MILES

Thomas F. Rutherford, University of Colorado

Contents

Abstract

MILES is a solver for nonlinear complementarity problems and nonlinear systems of equations. This solver can be accessed indirectly through GAMS/MPSGE or GAMS/MCP. This paper documents the solution algorithm, user options, and program output. The purpose of the paper is to provide users of GAMS/MPSGE and GAMS/MCP an overview of how the MCP solver works so that they can use the program effectively.

1 Introduction

MILES is a Fortran program for solving nonlinear complementarity problems and nonlinear systems of equations. The solution procedure is a generalized Newton method with a backtracking line search. This code is based on an algorithm investigated by Mathiesen (1985) who proposed a modeling format and sequential method for solving economic equilibrium models. The method is closely related to algorithms proposed by Robinson (1975), Hogan (1977), Eaves (1978) and Josephy (1979). In this implementation, subproblems are solved as linear complementarity problems (LCPs), using an extension of Lemke's almost-complementary pivoting scheme in which upper and lower bounds are represented implicitly. The linear solver employs the basis factorization package LUSOL, developed by Gill et al. (1991).

The class of problems for which MILES may be applied are referred to as "generalized" or "mixed" complementarity problems, which is defined as follows:

$$\begin{aligned} \text{Given:} \quad & F : R^n \to R^n \,, \quad \ell, u \in R^n \\ \text{Find:} \quad & z, w, v \in R^n \\ \text{such that} \quad & F(z) = w - v \\ & \ell \le z \le u, \;\; w \ge 0, \;\; v \ge 0 \\ & w^T (z - \ell) = 0 \,, \quad v^T (u - z) = 0. \end{aligned}$$

When $\ell = -\infty$ and $u = \infty$ MCP reduces to a nonlinear system of equations. When $\ell = 0$ and $u = +\infty$, the MCP is a nonlinear complementarity problem. Finite dimensional variational inequalities are also MCP. MCP includes inequality-constrained linear, quadratic and nonlinear programs as special cases, although for these problems standard optimization methods may be preferred. MCP models which are not optimization problems encompass a large class of interesting mathematical programs. Specific examples of MCP formulations are not provided here. See Rutherford (1992a) for MCP formulations arising in economics. Other examples are provided by Harker and Pang (1990) and Dirkse (1993).

There are two ways in which a problem may be presented to MILES:

I MILES may be used to solve computable general equilibrium models generated by MPSGE as a GAMS subsystem. In the MPSGE language, a model-builder specifies classes of nonlinear functions using a specialized tabular input format embedded within a GAMS program. Using benchmark quantities and prices, MPSGE automatically calibrates function coefficients and generates nonlinear equations and Jacobian matrices. Large, complicated systems of nonlinear equations may be implemented and analyzed very easily using this interface to MILES. An introduction to general equilibrium modeling with GAMS/MPSGE is provided by Rutherford (1992a).

II MILES may be accessed as a GAMS subsystem using variables and equations written in standard GAMS algebra and the syntax for "mixed complementarity problems" (MCP). If more than one MCP solver is available [1], the statement "`OPTION MCP = MILES;`" tells GAMS to use MILES as the MCP solution system. When problems are presented to MILES using the MCP format, the user specifies nonlinear functions using GAMS matrix algebra and the GAMS compiler automatically generates the Jacobian functions. An introduction to the GAMS/MCP modeling format is provided by Rutherford (1992b).

The purpose of this document is to provide users of MILES with an overview of how the solver works so that they can use the program more effectively. Section 2 introduces the Newton algorithm. Section 3 describes the implementation of Lemke's algorithm which is used to solve linear subproblems. Section 4 defines switches and tolerances which may be specified using the options file. Section 5 interprets the run-time log file which is normally directed to the screen. Section 6 interprets the status file and the detailed iteration reports which may be generated. Section 7 lists and suggests remedies for abnormal termination conditions.

2 The Newton Algorithm

The iterative procedure applied within MILES to solve nonlinear complementarity problems is closely related to the classical Newton algorithm for nonlinear equations. This first part of this section reviews the classical procedure. A thorough introduction to these ideas is provided by Dennis and Schnabel (1983). For a practical perspective, see Press et al. (1986).

Newton algorithms for nonlinear equations begin with a local (Taylor series) approximation of the system of nonlinear equations. For a point z in the neighborhood of $\hat{z}$, the system of nonlinear functions is linearized:

$$LF(z) = F(\bar{z}) + \nabla F(\bar{z})(z - \bar{z}).$$

Solving the linear system $LF(z) = 0$ provides the Newton direction from $\bar{z}$ which given by $d = -\nabla F^{-1}\, F(\bar{z})$.

Newton iteration k begins at point z^k. First, the linear model formed at z^k is solved to determine the associated "Newton direction", d^k. Second, a line search in direction d^k determines the scalar steplength and the subsequent iterate: $z^{k+1} = z^k + \lambda d^k$. An Armijo or "back-tracking" line search initially considers $\lambda = 1$. If $\|F(z^z + \lambda d^k)\| \leq \|F(z^k)\|$, the step size λ is adopted, otherwise is multiplied by a positive factor α, $\alpha < 1$, and the convergence test is reapplied. This procedure is repeated until either an improvement results or $\lambda < \underline{\lambda}$. When $\underline{\lambda} = 0$, a positive step is taken provided that [2]:

$$\frac{d}{d\lambda}\|F(z^k + \lambda d^k)\| < 0.$$

Convergence theory for this algorithm is quite well developed. See, for example, Ortega and Rheinbolt (1970) or Dennis and Schnabel (1983). The most attractive aspect of the Newton scheme with the backtracking line search is that in the

[1] There is one other MCP solver available through GAMS: PATH (Ferris and Dirkse, 1992)

[2] α and $\underline{\lambda}$ correspond to user-specified tolerances `DMPFAC` and `MINSTP`, respectively

neighborhood of a well-behaved fixed point, $\lambda = 1$ is the optimal step length and the rate of convergence can be quadratic. If this method finds a solution, it does so very quickly.

The application of Newton methods to nonlinear complementarity problems involves a modification of the search direction. Here, d solves a *linear complementarity* problem (LCP) rather than a linear system of equations. For iteration k, d solves:

$$F(z^k) + \nabla F(z^k)d - w + v = 0$$

$$\ell \leq d + z^k \leq u, \;\; w \geq 0, \;\; v \geq 0$$

$$w^T(d + z^k - \ell) = v^T(u - d - z^k) = 0.$$

Conceptually, we are solving for d, but in practice MILES solves the linear problem in terms of the original variables $z = z^k + d$:

$$F(z^k) - \nabla F(z^k)z_k + \nabla F(z^k)z = w - v$$

$$\ell \leq z \leq u, \;\; w \geq 0, \;\; v \geq 0$$

$$w^T(z - \ell) = 0\,, \quad v^T(u - z) = 0.$$

After computing the solution z, MILES sets $d^k = z - z^k$.

The linear subproblem incorporates upper and lower bounds on any or all of the variables, assuring that the iterative sequence always remains within the bounds: ($\ell \leq z^k \leq u$). This can be helpful when, as is often the case, $F()$ is undefined for some $z \in R^n$.

Convergence of the Newton algorithm applied to MCP hinges on three questions:

I Does the linearized problem always have a solution?

II If the linearized problem has a solution, does Lemke's algorithm find it?

III Is it possible to show that the computed direction d^k will provide an "improvement" in the solution?

Only for a limited class of functions $F()$ can all three questions be answered in the affirmative. For a much larger class of functions, the algorithm converges in practice but convergence is not "provable"[3].

The answer to question (III) depends on the choice of a norm by which an improvement is measured. The introduction of bounds and complementarity conditions makes the calculation of an error index more complicated. In MILES, the deviation associated with a candidate solution $z, \varepsilon(z)$, is based on a measure of the extent to which z, w and v violate applicable upper and lower bounds and complementarity conditions.

Evaluating Convergence

Let δ_i^L and δ_i^U be indicator variables for whether z_i is off its lower or upper bound. These are defined as [4]:

$$\delta_i^L = min(1, (z_i - \ell_i)^+) \;\text{ and }\; \delta_i^U = min(1, (u_i - z_i)^+).$$

Given z, MILES uses the value of $F(z)$ to implicitly define the slack variables w and v:

[3] Kaneko (1978) provides some convergence theory for the linearized subproblem

[4] In the following $x^+ = max(x, 0)$

$$w_i = F_i(z)^+ \ , \quad v_i = \Big(-F_i(z)\Big)^+ .$$

There are two components to the error term associated with index i, one corresponding to z_i's violation of upper and lower bounds:

$$\varepsilon_i^B = (z_i - u_i)^+ + (\ell_i - z_i)^+$$

and another corresponding to violations of complementarity conditions:

$$\varepsilon_i^C = \delta_i^L w_i + \delta_i^U v_i.$$

The error assigned to point z is then taken:

$$\varepsilon(z) = \|\varepsilon^B(z) + \varepsilon^C(z)\|_p$$

for a pre-specified value of $p = 1, 2$ or $+\infty$.[5]

3 Lemke's Method with Implicit Bounds

A mixed linear complementarity problem has the form:

$$\begin{array}{rl} \text{Given:} & M \in R^{n\times n}, \quad q, \ell, u \in R^n \\ \text{Find:} & z, w, v \in R^n \\ \text{such that} & Mz + q = w - v, \\ & \ell \le z \le u, \ \ w \ge 0, \ \ v \ge 0, \\ & w^T(z-\ell) = 0 \ , \quad v^T(u-z) = 0. \end{array}$$

In the Newton subproblem at iteration k, the LCP data are given by $q = F(z^k) - \nabla F(z^k) z^k$ and $M = \nabla F(z^k)$.

The Working Tableau

In MILES, the pivoting scheme for solving the linear problem works with a re-labeled linear system of the form:

$$Bx^B + Nx^N = q,$$

where $x^B \in R^n$, $\quad x^N \in R^{2n}$, and the tableau $[B|N]$ is a conformal "complementary permutation" of $[-M|\ I\ |-I]$. That is, every column i in B must either be the ith column of M, I or $-I$, while the corresponding columns i and $i+n$ in N must be the two columns which were not selected for B.

To move from the problem defined in terms of z, w and v to the problem defined in terms of x^B and x^N, we assign upper and lower bounds for the x^B variables as follows:

$$\underline{x}_i^B = \begin{cases} \ell_i, & \texttt{if} \quad x_i^B = z_i \\ 0, & \texttt{if} \quad x_i^B = w_i \ \texttt{or} \ v_i, \end{cases}$$

$$\overline{x}_i^B = \begin{cases} u_i, & \texttt{if} \quad x_i^B = z_i \\ \infty, & \texttt{if} \quad x_i^B = w_i \ \texttt{or} \ v_i \end{cases}$$

The values of the non-basic variables x_i^N and x_{i+n}^N are determined by the assignment of x_i^B:

[5] Parameter p may be selected with input parameter `NORM`. The default value for p is $+\infty$.

$$x_i^B = \begin{cases} z_i \;\Rightarrow\; \begin{cases} x_i^N = w_i = 0 \\ x_{i+n}^N = v_i = 0 \end{cases} \\ w_i \;\Rightarrow\; \begin{cases} x_i^N = z_i = \ell_i \\ x_{i+n}^N = v_i = 0 \end{cases} \\ v_i \;\Rightarrow\; \begin{cases} x_i^N = w_i = 0 \\ x_{i+n}^N = z_i = u_i. \end{cases} \end{cases}$$

In words: if z_i is basic then both w_i and v_i equal zero. If z_i is non-basic at its lower bound, then w_i is possibly non-zero and v_i is non-basic at zero. If z_i is non-basic at its upper bound, then v_i is possibly non-zero and w_i is non-basic at zero.

Conceptually, we could solve the LCP by evaluating 3^n linear systems of the form:

$$x^B = B^{-1}\Big(q - Nx^N\Big).$$

Lemke's pivoting algorithm provides a procedure for finding a solution by sequentially evaluating some (hopefully small) subsets of these 3^n alternative linear systems.

Initialization

Let B^0 denote the initial basis matrix [6] . The initial values for basic variables are then:

$$\hat{x}^B = (B^0)^{-1}(q - N\,\hat{x}^N).$$

If $\underline{x}^B \le \hat{x}^B \le \bar{x}^B$, then the initial basis is feasible and the complementarity problem is solved [7]. Otherwise, MILES introduces an artificial variable z_0 and an artificial column h. Basic variables are then expressed as follows:

$$x^B = \hat{x}^B - \tilde{h}z_0,$$

where $\tilde{h}$ is the "transformed artificial column" (the untransformed column is $h = B^0\tilde{h}$). The coefficients of $\tilde{h}$ are selected so that:

I The values of "feasible" basis variables are unaffected by z_0: ($\underline{x}_i^B \le x_i^B \le \bar{x}_i^B \Longrightarrow \tilde{h}_i = 0$).

II The "most infeasible" basic variable ($i = p$) is driven to its upper or lower bound when $z_0 = 1$:

$$\tilde{h}_p = \begin{cases} \hat{x}_p^B - \bar{x}_p^B, & if \quad \tilde{x}_p^B > \bar{x}_p^B \\ \hat{x}_p^B - \underline{x}_p^B, & if \quad \tilde{x}_p^B < \underline{x}_p^B \,. \end{cases}$$

III All other *infeasible* basic variables assume values between their upper and lower bounds when z_0 increases to 1:

$$x_i^B = \begin{cases} 1 + \underline{x}_i^B, & if \quad \underline{x}_i^B > -\infty, \quad \bar{x}_i^B = +\infty \\ \frac{\bar{x}_i^B + \underline{x}_i^B}{2}, & if \quad \underline{x}_i^B > -\infty, \quad \bar{x}_i^B < +\infty \\ \bar{x}_i^B - 1, & if \quad \underline{x}_i^B = -\infty, \quad \bar{x}_i^B < +\infty \,. \end{cases}$$

[6] In Miles, B^0 is chosen using the initially assigned values for z. When $z_i \le \ell_i$, then $x_i^B = w_i$; when $z_i \ge u_i$, then $x_i^B = v_i$; otherwise $x_i^B = z_i$.

[7] The present version of the code simply sets $B^0 = -I$ and $x^B = w$ when the user-specified basis is singular. A subsequent version of the code will incorporate the algorithm described by Anstreicher, Lee, and Rutherford [1992] for coping with singularity.

Pivoting Rules

When z_0 enters the basis, it assumes a value of unity, and at this point (baring degeneracy), the subsequent pivot sequence is entirely determined. The entering variable in one iteration is determined by the exiting basic variable in the previous iteration. For example, if z_i were in B^0 and introducing z_0 caused z_i to move onto its lower bound, then the subsequent iteration introduces w_i. Conversely, if w_i were in B^0 and z_0 caused w_i to fall to zero, the subsequent iteration increases z_i from ℓ_i. Finally, if v_i were in B^0 and z_0's introduction caused v_i to fall to zero, the subsequent iteration *decreases* z_i from u_i.

Table 1 Pivot Sequence Rules for Lemke's Algorithm with Implicit Bounds

N	Exiting Variable	Entering Variable	Change in Non-basic Values
I	z_i at lower bound	w_i increases from 0	$x_i^N = z_i = \ell_i$
II	z_i at upper bound	v_i increases from 0	$x_{i+n}^N = z_i = u_i$
III	w_i at 0	z_i increases from ℓ_i	$x_i^N = x_{i+n}^N = 0$
IV	v_i at 0	z_i decreases from u_i	$x_i^N = x_{i+n}^N = 0$

The full set of pivoting rules is displayed in Table 1. One difference between this algorithm and the original Lemke (type III) pivoting scheme (see Lemke (1965), Garcia and Zangwill (1981), or Cottle and Pang (1992)) is that structural variables (z_i's) may enter or exit the basis at their upper bound values. The algorithm, therefore, must distinguish between pivots in which the entering variable increases from a lower bound from those in which the entering variable decreases from an upper bound.

Another difference with the "usual" Lemke pivot procedure is that an entering structural variable may move from one bound to another. When this occurs, the subsequent pivot introduces the corresponding slack variable. For example, if z_i is increased from ℓ_i to u_i without driving a basic variable infeasible, then z_i becomes non-basic at u_i, and the subsequent pivot introduces v_i. This type of pivot may be interpreted as z_i entering and exiting the basis in a single step [8].

In theory it is convenient to ignore degeneracy, while in practice degeneracy is unavoidable. The present algorithm does not incorporate a lexicographic scheme to avoid cycling, but it does implement a ratio test procedure which assures that when there is more than one candidate, priority is given to the most stable pivot. The admissible set of pivots is determined on both an absolute pivot tolerance (`ZTOLPV`) and a relative pivot tolerance (`ZTOLRP`). No pivot with absolute value smaller than $min($ `ZTOLPV`, `ZTOLRP`$\|V\|)$ is considered, where $\|V\|$ is the norm of the incoming column.

Termination on a Secondary Ray

Lemke's algorithm terminates normally when the introduction of a new variable drives z_0 to zero. This is the desired outcome, but it does not always happen. The algorithm may be interrupted prematurely when no basic variable "blocks" an incoming variable, a condition known as "termination on a secondary ray". In anticipation of such outcomes, MILES maintains a record of the value of z_0 for successive iterations, and it saves basis information associated with the smallest observed value, z_0^*. (In Lemke's algorithm, the pivot sequence is determined without regard for the effect on z_0, and the value of the artificial variable may follow an erratic (non-monotone) path from its initial value of one to the final value of zero.)

When MILES encounters a secondary ray, a restart procedure is invoked in which the set of basic variables associated with z_0^* are reinstalled. This basis (augmented with one column from the non-basic triplet to replace z_0) serves as B^0, and the algorithm is restarted. In some cases this procedure permits the pivot sequence to continue smoothly to a solution, while in other cases may only lead to another secondary ray.

[8] If all structural variables are subject to finite upper and lower bounds, then no z_i variables may be part of a homogeneous solution adjacent to a secondary ray. This does not imply, however, that secondary rays are impossible when all z_i variables are bounded, as a ray may then be comprised of w_i and v_i variables.

4 The Options File

MILES accepts the same format options file regardless of how the system is being accessed, through GAMS/MPSGE or GAMS/MCP. The options file is a standard text file which is normally named *MILES.OPT* [9]. The following is a typical options file:

```
BEGIN SPECS
               ITLIMT = 50
               CONTOL = 1.0E-8
               LUSIZE = 16
          END SPECS
```

All entries are of the form "<keyword> = <value>", where keywords have at most 6 characters. The following are recognized keywords which may appear in the options file, identified by keyword, type and default value, and grouped according to function:

Termination control

Option	Description	Default
CONTOL	is the convergence tolerance. Whenever an iterate is encountered for which $\varepsilon(z) <$CONTOL, the algorithm terminates. This corresponds to the GAMS/MINOS parameter "Row tolerance".	1.0E-6
ITLIMT	is an upper bound on the number of Newton iterations. This corresponds to the GAMS/MINOS parameter "Major iterations".	25
ITERLIM	is an upper bound on the cumulative number of Lemke iterations. When MILES is invoked from within a GAMS program (either with MPSGE or GAMS/MCP), <model>.ITERLIM can be used to set this value. This corresponds to the GAMS/MINOS parameter "Iterations limit".	1000
NORM	defines the vector norm to be used for evaluating $\varepsilon(z)$. Acceptable values are 1, 2 or 3 which correspond to p = 1, 2 and $+\infty$.	3
NRFMAX	establishes an upper bound on the number of factorizations in any LCP. This avoids wasting lots of CPU if a subproblem proves difficult to solve.	3
NRSMAX	sets an upper bound on the number of restarts which the linear subproblem solver will undertake before giving up.	1

Basis factorization control

Option	Description	Default
FACTIM	indicates the maximum number of CPU seconds between recalculation of the basis factors.	120.0
INVFRQ	determines the maximum number of Lemke iterations between recalculation of the basis factors. This corresponds to the GAMS/MINOS parameter "Factorization frequency".	200
ITCH	indicates the frequency with which the factorization is checked. The number refers to the number of basis replacements operations between refinements. This corresponds to the GAMS/MINOS parameter "Check frequency".	30

[9] When invoking MILES from within GAMS it is possible to use one of several option file names. See the README documentation with GAMS 2.25 for details.

Pivot Selection

Option	Description	Default
PLINFY	is the value assigned for "plus infinity" ("+INF" in GAMS notation).	1.D20
ZTOLPV	is the absolute pivot tolerance. This corresponds, roughly, to the GAMS/MINOS parameter "Pivot tolerance" as it applies for nonlinear problems.	3.644E-11 (EPS**(2./3.)) [10]
ZTOLRP	is the relative pivot tolerance. This corresponds, roughly, to the GAMS/MINOS parameter "Pivot tolerance" as it applies for nonlinear problems.	3.644E-11 (EPS**(2./3.))
ZTOLZE	is used in the subproblem solution to determine when any variable has exceeded an upper or lower bound. This corresponds to GAMS/MINOS parameter "Feasibility tolerance".	1.E-6

Linearization and Data Control

Option	Description	Default
SCALE	invokes row and column scaling of the LCP tableau in every iteration. This corresponds, roughly, to the GAMS/MINOS switch "scale all variables".	.TRUE.
ZTOLDA	sets a lower bound on the magnitude of nonzeros recognized in the linearization. All coefficients with absolute value less than ZTOLDA are ignored.	1.483E-08 (EPS**(1/2))

Newton Iteration Control

Option	Description	Default
DMPFAC	is the Newton damping factor, represented by α above.	0.5
MINSTP	is the minimum Newton step length, represented by $\underline{\lambda}$ above.	0.03

Output Control

Option	Description	Default
INVLOG	is a switch which requests LUSOL to generate a report with basis statistics following each refactorization.	.TRUE.
LCPDMP	is a switch to generate a printout of the LCP data after scaling.	.FALSE.
LCPECH	is a switch to generate a printout of the LCP data before scaling, as evaluated.	.FALSE.
LEVOUT	sets the level of debug output written to the log and status files. The lowest meaningful value is -1 and the highest is 3. This corresponds, roughly, to the GAMS/MINOS parameter "Print level"	0
PIVLOG	is a switch to generate a status file listing of the Lemke pivot sequence.	.FALSE.

LUSOL parameters

(as with MINOS 5.4, except LUSIZE)

Option	Description	Default
LUSIZE	is used to estimate the number of LU nonzeros which will be stored, as a multiple of the number of nonzeros in the Jacobian matrix.	10
LPRINT	is the print level, < 0 suppresses output. = 0 gives error messages. = 1 gives debug output from some of the other routines in LUSOL. ≥ 2 gives the pivot row and column and the no. of rows and columns involved at each elimination step in lu1fac.	0
MAXCOL	in lu1fac is the maximum number of columns searched allowed in a Markowitz-type search for the next pivot element. For some of the factorization, the number of rows searched is maxrow = maxcol - 1.	5
ELMAX1	is the maximum multiplier allowed in L during factor.	10.0
ELMAX2	is the maximum multiplier allowed in L during updates.	10.0
SMALL	is the absolute tolerance for treating reals as zero. IBM double: 3.0d-13	EPS**(4./5.)
UTOL1	is the absolute tol for flagging small diagonals of U. IBM double: 3.7d-11	EPS**(2./3.)
UTOL2	is the relative tol for flagging small diagonals of U. IBM double: 3.7d-11	EPS**(2./3.)
USPACE	is a factor which limits waste space in U. In lu1fac, the row or column lists are compressed if their length exceeds uspace times the length of either file after the last compression.	3.0
DENS1	is the density at which the Markowitz strategy should search maxcol columns and no rows.	0.3
DENS2	is the density at which the Markowitz strategy should search only 1 column or (preferably) use a dense LU for all the remaining rows and columns.	0.6

5 Log File Output

The log file is intended for display on the screen in order to permit monitoring progress. Relatively little output is generated.

A sample iteration log is displayed in Table 2. This output is from two cases solved in succession. This and subsequent output comes from program *TRNSP.FOR* which calls the MILES library directly. (When MILES is invoked from within GAMS, at most one case is processed at a time.)

The first line of the log output gives the MILES program date and version information. This information is important for bug reports.

The line beginning "Work space..." reports the amount of memory which has been allocated to solve the model - 10K for this example. Thereafter is reported the initial deviation together with the name of the variable associated with the largest imbalance ($\varepsilon_i^B + \varepsilon_i^C$). The next line reports the convergence tolerance.

The lines beginning 0 and 1 are the major iteration reports for those iterations. the number following the iteration number is the current deviation, and the third number is the Armijo step length. The name of the variable complementary to the equation with the largest associated deviation is reported in parenthesis at the end of the line.

Following the final iteration is a summary of iterations, refactorizations, amd final deviation. The final message reports the solution status. In this case, the model has been successfully processed ("Solved.").

Table 2 Sample Iteration Log

```
MILES  (July 1993)                                    Ver:225-386-02

Thomas F. Rutherford
Department of Economics
University of Colorado

Technical support available only by Email:  TOM@GAMS.COM

Work space allocated           --    0.01 Mb

Initial deviation ........ 3.250E+02    P_01
Convergence tolerance .... 1.000E-06

   0    3.25E+02    1.00E+00 (P_01    )
   1    1.14E-13    1.00E+00 (W_02    )

Major iterations .........     1
Lemke pivots .............    10
Refactorizations .........     2
Deviation ................ 1.137E-13

Solved.

Work space allocated           --    0.01 Mb

Initial deviation ........ 5.750E+02    W_02
Convergence tolerance .... 1.000E-06

   0    5.75E+02    1.00E+00 (W_02    )
   1    2.51E+01    1.00E+00 (P_01    )
   2    4.53E+00    1.00E+00 (P_01    )
   3    1.16E+00    1.00E+00 (P_01    )
   4    3.05E-01    1.00E+00 (P_01    )
   5    8.08E-02    1.00E+00 (P_01    )
   6    2.14E-02    1.00E+00 (P_01    )
   7    5.68E-03    1.00E+00 (P_01    )
   8    1.51E-03    1.00E+00 (P_01    )
   9    4.00E-04    1.00E+00 (P_01    )
  10    1.06E-04    1.00E+00 (P_01    )
  11    2.82E-05    1.00E+00 (P_01    )
  12    7.47E-06    1.00E+00 (P_01    )
  13    1.98E-06    1.00E+00 (P_01    )
  14    5.26E-07    1.00E+00 (P_01    )

Major iterations .........    14
Lemke pivots .............    23
Refactorizations .........    15
Deviation ................ 5.262E-07

Solved.
```

6 Status File Output

The status file reports more details regarding the solution process than are provided in the log file. Typically, this file is written to disk and examined only if a problem arises. Within GAMS, the status file appears in the listing only following the GAMS statement "`OPTION SYSOUT=ON;`".

The level of output to the status file is determined by the options passed to the solver. In the default configuration, the status file receives all information written to the log file together a detailed listing of all switches and tolerances and a report of basis factorization statistics.

When output levels are increased from their default values using the options file, the status file can receive considerably more output to assist in debugging. Tables 3-6 present a status file generated with `LEVOUT=3` (maximum), `PIVLOG=T`, and `LCPECH=T`.

The status file begins with the same header as the log file. Thereafter is a complete echo-print of the user-supplied option file when one is provided. Following the core allocation report is a full echo-print of control parameters, switches and tolerance as specified for the current run.

Table 4 continues the status file. The iteration-by- iteration report of variable and function values is produced whenever `LEVOUT >= 2`. Table 4 also contains an LCP echo-print. This report has two sections: $ROWS and $COLUMNS. The four columns of numbers in the $ROWS section are the constant vector (q), the current estimate of level values for the associated variables (z), and the lower and upper bounds vectors (ℓ and u). The letters L and U which appear between the ROW and Z columns are used to identify variables which are non-basic at their lower and upper bounds, respectively. In this example, all upper bounds equal $+\infty$, so no variables are non-basic at their upper bound.

By convention, only variable (and not equation names) appear in the status file. An equation is identified by the corresponding variable. We therefore see in the $COLUMNS: section of the matrix echo-print, the row names correspond to the names of z variables. The names assigned to variables z_i, w_i and v_i are $z-$ <name i >, $w-$ <name i >, and $v-$ <name i >, as shown in the $COLUMNS section. The nonzeros for $w-<>$ and $v-<>$ variables are not shown because they are assumed to be $-/+I$.

The status file output continues on Table 5 where the first half of the table reports output from the matrix scaling procedure, and the second half reports the messages associated with initiation of Lemke's procedure.

The "lu6chk warning" is a LUSOL report. Thereafter are two factorization reports. Two factorizations are undertaken here because the first basis was singular, so the program install all the lower bound slacks in place of the matrix defined by the initial values, z.

Following the second factorization report, at the bottom of Table 5 is a summary of initial pivot. "Infeasible in 3 rows." indicates that $\tilde{h}$ contains 3 nonzero elements. "Maximum infeasibility" reports the largest amount by which a structural variable violates an upper or lower bound. "Artificial column with 3 elements." indicates that the vector $h = B^0\tilde{h}$ contains 3 elements (note that in this case $B^0 = -I$ because the initial basis was singular, hence the equivalence between the number of nonzeros in $\tilde{h}$ and h.).

Table 6 displays the final section of the status file. At the top of page 6 is the Lemke iteration log. The columns are interpreted as follows:

`ITER`	is the iteration index beginning with 0,
`STATUS`	is a statistic representing the efficiency of the Lemke path. Formally, status is the ratio of the minimum number of pivots from B_0 to the current basis divided by the actual number of pivots. When the status is 1, Lemke's algorithm is performing virtually as efficiently as a direct factorization (apart from the overhead of basis factor updates.)
`Z%`	indicates the percentage of columns in the basis are "structural" (z's).
`Z0`	indicates the value of the artificial variable. Notice that in this example, the artificial variable declines monotonically from its initial value of unity.
`ERROR`	is a column in which the factorization error is reported, when it is computed. For this run, ITCH=30 and hence no factorization errors are computed.

`INFEAS.` is a column in which the magnitude of the infeasibility introduced by the artificial column (defined using the box-norm) is reported. (In MILES the cover vector h contains many different nonzero values, not just 1's; so there may be a large difference between the magnitude of the artificial variable and the magnitude of the induced infeasibility.

`PIVOTS` reports the pivot magnitude in both absolute terms (the first number) and relative terms (the second number). The relative pivot size is the ratio of the pivot element to the norm of the incoming column.

`IN/OUT` report the indices (not names) of the incoming and outgoing columns for every iteration. Notice that Lemke's iteration log concludes with variable z_0 exiting the basis.

The convergence report for iteration 1 is no different from the report written to the log file, and following this is a second report of variable and function values. We see here that a solution has been obtained following a single subproblem. This is because the underlying problem is, in fact, linear.

The status file (for this case) concludes with an iteration summary identical to the log file report and a summary of how much CPU time was employed overall and within various subtasks. (Don't be alarmed if the sum of the last five numbers does not add up to the first number - some cycles are not monitored precisely.)

Table 3 Status File with Debugging Output (page 1 of 4)

```
MILES  (July 1993)                                        Ver:225-386-02
Thomas F. Rutherford
Department of Economics
University of Colorado

Technical support available only by Email:  TOM@GAMS.COM

User supplied option file:

>BEGIN
> PIVLOG = .TRUE.
> LCPECH = .TRUE.
> LEVOUT = 3
>END

Work space allocated           --    0.01 Mb

NEWTON algorithm control parameters:
    Major iteration limit ..(ITLIMT).          25
    Damping factor .........(DMPFAC).   5.000E-01
    Minimum step length ....(MINSTP).   1.000E-02
    Norm for deviation .....(NORM)...           3
    Convergence tolerance ..(CONTOL).   1.000E-06
LEMKE algorithm control parameters:
    Iteration limit .......(ITERLIM).        1000
    Factorization frequency (INVFRQ).         200
    Feasibility tolerance ..(ZTOLZE).   1.000E-06
    Coefficient tolerance ..(ZTOLDA).   1.483E-08
    Abs. pivot tolerance ...(ZTOLPV).   3.644E-11
    Rel. pivot tolerance ...(ZTOLRP).   3.644E-11
    Cover vector tolerance .(ZTOLZO).   1.000E-06
    Scale every iteration ...(SCALE).   T
    Restart limit ..........(NRSMAX).           1
Output control switches:
    LCP echo print ...........(LCPECH).   F
    LCP dump .................(LCPDMP).   T
    Lemke inversion log ......(INVLOG).   T
    Lemke pivot log ......... (PIVLOG).   T
Initial deviation ........ 3.250E+02    P_01
Convergence tolerance .... 1.000E-06

================================
Convergence Report, Iteration  0
================================================================
ITER   DEVIATION        STEP
   0    3.25E+02    1.00E+00 (P_01    )
================================================================
```

Table 4 Status File with Debugging Output (page 2 of 4)

```
 Iteration   0 values.
               ROW             Z                      F
              --------    ------------           ------------
              X_01_01  L   0.00000E+00            -7.75000E-01
              X_01_02  L   0.00000E+00            -8.47000E-01
              X_01_03  L   0.00000E+00            -8.38000E-01
              X_02_01  L   0.00000E+00            -7.75000E-01
              X_02_02  L   0.00000E+00            -8.38000E-01
              X_02_03  L   0.00000E+00            -8.74000E-01
              W_01     L   0.00000E+00             3.25000E+02
              W_02     L   0.00000E+00             5.75000E+02
              P_01         1.00000E+00            -3.25000E+02
              P_02         1.00000E+00            -3.00000E+02
              P_03         1.00000E+00            -2.75000E+02
 ==================================
 Function Evaluation, Iteration:  0
 ==================================
$ROWS:
X_01_01    -2.25000000E-01  0.00000000E+00  0.00000000E+00  1.00000000E+20
X_01_02    -1.53000004E-01  0.00000000E+00  0.00000000E+00  1.00000000E+20
X_01_03    -1.61999996E-01  0.00000000E+00  0.00000000E+00  1.00000000E+20
X_02_01    -2.25000000E-01  0.00000000E+00  0.00000000E+00  1.00000000E+20
X_02_02    -1.61999996E-01  0.00000000E+00  0.00000000E+00  1.00000000E+20
X_02_03    -1.25999998E-01  0.00000000E+00  0.00000000E+00  1.00000000E+20
W_01       -3.25000000E+02  0.00000000E+00  0.00000000E+00  1.00000000E+00
W_02       -5.75000000E+02  0.00000000E+00  0.00000000E+00  1.00000000E+00
P_01        3.25000000E+02  1.00000000E+00  0.00000000E+00  1.00000000E+20
P_02        3.00000000E+02  1.00000000E+00  0.00000000E+00  1.00000000E+20
P_03        2.75000000E+02  1.00000000E+00  0.00000000E+00  1.00000000E+20
     ...    0.00000000E+00  0.00000000E+00  0.00000000E+00  0.00000000E+00
$COLUMNS:
Z-X_01_01   W_01     -1.00000000E+00
            P_01      1.00000000E+00
Z-X_01_02   W_01     -1.00000000E+00
            P_02      1.00000000E+00
Z-X_01_03   W_01     -1.00000000E+00
            P_03      1.00000000E+00
Z-X_02_01   W_02     -1.00000000E+00
            P_01      1.00000000E+00
Z-X_02_02   W_02     -1.00000000E+00
            P_02      1.00000000E+00
Z-X_02_03   W_02     -1.00000000E+00
            P_03      1.00000000E+00
Z-W_01      X_01_01   1.00000000E+00
            X_01_02   1.00000000E+00
            X_01_03   1.00000000E+00
Z-W_02      X_02_01   1.00000000E+00
            X_02_02   1.00000000E+00
            X_02_03   1.00000000E+00
Z-P_01      X_01_01  -1.00000000E+00
            X_02_01  -1.00000000E+00
Z-P_02      X_01_02  -1.00000000E+00
            X_02_02  -1.00000000E+00
Z-P_03      X_01_03  -1.00000000E+00
            X_02_03  -1.00000000E+00
       ...       ...  0.00000000E+00
```

Table 5 Status File with Debugging Output (page 3 of 4)

```
SCALING LCP DATA
-------
            MIN ELEM    MAX ELEM       MAX COL RATIO
AFTER  0    1.00E+00    1.00E+00               1.00
AFTER  1    1.00E+00    1.00E+00               1.00
AFTER  2    1.00E+00    1.00E+00               1.00
AFTER  3    1.00E+00    1.00E+00               1.00
SCALING RESULTS:

      A(I,J) <= A(I,J) * R(I) / C(J)

  ROW         ROW    Z COLUMN    W COLUMN    V COLUMN
    1      1.0000      1.0000      1.0000      1.0000
    2      1.0000      1.0000      1.0000      1.0000
    3      1.0000      1.0000      1.0000      1.0000
    4      1.0000      1.0000      1.0000      1.0000
    5      1.0000      1.0000      1.0000      1.0000
    6      1.0000      1.0000      1.0000      1.0000
    7      1.0000      1.0000      1.0000      1.0000
    8      1.0000      1.0000      1.0000      1.0000
    9      1.0000      1.0000      1.0000      1.0000
   10      1.0000      1.0000      1.0000      1.0000
   11      1.0000      1.0000      1.0000      1.0000

lu6chk  warning.  The matrix appears to be singular.
    nrank =      8        rank of U
n - nrank =      3        rank deficiency
    nsing =      3        singularities
    jsing =     10        last singular column
    dumax =   1.00E+00    largest  triangular diag
    dumin =   1.00E+00    smallest triangular diag

LUSOL 5.4  FACTORIZATION STATISTICS
Compressns    0    Merit      0.00    LenL          0    LenU         14
Increase   0.00    M            11    UT           11    D1            0
Lmax    0.0E+00    Bmax    1.0E+00    Umax    1.0E+00    Umin    1.0E+00
Growth  1.0E+00    LT            0    BP            0    D2            0

LUSOL 5.4  FACTORIZATION STATISTICS
Compressns    0    Merit      0.00    LenL          0    LenU         11
Increase   0.00    M            11    UT           11    D1            0
Lmax    0.0E+00    Bmax    1.0E+00    Umax    1.0E+00    Umin    1.0E+00
Growth  1.0E+00    LT            0    BP            0    D2            0

CONSTRUCTING ARTIFICIAL COLUMN

--- Infeasible in   3 rows.
--- Maximum infeasibility:    3.250E+02
--- Artificial column with   3 elements.
--- Pivoting in row:   9 to replace column  20
--- Pivot element:   -3.250E+02
```

Table 6 Status File with Debugging Output (page 4 of 4)

```
                        LEMKE PIVOT STEPS
                        =================
ITER  STATUS   Z%   Z0     ERROR   INFEAS.  ---- PIVOTS ----  IN      OUT
   1    1.00   0  1.000                    3.E+02 1.E+00  1  Z0      W    9
   2    1.00   9  1.000                    1.E+00 1.E+00  2  Z    9  W    1
   3    1.00  18  0.997                    9.E-01 9.E-01  1  Z    1  W   10
   4    1.00  27  0.997                    1.E+00 1.E+00  1  Z   10  W    2
   5    1.00  36  0.996                    9.E-01 4.E-01  1  Z    2  W   11
   6    1.00  45  0.996                    1.E+00 1.E+00  1  Z   11  W    6
   7    1.00  55  0.479                    2.E+00 1.E+00  1  Z    6  W    7
   8    1.00  64  0.479                    1.E+00 1.E+00  1  Z    7  W    4
   9    1.00  73  0.000                    1.E+00 1.E+00  2  Z    4  W    8
  10    1.00  73  0.000                    1.E-03 2.E-03  1  V    8  Z0

================================
Convergence Report, Iteration  1
================================================================
ITER   DEVIATION        STEP
   0    3.25E+02    1.00E+00
   1    1.14E-13    1.00E+00 (W_02    )
================================================================

Iteration   1 values.

              ROW            Z                    F
             --------    ------------         ------------
             X_01_01      2.50000E+01         -8.32667E-17
             X_01_02      3.00000E+02         -5.55112E-17
             X_01_03  L   0.00000E+00          3.60000E-02
             X_02_01      3.00000E+02         -8.32667E-17
             X_02_02  L   0.00000E+00          8.99999E-03
             X_02_03      2.75000E+02          2.77556E-17
             W_01         1.00000E+00         -1.13687E-13
             W_02         1.00000E+00          1.13687E-13
             P_01         1.22500E+00          0.00000E+00
             P_02         1.15300E+00          0.00000E+00
             P_03         1.12600E+00          0.00000E+00

Major iterations ........     1
Lemke pivots ............    10
Refactorizations ........     2
Deviation ............... 1.137E-13
Solved.

Total solution time .:        0.6 sec.
Function & Jacobian..:        0.2 sec.
LCP solution ........:        0.2 sec.
Refactorizations ....:        0.1 sec.
FTRAN ...............:        0.0 sec.
Update ..............:        0.1 sec.
```

7 Termination Messages

`Basis factorization error in INVERT.` An unexpected error code returned by LUSOL. This should normally not occur. Examine the status file for a message from LUSOL [11].

`Failure to converge.` Two successive iterates are identical - the Newton search direction is not defined. This should normally not occur.

`Inconsistent parameters ZTOLZ0, ZTOLZE.` `ZTOLZ0` determines the smallest value loaded into the cover vector h, whereas `ZTOLZE` is the feasibility tolerance employed in the Harris pivot selection procedure. If `ZTOLZ0 < -ZTOLZE`, Lemke's algorithm cannot be executed because the initial basis is infeasible.

`Insufficient space for linearization.` Available memory is inadequate for holding the nonzeros in the Jacobian. More memory needs to be allocated. On a PC, you probably will need to install more physical memory - if there is insufficient space for the Jacobi matrix, there is far too little memory for holding the LU factors of the same matrix.

`Insufficient space to invert.` More memory needs to be allocated for basis factors. Increase the value of `LUSIZE` in the options file, or assign a larger value to `<model>.workspace` if MILES is accessed through GAMS.

`Iteration limit exceeded.` This can result from either exceeding the major (Newton) or minor (Lemke) iterations limit. When MILES is invoked from GAMS, the Lemke iteration limit can be set with the statement "`<model>.iterlim = xx;`" (the default value is 1000). The Newton iteration limit is 25 by default, and it can be modified only through the `ITLIMT` option.

`Resource interrupt.` Elapsed CPU time exceeds options parameter `RESLIM`. To increase this limit, either add `RESLIM = xxx` in the options file or (if MILES is invoked from within GAMS), add a GAMS statement "`<model>.RESLIM = xxx;`".

`Singular matrix encountered.` Lemke's algorithm has been interrupted due to a singularity arising in the basis factorization, either during a column replacement or during a refactorization. For some reason, a restart is not possible.

`Termination on a secondary ray.` Lemke's algorithm terminated on a secondary ray. For some reason, a restart is not possible.

`Unknown termination status.` The termination status flag has not been set, but the code has interrupted. Look in the status file for a previous message. This termination code should not happen often.

References

K.J. Anstreicher, J. Lee and T.F. Rutherford "Crashing a Maximum Weight Complementary Basis", Mathematical Programming. (1992)

A. Brooke, D. Kendrick, and A. Meeraus "GAMS: A User's Guide", Scientific Press, (1987).

R.W. Cottle and J.S. Pang "The Linear Complementarity Problem", Academic Press, (1992).

J.E. Dennis and R.B. Schnabel "Numerical Methods for Unconstrained Optimization and Nonlinear Equations", Prentice-Hall (1983).

S. Dirkse "Robust solution of mixed complementarity problems", Computer Sciences Department, University of Wisconsin (1992).

B.C. Eaves, "A locally quadratically convergent algorithm for computing stationary points," Tech. Rep., Department of Operations Research, Stanford University, Stanford, CA (1978).

P.E. Gill, W. Murray, M.A. Saunders and M.H. Wright "Maintaining LU factors of a general sparse matrix", Linear Algebra and its Applications 88/89, 239-270 (1991).

C.B. Garcia and W.I. Zangwill "Pathways to Solutions, Fixed Points, and Equilibria", Prentice-Hall (1981)

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

W.W. Hogan, "Energy policy models for project independence," Computation and Operations Research 2 (1975) 251-271.

[11] Within GAMS, insert the line "`OPTION SYSOUT=ON;`" prior to the solve statement and resubmit the program in order to pass the MILES solver status file through to the listing.

N.H. Josephy, ”Newton’s method for generalized equations”, Technical Summary Report #1965, Mathematical Research Center, University of Wisconsin - Madison (1979).

I. Kaneko, ”A linear complementarity problem with an n by 2n ’P’- matrix”, Mathematical Programming Study 7, pp. 120-141, (1978).

C.E. Lemke ”Bimatrix equilibrium points and mathematical programming”, Management Science 11, pp. 681-689, (1965).

L. Mathiesen, ”Computation of economic equilibria by a sequence of linear complementarity problems”, Mathematical Programming Study 23 (1985).

P.V. Preckel, ”NCPLU Version 2.0 User’s Guide”, Working Paper, Department of Agricultural Economics, Purdue University, (1987).

W.H. Press, B.P.Flannery, S.A. Teukolsky, W.T. Vetterling ”Numerical Recipes: The Art of Scientific Computing”, Cambridge University Press (1986).

J.M. Ortega and W.C. Rheinboldt, ”Iterative Solution of Nonlinear Equations in Several Variables”, Academic Press (1970).

S.M. Robinson, ”A quadratically-convergent algorithm for general nonlinear programming problems”, Mathematical Programming Study 3 (1975).

T.F. Rutherford ”Extensions of GAMS for variational and complementarity problems with applications in economic equilibrium analysis”, Working Paper 92-7, Department of Economics, University of Colorado (1992a).

T.F. Rutherford ”Applied general equilibrium modeling using MPS/GE as a GAMS subsystem”, Working Paper 92-15, Department of Economics, University of Colorado (1992b).

Table 7 Transport Model in GAMS/MCP (page 1 of 2)

```
*==>TRNSP.GMS
option mcp=miles;
$TITLE  LP TRANSPORTATION PROBLEM FORMULATED AS A ECONOMIC EQUILIBRIUM
*       ===============================================================
*       In this file, Dantzig's original transportation model is
*       reformulated as a linear complementarity problem.  We first
*       solve the model with fixed demand and supply quantities, and
*       then we incorporate price-responsiveness on both sides of the
*       market.
*
*       T.Rutherford  3/91  (revised 5/91)
*       ===============================================================
*
*       This problem finds a least cost shipping schedule that meets
*       requirements at markets and supplies at factories
*
*       References:
*              Dantzig, G B., Linear Programming and Extensions
*              Princeton University Press, Princeton, New Jersey, 1963,
*              Chapter 3-3.
*
*              This formulation is described in detail in Chapter 2
*              (by Richard E. Rosenthal) of GAMS: A Users' Guide.
*              (A Brooke, D Kendrick and A Meeraus, The Scientific Press,
*              Redwood City, California, 1988.
*
   SETS
        I   canning plants   / SEATTLE, SAN-DIEGO /
        J   markets          / NEW-YORK, CHICAGO, TOPEKA / ;

   PARAMETERS
       A(I)  capacity of plant i in cases (when prices are unity)
         /    SEATTLE     325
              SAN-DIEGO   575  /,
       B(J)  demand at market j in cases (when prices equal unity)
         /    NEW-YORK    325
              CHICAGO     300
              TOPEKA      275  /,
        ESUB(J)  Price elasticity of demand (at prices equal to unity)
         /    NEW-YORK    1.5
              CHICAGO     1.2
              TOPEKA      2.0  /;

   TABLE D(I,J)  distance in thousands of miles
                     NEW-YORK       CHICAGO      TOPEKA
       SEATTLE          2.5           1.7          1.8
       SAN-DIEGO        2.5           1.8          1.4  ;

  SCALAR F  freight in dollars per case per thousand miles  /90/ ;

  PARAMETER C(I,J)  transport cost in thousands of dollars per case ;

            C(I,J) = F * D(I,J) / 1000 ;

  PARAMETER PBAR(J) Reference price at demand node J;
```

Table 8 Transport Model in GAMS/MCP (page 2 of 2)

```
  POSITIVE  VARIABLES
       W(I)             shadow price at supply node i,
       P(J)             shadow price at demand node j,
       X(I,J)           shipment quantities in cases;

  EQUATIONS
       SUPPLY(I)        supply limit at plant i,
       FXDEMAND(J)      fixed demand at market j,
       PRDEMAND(J)      price-responsive demand at market j,
       PROFIT(I,J)      zero profit conditions;
PROFIT(I,J)..   W(I) + C(I,J)   =G= P(J);
SUPPLY(I)..     A(I) =G= SUM(J, X(I,J));
FXDEMAND(J)..   SUM(I, X(I,J))  =G=  B(J);
PRDEMAND(J)..   SUM(I, X(I,J))  =G=  B(J) * (PBAR(J)/P(J))**ESUB(J);
*       Declare models including specification of equation-variable association:
  MODEL FIXEDQTY / PROFIT.X, SUPPLY.W, FXDEMAND.P/ ;
  MODEL EQUILQTY / PROFIT.X, SUPPLY.W, PRDEMAND.P/ ;
*       Initial estimate:
  P.L(J) = 1;    W.L(I) = 1;
  PARAMETER REPORT(*,*,*)  Summary report;
  SOLVE FIXEDQTY USING MCP;
  REPORT("FIXED",I,J) = X.L(I,J);   REPORT("FIXED","Price",J) = P.L(J);
  REPORT("FIXED",I,"Price") = W.L(I);

*       Calibrate the demand functions:

  PBAR(J) = P.L(J);

*       Replicate the fixed demand equilibrium:

  SOLVE EQUILQTY USING MCP;

  REPORT("EQUIL",I,J) = X.L(I,J);  REPORT("EQUIL","Price",J) = P.L(J);
  REPORT("EQUIL",I,"Price") = W.L(I);

  DISPLAY "BENCHMARK CALIBRATION", REPORT;

*       Compute a counter-factual equilibrium:

  C("SEATTLE","CHICAGO") = 0.5 * C("SEATTLE","CHICAGO");

  SOLVE FIXEDQTY USING MCP;
  REPORT("FIXED",I,J) = X.L(I,J);  REPORT("FIXED","Price",J) = P.L(J);
  REPORT("FIXED",I,"Price") = W.L(I);

*       Replicate the fixed demand equilibrium:

  SOLVE EQUILQTY USING MCP;
  REPORT("EQUIL",I,J) = X.L(I,J);  REPORT("EQUIL","Price",J) = P.L(J);
  REPORT("EQUIL",I,"Price") = W.L(I);

  DISPLAY "Reduced Seattle-Chicago transport cost:", REPORT;
```