```
MODEL:
 ! Two level blending or pooling model(POOLINGH).
   Raw materials are selectively blended into pools, e.g., storage tanks,
   then pools are blended into finished products.
    We want to maximize the profit = sales of finished goods,
   minus cost of raw materials purchased,
    subject to satisfying quality requirements,
   such as density, sulfur fraction, Motor Octane Number,
   Research Octane Number, etc.
 ! Keywords: Blending, Haverly, LINGO, Pooling, Petroleum refining, Refinery;
SETS:
 ! Each raw material has an availability and cost/unit;
   RM : A, COST;

 ! There are a set of intermediate pools, e.g., storage tanks;
   POOL: BP, BPUB ;
 ! Each finished good has a min required, max sellable,pro-
   fit contribution/unit and batch size to be determined;
   FG: D, E, PRICE, BF;

  ! There are a set of quality measures;
   QM;

  ! For each combo of RM and QM there is a quality level;
   RXQ( RM, QM): QL;

  ! For each combo QM, FG there are upper and lower limits
   on quality, and slack on upper quality to be determined;
   QXF( QM, FG): U, L, S;

  ! For each combination of Pool and quality level, there is
     a quality level to be determined;
   PXQ( POOL, QM): QP;

  ! Set of which RM can go into which POOL;
   RXP( RM, POOL): R2P;

  ! Set of which POOL can go into which FG;
   PXF( POOL, FG): P2F;

  ! Set of which RM can go to a FG;
   RXF( RM, FG): R2F, R2FUB;
 ENDSETS

 DATA:
! Case 1: This data set based on a Haverly example;
! It has local optima at 0, 300, and 400 (Global);
!CH1;  RM=  RMA    RMB    RMC;  ! Set of raw materials;
!CH1;  A = 9999   9999   9999;  ! Raw matl availabilities;
!CH1;  COST = 6     16     10;  ! R. M. costs;

!CH1; QM = SULFUR;              ! Set of qualities;
!CH1; QL =  .03   .01    .02;  ! Quality by R.M.;


!CH1; FG =   FGX   FGY;
!CH1;  D =    0,    0;        ! Min needed of each F.G.;
!CH1;  E =   100   200;       ! Max sellable of each F.G;
!CH1;  PRICE=  9     15;      ! Selling price of each F.G.;

!CH1;  U =   .025  .015;      ! Upper limits on quality for each FG.;
!CH1;  L =      0    0 ;      ! Lower limits on quality...;

!CH1; POOL  = POOLAB  POOLC_; ! The names of the pools;
!CH1; BPUB  = 999999  999999; ! UL on amount in pool;

! Which R.M. can go into which POOLS;
!CH1; RXP=
```

```
     RMA   POOLAB
     RMB   POOLAB
     RMC   POOLC_;

! Which pools go into which F.G.;
!CH1; PXF = POOLAB FGX
            POOLAB FGY
            POOLC_ FGX
            POOLC_ FGY;

! Which RM can go into which FG;
!CH1;   RXF = ; ! An empty set;

! This data set has several local optima:
 ! Local optimum 1:
     Obj = 0,
     Do not buy or sell anything,
     BP( ) = BF( ) = 0,

   Local optimum 2:
     Obj = 300,
     R2P( RMA, POOLAB) =   50.0000
     R2P( RMB, POOLAB)  = 150.0000
     R2P( RMC, POOLC)   =   0.0000
     P2F( POOLAB, FGX)  =   0.0000
     P2F( POOLAB, FGY)  = 200.0000
     QP( POOLAB, SULFUR)=   0.0150

   Local optimum 3 (Global):
     Obj = 400,
     R2P( RMA, POOLAB)  =   0.0000
     R2P( RMB, POOLAB)  = 100.0000
     R2P( RMC, POOLC)   = 100.0000
     P2F( POOLAB, FGX)  =   0.0000
     P2F( POOLAB, FGY)  = 100.0000
     P2F( POOLC, FGX)   =   0.0000
     P2F( POOLC, FGY)   = 100.0000
     QP( POOLAB, SULFUR)=   0.0100
     QP( POOLC, SULFUR) =   0.0200
;
! This data set based on Audet et al.;
! Optimal solution is 4391.826;
 ! Set of raw materials, Availabilities and Cost;
!CH2   RM,      A,    COST =
       RMA    60.9756   49.2
       RMB   161.29    62
       RMC     5      300;

!CH2   QM = DEN  BNZ RON  MON;    ! Set of qualities;
! Quality by 1st R.M.;
!CH2   QL = .82   3  99.2 90.5
            .62   0  87.9 83.5
            .75   0  114  98.7;

 ! Finished goods, min/max needed, selling price;
!CH2   FG , D,   E,  PRICE =
       FG1  5  9999   190
       FG2  5  9999   230
       FG3  5  9999   150;

! FG Lower limits on qualities, 1, 2, ...;
!CH2    L =  0.74 0.74 0.74
              0     0    0
             95    96   91
             85    88    0;
! FG Upper limits Quality x FG.;
!CH2    U =  .79  .79  .79
            9999  0.9  9999
```

```
            9999  9999 9999
            9999  9999 9999;

!CH2   POOL  = POOL1  POOL2; ! The names of the pools;
!CH2   BPUB  = 12.5   17.5; ! UL on amount in pool;

! Which R.M. can go into which POOLS;
!CH2   RXP=
     RMA   POOL1
     RMB   POOL1
     RMC   POOL1
     RMA   POOL2
     RMB   POOL2
     RMC   POOL2
     ;
! Pairs of Which pools go into which F.G.;
!CH2   PXF = POOL1 FG1
            POOL1 FG2
            POOL1 FG3
            POOL2 FG1
            POOL2 FG2
            POOL2 FG3;

! RM directly to FG with upper bounds on flow;
!CH2     RXF,    R2FUB =
         RMA FG2  7.5
         RMB FG1 9999
         RMB FG3 9999
         RMC FG1  7.5;

 ENDDATA
!---------------------------------------------------------------;
SUBMODEL PoolingOpt:

! Variables:
   R2P(r,p) = amount of raw material r transferred to pool p,
   P2F(p,f) = amount of material transferred
              from pool p to finished good f,
   R2F(r,f) = amount from RM r to FG f,
   QP(p,q) = level of quality q (e.g., fraction sulfur)
             in pool p,
   BP(p) = batch size (or amount in) of pool p,
   BF(f) = batch size (or amount in) of finished good f;

! The model;
!  This is the so-called P formulation. Other formulations, e.g.,
   the Q formulation, may be tighter;
   ! Max revenues - cost of raw materials;
       [PROFIT] MAX = OBJ;
           @FREE( OBJ);
           OBJ = REVFG - COSTRM;
           REVFG = RevR2F + RevP2F; !@SUM( FG(f): PRICE(f) * BF(f));
           COSTRM = CostR2F + CostR2P; ! @SUM( RXP(r, p): COST( r)* R2P( r, p))
                  + @SUM( RXF(r, f): COST( r)* R2f( r, f));
          RevR2F  = @SUM( RXF(r, f): PRICE( f) * R2F( r, f));
          RevP2f  = @SUM( PXF( p, f): PRICE( f) * P2F( p, f));
          CostR2F = @SUM( RXF(r, f):  COST( r) * R2F( r, f));
          CostR2P = @SUM( RXP(r, p):  COST( r) * R2P( r, p));

   ! Raw materials stage: raw material availabilities;
    @FOR( RM( r):
         [RMLIM] @SUM( RXP( r, p): R2P( r, p))
               + @SUM( RXF( r, f): R2F( r, f)) <= A( r);
        );

   ! Pools stage;
      @FOR( POOL(p):
   !  How much ( the batch size) is in each pool;
```

```
      [INPOOLI]  BP( p) = @SUM( RXP( r, p): R2P( r, p)); !Flow in;
      [INPOOLO]  BP( p) = @SUM( PXF( p, f): P2F( p, f)); !Flow out;
      [INPOOLUB] BP( p) <= BPUB( p);

   ! The quality level, QP(p,q), of each pool.
      The model is nonlinear and nonconvex because
      of the product terms QP( p, q) * BP( p).
      If the RHS  (and BP) = 0, then we want QP = 0, thus, the epsilon term;
!  We should branch on BP rather than QP? ;
      @FOR( QM( q):
       [QPOOL] QP( p, q)* ( BP( p) + 0.0000001)  ! Nonlinear;
                 = @SUM( RXP( r, p): QL( r, q)* R2P( r, p));
          );
         );

! Raw Materials to Finished Goods upper bound on flow;
      @FOR( RXF( r, f): R2F( r, f) <= R2FUB( r, f));

   ! Finished goods stage;
    @FOR( FG( f):
     ! Batch size computation;
         [BDEFO] BF( f) = @SUM( PXF( p, f): P2F( p, f))
                        + @SUM( RxF( r, f): R2F( r, f));
     ! Batch size limits;
         [BLO] BF( f) >= D( f) ;
         [BHI] BF( f) <= E( f) ;

     ! Quality restrictions for each FG f and quality q,
      The model is nonlinear and nonconvex because
      of the product terms QP( p, q) * P2F( p, f);
       @FOR( QM( q)  :
!  If there are more qualities than finished goods, then one should probably
   branch on R2F;
          [QDF] @SUM( PXF( p, f): QP( p, q) * P2F( p, f))   !Nonlinear;
              + @SUM( RXF( r, f): QL( r, q) * R2F( r, f)) = S( q, f);
                @FREE( S( q, f));
          [QUP] S( q, f) <= U( q, f) * BF( f) ;
          [QDN] S( q, f) >= L( q, f) * BF( f) ;
         );
       );
  ENDSUBMODEL

  CALC:
    @SET( 'TERSEO',1); ! Output level (0:verbose, 1:terse, 2:only errors, 3:none);
    @SET( 'GLOBAL', 1);! 0:Do not use Global solver, 1:Use the Globasolver;
    @SET( 'TATSLV', 60); ! Solver time limit in seconds (0:no limit) for @SOLVE's;

 !   @DEBUG( PoolingOpt);
 !   @GEN( PoolingOpt);

    @APISET( 6411, 'INT', 6); ! LS_IPARAM_GOP_POSTLEVEL optimization based bound
tightening (OBBT);
    @SOLVE( PoolingOpt); ! Solve a specified model;
    @WRITE( ' Profit = ', @FORMAT( Obj, '12.2f'), @NEWLINE( 1));

    @WRITE( ' Flows from Raw Materials to Pools', @NEWLINE( 1));
    @FOR( RXP( r, p):
      @WRITE( '  ', RM( r), '  ', POOL( p),'  ', @FORMAT( R2P( r, p),'12.6f'), @NEWLINE(
1));
         );

    @WRITE( @NEWLINE(1),' Flows from Pools to Finished Goods', @NEWLINE( 1));
    @FOR( PXF( p, f):
      @WRITE( '  ', POOL( p), '  ', FG( f),'  ', @FORMAT( P2f( p, f), '12.8f'), @NEWLINE(
1));
         );

    @WRITE( @NEWLINE(1),' Flows from Raw Materials to Finished Goods', @NEWLINE( 1));
```

```
    @FOR( RXF( r, f):
      @WRITE( '  ', RM( r), '  ', FG( f),'  ', @FORMAT( R2f( r, f), '12.6f'), @NEWLINE(
1));
         );

    @WRITE( @NEWLINE(1),' Quality, Finished Good, Low Limit,   Actual, Upper Limit',
@NEWLINE( 1));

!   QXF( QM, FG): U, L, S;
    @FOR( QXF( q, f) | BF( f) #GT# 0:
      TEMP = S( q, f)/ BF( f);
      @WRITE( '  ', QM( q), '        ', FG( f), '     ',
        @FORMAT( L( q, f), '12.3f'), ' ', @FORMAT( TEMP, '12.6f'),' ', @FORMAT( U( q, f),
'12.3f'), @NEWLINE( 1));
         );

  ENDCALC
 END
```