```
! Blending gases based on the Wobbe index.
 When you turn on your gas stove (or gas turbine if you
are in industry) each morning and you set the dial to "5",
you would like to get the same amount of heat from the burner
from one day to the next, even though the composition of the
"natural" gas feeding your appliance may vary from day to day.
  If the Wobbe index of the gas remains constant from day to day
then the heat output of your appliance will remain the same from
day to day. The Wobbe index is a measure of the heat content
of the gas that flows through an orifice of constant size under
constant pressure per unit of time.
Based on elementary physics, Goffredo Wobbe in 1927 deduced
that said heat content is a) proportional to the heat content of
a cubic foot (or meter) of gas at
standard temperature and pressure (STP), and
b) inversely proportional to the square root of the density
(or specific gravity) of the gas.
I.e., at STP, lighter gas flows through the orifice
faster than heavier gas;
! Keywords:  Blending, BTU, Energy content, Fuel, Gas blending, Heat content, LINGO,
    Natural gas, Specific gravity, STP, Turbine, Wobbe index;
sets:
  gas: btupcf, spg, avail, x;
endsets
data:
 ! The gases of interest, their heat content in
     BTU per cubic foot, and specific gravity
     relative to air, cubic feet available per unit time;
   gas,     btupcf,  spg, avail=
   argon       0    1.38   10
   hydrogen  325    0.07   10
   helium      0    0.138  45
   methane  1012    0.55  120
   ethane   1773    1.04  235
   propane  2522    1.52  827
   butane   3225    2.01  395
   natgas   1025    0.65  690
   co2         0    1.52  150
   nitrogen    0    0.97   80
   ;
! We want the Wobbe index in this interval;
  WOBL = 1500;
  WOBU = 1600;
enddata

submodel Wobbe:
 ! Availability limits;
  @for( gas(i):
    x(i) <= avail(i);
      );

 ! batch size;
    blendtot = @sum( gas(i): x(i));
 ! Total heat content;
    heattot = @sum( gas(i): btupcf(i)*x(i));
 ! Total weight in spg;
    spgtot = @sum( gas(i): spg(i)*x(i));
 ! spg per cf at STP;
    spgcf = spgtot/blendtot;
 ! heat content per cf at STP;
    heatcf = heattot/blendtot;
 ! Wobbe index;
    wobbei = heatcf/(spgcf^0.5);
 ! bounds on the Wobbe index;
    wobbei >= WOBL;
    wobbei <= WOBU;
 ! A plausible objective is to maximize the energy content;
    max = heattot;
```

```
endsubmodel


CALC:
! Set some parameters before the solve;
  @SET( 'TERSEO',1); ! Output level (0:verbose, 1:terse, 2:only errors, 3:none);
  @SET( 'IPTOLR', 0.0001);! Set IP ending relative optimality tolerance(Should be >0);
  @SET( 'TATSLV', 900); ! Solver time limit in seconds (0:no limit) for @SOLVE's;
  @SET( 'GLOBAL', 0);! 0:Do not use Global solver, 1:Use the Globasolver;

  @SOLVE( Wobbe);  ! Solve a specified submodel;
  ISTAT = @STATUS();! 0: Optimal to tolerance. 1: infeasible, 2: unbounded,
                      3: undetermined, 4: Feasible,  5: Infeasible/unbounded in
preprocessor,
                      6: Local optimum, 7: locally infeasible, 8: Objective cutoff
reached,
                      9: numeric error;

 @WRITE( 'Solution Status = ', ISTAT, @NEWLINE( 1));
 ! Display a very simple solution report;
 @WRITE( @NEWLINE( 1), '  A Simple Customized Solution Report', @NEWLINE( 1));
 @WRITE( 'Objective= ', heattot, @NEWLINE(1));

 @WRITE( '       Gas    Amount to use ', @NEWLINE(1));
 @FOR( gas( i) | x( i) #GT# 0:
! Can format output if desired;
 @WRITE( @FORMAT( gas( i),'10s'), ' ', @FORMAT( x, '12.3f'), @NEWLINE(1));
     );

ENDCALC
```