - \{j \in E_0: \tilde{\pi}^T A^j \neq 0\}$. Return to Step 2.

Remark 1. As mentioned earlier, instead of letting $\tilde{\pi} \equiv \hat{\pi}^k$ in Step 1, we can initialize $\tilde{\pi} \equiv \pm\hat{d}^{k-1}$ to test if overstepping or understepping in the previous iteration would yield a nondifferentiable point. Alternatively, note that we could have applied the foregoing procedure only once at the very beginning of the overall algorithmic scheme to obtain a $\tilde{\pi}$ such that $\tilde{\pi}^T A^j \neq 0$, $\forall j = 1, \ldots, n$. Then, this same $\tilde{\pi}$ could be reused at each iteration of the perturbation routine since it would satisfy (3a). However, in practice, because $|E|$ tends to be small (often ≤ 2) and since the perturbation routine is run only for a few (≤ 100) iterations, while n can be quite large, it is computationally preferable to compute a tailored $\tilde{\pi}$ for each iteration. Also, for the selection of $r \in E_0$ in Step 3, we initially prioritize A^j, $j = 1, \ldots, n$, from the largest number of nonzero elements to the least. Then, $r \in E_0$ is chosen according to this priority in the hope that A^r will yield

$(A^r)^T A^j \neq 0$ for many $j \in E_0$, thereby potentially reducing $|E_0|$ faster (see Step 3 in the Perturbation Routine above).

Now, whenever we encounter a nondifferentiable update $\widehat{\pi}^k$, i.e., a nonempty index set E, we obtain a new iterate π^k via this perturbation routine at which a solution of the Lagrangian subproblem in (4) is uniquely determined, so the corresponding subdifferential is a singleton. This subgradient is indeed a steepest-ascent direction at π^k. Therefore, we can utilize conventional differentiable optimization strategies to obtain the next update $\widehat{\pi}^{k+1}$, as further exemplified in §4 below.

3. Barrier-Lagrangian Dual Reformulation

In this section, we propose to apply a Lagrangian dual approach to a barrier-function-based reformulation of LP, which we call the *barrier-Lagrangian reformulation* (BLR) method. For $\mu > 0$ and sufficiently small, consider the following barrier problem, based on Frisch's logarithmic barrier-function applied to the bounding constraints in (1c) (see Bazaraa et al. 2006, for example).

$$\text{BP:}\quad \text{Minimize}\ \ c^T x - \mu \sum_{j=1}^{n} [\ln(x_j - l_j) + \ln(u_j - x_j)]$$
$$\text{subject to}\ \ Ax = b. \tag{10}$$

We know from barrier-function theory that as $\mu \to 0^+$, the trajectory of optimal solutions to (10) approaches an optimum for LP.

Consider solving BP via a Lagrangian dual approach for a fixed, relatively small value of $\mu > 0$, given our intent to derive a near-optimal solution to LP. This yields the Lagrangian dual problem

$$\text{LDBP:}\quad \text{Maximize}\ \bar{\theta}(\pi), \tag{11a}$$

where

$$\bar{\theta}(\pi) \equiv b^T \pi + \min_x \left\{ (c^T - \pi^T A)x - \mu \sum_{j=1}^{n} [\ln(x_j - l_j) + \ln(u_j - x_j)] \right\}. \tag{11b}$$

Since the objective function of the Lagrangian subproblem (11b) is strictly convex, it has a unique optimum, given that a critical point exists as verified below. Differentiating and setting equal to zero to compute this critical point, we obtain

$$\bar{c}_j - \mu \left[\frac{1}{x_j - l_j} - \frac{1}{u_j - x_j} \right] = 0, \quad \forall\, j,$$

where $\bar{c}_j \equiv c_j - \pi^T A^j$, $\forall\, j$, and where A^j is the jth column of A. Simplifying, we have,

$$\bar{c}_j x_j^2 - x_j[\bar{c}_j(l_j + u_j) + 2\mu] + \bar{c}_j l_j u_j + \mu(l_j + u_j) = 0, \quad \forall\, j. \tag{12}$$

If $\bar{c}_j = 0$, then (12) yields

$$x_j = \frac{l_j + u_j}{2} \equiv \bar{x}_j, \quad \text{say.}$$

Otherwise, solving the quadratic equation (12), we get

$$x_j = \bar{x}_j \equiv \frac{\bar{c}_j(l_j + u_j) + 2\mu \pm \sqrt{\bar{c}_j^2(u_j - l_j)^2 + 4\mu^2}}{2\bar{c}_j}. \tag{13}$$

From (13), we obtain

$$\bar{x}_j - l_j = \frac{\bar{c}_j(u_j - l_j) + 2\mu \pm \sqrt{\bar{c}_j^2(u_j - l_j)^2 + 4\mu^2}}{2\bar{c}_j} \tag{14a}$$

and

$$u_j - \bar{x}_j = \frac{\bar{c}_j(u_j - l_j) - 2\mu \mp \sqrt{\bar{c}_j^2(u_j - l_j)^2 + 4\mu^2}}{2\bar{c}_j}. \tag{14b}$$

Because we must have $l_j < \bar{x}_j < u_j$, $\forall\, j$, at optimality to (11b), observe from (14) that if $\bar{c}_j > 0$, then either $\pm$ is valid for (14a) but we must use the plus case in (14b), which corresponds to the minus case in (13). Likewise, if $\bar{c}_j < 0$, then we must use the minus case in (14a), while either case is legitimate for (14b), again implying the minus case for (13). This leads to the following unique solution $x = \bar{x}$ for (11b).

$$\bar{x}_j = \frac{\bar{c}_j(l_j + u_j) + 2\mu - \sqrt{\bar{c}_j^2(u_j - l_j)^2 + 4\mu^2}}{2\bar{c}_j}, \quad \text{if } \bar{c}_j \neq 0 \tag{15a}$$

$$\bar{x}_j = \frac{l_j + u_j}{2}, \quad \text{if } \bar{c}_j = 0. \tag{15b}$$

Observe that $\bar{x}_j$ is a continuous function of $\bar{c}_j$ in that, using L'Hospital's rule, the limiting value of (15a) as $\bar{c}_j \to 0$ equals the value in (15b). Moreover, by uniqueness of the solution in (11b), $\bar{\theta}$ is a differentiable function of π, with

$$\nabla \bar{\theta}(\pi) = b - A\bar{x}, \tag{16}$$

where $\bar{x}$ is given by (15), with $\bar{c}_j \equiv c_j - \pi^T A^j$, $\forall\, j$. By viewing $\bar{x}$ as a function of π via (15a, b), or effectively, viewing $\bar{x}_j$ as a function of $\bar{c}_j$, we can also verify that $\bar{\theta}$ is continuously twice differentiable, since by applying the definition of differentiation and using L'Hospital's rule, we obtain

PROPOSITION 1.

$$\left.\frac{d\bar{x}_j(\bar{c}_j)}{d\bar{c}_j}\right|_{\bar{c}_j=0} = \frac{-(u_j - l_j)^2}{8\mu} = \lim_{\bar{c}_j \to 0} \frac{d\bar{x}_j}{d\bar{c}_j}.$$

PROOF. See the Online Supplement to this paper on the journal's website.

Nonetheless, in keeping with our motivation to maintain simplicity in implementation, we experiment with several conjugate gradient strategies to solve LDBP for a given $\mu > 0$, instead of using Newton or quasi-Newton methods that would require solving large linear systems at each iteration. Also, note that by using $\mu > 0$ small enough along with an optimal rounding scheme (see Bazaraa et al. 2005, for example), we could, in theory, obtain optimal primal and dual solutions to LP via the optimization of LDBP. However, we intend to perform only a few iterations (say, 100, as used in our computational study) of optimizing LDBP to provide a warm start to the VTVM-GPKC procedure as mentioned above.

4. Differentiable Optimization Strategies

We consider several conjugate gradient methods for solving LD via the perturbation technique and for solving LDBP. For both problems, let π^k, g^k, and d^k denote the iterate, gradient, and search direction, respectively, at iteration k. Then, commencing with some iterate π^1, we adopt the initial search direction $d^1 = g^1$. At any iteration $k \geq 2$, we derive a search direction via the conjugate gradient strategy

$$d^k = g^k + \alpha_k d^{k-1},$$

where α_k is a suitable deflection multiplier. The next iterate is then computed as

$$\pi^{k+1} = \pi^k + \lambda_k \frac{d^k}{\|d^k\|},$$

where a step-size $\lambda_k > 0$ is obtained via a line search along the direction d^k. We employ an inexact line search here via a single quadratic fit (Bazaraa et al. 2006). Note that this resulting solution is perturbed, as necessary, when using the perturbation technique.

Four choices of the deflection multiplier α_k are explored in our experiments in the next section. Specifically, based on a quadratic approximation of the objective function, Hestenes and Stiefel (1952) derived

$$\alpha_k^{HS} = -\frac{(g^k)^T q^k}{(d^{k-1})^T q^k},$$

where $q^k \equiv g^k - g^{k-1}$. Assuming exact line searches, this reduces to Polak and Ribiere's (1969) deflection parameter

$$\alpha_k^{PR} = \frac{(g^k)^T q^k}{\|g^{k-1}\|^2}.$$

Furthermore, under the assumption of quadraticity, this simplifies to Fletcher and Reeves' (1964) formula $\alpha_k^{FR} = \|g^k\|^2/\|g^{k-1}\|^2$. Based on a scaled quasi-Newton method, Sherali and Ulular (1990) alternatively prescribed the deflection parameter $\alpha_k^{SU} = -(g^k)^T(q^k + d^{k-1})/[(q^k)^T d^{k-1}]$. Finally, Sherali and Ulular (1989) proposed the conjugate (sub)gradient average direction strategy (ADS) $\alpha_k^{ADS} = \|g^k\|/\|d^{k-1}\|$, and demonstrated promising performance for solving the nondifferentiable Lagrangian dual of LP itself. In addition to these five conjugate gradient strategies, we consider the memoryless BFGS method of Shanno (1978) in our experiments, for which a search direction is prescribed as

$$d^k = -\frac{(p^k)^T q^k}{(q^k)^T q^k} g^k - \left(2\frac{(p^k)^T g^k}{(p^k)^T q^k} - \frac{(q^k)^T g^k}{(q^k)^T q^k}\right) p^k + \frac{(p^k)^T g^k}{(q^k)^T q^k} q^k,$$

where $p^k \equiv \pi^k - \pi^{k-1}$. Note that, for the perturbation technique, if $q^k = 0$ (or practically $\|q^k\| < \varepsilon$ for some tolerance $\varepsilon > 0$), we use the gradient as the next search direction, i.e., $d^k = g^k$. Also, we implement the restarting criteria of Powell (1977) for all the foregoing conjugate gradient direction and memoryless quasi-Newton strategies.

5. Overall Algorithmic Procedure

There are two sequential phases. In the first phase, the foregoing schemes that circumvent or obviate the nondifferentiability of θ are utilized for deriving a relatively good starting solution. Then, in the second phase, we switch to a theoretically convergent NDO algorithm, which is a composition of a modified variable target value method (VTVM) of Sherali et al. (2000) and a generalized Polyak-Kelley cutting-plane strategy (GPKC) as proposed by Lim and Sherali (2006). In the computational study of Lim and Sherali, this combination (VTVM-GPKC) revealed the most promising performance when extensively tested on a variety of problems.

At the end of the first phase, after some K iterations, say, we estimate the optimal objective value for LD using a quadratic approximation for the original Lagrangian dual function θ in (2). This estimate is used as the initial target value w_1 for the second phase. Suppose that $\hat{\theta}$ is a quadratic approximation for θ, given by

$$\hat{\theta}(\pi) = h_0 + h^T \pi + \tfrac{1}{2}\pi^T H \pi, \tag{17}$$

where H is a negative definite diagonal matrix. Maximizing $\hat{\theta}$ gives an optimal solution and the optimal objective value as

$$\hat{\pi} = -H^{-1}h \quad \text{and} \quad \hat{w} = \hat{\theta}(\hat{\pi}) = h_0 - \tfrac{1}{2}h^T H^{-1} h, \tag{18}$$

respectively. Now, suppose that we evaluate $\theta(\pi^i)$ and $g^i \in \partial\theta(\pi^i)$ for $i = 1$ and K, where $\partial\theta(\pi)$ is the subdifferential of θ at π. Using (17), we equate

$$g^K - g^1 = \nabla\hat{\theta}(\pi^K) - \nabla\hat{\theta}(\pi^1) = H(\pi^K - \pi^1), \qquad (19)$$

to estimate the assumed diagonal Hessian H. To maintain strict concavity of $\hat{\theta}$, if any diagonal element of H is nonnegative or indeterminable from (19), we take it as -1. This gives us the desired estimate for the diagonal negative definite matrix H. Furthermore, we estimate h_0 and h by equating θ_K and g^K to $\hat{\theta}(\pi^K)$ and $\nabla\hat{\theta}(\pi^K)$, respectively, according to

$$\begin{aligned} \hat{\theta}(\pi^K) = h_0 + h^T\pi^K + \tfrac{1}{2}(\pi^K)^T H\pi^K = \theta_K \quad \text{and} \\ \nabla\hat{\theta}(\pi^K) = h + H\pi^K = g^K. \end{aligned} \qquad (20)$$

Substituting h_0, h, and H as obtained from (19) and (20) into (18), we compute an initial target value $w_1 \equiv \widehat{w}$ for the VTVM-GPKC phase. Note from (20) that since $\nabla\hat{\theta}(\pi^K) = g^K \neq 0$ while $\nabla\hat{\theta}(\widehat{\pi}) = 0$ by definition, we have that $\widehat{w} = \hat{\theta}(\widehat{\pi}) > \hat{\theta}(\pi^K) = \theta_K$, i.e., $\widehat{w} > \theta_K$ as desired.

We also experimented with an initial target value given by $w_1 \equiv \theta_K + p(\theta_K - \theta_1)$, where $p > 0$, based on a simple further projected estimated improvement of $p \times 100\%$ in the objective value.

Denote these two foregoing initial-target-value-determination methods by QUAD and PROJ, respectively. For the sake of completeness, we describe the overall two-phase algorithmic procedure below, including the options of the perturbation technique (PT) or the barrier-Lagrangian reformulation (BLR) in Phase I, and the VTVM-GPKC method of Lim and Sherali (2006) used in Phase II.

Phase I (PT Method)

Step 1. Select an initial iterate π^1 at which θ is differentiable (use the perturbation routine in §2 if necessary). Pick a termination tolerance $\varepsilon_{\text{tol}} = 10^{-6}$ for the norm of the (sub)gradient, and a maximum number of iterations K (=100, say). Compute x^1 and g^1. Set the iteration counter $k = 1$ and the restart indicator RESET $= 0$.

Step 2. If $\|g^k\| \leq \varepsilon_{\text{tol}}$ or $k = K$, then proceed to Phase II with the initial solution $\pi^1 \equiv \pi^k$ and the initial target value w_1 as obtained above via the QUAD or PROJ methods.

Step 3. If RESET $= 0$, put $d^k = g^k$. Otherwise, compute a search direction based on the employed (deflected gradient) direction-finding strategy.

Step 4. If $k \geq 2$, RESET ≥ 1, and any of the following two restarting criteria is violated (see Bazaraa et al. 2006):

(i) $\quad |(g^k)^T g^{k-1}| \leq 0.2\|g^k\|^2$

(ii) $\quad 0.8\|g^k\|^2 \leq (d^k)^T g^k \leq 1.2\|g^k\|^2$,

then put RESET $= 04$ and return to Step 3. Else, perform a single quadratic-fit line search along d^k to find a new iterate π^{k+1}. If θ is nondifferentiable at π^{k+1}, then apply the perturbation routine to revise π^{k+1}. Increment $k \leftarrow k+1$ and RESET $\leftarrow$ RESET $+ 1$. If RESET $= n$, put RESET $= 0$.

Step 5. Compute x^k and g^k, and return to Step 2.

Phase I (BLR Method)

Step 1. Select an initial iterate π^1, a termination tolerance $\varepsilon_{\text{tol}} = 10^{-6}$ for the norm of the gradient, a barrier parameter $\mu > 0$, and a maximum number of iterations K (=100, say). Compute $\bar{x}^1$ and $\nabla\bar{\theta}(\pi^1)$ according to (15) and (16), respectively. Set the iteration counter $k = 1$ and the restart indicator RESET $= 0$.

Step 2. If $\|\nabla\bar{\theta}(\pi^k)\| \leq \varepsilon_{\text{tol}}$ or $k = K$, then proceed to Phase II with the initial solution $\pi^1 \equiv \pi^k$ and the initial target value w_1 as obtained above via the QUAD or PROJ methods.

Step 3. If RESET $= 0$, put $d^k = \nabla\bar{\theta}(\pi^k)$. Otherwise, compute a search direction based on the employed (deflected gradient) direction-finding strategy.

Step 4. If $k \geq 2$, RESET ≥ 1, and any of the following two restarting criteria is violated (see Bazaraa et al. 2006):

(i) $\quad |(\nabla\bar{\theta}(\pi^k))^T \nabla\bar{\theta}(\pi^{k-1})| \leq 0.2\|\nabla\bar{\theta}(\pi^k)\|^2$

(ii) $\quad 0.8\|\nabla\bar{\theta}(\pi^k)\|^2 \leq (d^k)^T \nabla\bar{\theta}(\pi^k) \leq 1.2\|\nabla\bar{\theta}(\pi^k)\|^2$,

then put RESET $= 0$ and return to Step 3. Else, perform a single quadratic-fit line search along d^k to find a new iterate π^{k+1}. Increment $k \leftarrow k+1$ and RESET $\leftarrow$ RESET $+ 1$. If RESET $= n$, put RESET $= 0$.

Step 5. Compute $\bar{x}^k$ and $\nabla\bar{\theta}(\pi^k)$ via (15) and (16), and return to Step 2.

Phase II (VTVM-GPKC Method)

Initialization Step. Select $\varepsilon_{\text{tol}} = 10^{-6}$, $\tilde{\varepsilon} > 0$, $k_{\max}$, $\beta \in (0, 1]$, $\sigma \in (0, 1/3]$, r, $\bar{r}$, $\bar{\tau}$, $\bar{\gamma}$, and $\eta \in (0, 1]$. (Recommended values for these parameters are $\tilde{\varepsilon} = 0.1$, $k_{\max} = 1{,}000$, $\beta = 0.8$, $\sigma = 0.15$, $r = 0.1$, $\bar{r} = 1.1$, $\bar{\tau} = 75$, $\bar{\gamma} = 20$, and $\eta = 0.75$.) Determine $\theta_1 \equiv \theta(\pi^1)$ and g^1. Set the incumbent vectors $\bar{\pi} = \pi^1$, $\bar{g} = g^1$, and the incumbent objective value $z_1 = \theta_1$. If $\|g^1\| \leq \varepsilon_{\text{tol}}$, terminate the algorithm. Initialize $l = 1$, $k = 1$, $\tau = 0$, $\gamma = 0$, $\Delta = 0$, and the reset indicator RESET $= 1$. Set $\varepsilon_1 = \sigma(w_1 - \theta_1)$.

Step 1. Call Subroutine GPKC (described below) to obtain the next iterate π^{k+1}. Increment $\tau \leftarrow \tau + 1$ and $k \leftarrow k+1$. Compute $\theta_k \equiv \theta(\pi^k)$ and g^k. Put RESET $= 0$.

Step 2. If $\theta_k > z_{k-1}$, set $\Delta \leftarrow \Delta + \theta_k - z_{k-1}$, $z_k = \theta_k$, $\bar{\pi} = \pi^k$, $\bar{g} = g^k$, $\gamma = 0$, and proceed to Step 3. Otherwise, put $z_k = z_{k-1}$, increment $\gamma \leftarrow \gamma + 1$, and go to Step 4.

Step 3. If $k > k_{\max}$ or $\|g^k\| \leq \varepsilon_{\text{tol}}$, terminate the algorithm. If $z_k \geq w_l - \varepsilon_l$, go to Step 5. If $\tau \geq \bar{\tau}$, go to Step 6. Else, return to Step 1.

Step 4. If $k > k_{\max}$, terminate the algorithm. If $\gamma \geq \bar{\gamma}$ or $\tau \geq \bar{\tau}$, go to Step 6. Else, return to Step 1.

Step 5. Compute $w_{l+1} = z_k + \max\{\varepsilon_l + \eta\Delta, r|z_k|\}$, and $\varepsilon_{l+1} = \max\{(w_{l+1} - z_k)\sigma, \tilde{\varepsilon}\}$. If $k \leq 500$, update $\eta \leftarrow 2\eta$; otherwise, set $\eta = 0.75$. If $r|z_k| > \varepsilon_l + \eta\Delta$, update $r \leftarrow r/\bar{r}$. Put $\tau = 0$, $\Delta = 0$, and increment $l \leftarrow l+1$. Return to Step 1.

Step 6. Compute $w_{l+1} = (z_k + \varepsilon_l + w_l)/2$, and $\varepsilon_{l+1} = \max\{(w_{l+1} - z_k)\sigma, \tilde{\varepsilon}\}$. If $k \leq 500$, update $\eta \leftarrow \max\{\eta/2, 0.75\}$; otherwise, set $\eta = 0.75$. If $\gamma \geq \bar{\gamma}$, then set $\bar{\gamma} \leftarrow \min\{\bar{\gamma} + 10, 50\}$. If $w_l - w_{l+1} \leq 0.1$, then set $\beta \leftarrow \max\{\beta/2, 10^{-6}\}$. Put $\gamma = 0$, $\tau = 0$, $\Delta = 0$, and $l \leftarrow l+1$. Reset $\pi^k = \bar{\pi}$, $\theta_k = z_k$, $g^k = \bar{g}$, and put RESET = 1. Return to Step 1.

Subroutine GPKC

Step i. Let $\delta = k - \bar{k}$, where $\bar{k}$ is the most recent iteration at which the variable target value algorithm was reset (i.e. RESET = 1), and accordingly, let $\delta'' = \min\{\delta, 4\}$. Compute $\hat{\theta}^* = \theta_k + \beta(w_l - \theta_k)$.

Step ii. Set $\psi_1 = (\hat{\theta}^* - \theta_k)/\|g^k\|^2$, $\psi_2 = 0$, and $\pi^{k+1} = \pi^k + \psi_1 g^k$. If RESET = 1, put $\bar{k} = k$ and exit the subroutine.

Step iii. Compute $\xi_1 = \hat{\theta}^* - \theta_k + (\pi^k)^T g^k$ and $\xi_2 = \hat{\theta}^* - \theta_{k-1} + (\pi^{k-1})^T g^{k-1}$. If $\xi_2 \leq (\pi^{k+1})^T g^{k-1}$, go to Step vi.

Step iv. Put $\psi_2 = (\xi_2 - (\pi^k)^T g^{k-1})/\|g^{k-1}\|^2$, and let $\pi^{k+1} = \pi^k + \psi_2 g^{k-1}$. If $\psi_2 > 0$ and $(\pi^{k+1})^T g^k \geq \xi_1$, go to Step vi.

Step v. Compute $\psi_0 = \|g^k\|^2\|g^{k-1}\|^2 - ((g^{k-1})^T g^k)^2$. If $\psi_0 < 10^{-6}$, put $\pi^{k+1} = \pi^k + \psi_1 g^k$ and exit the subroutine. Otherwise, compute

$$\psi_1 = [\|g^{k-1}\|^2(\xi_1 - (\pi^k)^T g^k) - ((g^{k-1})^T g^k) \cdot (\xi_2 - (\pi^k)^T g^{k-1})]/\psi_0, \quad \text{and}$$

$$\psi_2 = [\|g^k\|^2(\xi_2 - (\pi^k)^T g^{k-1}) - ((g^{k-1})^T g^k) \cdot (\xi_1 - (\pi^k)^T g^k)]/\psi_0.$$

Put $\pi^{k+1} = \pi^k + \psi_1 g^k + \psi_2 g^{k-1}$ and proceed to Step vi.

Step vi. If $\delta'' = 1$, exit the subroutine. Otherwise, put $i = 2$ and proceed to Step vii.

Step vii. Compute $\xi_3 = \hat{\theta}^* - \theta_{k-i} + (\pi^{k-i})^T g^{k-i}$ and let $\psi_3 = \xi_3 - (\pi^{k+1})^T g^{k-i}$. If $\psi_3 \leq 0$, go to Step ix. Otherwise, proceed to Step viii.

Step viii. Put $\bar{\pi}^{k,i} = \pi^{k+1} + (\psi_3/\|g^{k-i}\|^2)g^{k-i}$. If $(\bar{\pi}^{k,i})^T g^k \geq \xi_1$ and $(\bar{\pi}^{k,i})^T g^{k-1} \geq \xi_2$, put $\pi^{k+1} = \bar{\pi}^{k,i}$.

Step ix. If $i = \delta''$, exit the subroutine. Otherwise, increment $i \leftarrow i+1$ and return to Step vii.

Remark 2. Often, in the context of solving LP, a (near-) optimal primal solution is also desirable besides obtaining an optimal dual solution along with the optimal objective value. Commencing with $\bar{x}^K$ as obtained after performing some K iterations using the BLR option in Phase I, we could employ an ergodic-primal-recovery strategy while executing the VTVM-GPKC algorithm in Phase II to derive such a primal solution for LP. (See Shor 1985, Sherali and Choi 1996, and Larsson et al. 1999, for several ergodic-primal-recovery strategies.) We could also adopt a similar primal-recovery scheme in concert with the PT option in Phase I. However, unlike the BLR approach, because the PT option lacks any theoretical convergence properties, we confine our attention to the BLR option for this particular investigation. If a more accurate primal solution is desired at the end of this overall two-phase process, the current primal solution can be used as a warm start for an interior-point or simplex-based algorithm (after applying a purification scheme to obtain a vertex solution in the latter case as described in Bazaraa et al. 2005, for example). In the next section, we provide some computational results pertaining to such a primal-recovery process in combination with the proposed two-phase dual optimization scheme that employs the BLR option in Phase I.

6. Computational Results

The proposed algorithmic procedures were coded in C++ and implemented for solving two types of test problems: general linear programs and transportation problems. Our choice of the latter class was motivated by the fact that, as shown by Shor (1985), these problems are particularly challenging for nondifferentiable optimization methods. As described in Lim (2004), 20 general linear-programming problems, LP1–LP20, formulated as LP in (1) using $l = 0$ and $u = e$, and 20 transportation problems, TR1–TR20, were generated, having a variety of sizes and degrees of degeneracy. We did not exploit the structure of the constraints in the transportation problem when formulating the corresponding Lagrangian dual problem, i.e., we simply dualized both the demand and supply balance constraints in order to test the proposed procedures. Particular specifications for these test problems are summarized in Table 1, where the two percentage values given in parentheses respectively indicate the degrees of primal and dual degeneracy induced at optimality. We also display the CPU times consumed by the commercial linear-programming solver, CPLEX 8.1, to solve this test bed of problems using the dual simplex option. (This option was used to provide a more suitable comparison against our dual-based procedure, as also when both methods were terminated with the same near-optimality tolerance as described in the sequel.) All runs were made on a PentiumIII/667 MHz computer and $\pi = 0$ was used as the initial solution for each algorithm and problem combination. Also, based on our preliminary tests, we used $p = 0.6$ for the target-value-initialization method PROJ. To compare the quality of solutions produced by the different methods, we report a *percentage optimality ratio* (POR) that measures the relative optimality gap of the final solution produced

Table 1 Summary of Test Problems

Problem type (% primal, dual degeneracy)	Prob	Row/column	CPLEX solution: Optimal value	CPLEX solution: CPU time (sec)
Linear program (5%, 5%)	LP1	500/1,000	−1,703.83	44.89
	LP2	1,000/3,000	−4,724.01	1,226.86
	LP3	1,000/5,000	−1,222.18	1,479.31
	LP4	2,000/3,000	3,078.64	9,466.68
	LP5	2,000/5,000	3,105.01	12,986.7
Linear program (5%, 25%)	LP6	500/1,000	−623.999	48.75
	LP7	1,000/3,000	−2,748.89	1,154.44
	LP8	1,000/5,000	−5,306.73	1,708.88
	LP9	2,000/3,000	−4,899.56	10,375.8
	LP10	2,000/5,000	−2,492.58	23,079.6
Linear program (25%, 5%)	LP11	500/1,000	−1,672.58	43.78
	LP12	1,000/3,000	−271.81	966.13
	LP13	1,000/5,000	1,375.16	1,343.07
	LP14	2,000/3,000	−2,107.96	9,015.16
	LP15	2,000/5,000	−3,456.14	14,872.2
Linear program (25%, 25%)	LP16	500/1,000	−338.54	45.62
	LP17	1,000/3,000	1,099.29	1,173.57
	LP18	1,000/5,000	−11,369.5	1,553.11
	LP19	2,000/3,000	1,121.03	6,378.62
	LP20	2,000/5,000	−5,871.56	43,708.8
Transportation (5%, 5%)	TR1	800/160,000	2,215.08	128.56
	TR2	1,000/250,000	1,499.65	272.27
	TR3	1,200/360,000	−895.093	567.47
	TR4	1,400/490,000	305.108	881.82
	TR5	1,600/640,000	−204.755	1,335.25
Transportation (5%, 25%)	TR6	800/160,000	681.531	125.04
	TR7	1,000/250,000	−571.108	221.71
	TR8	1,200/360,000	−1,779.95	547.84
	TR9	1,400/490,000	−1,278.88	627.79
	TR10	1,600/640,000	1,614.92	1,126.82
Transportation (25%, 5%)	TR11	800/160,000	−601.262	79.58
	TR12	1,000/250,000	43.413	168.35
	TR13	1,200/360,000	−2,308.51	349.38
	TR14	1,400/490,000	1,440.23	573.34
	TR15	1,600/640,000	−1,163.07	986.80
Transportation (25%, 25%)	TR16	800/160,000	553.837	85.84
	TR17	1,000/250,000	−1,361.69	221.17
	TR18	1,200/360,000	354.879	296.13
	TR19	1,400/490,000	−2,139	402.69
	TR20	1,600/640,000	2,144.66	600.79

by each employed algorithm, defined as {[optimal objective value − best objective value produced by the employed algorithm] ÷ [optimal objective value − initial objective value]} × 100%.

Table 2 presents the average POR values obtained after executing Phase I for the PT and BLR procedures in conjunction with the six deflection strategies described in §4, using an iteration limit of $K = 100$ (see Lim 2004 for detailed results). For both PT and BLR, the rank-order of the overall performance was SU, HS, FR, ADS, PR, and BFGS from best to worst. The PT and BLR techniques respectively yielded overall average POR values of 0.74% and 0.73% when implemented in concert with the SU strategy. Moreover, the BLR technique revealed the best respective POR values of 1.17% and 0.29% when used in conjunction with SU for both the classes of LP and TR problems. Also, SU displayed the best POR value of 1.15% for LP problems when implemented with PT. However, with the PT procedure, the ADS strategy yielded the least POR value of 0.29% for TR problems, while SU was second best, producing a 0.03% greater POR value than ADS. Considering all the test cases together, the BLR technique slightly, but consistently, outperformed PT in terms of the overall average POR values.

Table 3 displays the average POR values produced by the implemented procedures at the end of Phase II for the two target-value-initialization methods, denoted by QUAD and PROJ as discussed in §5, as well as by using the VTVM-GPKC procedure by itself (likewise initialized at $\pi = 0$), for each class of test problems. Note that the Phase II runs of VTVM-GPKC were executed with an iteration limit $k_{\max} = 1{,}000$ (following the $K = 100$ iterations of Phase I), while for the stand-alone VTVM-GPKC runs, we used a larger iteration limit $k_{\max} = 2{,}000$. Despite this, the proposed two-phase procedures always revealed smaller overall average POR values. Among the implemented procedures, the PT-BFGS-PROJ combination for Phase I yielded the best overall average POR value of 0.028% at the end of Phase II, which is 12.5% of that produced by the stand-alone VTVM-GPKC method. The combination BLR-BFGS was the second best, with an overall average POR value of 0.03%.

Although SU was the best strategy for the Phase I runs, it consistently revealed the worst overall performance over the two-phase runs. On the other hand, although BFGS and PR displayed relatively worse performances in Phase I, they yielded the best and the second-best overall POR values among all combinations of Phase I techniques and target-value-initialization methods over the two-phase runs. This

Table 2 Average POR Values After Phase I (%)

	Perturbation technique (PT)						Barrier-Lagrangian dual reformulation (BLR)					
Prob	HS	PR	FR	SU	ADS	BFGS	HS	PR	FR	SU	ADS	BFGS
LP	1.19	2.65	1.85	1.15	2.62	4.10	1.19	2.55	1.79	1.17	2.50	4.11
TR	0.35	0.36	0.38	0.32	0.29	0.61	0.32	0.35	0.39	0.29	0.29	0.60
Overall	0.77	1.50	1.12	0.74	1.45	2.36	0.76	1.45	1.09	0.73	1.40	2.35

Table 3 Average POR Values After Phase II (%)

Average for	Phase I	Target value initialization method	Strategy in Phase I: HS	PR	FR	SU	ADS	BFGS	Stand-alone VTVM-GPKC
LP	PT	QUAD	0.100	0.066	0.075	0.131	0.070	0.059	0.155
		PROJ	0.100	0.063	0.075	0.128	0.065	0.053	
	BLR	QUAD	0.110	0.067	0.076	0.114	0.074	0.055	
		PROJ	0.110	0.068	0.068	0.131	0.070	0.056	
TR	PT	QUAD	0.010	0.006	0.005	0.005	0.008	0.005	0.048
		PROJ	0.010	0.006	0.005	0.006	0.006	0.002	
	BLR	QUAD	0.010	0.005	0.005	0.007	0.008	0.004	
		PROJ	0.010	0.005	0.004	0.006	0.008	0.004	
Overall	PT	QUAD	0.050	0.036	0.040	0.068	0.039	0.032	0.102
		PROJ	0.060	0.035	0.040	0.067	0.035	0.028	
	BLR	QUAD	0.060	0.036	0.040	0.060	0.041	0.030	
		PROJ	0.060	0.036	0.036	0.069	0.039	0.030	

can be explained by the computational characteristic of the Phase II procedure in response to the quality and nondifferentiability of the warm-start solution it receives from Phase I. One might generally presume that the performance of VTVM-GPKC can be improved by providing a better initial solution and initial target value via Phase I, but if the initial solution is close to nondifferentiability, the Phase II procedure experiences difficulties in attaining ascents and in properly adjusting the target values. To verify this, define a *degree of nondifferentiability* (R) at π^K as

$$R \equiv \frac{\sum_{j=1}^{n} |c_j - (\pi^K)^T A^j|}{n}.$$

Recall that if $|c_j - (\pi^K)^T A^j| = 0$ (or is small) for any $j = 1, \ldots, n$, θ is (near-) nondifferentiable at π^K. The average R values computed for each combination of strategy and problem type are in Table 4. The overall average R when SU was employed are relatively small (5.10 and 5.08 for PT and BLR, respectively), as compared with the R values for the other strategies, particularly for the more competitive methods PR and BFGS. Therefore, we conjecture that the performance of Phase II was affected by the degree of nondifferentiability as well as by the quality of the initial solution.

Next, to compare CPU times meaningfully, we measured run times until the implemented procedures (including CPLEX) produced objective values, called *near-optimal values*, corresponding to a POR value of 1% or less. (All procedures attained near-optimal values within the prescribed number of iterations.) The resulting average RCP (*relative CPU time percentage*) values, defined as {[CPU time of the implemented procedure to attain a near-optimal value] ÷ [CPU time of CPLEX 8.1 to attain a near-optimal value using the dual simplex option]} × 100%, are in Table 5. The composition of Phase I via PT-ADS and Phase II initialized by the QUAD method yielded the best overall average RCP value of 3.19%. This combination actually attained near-optimal values by the end of Phase I itself for 26 instances (6 out of 20 for LP and 20 out of 20 for TR). This procedure also yielded the smallest average RCP value (2.85%) for the class of TR problems. Furthermore, note that we attained near-optimal values for all TR instances during Phase I itself via PT. (This is the reason that, for the class of TR problems, the average RCP values for QUAD and PROJ in concert with PT were the same.) The best average RCP value of 3.13% for LP was obtained with PT-FR-QUAD.

Table 4 Average Degrees of Nondifferentiability (R) at π^K

Average for	Phase I	Strategy in Phase I: HS	PR	FR	SU	ADS	BFGS
LP	PT	5.91	8.95	7.34	6.16	8.80	9.93
	BLR	5.84	8.76	7.13	6.21	8.58	9.93
TR	PT	4.13	4.13	4.20	4.04	3.98	4.77
	BLR	4.10	4.10	4.24	3.95	3.98	4.75
Overall	PT	5.02	6.54	5.77	5.10	6.39	7.35
	BLR	4.97	6.43	5.68	5.08	6.28	7.34

The stand-alone VTVM-GPKC procedure yielded no advantage over CPLEX 8.1 when solving our test bed of transportation problems, as evident from the average RCP value of 132.34%. Hence, the proposed two-phase procedures can effectively provide near-optimal values with a significant reduction in effort.

We conclude that the PT-BFGS-PROJ combination yielded the best POR value (0.028%), while the composition of PT-ADS-QUAD produced the best RCP value (3.19%) among the implemented procedures. For benchmarking, we also compared the POR and the RCP values obtained for the PT-BFGS-PROJ combination with those produced by the heuristic subgradient method of Held et al. (1974), which is popular. In our runs for the Held et al. procedure, we experimented with two upper bounds on θ^*, namely, $\theta^* +$

Table 5 Average RCP Values (%)

Average for	Phase I	Target value initialization method	Strategy in Phase I HS	PR	FR	SU	ADS	BFGS	Stand-alone VTVM-GPKC
LP	PT	QUAD	3.28	3.48	3.13	3.14	3.53	4.32	5.54
		PROJ	3.89	4.13	3.71	3.53	3.95	4.72	
	BLR	QUAD	3.91	3.65	3.79	3.39	3.85	4.55	
		PROJ	4.43	4.26	4.12	3.94	4.43	5.13	
TR	PT	QUAD	3.46	3.58	3.83	4.05	2.85	6.40	132.34
		PROJ	3.46	3.58	3.83	4.05	2.85	6.40	
	BLR	QUAD	10.66	10.36	13.42	10.93	7.85	13.03	
		PROJ	10.90	10.36	13.42	10.93	7.85	13.03	
Overall	PT	QUAD	3.37	3.53	3.48	3.60	3.19	5.36	68.94
		PROJ	3.68	3.86	3.77	3.79	3.40	5.56	
	BLR	QUAD	7.29	7.01	8.60	7.16	5.85	8.79	
		PROJ	7.67	7.32	8.79	7.45	6.27	9.56	

$(\theta^* - \theta_1) \times 10\%$, and $\theta^* + (\theta^* - \theta_1) \times 20\%$, which are denoted by HWC10 and HWC20, respectively. Following a typical strategy, the step-length parameter for HWC10 and HWC20 was initialized as 2, and was halved at iterations 800, 1,200, 1,400, 1,500, and 1,550. Then, it was halved for every 25 iterations until $k_{\max} = 2{,}000$. Table 6 displays the average POR and RCP values for these algorithms along with the PT-BFGS-PROJ composition that yielded the best solution quality over the two-phase runs. Note that both HWC10 and HWC20 could not attain near-optimal values for most TR problems (18 and 20 instances, respectively) and for four LP instances (LP4, 9, 14, and 19). To penalize these cases, we ascribed a corresponding RCP value of 100%. Despite assuming prior knowledge of a tight upper bound based on the optimal value, HWC10 yielded a 11,436% larger overall average POR value than the PT-BFGS-PROJ combination, while its RCP value was 1,167% larger than that of PT-BFGS-PROJ. HWC20 displayed far worse performance. Thus, it appears that the two-phase procedure PT-BFGS-PROJ offers a much more effective methodology than the popular Held et al. scheme.

Finally, as commented in Remark 2 of §3, we experimented with the primal recovery schemes of Shor (1985) and Larsson et al. (1999) for the pure subgradient strategy and Sherali and Choi (1996) for the deflected subgradient strategy. (Denote these methods by SHOR, LPS, and SC, respectively.) For the sake of simplicity, we used optimal values instead of target values when computing step lengths for SHOR and SC. Moreover, the parameter values for LPS were set as in their numerical experiments, and the deflection parameter for SC was set to 1. Each procedure was run for 2,000 iterations with and without the BLR phase, to see the effect of providing a good starting point for the tested ergodic-primal-recovery schemes. Table 7 displays the average *degree of primal infeasibility* (ρ) attained at the final resulting point $\tilde{x}$ (which effectively measures the near-optimality of this solution), defined as $\rho \equiv \sum_{i=1}^{m} |A_i\tilde{x} - b_i|/m$, where A_i is the ith row of A. For the class of LP problems, SC in conjunction with BLR yielded the smallest average ρ value of 0.47, while SHOR revealed the best average value of 0.83 for the TR problems. Overall, SHOR, implemented along with the BLR phase, produced the smallest average ρ value of 0.68. Note that LPS, SHOR, and SC in conjunction with the BLR phase yielded overall average ρ values of 2.18, 0.68, and 0.90, respectively, which are only 20.7%, 20.5%, and 5.8% of the corresponding values produced by the same methods when not employing the BLR phase. Hence, we conclude that the BLR phase also significantly facilitates more rapid convergence of ergodic-primal-recovery schemes.

In conclusion, the proposed techniques can be highly useful in the context of Lagrangian relaxation

Table 6 Benchmarking with HWC

	Procedure	LP	TR	Overall
POR values	PT-BFGS-PROJ	0.053	0.002	0.028
	HWC10	4.575	1.784	3.179
	HWC20	9.234	3.016	6.125
RCP values	PT-BFGS-PROJ	4.718	6.402	5.560
	HWC10	30.297	99.465	64.881
	HWC20	33.694	100	66.847

Table 7 Average Degree of Primal Infeasibility

Recovery scheme	BLR	LP	TR	Overall
LPS	Without BLR	12.43	8.61	10.52
	With BLR	0.77	3.58	2.18
SHOR	Without BLR	4.30	2.34	3.32
	With BLR	0.53	0.83	0.68
SC	Without BLR	2.75	28.40	15.58
	With BLR	0.47	1.33	0.90

approaches. As such, it might be potentially beneficial to employ them in LP-based branch-and-cut algorithms for discrete problems by using suitable Lagrangian dual formulations for deriving bounds and generating cuts, in lieu of solving the underlying LPs via simplex or interior-point algorithms. Furthermore, VTVM-GPKC performs well when provided with a good initial point, so might be suitable for re-optimization in branch-and-cut methods or sensitivity analyses. We propose these investigations for future research.

Acknowledgments

This research has been supported by the National Science Foundation under Grant DMI-0094462.

References

Adams, W. P., H. D. Sherali. 1993. Mixed-integer bilinear-programming problems. *Math. Programming* **59** 279–305.

Barahona, F., R. Anbil. 2000. The volume algorithm: Producing primal solutions with a subgradient method. *Math. Programming* **87** 385–399.

Bazaraa, M. S., J. J. Jarvis, H. D. Sherali. 2005. *Linear Programming and Network Flows*, 3rd ed. Wiley, New York.

Bazaraa, M. S., H. D. Sherali, C. M. Shetty. 2006. *Nonlinear Programming: Theory and Algorithms*, 3rd ed. Wiley, New York.

Camerini, P. M., L. Fratta, F. Maffioli. 1975. On improving relaxation methods by modified gradient techniques. *Math. Programming Stud.* **3** 26–34.

Fisher, M. L. 1981. The Lagrangian relaxation method for solving integer programming problems. *Management Sci.* **27** 1–18.

Fletcher, R., C. M. Reeves. 1964. Function minimization by conjugate gradients. *Comput. J.* **7** 149–154.

Held, M., P. Wolfe, H. Crowder. 1974. Validation of subgradient optimization. *Math. Programming* **6** 62–88.

Hestenes, M. R., E. Stiefel. 1952. Methods of conjugate gradients for solving linear systems. *J. Res. National Bureau Standards, Section B* **48** 409–436.

Larsson, T., M. Patriksson, A. B. Stromberg. 1999. Ergodic, primal convergence in dual subgradient schemes for convex programming. *Math. Programming* **86** 283–312.

Lemarechal, C. 1977. Bundle methods in nonsmooth optimization. C. Lemarechal, R. Mifflin, eds. *Proc. IIASA Workshop, Laxenburg, Austria*, Vol 3, 79–109.

Lim, C. 2004. Nondifferentiable optimization of Lagrangian dual formulations for linear programs with recovery of primal solutions. Ph.D. dissertation, Grado Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA.

Lim, C., H. D. Sherali. 2006. Convergence and computational analyses for some variable target value and subgradient deflection methods. *Comput. Optim. Appl.* **34** 409–428.

Makela, M. M. 2002. Survey of bundle methods for nonsmooth optimization. *Optim. Methods Software* **17** 1–29.

Mifflin, R. 1977. An algorithm for constrained optimization with semismooth functions. *Math. Oper. Res.* **2** 191–207.

Nesterov, Y. 2005. Smooth minimization of non-smooth functions. *Math. Programming* **103** 127–152.

Polak, E., G. Ribiere. 1969. Note on the convergence of methods of conjugate directions. *Rev. Française d'Informatique Recherche Operationelle* **3** 35–43.

Polyak, B. T. 1967. A general method of solving extremum problems. *Soviet Math.* **8** 593–597.

Polyak, B. T. 1969. Minimization of unsmooth functionals. *USSR Comput. Math. Math. Phys.* **9** 14–29.

Powell, M. J. D. 1977. Restart procedures for the conjugate gradient method. *Math. Programming* **12** 241–254.

Shanno, D. F. 1978. Conjugate gradient methods with inexact searches. *Math. Oper. Res.* **3** 244–256.

Sherali, H. D., G. Choi. 1996. Recovery of primal solutions when using subgradient optimization methods to solve Lagrangian duals of linear programs. *Oper. Res. Lett.* **19** 105–113.

Sherali, H. D., D. C. Myers. 1988. Dual formulations and subgradient optimization strategies for linear programming relaxations of mixed-integer programs. *Discrete Appl. Math.* **20** 51–68.

Sherali, H. D., C. H. Tuncbilek. 1997. New reformulation linearization/convexification relaxations for univariate and multivariate polynomial programming problems. *Oper. Res. Lett.* **21** 1–9.

Sherali, H. D., O. Ulular. 1989. A primal-dual conjugate subgradient algorithm for specially structured linear and convex programming problems. *Appl. Math. Optim.* **20** 193–221.

Sherali, H. D., O. Ulular. 1990. Conjugate gradient methods using quasi-Newton updates with inexact line searches. *J. Math. Anal. Appl.* **150** 359–377.

Sherali, H. D., G. Choi, Z. Ansari. 2001. Limited memory space dilation and reduction algorithms. *Comput. Optim. Appl.* **19** 55–77.

Sherali, H. D., G. Choi, C. H. Tuncbilek. 2000. A variable target value method for nondifferentiable optimization. *Oper. Res. Lett.* **26** 1–8.

Shor, N. Z. 1970a. Utilization of the operation of space dilatation in the minimization of convex functions. *Kibernetika* **6** 6–12.

Shor, N. Z. 1970b. Convergence rate of the gradient descent method with dilatation of the space. *Kibernetika* **6** 80–85.

Shor, N. Z. 1985. *Minimization Methods for Non-Differentiable Functions*. Springer-Verlag, New York.