A SEQUENTIAL LINEARIZATION APPROACH FOR SOLVING MIXED-DISCRETE NONLINEAR DESIGN OPTIMIZATION PROBLEMS

by

Han Tong Loh

and

Panos Y. Papalambros

Technical Report UM-MEAM-89-08

Abstract

Design optimization models often contain variables that must take only discrete values, such as standard sizes. Nonlinear optimization problems with a mixture of discrete and continuous variables are very difficult, and existing algorithms are either computationally intensive or applicable to models with special structure. A new approach for solving nonlinear mixed-discrete problems with no particular structure is presented here, motivated by its efficiency for models with extensive monotonicities of the problem's objective and constraint functions with respect to the design variables. It involves solving a sequence of mixed-discrete linear approximations of the original nonlinear model. In this article, a review of previous approaches is followed by description of the resulting algorithm, its convergence properties and limitations. Several illustrative examples are given. A sequel article presents a detailed algorithmic implementation and extensive computational results

June 1989

DESIGN LABORATORY
THE UNIVERSITY OF MICHIGAN
ANN ARBOR

Introduction

A general mathematical model for engineering design optimization is the nonlinear programming (NLP) formulation

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) \\ &\text{subject to} && \mathbf{h}_j(\mathbf{x}) = 0 \qquad j = 1,\ldots, k \\ &&& \mathbf{g}_j(\mathbf{x}) \le 0 \qquad j = (k+1),\ldots, m \\ &&& \mathbf{l}_i \le \mathbf{x}_i \le \mathbf{u}_i \qquad i = 1,\ldots, n \end{aligned} \tag{1}$$

where f, $\mathbf{h}_j$, and $\mathbf{g}_j$ are scalar objective, equality, and inequality constraint functions respectively, and $\mathbf{u}_i$ and $\mathbf{l}_i$ are upper and lower bounds, respectively, for the design variables $\mathbf{x}$ (see Table of Notation and Abbreviations). The objective and constraint functions may be given explicitly by algebraic functions or implicitly by iterative computational procedures, such as finite element analysis. In NLP formulations, the variables are usually assumed to take real continuous values. However, variables that can take only integer or discrete values occur naturally and frequently in engineering design models. Examples are the number of teeth in a gear, the number of bars in a truss, and the size of components available only in standard sizes. A variable that can take only integer values is called an *integer variable*, while one that can take its value only from a set of discrete values is called a *discrete variable*. Since integer variables are also discrete, the term "discrete variables" will refer to both integer and discrete variables, and "integer variables" to integer variables only. When some variables are discrete and some are continuous, the problem is a *mixed-discrete* one. The presence of discrete variables makes solution of NLP problems substantially more difficult.

Equality constraints cannot be usually satisfied when the variables are discrete, or if satisfied, they can be eliminated by a variable reduction and elimination procedure, at least theoretically. For example, the equation $x_1^2x_2 + x_2^3 - 21 = 0$ has no solution when x_1 and x_2 must be integers. Hence, research has focused mostly on problems with only inequality constraints, and the term *Mixed-Discrete Nonlinear Programming* (MDNLP) problems is used here for mathematical models stated as

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) \\ &\text{subject to} && \mathbf{g}_j(\mathbf{x}) \le 0 \qquad j = 1,\ldots, m \\ &&& \mathbf{l}_i \le \mathbf{x}_i \le \mathbf{u}_i \qquad i = 1,\ldots, n \\ &&& \mathbf{x} \in \mathcal{X} \subset \mathcal{D}^d \times \mathcal{R}^{(n-d)} \\ &&& f : \mathcal{X} \rightarrow \mathcal{R},\ \mathbf{g}_j : \mathcal{X} \rightarrow \mathcal{R} \end{aligned} \tag{2}$$

In the above model, $\mathbf{x}$ is a design point in a n-dimensional design space and consists of d discrete variables and (n-d) continuous variables. $\mathcal{D}$ denotes a discrete set for each of the discrete variables and $\mathcal{R}$ is the real continuous space.

TABLE OF NOTATION AND ABBREVIATIONS

Notation

$\mathcal{S}$	- script uppercase denotes a set or vector space
$\mathbf{A}$	- bold uppercase denotes a matrix
$\mathbf{x}$	- bold lowercase denotes a point or a vector
$\mathbf{x}_1$	- bold lowercase with bold subscript refers to a particular point (or vector)
$\mathbf{x}_i$	- bold lowercase with normal subscript refers to a component of a vector
x_1	- normal lowercase with or without subscript refers to a variable

Abbreviations

LP	- linear programming
NLP	- nonlinear programming
MDLP	- mixed-discrete linear programming
MDNLP	- mixed-discrete nonlinear programming
QP	- quadratic programming
RMDLP	- restricted mixed-discrete linear program
SLP	- sequential linear programming
SQP	- sequential quadratic programming

We shall define now global and local minimizers for a MDNLP model.

Definition 1. A point $\mathbf{x}_1$ is said to be *feasible for the MDNLP problem*, if $\mathbf{g}(\mathbf{x}_1) \leq \mathbf{0}$ and $\mathbf{x}_1$ belongs to the set $\mathcal{X}$.

Definition 2. The value $f(\mathbf{x}*)$ is said to be the *global minimum*, and the point $\mathbf{x}*$ the *global minimizer*, for the MDNLP problem, if $\mathbf{x}*$ is feasible for the problem and $f(\mathbf{x}*) \leq f(\mathbf{x})$ for all feasible $\mathbf{x}$.

Definition 3. The *discrete neighborhood* of a point $\mathbf{x}$ is defined as the set of all points $\mathbf{y}$, whose discrete components differ +1, 0 or -1 discrete units from the corresponding components of $\mathbf{x}$, and whose continuous components are within a distance δ of the corresponding components of $\mathbf{x}$, $\mathbf{x}$ itself being excluded from its own discrete neighborhood. Formally, this set is then defined as

$$DN(\mathbf{x}) = \{\mathbf{y} : |y_i - x_i| = 1 \text{ or } 0 \text{ discrete units}, \; i = 1,..., d;$$
$$|y_j - x_j| \leq \delta, \; j = d+1,..., n; \; \mathbf{y} \neq \mathbf{x}\} \qquad (3)$$

Definition 4. The point $\mathbf{x}_l$ is said to be a *local minimizer* for problem MDNLP, if $\mathbf{x}_l$ is feasible for problem MDNLP and $f(\mathbf{x}_l) \leq f(\mathbf{y})$ for all feasible $\mathbf{y}$ contained in $DN(\mathbf{x}_l)$.

There are several reasons why MDNLP problems are more difficult to solve than their continuous counterparts. We briefly describe them below.

Since mixed-discrete problems have many local minimizers, it may be meaningless to find a local minimizer, unless the minimum found is within a certain (acceptable) distance from a known bound on the global one. Proliferation of local minima occurs even when the underlying continuous problem (obtained by relaxing the discreteness requirement) is convex.

Example1. Consider the model

$$\begin{aligned} &\text{minimize} && f = -x_1 - 1.8x_2 && x_1, x_2 \text{ integer} \\ &\text{subject to} && g_1: x_1^2 + (x_2 + 6)^2 - 85 \leq 0 && \\ & && g_2: 1 - x_1^2 \leq 0 && (4) \\ & && g_3: -x_2 \leq 0 && \end{aligned}$$

The continuous problem is convex and has only one global (and local) minimizer at $(4.477, 2.059)^T$. The discrete problem however has three local minimizers at $(2, 3)^T$, $(4, 2)^T$ and $(6, 1)^T$. See Figure 1. □

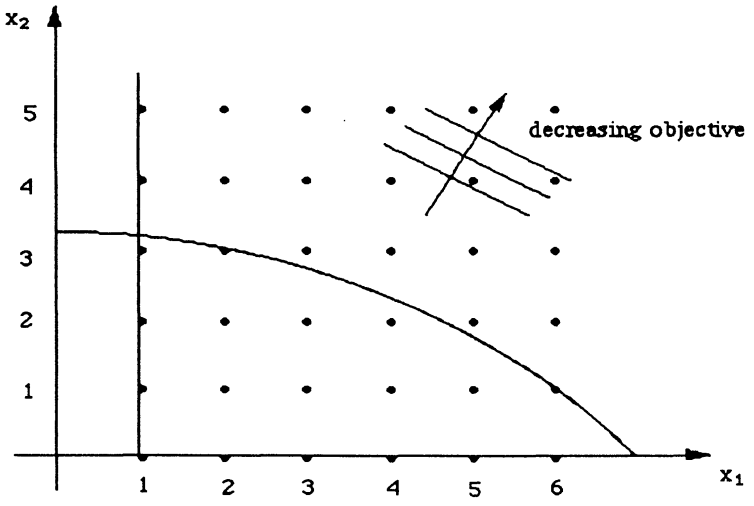


Figure 1. A convex continuous problem with three discrete minimizers (Example 1).

Derivation of optimality criteria for a global minimizer has not yet evolved beyond Definition 2, which really entails comparing all feasible discrete solutions. Unlike its

continuous counterpart, optimality criteria such as the sufficiency of the Karush-Kuhn - Tucker conditions for convex problems do not exist. This means that after a computational procedure terminates, it is difficult to guarantee that the point found is a global minimizer unless an explicit or implicit enumeration of all other points has been considered.

Further, MDNLP problems are NP-complete (Papadimitriou & Steiglitz, 1982), meaning that it is not possible to guarantee finding the solution in polynomial time. NP-complete problems are inherently hard to solve and in practice one has to be satisfied with procedures that find just a good feasible solution.

Finally, the global minimizer of a MDNLP problem may be far from that of the corresponding continuous problem, obtained when the discreteness requirement is *relaxed.* Procedures that solve the relaxed problem and search for a discrete solution in the neighborhood of the continuous solution, often will either find only a local discrete minimizer or fail to find a discrete solution at all. Furthermore, numerical techniques that tackle the continuous problem usually guarantee to find only a local continuous minimizer and not a global one. Hence, it becomes even less likely that techniques which simply couple continuous optimization with a local search strategy will find the global discrete minimizer.

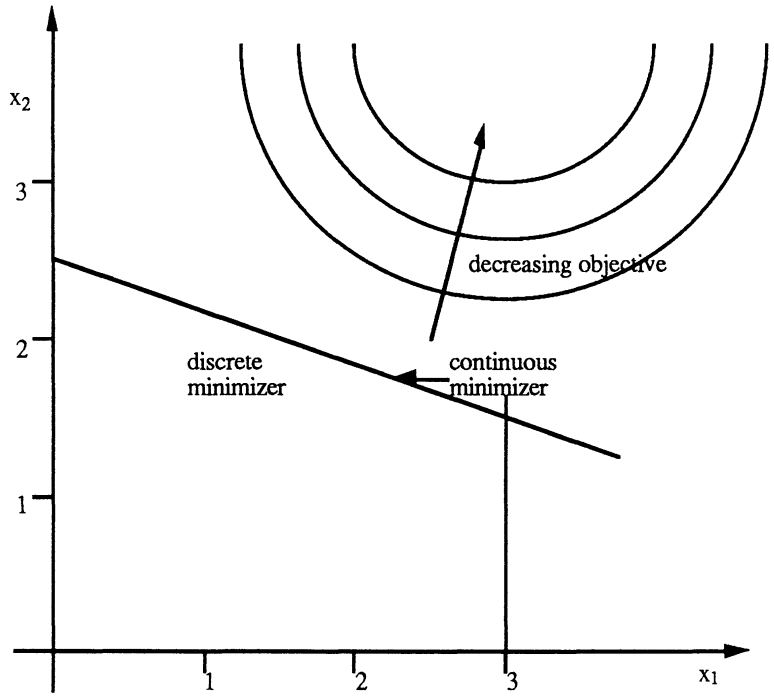


Figure 2. The discrete minimizer is not in the discrete neighborhood of the continuous minimizer (Example 2).

Example 2. The problem

$$\begin{aligned} \text{minimize} \quad & f = (x_1 - 3)^2 + (x_2 - 4)^2 \qquad x_1, x_2 \text{ integer} \\ \text{subject to} \quad & g_1: x_1 + 3x_2 - 7.5 \leq 0 \\ & g_2: x_1 - 3 \leq 0 \quad \text{(b)} \\ & g_3: -x_1 \leq 0 \\ & g_4: -x_2 \leq 0 \end{aligned} \tag{5}$$

has the continuous minimizer at $(2.251, 1.750)^T$ while the global discrete minimizer is at $(1, 2)^T$. A local neighborhood search near the continuous minimizer will produce the local discrete minimizer at $(3, 1)^T$. However, $f(1, 2) = 8$, while $f(3, 1) = 9$; see Figure 2. □

These difficulties imply that solution techniques are generally computationally expensive. Some techniques (e.g., dynamic programming, outer-approximation) are not so expensive but require special model structure, while others (e.g., Lagrangian relaxation, discrete descent) find only an approximate solution and not necessarily the global one even when the corresponding continuous model is convex. Techniques such as branch and bound, that apply to more general models and guarantee finding the global minimum, generally require a large number of function evaluations, and tend to be impractical for moderate- to large-size problems with computationally expensive function evaluations.

The present article investigates a strategy for solving MDNLP problems with no explicit special structure, but exploiting a specific model property, i.e., presence of monotonicities in the model functions. The approach is to solve a sequence of mixed-discrete linear approximations to the nonlinear problem. Total ordering of the discrete variable values is assumed, and the method seeks the global solution rather than an approximate one. Theoretical convergence arguments are given for MDNLP problems with convex objectives and constraints. Extensions to non-convex problems are heuristic, but effective in practice. The approach is motivated by the fact that, for the class of problems where the objective and constraint functions exhibit monotonicities with respect to the design variables (such as sizing problems in structural design), strategies based on sequential linearizations can be very effective (e.g., see Tzannetakis & Papalambros, 1987). For at least this class of problems, the proposed mixed-discrete strategy is particularly effective (Loh & Papalambros, 1989; Bremicker, Loh, and Papalambros, 1989).

In the remainder of this article, after a review of Mixed-Discrete Linear Programming (MDLP) and MDNLP solution strategies, we describe the sequential minimization approach, resulting basic algorithms and convergence properties. We also examine briefly how concepts in monotonicity analysis (Papalambros & Wilde, 1988) are

affected by the presence of discrete variables. In a sequel article (Loh & Papalambros, 1989), a detailed algorithmic implementation and extensive numerical results are presented. Both these articles are based on the dissertation by Loh (1989). Application of the algorithm to optimal structural design problems in described in Bremicker, Loh, and Papalambros (1989).

Previous Efforts for Solving Mixed-Discrete Problems

Many strategies for MDNLP problems derive their basic ideas from solving MDLP problems, which in general being also NP-complete are still difficult but more tractable. A classification of algorithms is made by the type of variables they can handle. The variable disticions we use here are: binary (zero-one) versus non-binary; purely discrete versus mixed-discrete; and integer versus arbitrarily discrete.

Several methods cover more than one classification either directly or indirectly through model transformation, but some methods can be applied strictly only to a specific model type. Model transformations are used to exploit one solution technique for different models, but such transformations may bring new difficulties. For example, a purely discrete linear model with non-binary variables that takes a long time to be solved by a cutting plane method can be transformed into a purely 0-1 discrete linear model that can be solved efficiently by Balas' additive algorithm. However, this will usually result in a large increase in the number of variables.

Methods for linear problems will be reviewed first, as the subsequently proposed SLP approach uses solutions of mixed-discrete linear subproblems.

MIXED-INTEGER LINEAR PROGRAMMING (MDLP)

An obvious way to try to solve a MDLP problem is by rounding the continuous optimizers to nearby discrete values. When the number of discrete choices in each discrete set is large and the feasible domain is also large, such an approach works to the extent that the answer obtained is optimal or near-optimal. However, it is well documented (Hillier & Lieberman, 1974; Glover & Sommers, 1975) that an optimal or even a feasible solution may not be produced, because there may not exist any feasible discrete solution in the neighborhood of the continuous optimizer. More elaborate methods, briefly reviewed here, are branch and bound, cutting plane, dynamic programming, implicit enumeration, and Lagrangian relaxation.

Branch and bound is most commonly used for purely discrete and MDLP problems without special structure. Its first use for MDLP is attributed to Land & Doig (1960), whose algorithm was subsequently modified by Dakin (1965). In branch and bound

procedures, subproblems are created by partitioning the feasible domain to force the integer variables to take integer values. The partitioning is done perpendicular to the axis of the integer variable chosen for the branching. In *cutting plane* algorithms, the feasible domain stays in one continuous piece, while trimming away corners of the domain that are solutions to the continuous linear program but violate the integer requirement. This is done by adding a linear constraint, called a *cut*, to the continuous problem, cuts not being perpendicular to any axis in general. The method is readily applicable to both purely integer and mixed-integer LP problems. The first cutting plane algorithm is attributed to Gomory (1958), with many variations following (Salkin, 1975). Though these algorithms generated much theoretical interest, they are not generally used in practice even for linear problems of moderate size. One difficulty is that as cuts are added they tend to become more parallel, leading to numerical difficulties. Another difficulty is that a large number of cuts is required to produce an integer or mixed-integer solution, even for small problems. An implementation of Dakin's algorithm for MDLP is used in the SLP algorithm proposed in the present article.

Certain multistage problems can be solved by *dynamic programming,* leading to extremely efficient methods for mixed-discrete models. For example, in the multiple-choice knapsack problem, dynamic programming (Bean, 1987) is orders of magnitude faster than branch and bound (Sinha & Zoltners, 1979). Difficulties exist also. Casting problems into a dynamic programming model requires existence of monotonicity and separability (Mitten, 1964). Further, dynamic programming works well when the number of constraints is small compared to the number of variables, making it easy to find minimal values that are also feasible for intermediate stages. In most mechanical engineering design optimization problems, the converse is true; there is usually a small number of variables and a relatively large number of constraints. Finally, dynamic programming is usually memory intensive due to storage of values for the intermediate stages.

In *implicit enumeration* methods, one may enumerate a partial tree and claim that non-enumerated nodes are infeasible or have objective function values worse than those enumerated. Branch and bound and dynamic programming are implicit enumeration methods, but because of their distinctive features they are often considered separately. The most well-known implicit enumeration is Balas' Additive Algorithm (Balas, 1965) used to solve 0-1 integer LPs. Lemke & Spielberg (1967) extended Balas' algorithm to handle continuous variables as well. At each node where the integer variables have been fixed, a LP-subproblem is solved to obtain the continuous variables.

The Lagrangian relaxation method by Geoffrion (1974) has its roots in the surrogate constraints concept of Glover (1968). It takes advantage of the fact that often a

difficult integer programming problem is an easy one complicated by a small number of troublesome constraints. Adding a scaled vector of these complicating constraints to the objective function and removing ("relaxing") these constraints from the constraint set yields an easier problem. The optimal objective function value for the relaxed problem is a lower bound to the original problem, but to obtain a tight bound, the dual of the relaxed problem must be solved. This dual is C^1 discontinuous and its solution requires methods for non-smooth optimization. Lagrangian relaxation is best used in conjunction with branch and bound. Unfortunately, the minimizer found for the dual of the relaxation can be very different from the true multipliers of the original problem, without even primal feasibility. This makes relaxation unsuitable for an SLP method. In addition, the selection of constraints to be considered as "easy" and "complicated" is subjective, and the choice greatly affects the effectiveness of the method. As there is no theory about which constraints to relax (except perhaps monotonicity analysis - if applicable), use of relaxation in a SLP approach with automatic generation of a MDLP subproblem appears difficult.

MIXED-INTEGER NONLINEAR PROGRAMMING (MDNLP)

Reiter & Rice (1966) proposed a modified *discrete gradient method* for quadratic mixed-discrete problems, similar to the gradient method for continuous problems. A distribution of results from different starting points gives an estimate of the global solution. Numerical tests indicated that a reasonable solution may be reached. The method can be modified for mixed-discrete and non-convex problems. However, only an approximate solution is produced and a global one is not guaranteed.

Another approach is to treat discrete requirements as explicit constraints and construct an objective function penalizing deviations from discrete values. Davydov & Sigal (1972) devised a number of penalty functions for 0-1 problems and convex problems with regularly spaced discrete intervals. Gisvold & Moe (1972) offered a similar method for discrete variables with arbitrarily spaced intervals. A recovery step is used to proceed, if the penalty method does not terminate near a discrete point. The algorithm was applied to two nonlinear structural problems with results comparable to those from branch and bound. There are two main difficulties that arise in employing this method, both hard to overcome. As in the continuous penalty method, penalty parameters are difficult to set *a priori*, with different parameters leading to different results. Unless one has some knowledge about the discrete solution, it is difficult to decide whether the result obtained is optimal or the parameters need adjustment to continue iterating. The more serious difficulty is that adding a penalty for moving away from a discrete point creates a local optimum for each discrete point. This makes finding global optima with a continuous method even more difficult.

Cooper & Cooper (1975) developed a search procedure for purely integer nonlinear problems, which uses dynamic programming as an intermediate step. The problems must have objectives that are separable, monotonic in every variable, and also integer functions themselves. The approach is simple and elegant, the algorithm terminates after a finite number of steps, and no storage of intermediate results is required. However, the obvious model limitations do not allow general utility. Cooper & Cooper showed very attractive computational times, but their examples were created by an automatic generation process and did not correspond to real-world problems.

The only application of Lagrangian relaxation to discrete nonlinear problems that the authors have found, was by Schmit & Fleury (1980). A structural optimization problem is solved by a sequence of convex and separable approximation subproblems. Each subproblem is solved by Lagrangian relaxation and the dual objective function is maximized using a subgradient method combined with explicit enumeration. This method was applied by Ringertz (1988) to solve six structural optimization problems, and the results were compared with those using branch and bound. Similar to applying Lagrangian relaxation to MDLP problems, minima obtained for these nonlinear structural problems are very close or equal to those given by branch and bound, but the minimizers obtained are different.

Fox & Liebman (1981) used the "complex" algorithm (Nelder & Mead, 1965; Box 1965) for continuous NLP problems, modified for discrete and mixed-discrete variables. Numerical tests on fifteen problems gave better results than both a discrete penalty algorithm and an algorithm using rounding off the continuous optimizer.

Gupta & Ravindran (1983) applied Dakin's method to nonlinear mixed-integer problems, using a generalized reduced gradient method to solve the nonlinear continuous subproblem at each node, and published results for 24 test problems. Computational effort is affected by rules for selecting nodes for expansion and variables for branching. Gupta (1980) considered three choices for node expansion: depth-first search, best-first search based on the objective function, and best-first search based on an estimation function combining the objective and weighted fractional parts of the variables. Three choices were also considered for branching variables: the most fractional part, the lowest-indexed-first, and a "pseudo-cost" index estimating deterioration of the objective per fractional change in each integer variable. No single choice appeared to offer a definite advantage. Gupta used a statistical method to compare speed of convergence using different combinations of these choices. Overall, he found that for selecting a branching variable, the best strategy is to branch from the variable with the most fractional part. For selecting the expansion node, the best strategy is either to use the node with the lowest objective function or use an

estimation function criterion. Sandgren (1988) used basically the same approach as Gupta, but included cases with equality constraints having 0-1 variables. He also included some heuristics in monitoring expansion of nodes and the solution process of subproblems.

The main advantage of branch and bound is that a proper fathoming rule guarantees finding the global minimum. The bounding process assumes that at a discrete minimizer the objective function value is a lower bound on the discrete minimum of any other node in the tree, an assumption valid for objectives that are at least pseudoconvex and constraints that are convex. For non-convex problems, branch and bound cannot guarantee finding the global minimizer since it may end up fathoming nodes that should not have been fathomed. However, branch and bound usually finds a good solution for these cases. The main disadvantage is that a large number of nodes and corresponding nonlinear subproblems is generated. Except for small problems, branch and bound is not a realistic alternative when solution of subproblems is computationally intensive.

For mixed-discrete problems, Cha & Mayne (1987) suggested coupling a sequential quadratic programming (SQP) method with a local search strategy. A continuous SQP algorithm moves the search quickly in the vicinity of the optimizer, and a local search based on gradient information of the objective finds the best discrete solution in this neighborhood. The local search uses first the projected gradient direction; if this fails to find an improved discrete point, the search is carried out along coordinate directions in order of decreasing value of gradient components. If this also fails, a heuristic "two variables at a time" search is made in a selected quadrant of the discrete neighborhood. The two phases, QP approximation and local discrete search, are applied in turn until no improved discrete points can be formed. Cha & Mayne (1988) showed that, instead of the popular rank two update formula for the Hessian matrix, a symmetric rank one update must be used that does not require quasi-Newton directions to have an accurate estimate of the Hessian. This is important, since due to the discrete nature of the variables, the update vectors cannot take quasi-Newton directions.

The algorithm handles discretization at regular and irregular intervals. Results for 25 test problems showed a low number of objective and constraint function evaluations. However, there is no guarantee for finding the global minimum, particularly when the discrete optimizer is at least a few discrete steps away from the continuous one. The algorithm brings the search to a neighborhood of the continuous optimizer, and settles on the local discrete minimizer in this neighborhood. Furthermore, as the local search depends on the direction of the gradient vector of the objective, when the objective function is not pseudoconvex the downhill gradient vector can point in the wrong direction for the local

search. For example, for the non-pseudoconvex objective shown in Figure 3, at x = 3, the downhill gradient vector suggests that, to decrease the objective, the search should be carried out in the positive x direction. Since f(4) > f(3), the local search stops at x = 3, whereas a better discrete optimizer is at x = 2.

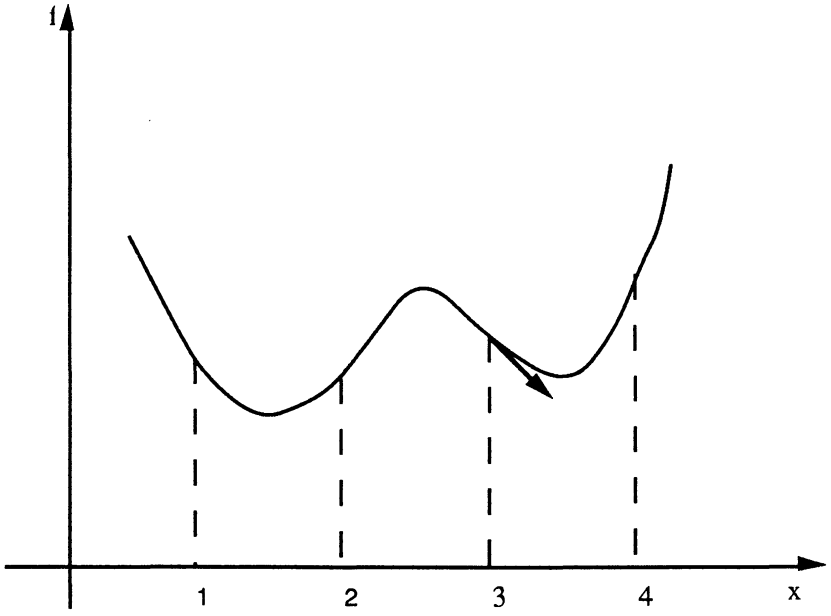


Figure 3. A non-pseudoconvex function

The method is not likely to fail if the interval of discretization is relatively small so that the discrete optimizer lies near the continuous one and the downhill gradient vector is unlikely to point in the wrong direction. For several of the test problems reported by Cha & Mayne (1987), the interval of variable discretization was 0.001, making the problems behave almost like continuous ones. Such criticisms not withstanding, this SQP approach appears useful for solving mixed-discrete nonlinear problems.

Lu (1988) solved purely discrete nonlinear optimization problems by Boolean algebra techniques. The objective and constraints are converted into equivalent 0-1 polynomial functions. Each Boolean polynomial is equivalent to the union of all its prime implicants (Hammer, 1974). Some Boolean variables are determined by contradiction, fixation and implication, and others by a heuristic search that decreases the objective function. When the objective cannot be reduced further, the point is taken as optimal. The main advantage of this method is that it is very fast. Some disadvantages are: (1)

Conversion to a 0-1 model causes a large increase in the number of variables. For complicated constraints, conversion is difficult and prone to error, though symbolic manipulators can be used. The prime implicants are also hard to generate; if the constraints come, say, from finite element analysis, the task may be impossible. (2) Reduction of the objective uses heuristics that may lead to a local optimizer. (3) No extension to the mixed-discrete case seems possible.

Duran & Grossmann (1986) proposed a linearization algorithm for mixed-integer NLP, when integer variables appear only in linear functions and the nonlinear functions involving continuous variables are convex. Separability of integer and continuous variables allows the feasible domains to be independently characterized. The nonlinear constraints are linearized to create a mixed-integer linear approximating "master" problem. This is solved to obtain values of integer variables. With integer values fixed, a continuous nonlinear subproblem is solved to get a new iterate. A new linearization is then taken and added to the master problem. Thus, the supporting half spaces, whose intersection confines the convex feasible region, are obtained. The algorithm consists then of solving an alternating sequence of nonlinear subproblems and a mixed-integer linear master problem. By the convexity assumption, the continuous subproblem provides an upper bound to the MDLP problem while the master problem provides a lower bound. When the two bounds meet or cross over, the process terminates. The method is similar to the generalized Bender's decomposition (Geoffrion, 1972). A detailed discussion of the relationship between the two is given in Duran's dissertation (1984).

Duran & Grossmann (op.cit.) reported results on four test problems with 0-1 variables. An extension by Kocis & Grossmann (1987) included nonlinear equality constraints with only continuous variables and linear equality constraints with only integer variables. Satisfactory results were reported for design optimization of chemical process flowsheets. Viswanathan & Grossmann (1989) further extended the algorithm for nonconvex constraints by augmenting objective and constraints in the mixed-integer linear master problem with penalty functions. Computational results on eighteen test problems show that the proposed extension has high reliability for finding the minimizers in these nonconvex problems.

Finally, an SLP method for purely discrete NLP problems in truss design was reported by John, Ramakrishnan, and Sharma (1988), with truss sizing examples.

In conclusion, no single method appears superior in all three criteria of general applicability, robustness, and efficiency. These desired characteristics must be traded off, and any special structure in the model should be exploited.

A Basic Sequential Linearization Algorithm

We now outline a primitive SLP algorithm for MDNLP problems, and in the next section we refine it in order to overcome convergence difficulties. First, let us define some terms.

Definition 5. Given a MDNLP model of the form

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) \\ &\text{subject to} && \mathbf{g}(\mathbf{x}) \leq \mathbf{0} \\ & && \mathbf{x} \in \mathcal{X} \subset \mathcal{D}^d \times \mathcal{R}^{(n-d)} \end{aligned} \tag{6}$$

the *mixed-discrete linearized model* (MDLP) about a point $\mathbf{x_0}$, MDLP($\mathbf{x_0}$), is:

$$\begin{aligned} &\text{minimize} && \nabla f(\mathbf{x_0})^T (\mathbf{x} - \mathbf{x_0}) \\ &\text{subject to} && \mathbf{g}(\mathbf{x_0}) + \nabla \mathbf{g}(\mathbf{x_0})^T(\mathbf{x} - \mathbf{x_0}) \leq \mathbf{0} \\ & && \mathbf{x} \in \mathcal{X} \subset \mathcal{D}^d \times \mathcal{R}^{(n-d)} \end{aligned} \tag{7}$$

Definition 6. A point $\mathbf{x_1}$ is said to be *MDNLP-feasible*, if $\mathbf{g}(\mathbf{x_1}) \leq \mathbf{0}$ and $\mathbf{x_1}$ satisfies all discreteness requirements of $\mathcal{X}$.

Definition 7. A point $\mathbf{x_1}$ is said to be *MDLP-feasible* with respect to model MDLP($\mathbf{x_0}$), if $\mathbf{g}(\mathbf{x_0}) + \nabla \mathbf{g}(\mathbf{x_0})^T(\mathbf{x} - \mathbf{x_0}) \leq \mathbf{0}$ and $\mathbf{x_1}$ satisfies all discreteness requirements of $\mathcal{X}$.

The basic SLP algorithm (SLP1) is given as follows:

Algorithm SLP1

1. Given $\mathbf{x_0}$, δ ($\delta > 0$). Set k = 0.
2. Construct MDLP($\mathbf{x_k}$).
3. Solve $\mathbf{x_{k+1}}$ = arg min (MDLP($\mathbf{x_k}$)) using a LP solver and Dakin's branch and bound rule.
4. If $\| \mathbf{x_{k+1}} - \mathbf{x_k} \| < \delta$, stop. Else set k = k + 1 and go to step 2.

Definition 8. The *SLP1 solution* to MDNLP is defined as the point obtained at the termination of algorithm SLP1.

Example 3. (Gupta, 1980; Test Problem 3.) Consider the problem

$$\begin{aligned} &\text{minimize} && f = (x_1 - 8)^2 + (x_2 - 2)^2 && x_1, x_2 \text{ integer} \\ &\text{subject to} && g_1 : 0.1x_1^2 - x_2 \leq 0 \\ & && g_2 : \tfrac{1}{3}x_1 + x_2 - 4.5 \leq 0 \end{aligned} \tag{8}$$

The minimizer of the continuous problem is at $(5.245, 2.75)^T$. Round it to $(5, 3)^T$ and note that this point is infeasible. Linearize the model in Eq.(8) at point $(5, 3)^T$ and solve the integer problem:

$$\begin{array}{lll} \text{minimize} & -6x_1 + 2x_2 & x_1, x_2 \text{ integer} \\ \text{subject to} & x_1 - x_2 - 2.5 \leq 0 & \\ & \frac{1}{3} x_1 + x_2 - 4.5 \leq 0 & \end{array} \qquad (9)$$

The solution $(4, 2)^T$ is feasible with $f(4, 2) = 16$. Linearize again at $(4, 2)^T$ to get the problem:

$$\begin{array}{lll} \text{minimize} & -8x_1 & x_1, x_2 \text{ integer} \\ \text{subject to} & 0.8x_1 - x_2 - 1.6 \leq 0 & \\ & \frac{1}{3} x_1 + x_2 - 4.5 \leq 0 & \end{array} \qquad (10)$$

Again $(4, 2)^T$ is the minimizer and the linearization process terminates. (This is the same discrete solution obtained by Gupta.) It will be shown in Theorem 4 of the next section that this point must be the global discrete minimizer for the nonlinear problem. Graphically, the process is depicted in Figure 4. □

Convergence and Refinement

We now derive some theoretical results and show how a modified SLP algorithm may converge to the discrete optimizer for convex problems.

Definition 9. (Ponstein, 1967.) Let $\mathcal{S}$ be a nonempty set in $\mathcal{R}^n$. Let a function f: $\mathcal{S} \rightarrow \mathcal{R}$ be differentiable on $\mathcal{S}$. The function f is said to be (*strictly*) *pseudoconvex*, if and only if for every $\mathbf{x_1}, \mathbf{x_2} \in \mathcal{S}$,

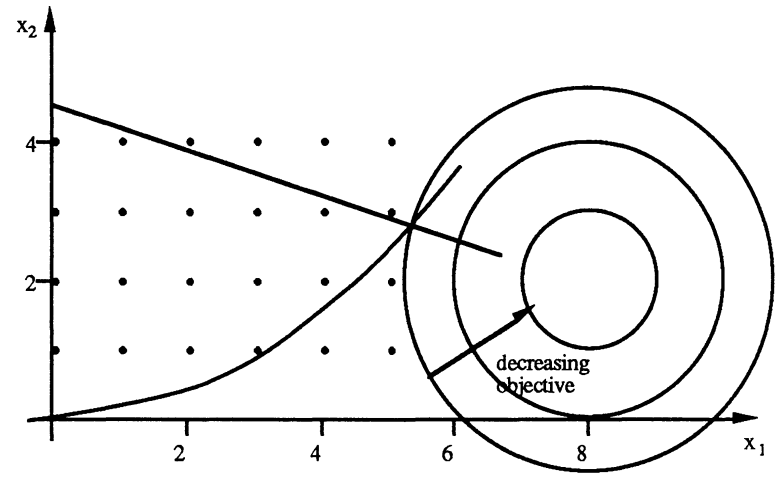
$$(\nabla f(\mathbf{x_1})^T(\mathbf{x_2}-\mathbf{x_1}) \geq 0 \implies f(\mathbf{x_2}) > f(\mathbf{x_1})) \qquad (11a)$$

$$\nabla f(\mathbf{x_1})^T(\mathbf{x_2}-\mathbf{x_1}) \geq 0 \implies f(\mathbf{x_2}) \geq f(\mathbf{x_1}) \qquad (12a)$$

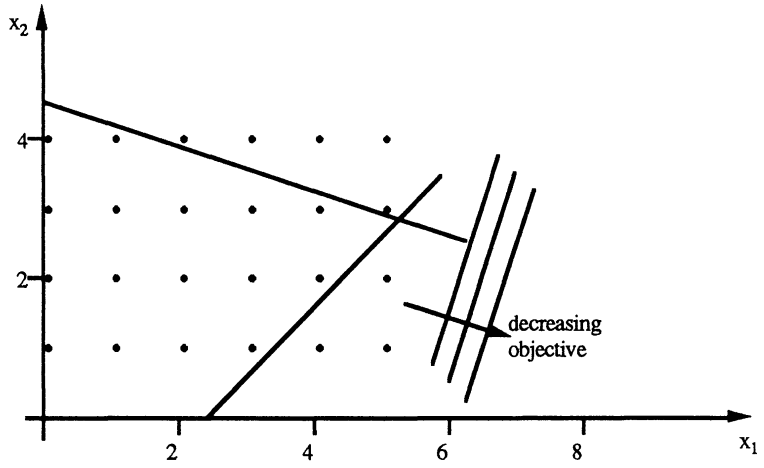
or equivalently

$$(f(\mathbf{x_2}) \leq f(\mathbf{x_1}) \implies \nabla f(\mathbf{x_1})^T(\mathbf{x_2}-\mathbf{x_1}) < 0) \qquad (11b)$$

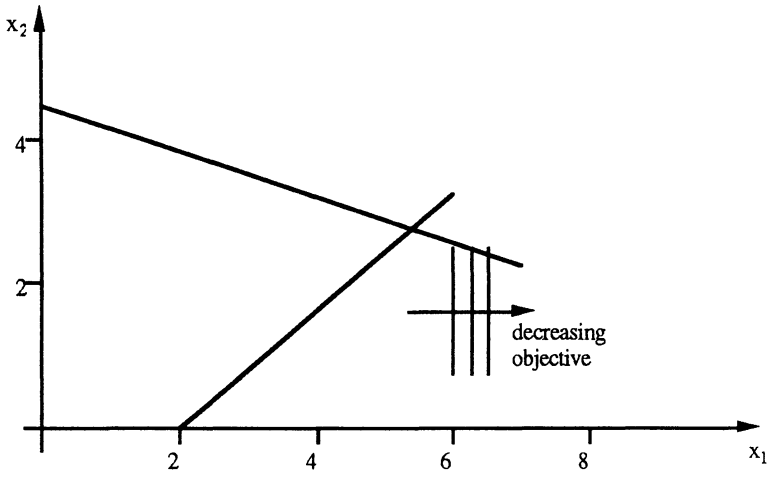
$$f(\mathbf{x_2}) < f(\mathbf{x_1}) \implies \nabla f(\mathbf{x_1})^T(\mathbf{x_2}-\mathbf{x_1}) < 0 \qquad (12b)$$



(a)



(b)



(c)

Figure 4. The linearization process for Example 3. (a) Nonlinear problem. (b) LP approximation at $(5, 3)^T$. (c) LP approximation at $(4, 2)^T$.

LINEAR CONSTRAINTS CASE

Theorem 1. Let f: $\mathcal{R}^n \rightarrow \mathcal{R}$ be a pseudoconvex function and g_i: $\mathcal{R}^n \rightarrow \mathcal{R}$ be linear functions for all i, i = 1,..., m. If the SLP1 solution of the problem

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) && \mathbf{x} \in \mathcal{X} \\ &\text{subject to} && g_i(\mathbf{x}) \leq 0 && i = 1, ..., m \end{aligned} \tag{13}$$

converges to a point $\mathbf{x_1}$, then $\mathbf{x_1}$ is the global discrete minimizer.

Proof: (By contradiction.) Since g_i is linear, its linearization is g_i itself. Hence, any feasible point $\mathbf{x}$ in any of the linear approximation programs is also feasible for the nonlinear problem in Eq.(13). Assume the algorithm has terminated at point $\mathbf{x_1}$. Then $\mathbf{x_1}$ is also the minimizer of the LP problem

$$\begin{aligned} &\text{minimize} && \nabla f(\mathbf{x_1})^T \mathbf{x} \\ &\text{subject to} && g_i(\mathbf{x}) \leq 0 \\ & && \mathbf{x} \in \mathcal{X} \end{aligned} \tag{14}$$

Assume further there exists a point $\mathbf{x_2}$ such that

$$f(\mathbf{x_2}) < f(\mathbf{x_1}) \tag{15}$$

with $\mathbf{x_1}$, $\mathbf{x_2}$ both feasible. Since $\mathbf{x_1}$ is the minimizer of LP problem (14), we have

$$\nabla f(\mathbf{x_1})^T \mathbf{x_1} \leq \nabla f(\mathbf{x_1})^T \mathbf{x_2} \implies 0 \leq \nabla f(\mathbf{x_1})^T(\mathbf{x_2}-\mathbf{x_1}) \tag{16}$$

From the definition of pseudoconvexity in Eq.(12a), this implies $f(\mathbf{x_2}) \geq f(\mathbf{x_1})$ which contradicts the assumption in Eq.(15). □

So far we proved is that if the algorithm converges, then it has to converge to the global minimizer. We have not yet shown that the algorithm is a descent one, and that it will not cycle or oscillate between two fixed points.

Example 4. In one-dimension, for the model

$$\begin{aligned} &\text{minimize} && f = (x - 3)^2 && x \text{ integer} \\ &\text{subject to} && x \leq 4 \\ & && x \geq 2 \end{aligned} \tag{17}$$

Algorithm SLP1 will cycle between points x = 2 and x = 4. □

To have descent and to avoid cycling, we need two addditional steps in the algorithm:

1. In generating the successive linear programs, move to another point $\mathbf{x_2}$ from $\mathbf{x_1}$ only if there is a strict improvement in the *nonlinear* objective, i.e., only if $f(\mathbf{x_2}) < f(\mathbf{x_1})$.

2. Incorporate decreasing step bounds (Palacios-Gomez, 1980), i.e., linearize the MDNLP model in the form

$$\begin{aligned} &\text{minimize} && \nabla f(\mathbf{x_0})^T\mathbf{x} \\ &\text{subject to} && \nabla g(\mathbf{x_0})^T(\mathbf{x}-\mathbf{x_0}) + g(\mathbf{x_0}) \le 0 \\ & && -\mathbf{t_0} \le \mathbf{x}-\mathbf{x_0} \le \mathbf{t_0} \\ & && \mathbf{x} \in \mathcal{X} \end{aligned} \tag{18}$$

and decrease $\mathbf{t_0}$ as the algorithm progresses.

Definition 10. Model (18) is called the *restricted mixed-discrete linearized program* (RMDLP).

An improved SLP algorithm (SLP2) is as follows:

Algorithm SLP2

1. Given $\mathbf{x_0}$, $\mathbf{t_0}$, r_t, δ ($r_t > 0$). Set k = 0.
2. Construct RMDLP($\mathbf{x_k}$).
3. Solve $\mathbf{z}$ = arg min (RMDLP($\mathbf{x_k}$)) by using a LP solver and Dakin's branch and bound rule.
4. If $\| \mathbf{z} - \mathbf{x_k} \| < \delta$, stop.
5. If $f(\mathbf{z}) < f(\mathbf{x_k})$, set k=k+1, $\mathbf{x_k} = \mathbf{z}$, go to step 2. Else set $\mathbf{t_0} = \mathbf{t_0}/r_t$, go to step 2.

Example 5. Consider the problem

$$\begin{aligned} &\text{minimize} && f = 7x_1^2 + 6x_2^2 + 8x_3^2 - 6x_1x_3 + 4x_2x_3 - 15.8x_1 - 93.2x_2 - 63x_3 + 500 \\ &\text{subject to} && g_1: \ 142x_1 + 172x_2 + 118x_3 \le 1992 \\ & && g_2: \ 98x_1 + 114x_2 + 44x_3 \le 1162 \\ & && g_3: \ 40x_1 + 72x_2 + 34x_3 \le 703 \end{aligned} \tag{19}$$

with x_1, x_2, x_3 integer. Linearize initially at $(3, 6, 3)^T$ and impose a step bound of $t_{0i} = 5$:

$$\begin{aligned} &\text{minimize} && 8.221x_1 - 9.164x_2 - 8.976x_3 \\ &\text{subject to} && g_1: \ 142x_1 + 172x_2 + 118x_3 \le 1992 \\ & && g_2: \ 98x_1 + 114x_2 + 44x_3 \le 1162 \\ & && g_3: \ 40x_1 + 72x_2 + 34x_3 \le 703 \\ & && -5 \le x_1 - 3 \le 5 \\ & && -5 \le x_2 - 6 \le 5 \\ & && -5 \le x_3 - 3 \le 5 \\ & && x_1, x_2, x_3 \text{ integer} \end{aligned} \tag{20}$$

Point $(3, 6, 3)^T$ is MDNLP-feasible and f(3, 6, 3) = 83.4. The solution to the LP subproblem in Eq.(20) is $(1, 5, 8)^T$ with a nonlinear objective value of f(1, 5, 8) = 296.8.

Since this is worse than the incumbent, we reject it and decrease the step bounds by some ratio, say 2, from $t_{0i} = 5$ to $t_{0i} = 2.5$. Solving again, we get point $(1, 7, 4)^T$. Since f(1, 7, 4) = 96.8 is worse than the incumbent, we reduce t_0 to 1.25 and try again. This time we obtain the point $(2, 7, 3)^T$ with f(2, 7, 3) = 69.0. Since this improves the nonlinear objective, we accept $(2, 7, 3)^T$ as a better point to replace the incumbent and linearize again, restoring the original step bounds. The new subproblem is

$$\begin{aligned} \text{minimize} \quad & -5.786x_1 + 2.842x_2 + 1.024x_3 \\ \text{subject to} \quad & g_1: \ 142x_1 + 172x_2 + 118x_3 \le 1992 \\ & g_2: \ 98x_1 + 114x_2 + 44x_3 \le 1162 \\ & g_3: \ 40x_1 + 72x_2 + 34x_3 \le 703 \\ & -5 \le x_1 - 2 \le 5 \\ & -5 \le x_2 - 7 \le 5 \\ & -5 \le x_3 - 3 \le 5 \\ & x_1, x_2, x_3 \text{ integer} \end{aligned} \tag{21}$$

The solution to this LP is $(7, 2, 1)^T$, with f(7, 2, 1) = 421. We reject it, decrease t_0 to 2.5, and solve again. We now get point $(4, 5, 1)^T$, with f(4, 5, 1) = 173.2, and rejecting it we reduce t_0 to 1.25. The new solution is $(3, 6, 2)^T$, with f(3, 6, 2) = 90.4. We reject it and reduce t_0 to 0.75. Finally, we get $(2, 7, 3)^T$ and the algorithm has converged. □

Theorem 2. Algorithm SLP2 is a descent algorithm and will terminate.

Proof: Assume the linearization is done at a discrete point $\mathbf{x_1}$ (the current incumbent). The LP subproblem is

$$\begin{aligned} \text{minimize} \quad & \nabla f(\mathbf{x_1})^T \mathbf{x} \\ \text{subject to} \quad & \nabla \mathbf{g}(\mathbf{x}-\mathbf{x_1}) + \mathbf{g}(\mathbf{x_1}) \le \mathbf{0} \\ & \mathbf{x} \in \mathcal{X} \end{aligned} \tag{22}$$

Let $\mathbf{x_2}$ be the solution to this LP, and assume that $f(\mathbf{x_2}) \ge f(\mathbf{x_1})$, so that descent does not occur naturally without imposition of some step bounds. Let $\mathbf{t_0}'$ be half the length of the side of the smallest hypercube with $\mathbf{x_1}$ as center that includes $\mathbf{x_2}$, i.e., $| x_{1i}-x_{2i} | = t_{0i}'$ (see Figure 5). Then $\mathbf{x_2}$ is the global minimizer of the problem

$$\begin{aligned} \text{minimize} \quad & \nabla f(\mathbf{x_1})^T \mathbf{x} \\ \text{subject to} \quad & \nabla \mathbf{g}(\mathbf{x}-\mathbf{x_1}) + \mathbf{g}(\mathbf{x_1}) \le \mathbf{0} \\ & \mathbf{x} \in \mathcal{X} \\ & -\mathbf{t_0}' \le \mathbf{x}-\mathbf{x_1} \le \mathbf{t_0}' \end{aligned} \tag{23}$$

Since $f(\mathbf{x_2}) \ge f(\mathbf{x_1})$, exactly one of the following three cases must happen:

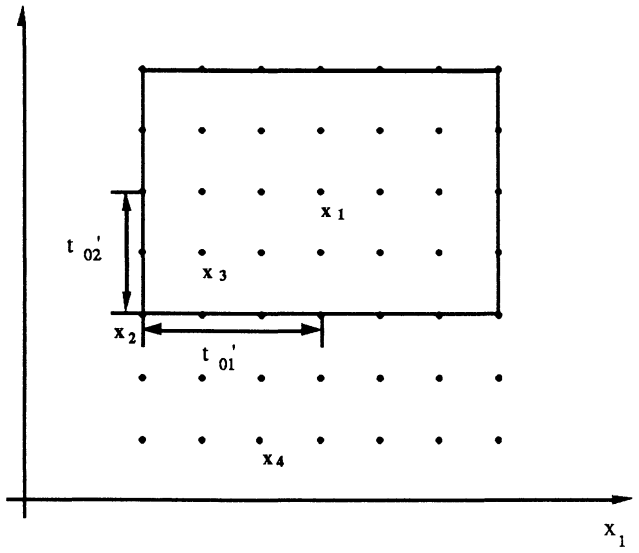


Figure 5. The smallest hypercube centered at $\mathbf{x_1}$ which includes $\mathbf{x_2}$.

Case 1: $\mathbf{x_1}$ is the global minimizer of the mixed-discrete nonlinear program.

Case 2: There exists a point $\mathbf{x_3}$ such that $|\mathbf{x_3} - \mathbf{x_1}| < \mathbf{t_0}'$, $\mathbf{x_3}$ is MDNLP-feasible, and $f(\mathbf{x_3}) < f(\mathbf{x_1})$.

Case 3: There does not exist a point $\mathbf{x_3}$ such that $|\mathbf{x_3} - \mathbf{x_1}| < \mathbf{t_0}'$, $\mathbf{x_3}$ is MDNLP-feasible, and $f(\mathbf{x_3}) < f(\mathbf{x_1})$, but there exists a point $\mathbf{x_4}$ such that $|\mathbf{x_4} - \mathbf{x_1}| \geq \mathbf{t_0}'$, $\mathbf{x_4}$ is MDNLP-feasible, and $f(\mathbf{x_4}) < f(\mathbf{x_1})$.

In Case 1, when we progressively shrink the hypercube from its initial value of $\mathbf{t_0}'$, we find that any discrete feasible points in the resulting problem must have objective function values greater than $f(\mathbf{x_1})$, and so they are rejected. The hypercube is eventually reduced to a size smaller than the discretization interval, and thus we conclude that $\mathbf{x_1}$ is the global minimizer. In Case 2, if the hypercube is decreased from $\mathbf{t_0}'$ by a small amount, point $\mathbf{x_2}$ is rejected as infeasible. The next feasible discrete point will either have $f(\mathbf{x}) \geq f(\mathbf{x_1})$ or $f(\mathbf{x}) < f(\mathbf{x_1})$. In the latter case, we will have achieved descent of the objective function. In the former case, this point is rejected and the hypercube is shrunk further, until eventually a point $\mathbf{x_3}$ is reached, $\mathbf{x_3}$ being MDNLP-feasible with $f(\mathbf{x_3}) < f(\mathbf{x_1})$. In Case 3, we will reject any intermediate points found as we shrink the hypercube to a size smaller than the discretization interval. We will conclude *erroneously* that $\mathbf{x_1}$ is the global minimizer when there exists a point $\mathbf{x_4}$ such that $\mathbf{x_4}$ is MDNLP-feasible with $f(\mathbf{x_4}) < f(\mathbf{x_1})$. In all three cases, Algorithm SLP2 will terminate. In Case 1, the algorithm terminates at the global discrete minimizer. In Case 2, the algorithm descends and terminates when the best discrete minimizer in the hypercube is found. In Case 3, the algorithm terminates but not at the global discrete minimizer. □

Case 3 is much less likely to occur than Case 1 or Case 2. For pseudoconvex f, the graph of f along $\mathbf{x_1x_2}$ must be as shown in Figure 6 with $f(\mathbf{x_1}) \leq f(\mathbf{x_2})$, and the objective in the neighborhood of $\mathbf{x_1}$ is decreasing in the direction $\mathbf{x_1x_2}$. (See Theorem 3 below.) The graph of f along $\mathbf{x_1x_4}$ must be as shown in Figure 7a or 7b with $f(\mathbf{x_1}) > f(\mathbf{x_4})$, and the objective in the neighborhood of $\mathbf{x_1}$ is also decreasing in the direction $\mathbf{x_1x_4}$. (See Theorem 3.) Since in both directions $\mathbf{x_1x_2}$ and $\mathbf{x_1x_4}$ the objective function is decreasing, it is unlikely that all intermediate discrete points found in the hypercube have $f(\mathbf{x}) \geq f(\mathbf{x_1})$, such that there does not exist a point $\mathbf{x_3}$ within the hypercube that is MDNLP-feasible with $f(\mathbf{x_3}) < f(\mathbf{x_1})$, and yet have a point $\mathbf{x_4}$ such that $f(\mathbf{x_4}) < f(\mathbf{x_1})$, $|\mathbf{x_4} - \mathbf{x_1}| \geq \mathbf{t_0}'$. This can happen only if the interval of discretization is relatively large and the level sets of f are ellipsoids of high eccentricity, aligned in such a way that level sets lower than $f(\mathbf{x_1})$ avoid all discrete points in the cone spanned by vectors $\mathbf{x_1x_2}$ and $\mathbf{x_1x_4}$.

Furthermore, if Case 3 occurs for a quadratic objective, the difference of the objective function value of a point like $\mathbf{x_4}$ and the objective function value of the continuous minimizer is bounded from below. The bound depends on the smallest discretization interval and the maximum and minimum eigenvalues of the Hessian. (The exact formula for this bound is currently under investigation.) This implies that, if the difference of the objective function value of a point found within the hypercube from that of the continuous minimizer is smaller than this bound, we can guarantee that Case 3 does not occur.

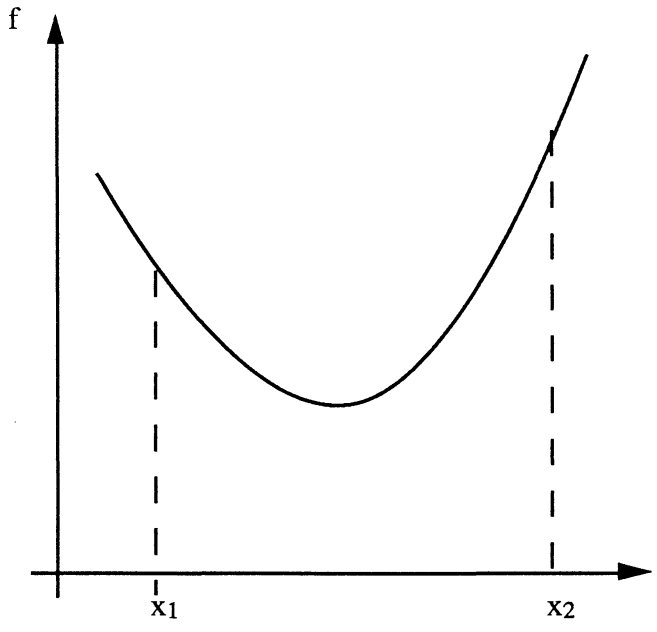


Figure 6. The graph of f along $\mathbf{x_1x_2}$.

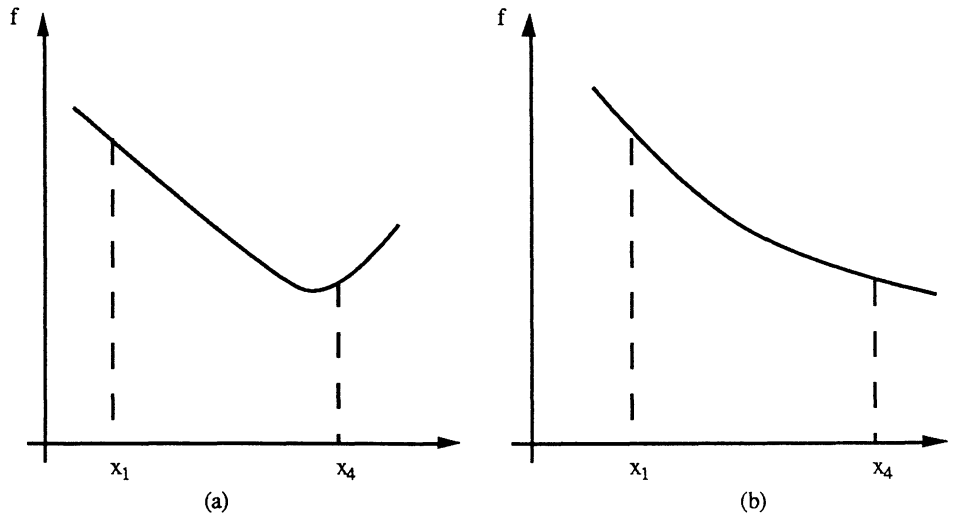


Figure 7. The two possible graphs of f along $\mathbf{x_1x_4}$.

Theorem 3. Let $\mathbf{x_1}$, $\mathbf{x_2}$, $\mathbf{x_4}$ be defined as in the proof of Theorem 2. Then in the neighborhood of $\mathbf{x_1}$ the following are true:

(a) the objective function is decreasing in the $\mathbf{x_1x_2}$ direction;

(b) the objective function is decreasing in the $\mathbf{x_1x_4}$ direction.

Proof: (a) Since $\mathbf{x_2}$ is minimizer of $\nabla f(\mathbf{x_1})^T \mathbf{x}$, we have

$$\nabla f(\mathbf{x_1})^T(\mathbf{x_2}) < \nabla f(\mathbf{x_1})^T(\mathbf{x_1}) \implies \nabla f(\mathbf{x_1})^T(\mathbf{x_2}-\mathbf{x_1}) < 0$$

(b) Since $f(\mathbf{x_4}) < f(\mathbf{x_1})$, then from the definition of pseudoconvexity

$$\nabla f(\mathbf{x_1})^T(\mathbf{x_4}-\mathbf{x_1}) < 0.$$

□

CONVEX CONSTRAINTS CASE

Let us now extend the above results to nonlinear **g**, specifically to **g** that are convex.

Theorem 4. Let $f: \mathcal{R}^n \rightarrow \mathcal{R}$ be a pseudoconvex function and $g_i: \mathcal{R}^n \rightarrow \mathcal{R}$ be convex functions for all i, i = 1,..., m. If the SLP1 solution of the problem

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) \\ &\text{subject to} && g_i(\mathbf{x}) \le 0 \qquad i = 1, \ldots, m \\ & && \mathbf{x} \in \mathcal{X} \end{aligned} \tag{24}$$

converges to a MDNLP-feasible point $\mathbf{x_1}$, then $\mathbf{x_1}$ is the global discrete minimizer.

Proof: The proof follows that for Theorem 1. Since **g** is convex, its linearization $\mathbf{g}(\mathbf{x_1}) + \nabla \mathbf{g}(\mathbf{x_1})\,(\mathbf{x} - \mathbf{x_1}) \le \mathbf{0}$ is an outer-linearization, i.e., the LP approximation overestimates the NLP feasible domain. Therefore, any **x** feasible for the MDNLP problem in Eq.(24) is

feasible for its LP approximation. By the same contradiction as in Theorem 1, it can be shown that if $\mathbf{x_1}$ is feasible for the MDNLP problem, then it must be the global discrete minimizer. □

Extension of this result to Algorithm SLP2 requires use of decreasing step bounds. Note that although any $\mathbf{x}$ feasible for the NLP problem is also feasible for its LP approximation, the converse is not true. To deal with this difficulty we introduce the concept of *ε-feasibility*, similar to that introduced by Palacios-Gomez (1980) for NLP problems, but applied to mixed-discrete points.

Definition 11. The sum of infeasibilities (SUMINF) of a point $\mathbf{x_1}$ is defined as $\text{SUMINF}(\mathbf{x_1}) = \Sigma g_j(x_1)$, $g_j > 0$. A point $\mathbf{x_1}$ is said to be *ε-feasible* for a MDNLP model, if $\mathbf{x_1}$ satisfies the discreteness requirement and $\text{SUMINF}(\mathbf{x_1}) \leq \varepsilon$.

The parameter ε essentially enlarges the feasible domain of a NLP model by an amount ε for each constraint. For a given ε, we accept a new point $\mathbf{x_{new}}$ as better than the incumbent $\mathbf{x_{old}}$ under one of two conditions:

(1) $\text{SUMINF}(\mathbf{x_{new}}) < \text{SUMINF}(\mathbf{x_{old}})$, if $\mathbf{x_{old}}$ is ε-infeasible, or

(2) $f(\mathbf{x_{new}}) < f(\mathbf{x_{old}})$, if $\mathbf{x_{old}}$ is ε-feasible and $\mathbf{x_{new}}$ is ε-feasible. (25)

This acceptance rule mandates that for a given ε, a new point is accepted if it improves ε-feasibility of an ε-infeasible incumbent, or if it strictly improves the objective without sacrificing ε-feasibility. For each ε value, there is only a finite number of discrete points that yield successively better objective function values. As ε is progressively tightened (ε—>0), points are forced to be feasible for the MDNLP model and those intermediate points that are LP-feasible but MDNLP-infeasible are rejected. For each ε value, the algorithm descends by virtue of the linearization process with decreasing step bounds, and if we tighten ε gradually, we ensure we will not miss intermediate points that may be part of a new linearization. Thus we have shown

Theorem 5. The results of Theorem 4 apply to Algorithm SLP2, modified by the ε-feasibility acceptance rules of Eq.(24).

Incorporation of ε-feasibility into Algorithm SLP2 and other implementation details are further described in the sequel article (Loh & Papalambros, 1989) where an algorithm called MINSLIP is described.

Pitfalls of the SLP Approach

In the present section we consider some situations where the SLP approach will fail to converge to the global minimizer.

NON-PSEUDOCONVEX OBJECTIVE

When f is not pseudoconvex, the algorithm can converge to a point $\mathbf{x}_1$ whereby $-\nabla f(\mathbf{x}_1)$ points in a direction opposite to where the optimizer $\mathbf{x}_*$ may be.

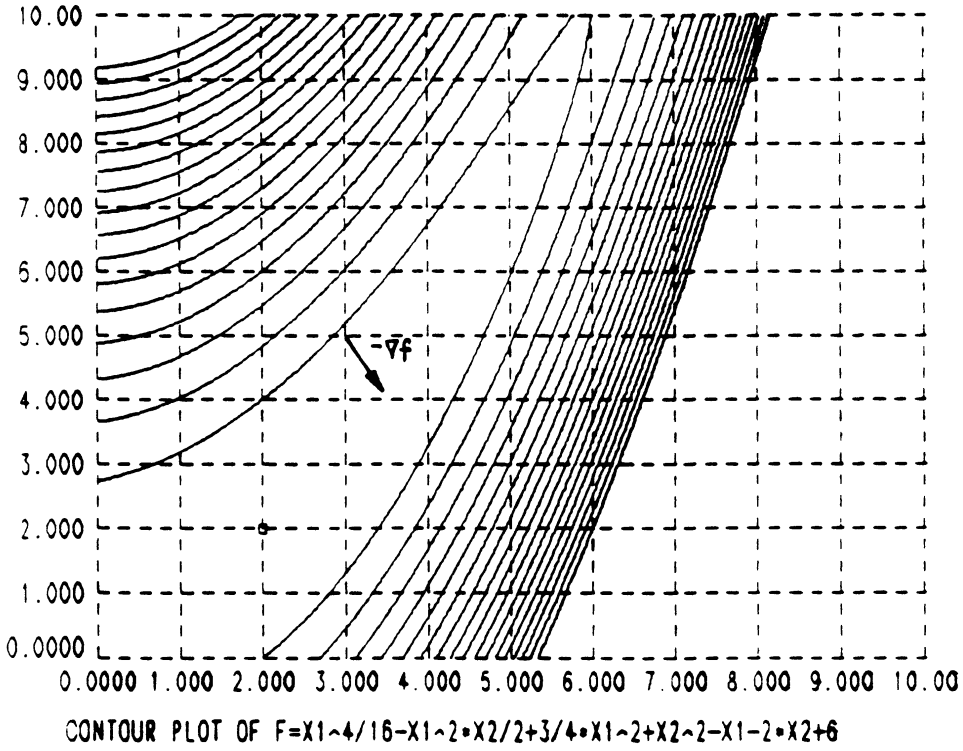


Figure 8. A nonconvex f with the downhill gradient pointing away from the global discrete minimizer

Example 6: Consider the problem (see Figure 8)

$$\text{minimize} \quad f = 0.0625x_1^4 - 0.5x_1^2x_2 + 0.75x_1^2 + x_2^2 - x_1 - 2x_2 + 6 \qquad (26)$$

The gradient of the objective is given by

$$\nabla f = (0.25x_1^3 - x_1x_2 + 1.5x_1 + x_2^2 - 1, -0.5x_1^2 + 2x_2 - 2)^T$$

At $(3, 5)^T$, $\nabla f = (-4.25, 3.5)^T$. Linearizing at $(3, 5)^T$, the linear subproblem is $\{\min -4.25x_1 + 3.5x_2\}$. A search along the $-\nabla f$ direction will not find the optimizer $(2, 2)^T$ as the vector $-\nabla f$ is pointing in the wrong direction. The algorithm will wrongly conclude that the function cannot be improved further from point $(3, 5)^T$. □

NONCONVEX CONSTRAINTS

When a constraint is not convex, its linearization is not an outer approximation and part of the NLP feasible region can be cut off by the linearization. See Figure 9.

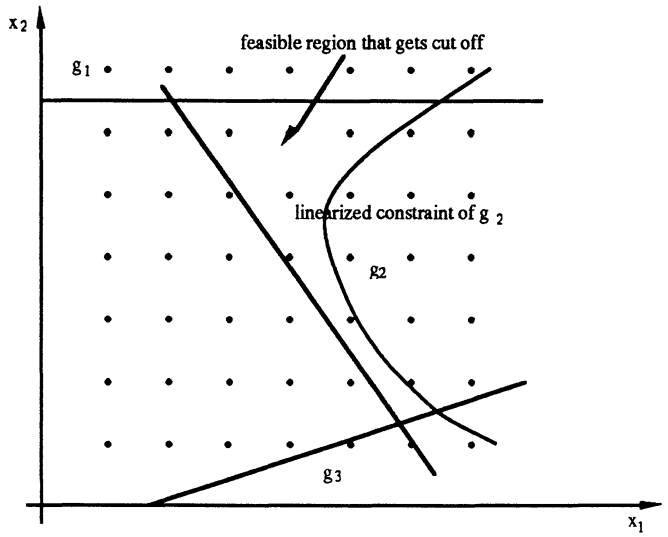


Figure 9. Cutting off feasible regions when g is not convex.

NONLINEAR EQUALITY CONSTRAINTS

The inclusion of nonlinear equality constraints in general turns the problem into a nonconvex one. In principle, equality constraints can be eliminated by variable elimination and substitution but in practise this may be hard to do. The requirement that the variables take only discrete values makes it even harder to solve for any nonlinear equality. In a SLP approach, the linearization may lead to a situation where there is no hope of finding any other discrete answer, as in Figure 10.

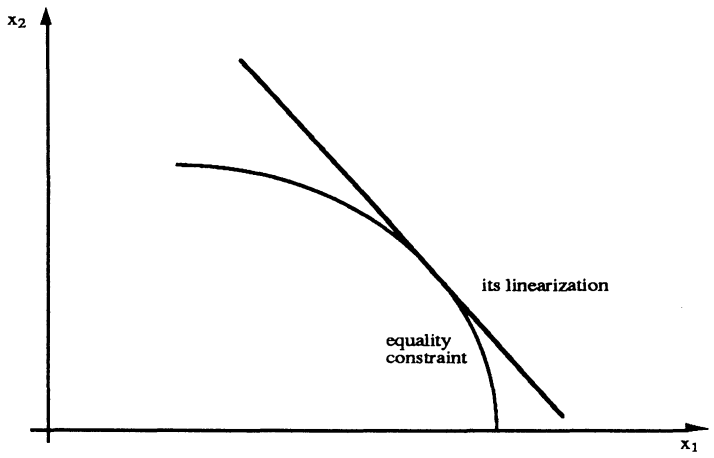


Figure 10. A nonlinear equality constraint and its linearization.

Monotonicity and Mixed-Discrete Variables

In this section we will consider some techniques that may assist in identifying a global minimizer or prove that a point cannot be a global minimizer. We will consider optimality conditions for discrete and mixed-discrete variables for problems that have objective and constraint functions displaying monotonicity properties. Such monotonicity analysis has been developed for problems with continuous variables (see, e.g., Papalambros & Wilde, 1988). We shall briefly examine the extension of monotonicity principles to purely discrete and mixed-discrete problems.

We start with some definitions.

Definition 12. A (differentiable) function f is said to be *monotonically increasing (decreasing)* with respect to a continuous variable x_i, if $(\partial f/\partial x_i) \geq 0$ (≤ 0) for the range of x_i applicable (Papalambros & Wilde, 1988).

Definition 13. A function f is said to be *monotonically increasing (decreasing)* with respect to a discrete variable x_i , if given points $\mathbf{x}' = (x_1,..., x_{i-1}, x_i', x_{i+1},...,x_n)^T$ and $\mathbf{x}'' = (x_1,..., x_{i-1}, x_i'', x_{i+1},...,x_n)^T$ for each $x_i', x_i'' \in \mathcal{D}(x_i)$, we have

$$x_i' > x_i'' \Longrightarrow f(\mathbf{x}') \geq (\leq) f(\mathbf{x}'') \qquad (27)$$

Strictly monotonic functions are defined with strict inequality.

Definition 14. An inequality constraint is said to be *active* (binding) at the optimizer, if removing it from the model changes the optimum of the relaxed problem ("relaxed" here meaning that the constraint is omitted [Papalambros & Wilde, 1988]).

Definition 15. An inequality constraint is said to be *tight* at a point, if it is satisfied as an equality at that point.

For continuous problems, constraints that are active are also tight at the optimum and constraints that are tight at the optimum are also active, if there is no redundant or degenerate constraint at the optimum. For discrete problems, the situation is more complicated. Inequalities are generally not satisfied as equalities at the optimizer. There may be more than one constraint acting on each discrete variable. For example, in Figure 11a, either constraint 1 or constraint 2 can be removed without changing the discrete optimizer, but not both. This situation does not occur in the case of the continuous problem, Figure 11b. If constraint 1 is removed, the optimizer will move from $\mathbf{x_2}$ to $\mathbf{x_3}$.

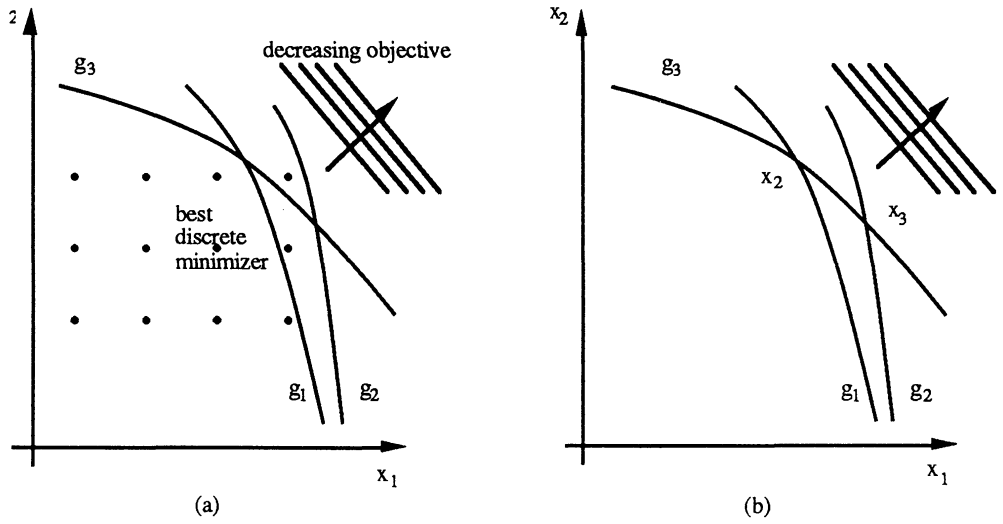


Figure 11. The occurrence of multiple active constraints for discrete problems.
(a) discrete domain; (b) continuous domain.

We will adopt the following definition for a "binding" constraint in the case of discrete variables:

Definition 16. A constraint g_i is *binding* with respect to a variable x_i at point $\mathbf{x}$, if $f(\mathbf{x}+\Delta\mathbf{x}_i) < f(\mathbf{x})$ and $g_j(\mathbf{x}+\Delta\mathbf{x}_i) \geq 0$, and g_j dominates all other constraints in the positive x_i direction, or $f(\mathbf{x}-\Delta\mathbf{x}_i) < f(\mathbf{x})$ and $g_j(\mathbf{x}-\Delta\mathbf{x}_i) \geq 0$ and g_j dominates all other constraints in the negative x_i direction.

(For discussion on constraint dominance see Papalambros & Wilde, op. cit.) The first monotonicity principle (MP1) now states that in a well-bounded problem, every increasing (decreasing) variable in the objective must be bounded below (above) by at least one binding constraint. The second monotonicity principle (MP2) states that every monotonic variable not occurring in the objective function of a well-bounded problem is either (a) irrelevant and can be deleted from the problem with all constraints in which it occurs; or (b) relevant and bounded by two binding constraints, one from above and one from below. With the change in the definition of a binding constraint, MP1 and MP2 remain as necessary optimality criteria for monotonic functions whether the variables are discrete or continuous.

Though these principles hold for continuous as well as discrete variables, they are more powerful for continuous variables than for discrete. variables. In the case of continuous variables, inequality constraints are satisfied as exact equalities at the optimizer

allowing problem reduction. For constraint-bound problems identifying all active constraints leads to solving only a system of nonlinear constraints. For purely discrete problems, the active constraints are not equalities at the optimizer but only pass through the discrete neighborhood of the optimizer. This means that the active constraint cannot be used to solve for the optimizer directly, but only to confirm the solution found by some other means.

For mixed-discrete problems, MP1 and MP2 must hold for the continuous variables. In this case, constraints binding for such variables must also be tight. We would expect then that, in an algorithm such as SLP2, an active set strategy may be used where constraints identified as active for the LP subproblem are taken as (locally) active for the NLP problem, even when the variables are non-monotonic. This is borne out numerically from a study of the test problems in Gupta (1980) and Cha (1987) that are mixed-discrete. The implication is that a local-global acive set strategy can be set up, similar to that in (Li, 1985).

Example 7. (Gupta, 1980; Test Problem 8) Consider the problem

$$\begin{aligned} &\text{minimize} && f = (x_1 - 3)^2 + (x_2 - 2)^2 + (x_3 + 4)^2 && x_1, x_2 \text{ integer} \\ &\text{subject to} && g_1 : \ 10 - x_1 - x_2^2 - x_3^{0.5} \le 0 \\ & && g_2 : -\frac{x_1^2}{4.166} + x_2 - \frac{x_3}{3.921} - 3 \le 0 && (28) \\ & && g_3 : \ 4x_1 - x_2^2 - x_3^{-3.5} - 12 \le 0 \end{aligned}$$

Linearize at $(3, 3, 0.527)^T$ obtaining the LP subproblem

$$\begin{aligned} &\text{minimize} && 2x_2 + 9.054x_3 && x_1, x_2 \text{ integer} \\ &\text{subject to} && -x_1 - 2x_2 - 0.6888x_3 \le -9.637 \\ & && -1.44x_1 + x_2 - 0.2550x_3 \le 0.8375 && (29) \\ & && 4x_1 - 6x_2 - 62.506x_3 \le -39.352 \end{aligned}$$

The solution to this LP is $(4, 3, 0.598)^T$. Only the third constraint is tight in this solution. With the discrete value of $(4, 3)^T$ and g_3 binding, we solve the continous problem obtaining $(4, 3, 0.631)^T$, which is the same minimizer given in the cited reference. □

Conclusion

The sequential linearization approach overcomes one of the main shortcomings of branch and bounb, namely the large number of node evaluations. This is confirmed by the numerical results reported in the sequel article. Here it was shown that if a sequence of mixed-discrete solutions, generated by successive linear approximations of the MDNLP problem, converge to a MDNLP-feasible point, then this point is the global discrete

minimizer. Introducing decreasing step bounds forces the linearization to find interior solutions. Algorithm SLP2 that incorporates this idea was shown to converge to the global discrete minimizer for problems having pseudoconvex objective functions and linear constraints, except in rare cases (Case 3 in Theorem 2). Introducing further the idea of ε-feasibility of a MDNLP problem, it was shown that progressively reducing the ε-feasibility, and rejecting points that exceed the allowable ε-feasibility, solutions generated by successive linearizations will likely converge for pseudoconvex objectives and convex constraints. Finally, the monotonicity principles were extended for problems with mixed-discrete variables, suggesting a correspondence between active constraints in a MDNLP model and active constraints in its linear approximations.

Acknowledgements

This research was partially supported by NSF Grants DMC 85-14721 and DMC 86-11916 at the University of Michigan. The first author was also supported by a Fellowship from the National University of Singapore. This support is gratefully acknowledged. The authors also express their thanks to John Birge for his insights during the course of this research.

References

Balas, E. 1965, "An Additive Algorithm for Solving Linear Programs with Zero-One Variables", *Operations Research*, Vol 13, pp517-526.

Bean, J. 1987, "Multiple Choice Knapsack Functions", *Technical Report 87-26*, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor.

Box, M.J. 1965, "A New Method of Constrained Optimization and a Comparison with Other Methods", *Computer Journal*, Vol 8, no. 1, pp42-52.

Bremicker, M., Loh, H.T., & Papalambros, P.Y., 1989, " Solution of Mixed-Discrete Structural Optimization Problems with a New Sequential Linearization Algorithm", *Computers and Structures* (submitted). Also, Report UM-MEAM-89-10, The University of Michigan, Ann Arbor.

Cha, J.Z. & Mayne, R.W. 1987, "Optimization with Discrete Variables via Recursive Quadratic Programming: Part I: Concepts and Definitions; Part II: Algorithm and Results", *Proceedings of the 1987 ASME Design Automation Conference,* Boston, Massachusetts, Vol 1, pp7-22.

Cha, J.Z. & Mayne, R.W. 1988, "The Symmetric Rank One Formula And Its Application in Discrete Nonlinear Optimization", *Proceedings of 1988 ASME Design Technology Conference*, Kissimmee, Florida, Sept 25th 1988, pp 95-105.

Cooper, L. & Cooper, M. 1975, "Nonlinear Integer Programming", *Computers And Mathematics with Applications*, Vol 1, pp215-222.

Dakin, R.J. 1965, "A Tree-Search Algorithm for Mixed Integer Programming Problems", *Computer Journal*, Vol 8, no. 3, pp 250-255.

Davydov, E.G. & Sigal, I. 1972, "Application of Penalty Function Method in Integer Programming Problems", *Engineering Cybernetics*, Vol 10, no. 1, pp21-24.

Duran, M.A. & Grossmann, I.E. 1986, "An Outer-Approximation Algorithm for a Class of Mixed-integer Nonlinear Programs", *Mathematical Programming*, Vol 36, pp307-339.

Duran, M.A. 1984, *A Mixed-integer Nonlinear Programming Approach for the Systematic Synthesis of Engineering Systems*, PhD Thesis, Carnegie-Mellon University.

Fox, D.B. & Liebman, J.S. 1981, "A Discrete Nonlinear Simplex Method for Optimized Engineering Design", *Engineering Optimization*, Vol 5, pp129-149.

Geoffrion, A.M. 1972, "Generalized Bender's Decomposition", *Journal of Optimization Theory and Applications*, Vol 10, no. 4, pp237-260.

Geoffrion, A.M. 1974, "Lagragian Relaxation For Integer Programming", *Mathematical Programming Study 2*, pp 82-114, North-Holland Publishing Company.

Gisvold, K.M. & Moe, J. 1972, " A Method for Nonlinear Mixed Integer Programming and Its Application to Design Problems", *Journal of Engineering for Industry*, Vol 94, pp 353-364.

Glover, F. & Sommers, D. 1975, "Pitfalls of Rounding in Discrete Management Decision Problems", *Decision Science* , Vol 22, pp 42-50.

Glover, F. 1968, "Surrogate Constraints", *Operations Research*, Vol 16, pp741-749.

Gomory, R.E. 1958,"An Algorithm for Integer Solutions to Linear Programs", *Princeton-IBM Mathematical Research Project, Technical Report 1.*

Gupta, O.K. & Ravindran, A. 1983, "Nonlinear Integer Programming and Discrete Optimization", *ASME Journal of Mechanism, Transmission and Automation in Design*, Vol 105, pp160-164.

Gupta, O.K.1980, *Branch and Bound Experiments in Nonlinear Integer Programming*, PhD Thesis, Purdue University.

Hammer, P.L. 1974, "Boolean Procedure for Bivalent Programming", ed P.L. Hammer and G . Zoutendijk: *Mathematical Programming in Theory and Applications*, North-Holland.

Hillier, F.S. & Liberman, G. J. 1974, *Introduction to Operations Research.*, Holden-Day, San Francisco.

John, K.V., Ramakrishnan, C.V., and Sharma, K.G., " Optimum Design of Trusses from Available Sections-Use of Sequential Linear Programming with Branch and Bound Algorithm", *Engineering Optimization*, Vol.13, pp 119-145.

Kocis, G.R. & Grossmann, I.E. 1987, "Relaxation Strategy for the Structural optimization of Process Flowsheets", *Ind. Eng. Chem. Research*, Vol 26, no 9, pp1869-1880.

Land, A.M. & Doig, A.G. 1960, "An Automatic Method of Solving Discrete Programming Problems", *Econometrica*, Vol 28, pp 497-520.

Lemke, C. & Spielberg, K. 1967, "Direct Search Algorithms for Zero-One and Mixed Integer Programming", *Operations Research*, Vol 15, pp892-914.

Li, H.L. 1985, *Design Optimization Strategies with Global and Local Knowledge*, PhD Thesis, The University of Michigan, Ann Arbor.

Loh, H.T., 1989, *A Sequential Linearization Approach for Mixed-Discrete Nonlinear Design Optimization*, Doctoral Dissertation, Dept. of Mechanical Engineering and Applied Mechanics, The University of Michigan, Ann Arbor.

Loh, H.T., & Papalambros, P.Y., 1989," Computational Implementation and Tests of a Sequential Linearization Algorithm for Mixed-Discrete Nonlinear Design Optimization", *Trans. ASME, J. of Mechanical Design* (in review). Also, Report UM-MEAM-89-09, The University of Michigan, Ann Arbor.

Lu, P. 1988, *Boolean Techniques in Discrete Optimization and Expert Systems*, PhD Thesis, McMaster University, Hamilton, Ontario.

Mitten, L.G. 1964, "Composition Principles for Synthesis of Multistage Processes", *Operations Research*, Vol 12, pp610-619.

Nelder, J.A. & Mead, R. 1965, "A Simplex Method for Function Minimization ", *Computer Journal*, Vol 7, no. 4, pp308-313.

Palacios-Gomez, F. 1980, *The Solution of Nonlinear Optimization problems using Successive Linear Programming: Theoretical and Computational Results*, Phd Thesis, University of Texas, Austin.

Papadimitriou, C.H. & Steiglitz, K. 1982, *Combinatorial Optimization*, Prentice-Hall, New Jersey.

Papalambros, P. & Wilde, D. J. 1988, *Principles of Optimal Design: Modelling and Computation,* Cambridge University Press, Cambridge.

Ponstein, J. 1967, "Seven Kinds of Convexity", *SIAM Review*, vol 9, pp115-119.

Reiter, S. & Rice, D.B. 1966, "Discrete Optimizing Solution Procedures for Linear and Nonlinear Integer Programming Problems", *Management Science*, vol 12, no. 11, pp 829-850.

Ringertz, U. T. 1988, "On Methods for Discrete Structural Optimization", *Engineering Optimization*, Vol 13, pp47-64.

Salkin, H.M. 1975, *Integer Programming*, Addison-Wesley.

Sandgren, E. 1988, "Nonlinear Integer and Discrete Programming in Mechanical Design", *Proceedings of 1988 ASME Design Technology Conference*, Kissimmee, Florida, Sept 25th 1988, pp 95-105.

Schmit, L.A. & Fleury, C. 1980, "Discrete-Continuous Variable Structural Synthesis Using Dual Methods", *AIAA Journal,* Vol 18, no. 12, pp1515-1524.

Sinha, P. & Zoltners, A. 1979, "The Multiple Choice Knapsack Problem", *Operations Research*, Vol 27, no.3, pp 503-515.

Tzannetakis, N. & Papalambros, P. 1987, "An Active Set Sequential Linearization Algorithm for Nonlinear Design Optimization", *Proceedings of the 1987ASME Design Automation Conference,* Boston, Massachusetts, Vol 1, pp1-6.

Viswanathan, J. & Grossman, I.E. 1989, "A Combined Penalty Function and Outer-Approximation Method for MINLP Optimization", presented at the CORS/TIM/ORSA Meeting, Vancouver, May 1989.