Homework 2 Solution: Project Independence

AAE 526: Quantitative Methods
Fall Semester, 2019

rutherford@aae.wisc.edu

Due 11 October, 2019

Project Independence for Energy Security (PIES) was an initiative announced by U.S. President Richard Nixon on November 7, 1973, in reaction to the OPEC oil embargo and the resulting 1973 oil crisis. Recalling the Manhattan Project, the stated goal of Project Independence was to achieve energy self-sufficiency for the United States by 1980 through a national commitment to energy conservation and development of alternative sources of energy. Nixon declared that American science, technology and industry could free America from dependence on imported oil (energy independence).

For this homework we implement a stylized PIES model in GAMS and attempt to reproduce results from the paper, "Energy Policy Models for Project Independence" by William Hogan in *Computers and Operations Research* Vol 2, pp 251–271, 1975.

i. Formulate the prototype PIES model as a quadratic program in GAMS which can produce two equilibria, one without constraints associated with capital or steel and another which accounts for these constraints.

ii. Reformulate your PIES model as a linear complementarity program in GAMS and demonstrate that you obtain the same results as in the quadratic program.

iii. Compare your results with those presented in Tables 8 and 9 of Hogan's paper. Can you explain the discrepency?

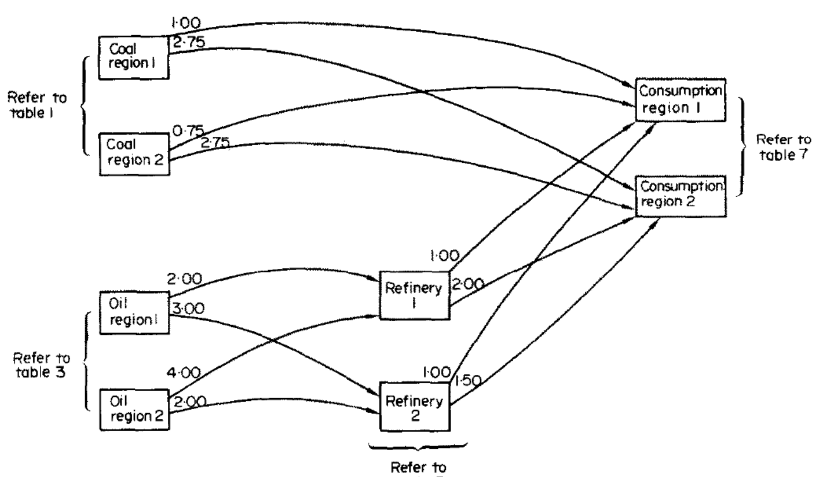

Fig. A1. Example energy system network.

We begin by choosing notation for sets and parameters. We use the following symbols to represent model dimensions.

```
set     j       Consumption regoins /j1, j2/
        i       Coal regions        /i1, i2/
        k       Oil regions         /k1, k2/
        r       Refineries          /r1, r2/

        c       Supply increments for coal regions /L, M, H/,
        o       Supply increments for oil regoins  /L, H/,

        p       Energy products         /Coal, Light, Heavy/,
        g(p)    Grades of refined oil   / Light, Heavy/,

        res     Resources               /steel, newcap/;

alias (p,pp,ip,jp);
```

Next we initialize parameters as data tables from the stylized PIES model described in Hogan's paper:

```
table table1(i,c,*) Resource requirements for production levels (p 257)

*       cap     Production capacity (tons per day)
*       c0      Minimum price / ton ($)
*       newcap  New capital / ton
*       steel   Steel / ton

        cap     c0      newcap  steel
i1.L    300     5       1       1
i1.M    300     6       5       2
i1.H    400     8       10      3

i2.L    200     4       1       1
i2.M    300     5       5       4
i2.H    600     7       6       5;

table table2(i,j)  Transport costs ($ per ton)

                j1      j2
        i1      1.00    2.50
        i2      0.75    2.75;


table table3(k,o,*)  Oil resource requirements

                cap     c0      newcap  steel
        k1.L    1100    1       0       0
        k1.H    1200    1.5     10      4
        k2.L    1300    1.25    0       0
        k2.H    1100    1.50    15      2;

table table4(k,r)  Oil transport costs ($ per barrel)
                r1      r2
        k1      2       3
        k2      4       2;

table table5(*,r)       Refinery yields and cost
                        r1      r2
        Light           0.6     0.5
        Heavy           0.4     0.5
        cost            6.5     5.0;

table table6(r,j)  Transport costs for refined products ($ per barrel)
                j1      j2
        r1      1       1.2
        r2      1       1.5;
```

```
table  table7(p,*)  Elasticities of final demand

                RefP    RefQ    Light   Heavy   Coal
        Light   16      1200    -0.5     0.2     0.1
        Heavy   12      1000     0.1    -0.5     0.2
        Coal    12      1000     0.1     0.2    -0.75;

table  table8(p,j) Demand without K or S constraints


          j1        j2
light    1252      1266
heavy    1041      1055
coal     1102       998   ;

table  table9(p,j) Demand with  K or S constraints

          j1        j2
light    1205      1229
heavy     996      1020
coal      996       910   ;
```

The Demand System

The demand function provided in Table 7 describes an asymmetric demand function of the form:

$$D_i(p) = \bar{q}_i + \sum_j \sigma_{ij}\left(p_j - \bar{p}_j\right)$$

in which $\bar{p}$ and $\bar{q}$ correspond to the `RefP` and `RefQ` columns in Table 7, and the asymmetric Slutsky matrix, σ, is computed on the basis of the demand elasticity matrix ϵ_{ij} given in Table 7:

$$\sigma_{ij} = \epsilon_{ij}\frac{\bar{q}_i}{\bar{p}_j}$$

In quadratic programming formulation we need an integrable demand function. In this case, we use Slutsky matrix S, a symmetric version of σ:

$$S_{ij} = \frac{\sigma_{ij} + \sigma_{ji}}{2}$$

The calculation of σ, S and S^{-1} is peformed in the following GAMS code:

```
parameter       pref(p)         Reference price,
                qref(p)         Reference demand;

pref(p) = table7(p,"refp");
qref(p) = table7(p,"refq");


parameter       sigma(p,pp)     Asymmetric slutsky matrix used for the AS model
                slutsky(p,pp)   Symmetric slutsky matrix used for CP model;

*       Asymmetric demand system:

sigma(p,pp) = table7(p,pp)*qref(p)/pref(pp);

*       Related symmetric demand system:

slutsky(p,pp) = 0.5*(sigma(p,pp)+sigma(pp,p));
```

```
*       Calculate the inverse demand coefficients:

variable        SLUTSKYINV(p,p) Inverse matrix for SLUTSKY;

equation        invdef;

invdef(ip,jp).. sum(p, SLUTSKYINV(ip,p)*slutsky(p,jp)) =e= 1$sameas(ip,jp);

model invert /invdef/;
solve invert using mcp;
```

Representation of the Network

It is helpful to initially approach the model formulation from a linear programming perspective, taking final demand as given. This permits us to see that we have the network logic in place without having to worry about the representation of demand functions through the quadratic form in the objective function. In the LP model we simply minimize the cost of meeting demand as reported in `table8`. The equations incorporate parameters which are not employed in the LP solution but which are required subsequently. These include a demand adjustment term (`delta`) and resource supply constraints (`rs`). We use the resource supply parameter to control whether the resource constraint enters the model. When `rs(res)=0`, the constraint for resource `res` is omitted.

The demand function associated with the *symmetric* Slutsky matrix is:

$$\hat{D}_i(p) = \bar{q}_i + \sum_j S_{ij}\left(p_j - \bar{p}_j\right)$$

and the inverse demand function associated with the symmetric Slutsky matrix is:

$$P_i(q) = \bar{p}_i + \sum_j S^{-1}_{ij}\left(q_j - \bar{q}_j\right)$$

Asymmetric demand may then be written as:

$$D_i(p) = \hat{D}_i(p) + \Delta_i$$

We Δ_i is assigned to the difference between $D_i(p)$ and $\hat{D}_i(p)$, the symmetric model lines up with the asymmetric model. Below we use this observation to implement a *diagonalization* strategy for solving the market equilibrium model with asymmetric demand through a sequence of quadratic programming problems.

```
parameter       delta(p,j)      Demand adjustment (off-diagonal terms),
                rs(res)         Resource supply,
                resutil         Resource utilization,
                report          Pivot report data,
                op              Flag for the diagonal own-price demand model (QP or MCP) /1/
                cp              Flag for symmetric cross-price demand model (QP or MCP) /0/
                as              Flag for the asymmetric cross-price demand model (MCP) /0/;
rs(res) = 0;
delta(p,j) = 0;

NONNEGATIVE
VARIABLES
        QC(i,c)         Quantity of coal extracted by region i at cost level c,
        QO(k,o)         Quantity of oil resources extracted -- region k at cost level o
        QR(r)           Quantity of oil refined -- refinery r,

        D(p,j)          Demand -- energy product p in consumption region j,

        XC(i,j)         Quantity of coal transported from region i to market j
        XO(k,r)         Quantity of oil resources shipped from region k to refinery r
        XR(r,g,j)       Quantity of oil grade g transported from refinery r to market j;
```

```
variable        COST            Total cost;

equations       oilresource, crudeoil, refinedoil, coalsupply, demand, costdef, resource;

*       Oil supply from region (k) by cost increment (o) equals oil shipments to refineries (r):

oilresource(k).. sum(o, QO(k,o)) =g= sum(r, XO(k,r));

*       Oil supply from regions (k) equals refinery (r) output:

crudeoil(r)..   sum(k, XO(k,r)) =g= QR(r);

*       Refined oil supply equals refined oil shipments:

refinedoil(r,g).. QR(r)*table5(g,r) =g= sum(j, XR(r,g,j));

*       Coal supply equals coal demand:

coalsupply(i).. sum(c, QC(i,c)) =g= sum(j, XC(i,j));


demand(j,p)..   sum((r,g(p)), XR(r,g,j)) + sum(i, XC(i,j))$sameas(p,"coal") =g= D(p,j) + delta(p,j);


resource(res)$rs(res).. rs(res) =g= sum((i,c), table1(i,c,res)*QC(i,c))  +
                                    sum((k,o), table3(k,o,res)*QO(k,o));

costdef..       COST =e=  sum((r,g,j), XR(r,g,j)*table6(r,j)) +
                          sum((k,r),   XO(k,r)*table4(k,r)) +
                          sum((i,j),   XC(i,j)*table2(i,j)) +

                          sum((k,o),   QO(k,o)*table3(k,o,"c0")) +
                          sum((i,c),   QC(i,c)*table1(i,c,"c0")) +

                          sum(r,       QR(r)*table5("cost",r));

*       Upper bounds on coal and oil production:

QC.UP(i,c) = table1(i,c,"cap");
QO.UP(k,o) = table3(k,o,"cap");

*       Solve the model as a linear program with fixed product demand
*       in each market and no resource constraints:

D.FX(p,j) = table8(p,j);

model pies_lp /all/;
solve pies_lp using LP minimizing COST;

report("price",p,j,"LP") = demand.m(j,p);
report("quantity",p,j,"LP") = D.l(p,j);
resutil(res,"LP") = sum((i,c), table1(i,c,res)*QC.L(i,c))  +
                        sum((k,o), table3(k,o,res)*QO.L(k,o));

report("price",p,j,"LP") = demand.m(j,p);
report("quantity",p,j,"LP") = D.l(p,j);
```

Two Integrable Demand Models

The next step in the computations is to add the quadratic term representing consumer surplus to the object function. This code includes two alternative demand functions, one based on the symmetric demand system with own- and cross-price effects (the *cross-price model,* `cp`), the other based on a diagonal demand function in which cross-price effects are assumed to be zero (the *own-price model*, op).

```
$macro  CS_op(p,j)      (D(p,j)*(pref(p) + 1/sigma(p,p) * (D(p,j)/2-qref(p))))

$macro  CS_cp(p,j)      (D(p,j)*(pref(p) + sum(pp, slutskyinv.L(p,pp) * (D(pp,j)/2-qref(pp)))))

variable        NSS     Negative of social surplus;

equation        nssdef;

nssdef..        NSS =e= COST - sum((p,j), CS_op(p,j)$op + CS_cp(p,j)$cp);

model pies_qcp /all/;

*       Solve the model with own-price demand function, ignoring
*       cross-price effects:

op = yes;
cp = no;
as = no;

delta(p,j) = 0;
D.UP(p,j) = +inf;
D.LO(p,j) = 0;
rs(res) = 0;
solve pies_qcp using QCP minimizing NSS;

resutil(res,"op") = sum((i,c), table1(i,c,res)*QC.L(i,c))  +
                                sum((k,o), table3(k,o,res)*QO.L(k,o));
report("price",p,j,"op") = demand.m(j,p);
report("quantity",p,j,"op") = D.l(p,j);

*       Solve the market equilibrium with symmetric cross-price demand:

op = no;
cp = yes;
as = no;

delta(p,j) = 0;
D.UP(p,j) = +inf;
D.LO(p,j) = 0;
rs(res) = 0;
solve pies_qcp using QCP minimizing NSS;

resutil(res,"cp") = sum((i,c), table1(i,c,res)*QC.L(i,c))  +
                                sum((k,o), table3(k,o,res)*QO.L(k,o));
report("price",p,j,"cp") = demand.m(j,p);
report("quantity",p,j,"cp") = D.l(p,j);
```

The Complementarity Model

As a cross check on the consumer surplus calculation we formulate the model as a complementarity problem. The complementarity problem shares the primal constraints with the optimization problem. In addition, it includes arbitrage conditions – dual feasibility constraints from the linear programming model and it incorporates an explicit primal demand function. Three alternative demand functions are included in the model. `D_op` defines the *own-price* (diagonal) demand in which the demand for product p depends only on the price of product p. `D_cp` defines the *symmetric cross-price* demand function in which the demand for product p depends on the prices of all goods as defined by parameter `slutsky(p,pp)`. Finally, `D_as` defines the *asymmetric cross-price* demand function as defined by `sigma(p,pp)`.

In a complementarity problem equations are associated with variables, and complementary slackness conditions require that when a variable is off its bounds, the corresponding equation is binding. The equation-variable associations are defined in the `model pies_mcp` statement.

```
NONNEGATIVE VARIABLES
        P_O(k)          Supply price of oil,
        P_XO(r)         Delivered price of oil,
        P_R(r,g)        Price of refinery outputs,

        P_C(i)          Supply price of coal,
        P_D(j,p)        Demand price of all products (oil and coal)

        PR(res)         Resource price;

equations
        prf_QC(i,c)             Quantity of coal extracted by region i at cost level c,
        prf_QO(k,o)             Quantity of oil resources extracted -- region k at cost level o
        prf_QR(r)               Quantity of oil refined -- refinery r,
        def_D(p,j)              Demand -- energy product p in consumption region j,
        prf_XR(r,g,j)           Quantity of oil transported from refinery r to market j
        prf_XC(i,j)             Quantity of coal transported from region i to market j
        prf_XO(k,r)             Quantity of oil resources shipped from region k to refinery r
        coalprice(j,p)          Coal price constraint;

prf_QC(i,c)..           table1(i,c,"c0") + sum(res,PR(res)*table1(i,c,res)) =e= P_C(i);

prf_QO(k,o)..           table3(k,o,"c0") + sum(res,PR(res)*table3(k,o,res)) =e= P_O(k) ;

prf_QR(r)..             P_XO(r) + table5("cost",r) =g= sum(g,P_R(r,g)*table5(g,r));

prf_XR(r,g,j)..         P_R(r,g) + table6(r,j)  =g= P_D(j,g);

prf_XC(i,j)..           P_C(i) + table2(i,j) =G= P_D(j,"coal");

prf_XO(k,r)..           P_O(k) + table4(k,r) =G= P_XO(r);

*       Define the own-price demand function:

$macro  D_op(p,j,price)  (qref(p) + slutsky(p,p)*(price(j,p)-pref(p)))

*       Define the symmetric cross-price demand function:

$macro  D_cp(p,j,price)  (qref(p) + sum(pp, slutsky(p,pp)*(price(j,pp)-pref(pp))))

*       Define the asymmetric cross-price demand function:

$macro  D_as(p,j,price)  (qref(p) + sum(pp, sigma(p,pp)*(price(j,pp)-pref(pp))))

def_D(p,j)..    D(p,j) =e= D_op(p,j,P_D)$op + D_cp(p,j,P_D)$cp + D_as(p,j,P_D)$as;

model pies_mcp /
        oilresource.P_O, crudeoil.P_XO, refinedoil.P_R, coalsupply.P_C, demand.P_D,
        prf_QC.QC, prf_QO.QO, prf_QR.QR, def_D.D, prf_XR.XR, prf_XC.XC, prf_XO.XO,
        resource.PR /;
```

After the complementarity problem is defined, it can be used to verify that the equilibrium found through social surplus optimization solves the corresponding MCP model. We do this check both for the own-price demand system (`op = yes`) and for the symmetric cross-price model (`cp = yes`).

```
op = yes;
cp = no;
as = no;

delta(p,j) = 0;
solve pies_qcp using QCP minimizing NSS;
```

```
PR.FX(res) = 0;
P_O.L(k) = oilresource.M(k);
P_XO.L(r) = crudeoil.M(r);
P_R.L(r,g) = refinedoil.M(r,g);
P_C.L(i) = coalsupply.m(i);
P_D.L(j,p) = demand.M(j,p);

pies_mcp.iterlim = 0;
solve pies_mcp using mcp;
abort$round(pies_mcp.objval) "MCP fails to replicate the QP model (own-price demand)";

op = no;
cp = yes;
as = no;

delta(p,j) = 0;
solve pies_qcp using QCP minimizing NSS;

PR.FX(res) = 0;
P_O.L(k) = oilresource.M(k);
P_XO.L(r) = crudeoil.M(r);
P_R.L(r,g) = refinedoil.M(r,g);
P_C.L(i) = coalsupply.m(i);
P_D.L(j,p) = demand.M(j,p);

pies_mcp.iterlim = 0;
solve pies_mcp using mcp;
abort$round(pies_mcp.objval) "MCP fails to replicate the QP model (cross-price demand)";
```

Diagonalization

A diagonalization algorithm involves solving a nonlinear system of equations of the form:

$$x = f(x)$$

throgh iterative assignment for iterations $k = 1, 2, \dots$

$$x^{k+1} = f(x^k)$$

We can use diagonalization to find Δ through the iterations:

i. Compute prices through solution of the quadratic program, implicitly compute $p^{k+1} = f(\Delta^k)$

ii. Re-compute the demand function perturbation, i.e. $\Delta^{k+1} = D(p^{k+1}) - \hat{D}(p^{k+1})$

iii. Stop when $\delta^k = ||\Delta^{k+1} - \Delta^k||$ is smaller than a given tolerance τ

The following GAMS codes implements this diagonalization algorithm using the own-price demand system:

```
set     iter    Iterations for demand adjustments /iter0*iter10/;

parameter       iterlog         Iteration log for iterative demand adjustments,
                qd(p,j)         Exact demand at current prices,
                dev             Maximum deviation /1/;

op = yes;
cp = no;
as = no;

dev = 1;
loop(iter$round(dev,4),
        iterlog(iter,p) = sum(j, sqr(delta(p,j) - (D_as(p,j,demand.M) - D.L(p,j))));
```

```
        iterlog(iter,j) = sum(p, sqr(delta(p,j) - (D_as(p,j,demand.M) - D.L(p,j))));
        delta(p,j) = D_as(p,j,demand.M) - D.L(p,j);
        solve pies_qcp using QCP minimizing NSS;
        dev = sum(p,iterlog(iter,p)) + sum(j,iterlog(iter,j));
);
display "Iteration log with diagonal (own-price) demand system:", iterlog;
```

```
----    392 Iteration log with diagonal (own-price) demand system:

----    392 PARAMETER iterlog  Iteration log for iterative demand adjustments

               j1          j2        Coal       Light       Heavy

iter0    1496.372    1080.593      86.203    1331.809    1158.953
iter1    7202.339    2822.173    3632.534     199.765    6192.213
iter2    1674.564    1570.186     930.695    2027.999     286.056
iter3     316.545     301.256     204.064     169.929     243.808
iter4      29.783      28.784       9.871      30.839      17.857
iter5       0.442       0.464       0.005       0.146       0.755
iter6       0.017       0.018       0.009       0.022       0.004
iter7       0.002       0.002 9.588287E-4       0.002       0.001
iter8  2.883785E-4 2.908058E-4 1.426338E-4 2.506698E-4 1.858807E-4
iter9  4.106105E-5 4.142413E-5 2.025582E-5 3.508095E-5 2.714841E-5
iter10 5.871390E-6 5.922886E-6 2.897648E-6 5.030134E-6 3.866495E-6
```

The symmetric cross-price model defined by `slutsky` is a closer approximation to the asymmetric cross-price model defined by `sigma`, and when this model is used, convergence of the diagonalization algorithm is considerably quicker:

```
iterlog(iter,p) = 0;
iterlog(iter,j) = 0;
delta(p,j) = 0;

op = no;
cp = yes;
as = no;

dev = 1;
loop(iter$round(dev,4),
        iterlog(iter,p) = sum(j, sqr(delta(p,j) - (D_as(p,j,demand.M) - D.L(p,j))));
        iterlog(iter,j) = sum(p, sqr(delta(p,j) - (D_as(p,j,demand.M) - D.L(p,j))));
        delta(p,j) = D_as(p,j,demand.M) - D.L(p,j);
        solve pies_qcp using QCP minimizing NSS;
        dev = sum(p,iterlog(iter,p)) + sum(j,iterlog(iter,j));
);
display "Iteration log with symmetric cross-price demand system:", iterlog;
```

```
----    417 Iteration log with symmetric cross-price demand system:

----    417 PARAMETER iterlog  Iteration log for iterative demand adjustments

              j1          j2        Coal       Light       Heavy

iter0   18398.117    9330.138    9726.247   11242.314    6759.693
iter1   19386.580   10155.725   11656.915    4433.775   13451.615
iter2      37.187      37.750       0.931      61.491      12.515
iter3       0.051       0.034       0.003       0.044       0.038
iter4 8.421629E-6 5.469478E-6 1.634557E-7 1.153012E-5 2.197526E-6
```

We verify that the equilibrium returned through diagonalization solves the corresponding complementarity model based on the asymmetric demand model.

```
op = no;
cp = no;
as = yes;

D.L(p,j) = D_as(p,j,demand.M);

delta(p,j) = 0;

PR.FX(res) = 0;
P_O.L(k) = oilresource.M(k);
P_XO.L(r) = crudeoil.M(r);
P_R.L(r,g) = refinedoil.M(r,g);
P_C.L(i) = coalsupply.m(i);
P_D.L(j,p) = demand.M(j,p);

pies_mcp.iterlim = 0;
solve pies_mcp using mcp;
abort$round(pies_mcp.objval) "MCP fails to replicate asymmetric equilibrium.";
```

Resource Constraints on Steel and New Capital

Finally, we install constraints on steel and new capital through assignment of parameter `rs(res)` and solve the model through diagonalization.

```
rs("steel")  = 12000;
rs("newcap") = 35000;

op = yes;
cp = no;
as = no;

delta(p,j)    = 0;

solve pies_qcp using QCP minimizing NSS;

resutil(res,"con_op") = sum((i,c), table1(i,c,res)*QC.L(i,c))  +
                        sum((k,o), table3(k,o,res)*QO.L(k,o));
report("price",p,j,"con_op") = demand.m(j,p);
report("quantity",p,j,"con_op") = D.l(p,j);

op = no;
cp = yes;
as = no;

delta(p,j)    = 0;

solve pies_qcp using QCP minimizing NSS;

resutil(res,"con_cp") = sum((i,c), table1(i,c,res)*QC.L(i,c))  +
                        sum((k,o), table3(k,o,res)*QO.L(k,o));
report("price",p,j,"con_cp") = demand.m(j,p);
report("quantity",p,j,"con_cp") = D.l(p,j);

*       Solve the QP model iteratively with cross-price elasticities of demand:

op = no;
cp = yes;
as = no;
dev = 1;
iterlog(iter,p) = 0;
iterlog(iter,j) = 0;

loop(iter$round(dev,4),
```

```
        iterlog(iter,p) = sum(j, sqr(delta(p,j) - (D_as(p,j,demand.M) - D.L(p,j))));
        iterlog(iter,j) = sum(p, sqr(delta(p,j) - (D_as(p,j,demand.M) - D.L(p,j))));
        delta(p,j) = D_as(p,j,demand.M) - D.L(p,j);
        solve pies_qcp using QCP minimizing NSS;
        dev = sum(p,iterlog(iter,p)) + sum(j,iterlog(iter,j));
);
display "Iteration log with resource constraints:", iterlog;

resutil(res,"iter_con") = sum((i,c), table1(i,c,res)*QC.L(i,c))  +
                                sum((k,o), table3(k,o,res)*QO.L(k,o));
report("price",p,j,"iter_con") = demand.m(j,p);
report("quantity",p,j,"iter_con") = D.l(p,j) + delta(p,j);
report("delta%",p,j,"iter_con") = 100 * delta(p,j) /(D.l(p,j) + delta(p,j));

*       Verify consistency with the MCP model:

op = no;
cp = no;
as = yes;

D.L(p,j) = D_as(p,j,demand.M);
delta(p,j) = 0;

PR.UP(res) = +inf;
PR.L(res) = resource.M(res);
P_O.L(k) = oilresource.M(k);
P_XO.L(r) = crudeoil.M(r);
P_R.L(r,g) = refinedoil.M(r,g);
P_C.L(i) = coalsupply.m(i);
P_D.L(j,p) = demand.M(j,p);

pies_mcp.iterlim = 0;
solve pies_mcp using mcp;
abort$round(pies_mcp.objval) "MCP fails to replicate constrained equilibrium.";

option resutil:1;
display resutil;

option report:2:2:1;
display report;
```

Replication of Hogan's Results

The equilibrium demand quantities *do not agree* with the values reported by Hogan in Tables 8 and 9.

```
----    528 PARAMETER resutil  Resource utilization

               LP          op          cp     iter_op     iter_cp      con_op      con_cp    iter_con

steel     13156.0     13500.0     13286.2     13327.7     13327.7     12000.0     12000.0     12000.0
newcap    38740.0     39600.0     39065.6     39169.3     39169.3     35000.0     35000.0     35000.0


----    531 PARAMETER report  Pivot report data

INDEX 1 = price

                 LP          op          cp     iter_op     iter_cp      con_op      con_cp    iter_con

Coal .j1       9.00       10.40        9.03        9.20        9.20       11.52       11.60       11.64
Coal .j2      10.50       12.00       10.78       10.97       10.97       13.52       13.60       13.64
Light.j1      12.40       13.71       12.40       11.98       11.98       15.63       15.70       15.75
Light.j2      12.60       13.94       12.60       12.32       12.32       15.83       15.90       15.95
```

Heavy.j1	8.60	10.10	8.60	9.02	9.02	11.67	11.80	11.85
Heavy.j2	9.10	10.60	9.10	9.52	9.52	12.17	12.30	12.35

INDEX 1 = quantity

	LP	op	cp	iter_op	iter_cp	con_op	con_cp	iter_con
Coal .j1	1102.00	1100.00	1100.00	1100.00	1100.00	1030.16	1019.33	1018.11
Coal .j2	998.00	1000.00	1000.00	1000.00	1000.00	905.16	904.29	902.69
Light.j1	1252.00	1285.77	1266.21	1263.30	1263.30	1213.92	1205.44	1202.55
Light.j2	1266.00	1277.07	1279.56	1278.20	1278.20	1206.42	1220.75	1225.05
Heavy.j1	1041.00	1078.99	1044.84	1052.34	1052.34	1013.55	997.76	998.97
Heavy.j2	1055.00	1058.16	1055.95	1063.09	1063.09	992.71	1012.89	1012.72

INDEX 1 = delta%

	iter_op	iter_cp	iter_con
Coal .j1	-6.80	0.69	0.05
Coal .j2	-6.43	0.69	9.377400E-3
Light.j1	-6.93	-2.04	-0.14
Light.j2	-4.68	-1.48	0.45
Heavy.j1	-6.82	2.63	0.17
Heavy.j2	-3.78	2.38	0.03