EXTENDING MODELING SYSTEMS: STRUCTURE AND SOLUTION

Michael C. Ferris
Computer Sciences Department, University of Wisconsin, Madison, WI 53706

Steven P. Dirkse
GAMS Corporation, 1217 Potomac Street, Washington, DC 20007

Jan Jagla
GAMS Software GmbH, Eupener Str. 135–137, 50933 Cologne, Germany

Alexander Meeraus
GAMS Corporation, 1217 Potomac Street, Washington, DC 2007

Abstract

Some extensions of a modeling system are described that facilitate higher level structure identification within a model. These structures, which often involve complementarity relationships, can be exploited by modern large scale mathematical programming algorithms. A specific implementation of an extended mathematical programming tool is outlined that communicates structure in a computationally beneficial manner from the GAMS modeling system to an appropriate solver.

Keywords

Modeling, complementarity, nonlinear programming, variational inequalities

Introduction

Chemical engineering has been a user of optimization techniques and technology for several decades. Many chemical engineers have made significant contributions to the field of optimization, and in several cases have developed software that has had impact far beyond the chemical engineering discipline. Packages such as DICOPT (Duran and Grossman, 1986), BARON (Tawarmalani and Sahinidis, 2004), ALPHAECP (Westerlund et al., 1998) and IPOPT (Wächter and Biegler, 2006) have influenced the debate on tractability of hard, practical nonlinear optimization problems, setting the gold standard for the solution of nonconvex, global optimization problems.

Accessing these solvers, and many of the other algorithms that have been developed over the past three decades has been made easier by the advent of modeling languages. A modeling language (Bisschop and Meeraus, 1982, Fourer et al., 1990) provides a natural, convenient way to represent mathematical programs. They typically have efficient procedures to handle vast amounts of data, take advantage of the numerous options for solvers and model types, and can quickly generate a large number of models. For this reason, modeling languages are heavily used in practical applications. Although, we will use GAMS (Brooke et al., 1988), the system we are intimately familiar with, most what will be said could as well be applied to other algebra based modeling systems like AIMMS, AMPL, MOSEL, MPL and OPL.

We believe that further advancements in applications of optimization can be achieved via identification of specific problem structures within a model. In some cases, such structures can be automatically extracted from a model that is formulated in a modeling system, but in general it may be difficult to tease out particular structures from a large complex model. As a concrete example, we cite the domain of applied general equilibrium modeling within economics, where a model consists of a collection of interacting agents each of which maximizes some utility function of some demanded goods, which in turn are generated from an optimization of production functions. Since current solvers require the (economic) modeler to formulate the problem as a complementarity problem or a system of nonlinear equations, the utility function is maximized (by hand) to derive demands which are normally rather complex nonlinear functions; similarly for the supplies that are derived from a cost minimization involving the production functions. It is typically difficult to determine the structure of the underlying model from the resulting nonlinear complementarity system. This paper aims to alleviate this difficulty.

The purpose of our work is to extend the classical nonlinear program from the traditional model:

$$\min_x f(x) \text{ s.t. } g(x) \le 0, h(x) = 0, \tag{1}$$

where f, g and h are assumed sufficiently smooth, to a more general format that allows new constraint types to be specified precisely. Some extensions of this format have already been incorporated into modeling systems. There is support for integer, sos, and semicontinuous variables, and some limited support for logical constructs. GAMS (Ferris and Munson, 2000), AMPL (Ferris et al., 1999) and AIMMS have support for complementarity constraints, and there are some extensions that allow the formulation of second-order cone programs within GAMS. AMPL (Fourer et al., 1993) has facilities to model piecewise linear functions. Much of this development is tailored to specific structures. We aim to develop more general annotation schemes to allow extended mathematical programs to be written clearly and succinctly.

In the following sections we outline some new features of GAMS that we refer to as extended mathematical programs (EMP), that incorporate many of the extensions menitoned above but also allows a variety of other structures to be described at the modeling level. We believe such extensions may have benefits on several levels. Firstly, we think this will make the modelers task easier, in that the model can be described more naturally and perhaps at a higher level. (Of course, there are several examples of this already in the literature including the use specialized languages such as MPSGE (Rutherford, 1999) to facilitate general equilibrium models, and specialized (graphical) interfaces to allow queueing system or process system design.) Secondly, the automatic generation of the model can be done more reliably based on techniques such as automatic differentiation and problem reformulation - duality constructs, or specific ways to write certain constraints. Thirdly, if an algorithm is given additional structure, it may be able to exploit that in an effective computational manner; knowing the problem is a cone program, or the problem involves the optimality conditions of a nonlinear program can be treated in a variety of different ways, some of which may be distinctly superior to others in certain settings. Indeed, the availability of such structure to a solver may well foster the generation of new features to existing solvers or drive the development of new classes of algorithms.

Extended Mathematical Programs

Complementarity Problems

The necessary and sufficient optimality conditions for the linear program

$$\begin{aligned} &\min_x c^T x \\ &\text{s.t. } Ax \ge b,\ x \ge 0 \end{aligned} \tag{2}$$

are that x and some λ satisfy

$$\begin{aligned} 0 \le c - A^T\lambda \quad &\perp \quad x \ge 0 \\ 0 \le Ax - b \quad &\perp \quad \lambda \ge 0. \end{aligned} \tag{3}$$

Here, the "$\perp$" sign signifies (for example) that in addition to the constraints $0 \le Ax - b$ and $\lambda \ge 0$, each of the products $(Ax - b)_i \lambda_i$ is constrained to be zero. An equivalent viewpoint is that either $(Ax - b)_i = 0$ or $\lambda_i = 0$. Within GAMS, these constraints can be modeled simply as

```
positive variables lambda, x;
model complp / defd.x, defp.lambda /;
```

where defp and defd are the equations that define primal and dual feasibility ($Ax \ge b$, $c \ge A^T\lambda$) respectively.

Other linear programs with specialized constraint structure are just as easy to specify. For example

$$\begin{aligned} &\min_x c^T x \\ &\text{s.t. } Ax = b,\ x \in [l, u] \end{aligned}$$

has similarly expressed optimality conditions:

$$\begin{aligned} &0 \le (c - A^T\lambda)_j \quad \text{if } x_j = l_j \\ &0 = (c - A^T\lambda)_j \quad \text{if } l_j < x_j < u_j \\ &0 \ge (c - A^T\lambda)_j \quad \text{if } x_j = u_j \\ &0 = Ax - b \qquad \perp \quad \lambda \text{ free.} \end{aligned} \tag{4}$$

Note that the first three complementarity relationships in (4) can be written more succinctly as $(c - A^T\lambda) \perp x \in [l, u]$. This is translated into GAMS syntax as follows:

```
variables lambda, x;
x.lo(i) = l(i); x.up(i) = u(i);
model complp / defd.x, defp.lambda /;
```

Such a problem is an instance of a linear mixed complementarity problem, for which we use the acronym MCP. Note that the bounds on the variables x determine the nature of the relationship on $c - A^T\lambda$ at the solution. (It is possible to introduce explicit multipliers on the constraints $x \geq l$ and $x \leq u$, and to rewrite the optimality conditions in terms of x, λ, and these multipliers. The "$\perp$" notation enables us to write these relationships much more succinctly.)

Complementarity problems do not have to arise as the optimality conditions of a linear program; the optimality conditions of the nonlinear programs (1) are the following MCP:

$$\begin{aligned} 0 &= \nabla f(x) + \lambda^T \nabla g(x) + \mu^T \nabla h(x) && \perp \quad x \text{ free} \\ 0 &\leq -g(x) && \perp \quad \lambda \geq 0 \\ 0 &= -h(x) && \perp \quad \mu \text{ free.} \end{aligned} \tag{5}$$

Many examples are no longer simply the optimality conditions of an optimization problem. A specific example arises in chemical phase equilibrium. In this setting, different conditions are satisfied at an equilibrium depending on whether we are in vapor, liquid or two-phase state. Letting α represent the fraction in vapor, the problem is to find $f(\alpha) \perp \alpha \in [0,1]$ where

$$f(\alpha) = \sum_i (x_i - K_i x_i),\ x_i = \frac{z_i}{K_i\alpha + 1 - \alpha},$$

for given data K_i and z_i. Ferris and Pang (1997) catalogue a number of other applications both in engineering and economics that can be written in a similar format.

It should be noted that robust large scale solvers exist for such problems; see for example Ferris and Munson (2000), where a description is given of the PATH solver.

Mathematical Programs with Complementarity Constraints

A mathematical program with complementarity constraints is the following:

$$\min_{x\in\mathbf{R}^n, y\in\mathbf{R}^m} f(x,y) \tag{6}$$

$$\text{s.t. } g(x,y) \leq 0 \tag{7}$$

$$0 \leq y \ \perp\ h(x,y) \geq 0. \tag{8}$$

The objective function (6) needs no further description, except to state that the solution techniques we are intending to apply require that f (g and h) are at least once differentiable, and for some solvers twice differentiable.

The constraints (7) are intended to represent standard nonlinear programming constraints. Clearly, these could involve equalities with a slight increase in exposition complexity.

The constraints that are of interest here are the complementarity constraints (8). Essentially, these are parametric constraints (parameterized by x) on the variable y, and encode the structure that y is a solution to the nonlinear complementarity problem defined by $h(x,\cdot)$. Within the GAMS modeling system, this can be written simply and directly as:

```
model mpecmod / deff, defg, defh.y /;
option mpec=nlpec;
solve mpecmod using mpec minimizing obj;
```

Here it is assumed that the objective (6) is defined in the equation deff, the general constraints (7) are defined in defg and the function h is described by defh. The complementarity relationship is defined by the bounds on y and the orthogonality relationship shown in the model declaration using ".". AMPL provides a slightly different but equivalent syntax for this, see Ferris et al. (1999). The problem is frequently called a mathematical program with complementarity constraints (MPCC).

Some solvers can process complementarity constraints explicitly. In many cases, this is achieved by a reformulation of the constraints (8) into the classical form given within (1). Dirkse and Ferris (2008) outlines a variety of ways to carry this out, all of which have been encoded in a solver package called NLPEC. While there are large numbers of different reformulations possible, the following parametric approach, coupled with the use of the nonlinear programming solver CONOPT or SNOPT, has proven effective in a large number of applications:

$$\begin{aligned} \min_{x\in\mathbf{R}^n, y\in\mathbf{R}^m, s\in\mathbf{R}^m} & f(x,y) \\ \text{s.t. } & g(x,y) \leq 0 \\ & s = h(x,y) \\ & y \geq 0, s \geq 0 \\ & y_i s_i \leq \mu, 1 = 1, \ldots, m. \end{aligned}$$

Note that a series of approximate problems are produced, parameterized by $\mu > 0$; each of these approximate problems have stronger theoretical properties than the problem with $\mu = 0$ (Ralph and Wright, 2004). A solution procedure whereby μ is successively reduced can be implemented as a simple option file to NLPEC, and this has proven remarkably effective. Further details can be found in the NLPEC documentation (Dirkse and Ferris, 2008). The approach has been used to effectively optimize the rig in a sailboat design (Wallace et al., 2006).

It is also possible to generalize the above complementarity condition to a mixed complementarity condition; details can be found in Ferris et al. (2005). Underlying the NLPEC "solver package" is an automatic conversion of the original problem into a standard nonlinear program which is carried out at the scalar model level. The technology to perform this conversion forms the core of the codes that we use to implement the model extensions of the sequel.

Variational Inequalities

A variational inequality $\mathrm{VI}(F, X)$ is to find $x \in X$:

$$F(x)^T(z - x) \geq 0, \text{ for all } z \in X.$$

Here X is a closed (frequently assumed convex) set, defined for example as $X = \{x \mid x \geq 0, g(x) \leq 0\}$. Note that the first-order (minimum principle) conditions of a nonlinear program

$$\min_{z \in X} f(z)$$

are precisely of this form with $F(x) = \nabla f(x)$. For concreteness, these conditions are necessary and sufficient for optimality of the linear programming. Solving the linear program (2) is equivalent to solving the variational inequality given by

$$F(x) = c, \quad X = \{x \mid Ax \geq b, x \geq 0\}. \tag{9}$$

In this case, F is simply a constant function. While there are a large number of instances of the problem that arise from optimization applications, there are many cases where F is not the gradient of any function f. For example, asymmetric traffic equilibrium problems have this format, where the asymmetry arises for example due to different costs associated with left or right hand turns. A complete treatment of the theory and algorithms in this domain can be found in Facchinei and Pang (2003).

Variational inequalities are intimately connected with the concept of a normal cone to a set S, for which a number of authors have provided a rich calculus. Instead of overloading a reader with more notation, however, we simply refer to the seminal work in this area, Rockafellar and Wets (1998).

Such a problem can be reformulated as a complementarity problem by introducing multipliers λ on the constraints g:

$$\begin{aligned} 0 \leq F(x) + \lambda^T \nabla g(x) \quad &\perp \quad x \geq 0 \\ 0 \leq -g(x) \quad &\perp \quad \lambda \geq 0. \end{aligned}$$

If X has a different representation, this construction would be modified appropriately. In the linear programming example above (9), these conditions are precisely those given as (3).

When X is the nonnegative orthant, the VI is just an alternative way to state a complementarity problem. However, when X is a more general set, it may be possible to treat it differently than simply introducing multipliers, see for example Cao and Ferris (1996). For example when X is a polyhedral set, algorithms may wish to generate iterates via projection onto X.

A simple example two dimensional may be useful. Let

$$F(x) = \begin{bmatrix} x_1 + 2 \\ x_1 + x_2 - 3 \end{bmatrix}, \; X = \{x \geq 0 \mid x_1 + x_2 \leq 1\},$$

so that F is an affine function, but F is not the gradient of any function $f : \mathbf{R}^2 \to \mathbf{R}$. For this particular data, $\mathrm{VI}(F, X)$ has a unique solution $x = (0, 1)$. Such a variational inequality can be described in GAMS via the model statement

```
model vi / deff, defg /;
```

combined with an annotation file that indicates certain equations are to be treated differently by the EMP tool. In this case, the "empinfo" file states that the model equations deff define a function F that is to be part of a variational inequality, while the equations defg define constraints on X. Details on specific syntax can be found in Jagla et al. (2008).

Mathematical Programs with Equilibrium Constraints

Mathematical programs with equilibrium constraints are a generalization of the aforementioned MPCC problem class. The difference is that the lower level problem, instead of being a complementarity problem, is now a variational inequality. Coupling the two approaches mentioned above, results in the ability to model and solve such problems.

Bilevel Programs

Heirarchical optimization has recently become important for a number of different applications. New codes are being developed that exploit this structure, at least for simple heirarchies, and attempt to define and implement algorithms for their solution.

The simplest case is that of bilevel programming, where an upper level problem depends on the solution of a lower level optimization. For example:

$$\begin{aligned}
&\min_{x,y} f(x,y)\\
&\text{s.t. } g(x,y) \le 0,\\
&\qquad y \text{ solves } \min_{s} v(x,s)\\
&\qquad\qquad\qquad \text{s.t. } h(x,s) \le 0.
\end{aligned}$$

This problem can be reformulated as an MPEC by replacing the lower level optimization problem by its optimality conditions:

$$\begin{aligned}
&\min_{x,y} f(x,y)\\
&\text{s.t. } g(x,y) \le 0,\\
&\qquad y \text{ solves } \mathrm{VI}(\nabla_s v(x,s), X),
\end{aligned}$$

where $X = \{s \mid h(x,s) \le 0\}$.

This approach then allows such problems to be solved using the NLPEC code, for example. However, there are several possible deficiencies that should be noted. Firstly, the optimality conditions encompassed in the VI may not have a solution, or the solution may only be necessary (and not sufficient) for optimality. Secondly, the MPEC solver may only find local solutions to the problem. The quest for practical optimality conditions and robust global solvers remains an active area of research. Importantly, the EMP tool will provide the underlying structure of the model to a solver if these advances determine appropriate ways to exploit this.

We can model this bilevel program in GAMS by

```
model bilev /deff,defg,defv,defh/;
```

along with some extra annotations to a subset of the model defining equations. Specifically, within an "empinfo" file we state that the bilevel problem involves the objective v which is to be minimized subject to the constraints specified in defh. The specific syntax is described in Jagla et al. (2008). A point that has been glossed over here but which is described carefully in the user manual is the process whereby multiple lower level problems are specified.

Extended Nonlinear Programs

Optimization models have traditionally been of the form (1). Specialized codes have allowed certain problem structures to be exploited algorithmically, for example simple bounds on variables.

In a series of papers, Rockafellar and colleagues have introduced the notion of extended nonlinear programming, where the (primal) problem has the form:

$$\min_{x \in X} f_0(x) + \theta(f_1(x), \ldots, f_m(x)). \tag{10}$$

In this setting, X is assumed to be a nonempty polyhedral set, and the functions $f_0, f_1, \ldots, f_m$ are smooth. The function θ can be thought of as a generalized penalty function. Specifically, when θ has the following form

$$\theta(u) = \sup_{y \in Y}\{y^T u - k(y)\}, \tag{11}$$

we obtain a computationally exploitable and theoretically powerful framework. A key point for computation and modeling is that the function θ can be fully described by defining the set Y and the function k. Furthermore, as we show below, different choices lead to a rich variety of functions θ, many of which are extremely useful for modeling. In the above setting θ can take on the value of ∞ and may well be nonsmooth, but it is guaranteed to be convex (proper and lower semicontinuous when $Y \neq \emptyset$ and k is smooth and convex).

Furthermore, from a modeling perspective, an extended nonlinear program can be specified simply by defining the functions $f_0, f_1, \ldots, f_m$ as is done already, with the additional issue of simply defining Y and k. Conceptually, this is not much harder that what is done already, but leads to significant enhancements to the types of models that are available. This paper outlines an approach to do this within the GAMS modeling system for a number of different choices of Y and k.

The tool we provide works in this setting by providing a library of functions θ that specify a library of choices for k and Y. Once a modeler determines which constraints are treated via which choice of k and Y, the tool automatically forms an equivalent variational inequality or complementarity problem. As we show later, there may be alternative formulations that are computationally more appealing; such reformulations can be generated using different options to our tool.

Forms of θ

The EMP tool makes this problem format available to users in GAMS. As special cases, we can model piecewise linear penalties, least squares and L_1 approximation problems, as well as the notion of soft and hard constraints. We allow modelers to utilize cone constraints and pass on the underlying geometric structure to solvers. Particular examples show enormous promise both from a modeling and solution perspective.

For ease of exposition, we now describe a subset of the types of functions θ that can be generated by particular choices of Y and k. In many cases, the function θ is separable, that is

$$\theta(u) = \sum_{i=1}^{m} \theta_i(u_i).$$

so we can either specify θ_i or θ itself.

Extended nonlinear programs include the classical nonlinear programming form (1) as a special case. This follows from the observation that if K is a closed convex cone, and we let ψ_K denote the "indicator function" of K defined by:

$$\psi_K(u) = \begin{cases} 0 & \text{if } u \in K \\ \infty & \text{else,} \end{cases}$$

then (1) can be rewritten as:

$$\min_x f(x) + \psi_K((g(x), h(x)), \; K = \mathbf{R}^m_- \times \{0\}^p,$$

where m and p are the dimensions of g and h respectively and $\mathbf{R}^m_- = \{u \in \mathbf{R}^m \mid u \leq 0\}$. An elementary calculation shows that

$$\psi_K(u) = \sup_{v \in K^\circ} u^T v,$$

where $K^\circ = \left\{u \mid u^T v \leq 0, \forall v \in K\right\}$ is the polar cone of the given cone K. Thus, when $\theta(u) = \psi_K(u)$ we simply take

$$k \equiv 0 \text{ and } Y = K^\circ. \tag{12}$$

In our example, $K^\circ = \mathbf{R}^m_+ \times \mathbf{R}^p$. To some extent, this is just a formalism that allows us to claim the classical case as a specialization; however when we take the cone K to be more general than the polyhedral cone used above, we can generate conic programs (see below) for example.

The second example involves a piecewise linear function θ: Formally, for $u \in \mathbf{R}$,

$$\theta(u) = \begin{cases} \rho u & \text{if } u \geq 0 \\ \sigma u & \text{else.} \end{cases}$$

In this case, simple calculations prove that θ has the form (11) for the choices:

$$k \equiv 0 \text{ and } Y = [\sigma, \rho].$$

The special case where $\sigma = -\rho$ results in

$$\theta(u) = \rho \left|u\right|.$$

This allows us to model nonsmooth L_1 approximation problems. Another special case results from the choice of $\sigma = 0$, whereby

$$\theta(u) = \rho \max\{u, 0\}.$$

This formulation corresponds to a soft penalization on an inequality constraint, namely if $\theta(f_1(x))$ is used then nothing is added to the objective function if $f_1(x) \leq 0$, but $\rho f_1(x)$ is added if the constraint $f_i(x) \leq 0$ is violated. Contrast this to the classical setting above, where ∞ is added to the objective if the inequality constraint is violated. It is interesting to see that truncating the set Y, which amounts to bounding the multipliers, results in replacing the classical constraint by a linearized penalty.

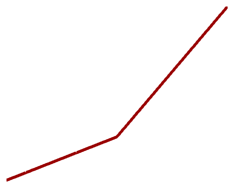
The third example involves a more interesting choice of k. If we wish to replace the "absolute value" penalization given above by a quadratic penalization (as in classical least squares analysis), that is

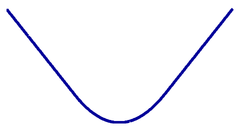
$$\theta(u) = \gamma u^2$$

then a simple calculation shows that we should take

$$k(y) = \frac{1}{4\gamma} y^2 \text{ and } Y = \mathbf{R}.$$

By simply specifying this different choice of k and Y we can generate such models easily and quickly within the modeling system: note however that the reformulation we would use in these two cases are very different as we shall explain in the simple example below. Furthermore, in many applications it has become popular to penalize violations using a quadratic penalty only within a certain interval, afterwards switching to a linear penalty (chosen to make the penalty fucntion θ continuously differentiable Huber (1981). That is:





$$\text{i.e. } \theta(u) = \begin{cases} \gamma u - \frac{1}{2}\gamma^2 & \text{if } u \geq \gamma \\ \frac{1}{2}u^2 & \text{if } u \in [-\gamma, \gamma] \\ -\gamma u - \frac{1}{2}\gamma^2 & \text{else.} \end{cases}$$

Such functions arise from quadratic k and simple bound sets Y. In particular, the function

$$\theta(u) = \begin{cases} \gamma\beta^2 + \rho(u - \beta) & \text{if } u \geq \beta \\ \gamma u^2 & \text{if } u \in [\alpha, \beta] \\ \gamma\alpha^2 + \sigma(u - \alpha) & \text{else} \end{cases}$$

arises from the choice of

$$k(y) = \frac{1}{4\gamma}y^2 \text{ and } Y = [\sigma, \rho],$$

with $\alpha = \frac{\sigma}{2\gamma}$ and $\beta = \frac{\rho}{2\gamma}$.

The final example that we give is that of L_∞ penalization. This example is different to the examples given above in that θ is not separable. However, straightforward calculation can be used to show

$$\theta(u) = \max_{i=1,\ldots,m} u_i$$

results from the choice of

$$k \equiv 0 \text{ and } Y = \left\{ y \in \mathbf{R}^m \mid y \geq 0, \sum_{i=1}^{m} y_i = 1 \right\},$$

that is Y is the unit simplex.

Underlying theory

The underlying structure of θ leads to a set of extended optimality conditions and an elegant duality theory. This is based on an extended form of the Lagrangian:

$$\mathcal{L}(x, y) = f_0(x) + \sum_{i=1}^{m} y_i f_i(x) - k(y)$$
$$x \in X, y \in Y$$

Note that the Lagrangian $\mathcal{L}$ is smooth - all the nonsmoothness is captured in the θ function. The theory is an elegant combination of calculus arguments related to f_i and its derivatives, and variational analysis for features related to θ.

It is shown in Rockafellar (1993) that under a standard constraint qualification, the first-order conditions of (10) are precisely in the form of the following variational inequality:

$$\text{VI}\left(\begin{bmatrix} \nabla_x \mathcal{L}(x, y) \\ -\nabla_y \mathcal{L}(x, y) \end{bmatrix}, X \times Y \right). \tag{13}$$

When X and Y are simple bound sets, this is simply a complementarity problem.

Note that EMP exploits this result. In particular, if an extended nonlinear program of the form (10) is given to EMP, then the optimality conditions (13) are formed as a variational inequality problem and can be processed as outlined above. For a specific example, we cite the fact that if we use the (classical) choice of k and Y given in (12), then the optimality conditions of (10) are precisely the standard complementarity problem given as (5). While this is of interest, we believe that other choices of k and Y may be more useful and lead to models that have more practical significance.

Under appropriate convexity assumptions on this Lagrangian, it can be shown that a solution of the VI (13) is a saddle point for the Lagrangian on $X \times Y$. Furthermore, in this setting, the saddle point generates solutions to the primal problem (10) and its dual problem:

$$\max_{y \in Y} g(y), \quad \text{where } g(y) = \inf_{x \in X} \mathcal{L}(x, y),$$

with no duality gap.

A simple example

As an example, consider the problem

$$\begin{aligned} \min_{x_1, x_2, x_3} \quad & \exp(x_1) + 5\left\|\log(x_1) - 1\right\|^2 + 2\max(x_2^2 - 2, 0) \\ \text{s.t.} \quad & x_1/x_2 = \log(x_3), \\ & 3x_1 + x_2 \leq 5, x_1 \geq 0, x_2 \geq 0. \end{aligned}$$

In this problem, we would take

$$X = \left\{ x \in \mathbf{R}^3 \mid 3x_1 + x_2 \leq 5, x_1 \geq 0, x_2 \geq 0 \right\}.$$

The function θ essentially treats 3 separable pieces:

$$\begin{aligned} f_1(x) &= \log(x_1) - 1, \\ f_2(x) &= x_2^2 - 2, \\ f_3(x) &= x_1/x_2 - \log(x_3). \end{aligned}$$

A classical problem would force $f_1(x) = 0$, $f_2(x) \leq 0$ and $f_3(x) = 0$, while minimizing $f_0(x) = \exp(x_1)$. In our problem, we still force $f_3(x) = 0$, but apply a (soft) least squares penalty on $f_1(x)$ and a smaller one-sided penalization on $f_2(x)$. The above formulation is nonsmooth due to the max term in the objective function; in practice we could replace this by:

$$\begin{aligned} \min_{x_1, x_2, x_3, w} \quad & \exp(x_1) + 5\left\|\log(x_1) - 1\right\|^2 + 2w \\ \text{s.t.} \quad & x_1/x_2 = \log(x_3), \\ & 3x_1 + x_2 \leq 5, x_1 \geq 0, x_2 \geq 0 \\ & w \geq x_2^2 - 2, w \geq 0 \end{aligned}$$

and recover a standard form NLP. If the penalty on $f_1(x)$ would be replaced by a one-norm penalization (instead of least squares), we would have to play a similar game, moving the function $f_1(x)$ into the constraints and adding additional variable(s). To some extent, this seems unnatural - a modeler should be able to interchange the penalization without having to reformulate the problem from scratch.

The proposed extended NLP would not be reformulated at all by the modeler, but allow all these "generalized constraints" to be treated in a similar manner within the modeling system. The actual formulation would take:

$$\theta(u) = \theta_1(u_1) + \theta_2(u_2) + \theta_3(u_3)$$

where

$$\begin{aligned} \theta_1(u_1) &= 5u_1^2, \\ \theta_2(u_2) &= 2\max(u_2, 0), \\ \theta_3(u_3) &= \psi_{\{0\}}(u_3). \end{aligned}$$

The discussion above allows us to see that

$$\begin{aligned} Y &= \mathbf{R} \times [0,2] \times \mathbf{R}, \\ k(y) &= \frac{1}{20}y_1^2 + 0 + 0. \end{aligned}$$

The corresponding Lagrangian is the smooth function:

$$\mathcal{L}(x,y) = f_0(x) + \sum_{i=1}^{3} y_i f_i(x) - k(y).$$

The corresponding VI (13) can almost be formulated in GAMS (except that the linear constraint in X cannot be handled currently except by introducing a $\theta_4(x)$). Thus

$$f_4(x) = 3x_1 + x_2 - 5,\ \theta_4(u) = \psi_{\mathbf{R}_-}$$

resulting in the following choices for Y and k:

$$\begin{aligned} Y &= \mathbf{R} \times [0,2] \times \mathbf{R} \times \mathbf{R}_+, \\ k(y) &= \frac{1}{20}y_1^2 + 0 + 0 + 0. \end{aligned}$$

Since X and Y are now simple bound sets, (13) is now a complementarity problem and can be solved for example using PATH. A simple "empinfo" file details the choices of Y and k from the implemented library. The full model and option files are available from the authors.

Reformulation as a classical NLP

Suppose

$$\theta(u) = \sup_{y \in Y}\{u^T y - \frac{1}{2}y^T Q y,\}$$

for a polyhedral set $Y \in \mathbf{R}^m$ and a symmetric positive semidefinite $Q \in \mathbf{R}^{m\times m}$ (possibly $Q = 0$). Suppose further that

$$\begin{aligned} &X = \{x \mid Rx \le r\},\ \ Y = \{y \mid S^T y \le s\}, \\ &Q = DJ^{-1}D^T,\ \ F(x) = (f_1(x), \ldots, f_m(x)), \end{aligned}$$

where J is symmetric and positive definite (for instance $J = I$). Then the optimal solutions $\bar{x}$ of (10) are the $\bar{x}$ components of the optimal solutions $(\bar{x}, \bar{z}, \bar{w})$ to

$$\begin{array}{ll} \min & f_0(x) + s^T z + \frac{1}{2}w^T J w \\ \text{s.t.} & Rx \le r, z \ge 0, F(x) - Sz - Dw = 0. \end{array}$$

The multiplier on the equality constraint in the usual sense is the multiplier associated with $\bar{x}$ in the extended Lagrangian for (10). (Note that a Cholesky factorization may be needed to determine D.)

It may be better to solve this reformulated NLP than to solve (13). However, it is important that we can convey all types of nonsmooth optimization problems to a solver as smooth optimization problems, and hence it is important to communicate the appropriate structure to the solver interface. We believe that specifying Y and k is a theoretically sound way to do this.

Another example showing formulation of an extended nonlinear program as a complementarity problem within GAMS can be found in Dirkse and Ferris (1995).

Conic Programming

A problem of significant recent interest involves conic constraints:

$$\min_{x \in X} p^T x \text{ s.t. } Ax - b \le 0, x \in C,$$

where C is a convex cone. Using the notation outlined above, this can be expressed as an EMP:

$$\min_{x \in X} p^T x + \psi_{\mathbf{R}^m_-}(Ax - b) + \psi_C(x)$$

For specific cones such as the Lorentz (ice-cream) cone, or the rotated quadratic cone, there are efficient implementations of interior point algorithms for their solution (Andersen et al., 2003). It is also possible to reformulate the problem in the form (1). Annotating the variables that must lie in a particular cone using a "empinfo" file allows solvers like MOSEK (Andersen and Andersen, 2000) to receive the problem as a cone program, while standard NLP solvers would see a reformulation of the problem as a nonlinear program. Details on this approach can be found in the EMP manual.

Embedded Complementarity Systems

A different type of embedded optimization model that arises frequently in applications is:

$$\begin{array}{lll} \min\limits_x & f(x,y) & \\ \text{s.t.} & g(x,y) \le 0 & (\perp \lambda \ge 0) \end{array}$$

$$H(x,y,\lambda) = 0 \qquad (\perp y \text{ free})$$

Note the difference here: the optimization problem is over the variable x, and is parameterized by the variable y. The choice of y is fixed by the (auxiliary) complementarity relationships depicted here by H.

Within GAMS, this is modeled as:

```
model emp /deff,defg,defH/;
```

Again, so this model can be processed correctly as an EMP, the modeler provides additional annotations to the model defining equations in an "empinfo" file, namely that the function H that is defined in defH is complementary to the variable y (and hence the variable y is a parameter to the optimization problem), and furthermore that the dual variable associated with the equation defg in the optimization problem is one and the same as the variable λ used to define H. Armed with this additional information, the EMP tool automatically creates the following MCP:

$$\begin{array}{lll} 0 = \nabla_x \mathcal{L}(x,y,\lambda) & \perp & x \text{ free} \\ 0 \le -\nabla_\lambda \mathcal{L}(x,y,\lambda) & \perp & \lambda \ge 0 \\ 0 = H(x,y,\lambda) & \perp & y \text{ free}, \end{array}$$

where the Lagrangian is defined as

$$\mathcal{L}(x,y,\lambda) = f(x,y) + \lambda^T g(x,y).$$

Perhaps the most popular use of this formulation is where competition is allowed between agents. A standard method to deal with such cases is via the concept of Nash Games. In this setting x^* is a Nash Equilibrium if

$$x_i^* \in \arg\min_{x_i \in X_i} \ell_i(x_i, x_{-i}^*, q), \forall i \in \mathcal{I},$$

where x_{-i} are other players decisions and the quantities q are given exogenously, or via complementarity:

$$0 \le H(x,q) \quad \perp \quad q \ge 0.$$

This mechanism is extremely popular in economics, and Nash famously won the Nobel Prize for his contributions to this literature.

This format is again an EMP, more general than the example given above in two respects. Firstly, there is more than one optimization problem specified in the embedded complementarity system. Secondly, the parameters in each optimization problem consist of two types. Firstly, there are the variables q that are tied down by the auxiliary complementarity condition and hence are treated as parameters by the ith Nash player. Also there are the variables x_{-i} that are treated as parameters by the ith Nash player, but are treated as variables by a different player j. While we do not specify the syntax here for these issues, the manual provides examples that outline how to carry out this matching within GAMS. Finally, two points of note: first it is clear that the resulting model is a complementarity problem and can be solved using PATH, for example. Secondly, performing the conversion from an embedded complementarity system or a Nash Game automatically is a critical step in making such models practically useful.

We note that there is a large literature on discrete-time finite-state stochastic games: this has become a central tool in analysis of strategic interactions among forward-looking players in dynamic environments. The Ericson and Pakes (1995) model of dynamic competition in an oligopolistic industry is exactly in the format described above, and has been used extensively in applications such as advertising, collusion, mergers, technology adoption, international trade and finance.

Conclusions

A number of new modeling formats involving complementarity and variational inequalities have been described and a new tool, EMP, that allows such problems to be specified in the GAMS modeling systems has been outlined. We believe this will make a modelers task easier by allowing model structure to be described succinctly. Furthermore, model generation can be done more reliably and automatically, and algorithms can exploit model structure to improve solution speed and robustness.

Acknowledgements

This work is supported in part by Air Force Office of Scientific Research Grant FA9550-07-1-0389, and National Science Foundation Grants DMI-0521953, DMS-0427689 and IIS-0511905. The authors wish to

thank Todd Munson and Tom Rutherford for insightful comments on the material presented here.

References

Andersen, E. D. and Andersen, K. D. (2000). The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In H. Frank et al., editors, *High Performance Optimization*, pages 197–232. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Andersen, E. D., Roos, C., and Terlaky, T. (2003). On implementing a primal-dual interior-point method for conic quadratic optimization. *Mathematical Programming*, 95(2):249–277.

Bisschop, J. and Meeraus, A. (1982). On the development of a general algebraic modeling system in a strategic planning environment. *Mathematical Programming Study*, 20:1–29.

Brooke, A., Kendrick, D., and Meeraus, A. (1988). *GAMS: A User's Guide*. The Scientific Press, South San Francisco, California.

Cao, M. and Ferris, M. C. (1996). A pivotal method for affine variational inequalities. *Mathematics of Operations Research*, 21:44–64.

Dirkse, S. and Ferris, M. (2008). *NLPEC,* User's Manual. http://www.gams.com/solvers/nlpec.pdf.

Dirkse, S. P. and Ferris, M. C. (1995). MCPLIB: A collection of nonlinear mixed complementarity problems. *Optimization Methods and Software*, 5:319–345.

Duran, M. A. and Grossman, I. E. (1986). An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36:307–339.

Ericson, R. and Pakes, A. (1995). Markov perfect industry dynamics: A framework for empirical analysis. *Review of Economic Studies*, 62:53–82.

Facchinei, F. and Pang, J. S. (2003). *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Springer-Verlag, New York, New York.

Ferris, M. C., Dirkse, S. P., and Meeraus, A. (2005). Mathematical programs with equilibrium constraints: Automatic reformulation and solution via constrained optimization. In Kehoe, T. J., Srinivasan, T. N., and Whalley, J., editors, *Frontiers in Applied General Equilibrium Modeling*, pages 67–93. Cambridge University Press.

Ferris, M. C., Fourer, R., and Gay, D. M. (1999). Expressing complementarity problems and communicating them to solvers. *SIAM Journal on Optimization*, 9:991–1009.

Ferris, M. C. and Munson, T. S. (2000). Complementarity problems in GAMS and the PATH solver. *Journal of Economic Dynamics and Control*, 24:165–188.

Ferris, M. C. and Pang, J. S. (1997). Engineering and economic applications of complementarity problems. *SIAM Review*, 39:669–713.

Fourer, R., Gay, D. M., and Kernighan, B. W. (1990). A modeling language for mathematical programming. *Management Science*, 36:519–554.

Fourer, R., Gay, D. M., and Kernighan, B. W. (1993). *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press, Pacific Grove, California.

Huber, P. J. (1981). *Robust Statistics*. John Wiley & Sons, New York.

Jagla, J., Meeraus, A., Dirkse, S., and Ferris, M. (2008). *EMP,* User's Manual. In preparation, http://www.gams.com/solvers/emp.pdf.

Ralph, D. and Wright, S. J. (2004). Some properties of regularization and penalization schemes for MPECs. *Optimization Methods and Software*, 19(5):527–556.

Rockafellar, R. T. (1993). Lagrange multipliers and optimality. *SIAM Review*, 35:183–238.

Rockafellar, R. T. and Wets, R. J. B. (1998). *Variational Analysis*. Springer-Verlag.

Rutherford, T. F. (1999). Applied general equilibrium modeling with MPSGE as a GAMS subsystem: An overview of the modeling framework and syntax. *Computational Economics*, 14:1–46.

Tawarmalani, M. and Sahinidis, N. V. (2004). Global Optimization of Mixed Integer Nonlinear Programs: A Theoretical and Computational Study. *Mathematical Programming*, 99(3):563–591.

Wächter, A. and Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57.

Wallace, J., Philpott, A., O'Sullivan, M., and Ferris, M. (2006). Optimal rig design using mathematical programming. In *2nd High Performance Yacht Design Conference, Auckland, 14–16 February, 2006*, pages 185–192.

Westerlund, T., Skrifvars, H., Harjunkoski, I., and Pörn, R. (1998). An extended cutting plane method for solving a class of non-convex minlp problems. *Computers and Chemical Engineering*, 22(3):357–365.