Bruce McCarl's GAMS Newsletter Number 41

July 2017

This Newsletter addresses the following

Contents

1 Scaling

Sometimes models do odd things like reporting problems as infeasible, stuck or falsely optimal when scaling is the real issue. To avoid this or correct such issues it is often desirable to check scaling and in turn rescale the model or ask the solvers to employ more aggressive scaling.

In terms of solver scaling most LP/MIP solvers do automatic scaling and a number have the option to apply a more aggressive scaling to numerically difficult models, e.g. Cplex option scanind, Gurobi option scaleflag, or Xpress option scaling. However, modelers can typically do better because they know how certain variables and equations are interrelated and can scale with common factors while the solvers do not have that knowledge. Generally, for difficult problems it is desirable to do the scaling as discussed below then let the solver scale as well. Scaling is typically not a concern for small problems.

Ideally, in doing a scaling exercise, one should modify the model so the absolute value of the constraint matrix coefficients are centered around one including the derivatives of any nonlinear terms (at the starting point). This involves manipulating variable and equation scaling. The target of this manipulation is after scaling when dividing the largest matrix coefficient by the smallest the ratio should be no more than 1000 to 10000 ie the coefficient would span from 0.01 to 100 or something like that. Arne Drud author of CONOPT in his CONOPT3 solver manual indicates

- Basic and superbasic solution values are expected to be around 1, e.g. from 0.01 to 100. Nonbasic variables will be at a bound, and the bound values should not be larger than say 100.
- Dual variables (or marginals) on active constraints are expected to be around 1, e.g. from 0.01 to 100. Dual variables on non-binding constraints will of course be zero.
- Derivatives (or Jacobian elements) are expected to be around 1, e.g. from 0.01 to 100.
- You should select the unit of measurement for the variables so their expected value is around unity.
- After you have selected units for the variables you should select the unit of measurement for the equations so the expected values of the individual terms are around one.

One can do this external to GAMS or can employ scaling within GAMS. The basic concept is discussed in chapter 17 of McCarl and Spreen here and involves altering all coefficients for a variable by the same amount and doing the same for all coefficients in an equation. Consider the following example

$$\begin{array}{llrcrcrcrcr}
\text{Max} & X_1 - & 500X_2 & - & 400X_3 & - & 500X_4 & & \\
\text{s.t.} & X_1 - & 10000X_2 & - & 8000X_3 & & & \leq & 0 \\
 & & 5X_2 & + & 4X_3 & - & 50X_4 & \leq & 0 \\
 & & 1500X_2 & + & 2000X_3 & & & \leq & 1200 \\
 & & 50X_2 & + & 45X_3 & & & \leq & 60 \\
 & X_1 & X_2 & & X_3 & & X_4 & \geq & 0
\end{array}$$

which can be scaled to become

$$\begin{array}{llcrcrcrcr}
\text{Max} & 20X_1 & - & X_2 & - & X_3 & - & 0.1X_4 & & \\
\text{s.t.} & X_1 & - & X_2 & - & X_3 & & & \leq & 0 \\
 & & & X_2 & + & X_3 & - & X_4 & \leq & 0 \\
 & & & X_2 & + & 1.667X_3 & & & \leq & 0.8 \\
 & & & X_2 & + & 1.125X_3 & & & \leq & 1.2 \\
 & X_1, & & X_2, & & X_3, & & X_4 & \geq & 0
\end{array}$$

This would be done by

- Dividing the coefficients in the first constraint by 1000.
- Dividing the coefficients in the second constraint by 5
- Dividing the coefficients in the third constraint by 1500,
- Dividing the coefficients in the fourth constraint by 50.
- Dividing the coefficients in the objective coefficient by 50.
- Multiplying the coefficients for X_1 by 1000
- Leaving the coefficients for X_2 alone
- Multiplying the coefficients for X_3 by 1.25
- Multiplying the coefficients for X_4 by 1/10

Both the original model and the manually scaled one are in the example scale2.gms.

In turn after solution of this scaled model, one can recover the scaled solution by appropriately multiplying and dividing by the scaling factors as explained in McCarl and Spreen. Specifically one would recover the solution to the original model by multiplying the scaled model solution values for variables by the variable scaling factor (so X_1 would be the scaled solution level for X_1 times 1000). Similarly the variable marginals would be multiplied by the objective function scaling factor in turn divided by the variable scaling factor (so for X_1 the marginal would be the scaled marginal times 50/10000). At the same time, the equation slacks or levels would be multiplied by the row scaling factor (for the first equation multiplied by 1000) and the equation marginals times the objective function scaler divided by the constraint scaler (for equation 1 multiply by 50/1000). Finally, the objective function value would be multiplied by the objective function scalar.

Two questions arise from this. First, do you really need to do the manual scaling and descaling? Second, how do I form the scaling factors?

On the first question, you are in luck. GAMS has an easy way of specifying the scaling factors and automatically descales the solution so no manual procedures are required. This is covered in the McCarl Expanded User Guide in the scaling section. For the example above the example scale2.gms contains an implementation of the GAMS scaling procedure and the relevant part is.

```
PARAMETER SCALEPROC(PROCESS)  VARIABLE SCALING
           / x1 10000, x2        1, x3        1.25,x4        0.1/
PARAMETER SCALERESOR(RESOURCE)  EQUATION SCALING
   /c1 10000, c2 5, c3 1500,c4 50/;
scalar rhsscale /1/
scalar objscale /500/;
PRODUCTION.scale(PROCESS)=SCALEPROC(PROCESS);
objt.scale=objscale;
AVAILABLE.scale(RESOURCE)=SCALERESOR(RESOURCE);
RESALLOC.scaleopt=1;
```

Where the code specifies the scaling factors, copies them into the GAMS scaling mechanism in turn specifies a model attribute that tells GAMS to solve the scaled model.

On the second question, one can get support on identifying coefficient magnitudes via the following approaches:

- One can following Drud's CONOPT3 document, use LIMROW, LIMCOL and search for large or small magnitude numbers within columns and rows then devise scaling factors.
- One can use GAMSCHK (see the solver manual on this here) with first the BLOCKLIST or BLOCKPIC commands to find blocks of variables or equations with large or small numbers or consistent departures from values of one and then later MATCHIT to find individual cases. These GAMSCHK procedures give largest and smallest magnitudes of coefficients in equations and variables so you don't have to search. This can also be supported by use of DISPLAYCR.
- Additionally GAMS personnel mentioned use of the CONVERT solver with the option 'jacobian'. This gives a gdx file with that contains the matrix coefficients which for nonlinear terms the gradients of the nonlinear terms (evaluated at the starting point- as limrow/col and GAMSCHK does). Then one can export that file to Excel and manipulate. Note when doing this Convert replaces all the internal variable and equation names. Subsequently the variables are named x1,x2,x3,... and the equations e1,e2,e3,...> To get back to the original names one needs to manually back translate using the CONVERT generated dictionary file. I have no experience with this but it just does not seem practical in large models as the GDX and Excel might be unwieldy and the back translation awkward.
- GAMS personnel also told me about a new option in Cplex 12.7 has a new option (datacheck=2) that "looks and reports" suspicious item in a model. In turn the log file (not the LST file) contains information about large and small coefficients, rhs values and

anything else Cplex thinks can cause numerical instability. I tried this on what I considered an intermediate sized model and got messages like

```
CPLEX Warning  1043: Detected righthand side <= CPX_FEAS_TOL at constraint 'AGTILLSTART(US.TxTranspec.Cropland.Zero)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_White_Fla.2)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_White_Fla.18)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_White_Fla.26)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_White_Fla.41)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_White_Fla.45)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_White_Fla.49)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_White_Fla.60)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_White_Fla.71)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_Red_Tex.2)'.
CPLEX Warning  1045: Detected nonzero <= CPX_CANCEL_TOL at constraint 'WELFAR', variable 'AGDEMANDS(2015.US.dom_demand.GrpfrtFrsh_Red_Tex.18)'.
CPLEX Warning  1045: Too many warnings of this type have been detected.  All further warnings of this type will be ignored.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'WELFAR' are fractions and can be scaled with 17/1.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'AGPRODBAL(2015.US.CB.RefSugar.base)' are fractions and can be scaled with 47/1.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'AGPRODBAL(2015.US.CB.CornforDairyCattle.base)' are fractions and can be scaled with 70/1.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'AGPRODBAL(2015.US.GP.Oats.base)' are fractions and can be scaled with 71/1.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'AGPRODBAL(2015.US.GP.Canola.base)' are fractions and can be scaled with 11/1.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'AGPRODBAL(2015.US.GP.RefSugar.base)' are fractions and can be scaled with 47/1.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'AGPRODBAL(2015.US.GP.CornforDairyCattle.base)' are fractions and can be scaled with 70/1.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'AGPRODBAL(2015.US.LS.Oats.base)' are fractions and can be scaled with 71/1.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'AGPRODBAL(2015.US.LS.RefSugar.base)' are fractions and can be scaled with 47/1.
CPLEX Warning  1047: Decimal part of coefficients in constraint 'AGPRODBAL(2015.US.LS.CornforDairyCattle.base)' are fractions and can be scaled with 70/1.
CPLEX Warning  1047: Too many warnings of this type have been detected.  All further warnings of this type will be ignored.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'WELFAR' the ratio of largest and smallest (in absolute value) coefficients is 4.48036e+020.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'AGPRODBAL(2015.US.CB.Hay.base)' the ratio of largest and smallest (in absolute value) coefficients is 100000.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'AGPRODBAL(2015.US.CB.CottonseedMeal.base)' the ratio of largest and smallest (in absolute value) coefficients is 100000.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'AGPRODBAL(2015.US.CB.CottonseedOil.base)' the ratio of largest and smallest (in absolute value) coefficients is 776300.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'AGPRODBAL(2015.US.GP.Hay.base)' the ratio of largest and smallest (in absolute value) coefficients is 100000.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'AGPRODBAL(2015.US.GP.CottonseedMeal.base)' the ratio of largest and smallest (in absolute value) coefficients is 100000.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'AGPRODBAL(2015.US.GP.CottonseedOil.base)' the ratio of largest and smallest (in absolute value) coefficients is 776300.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'AGPRODBAL(2015.US.LS.Silage.base)' the ratio of largest and smallest (in absolute value) coefficients is 101350.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'AGPRODBAL(2015.US.LS.CottonseedOil.base)' the ratio of largest and smallest (in absolute value) coefficients is 776300.
CPLEX Warning  1048: Detected constraint with wide range of coefficients. In constraint 'AGPRODBAL(2015.US.NE.CottonseedOil.base)' the ratio of largest and smallest (in absolute value) coefficients is 776300.
CPLEX Warning  1048: Too many warnings of this type have been detected.  All further warnings of this type will be ignored.
```

Across these, not surprisingly since I wrote it, I prefer the GAMSCHK where BLOCKLIST and MATCHIT tell you where the big and small numbers are and then I use targeted displays (through DISPLAYCR) to look at things. The CPLEX information backed by GAMSCHK DISPLAYCR also looks god but is not available for other solver.

Note when using the GAMS internal scaling feature the all of the above mentioned procedures report out the coefficients after the GAMS internal scaling has been applied.

The above GAMSCHK features are implemented at the bottom of the scale2.gms example and LIMROW is set large enough in there to display the model.

2 SCIP solver

In recent releases GAMS has modified the default LP solver that will be used with SCIP. Originally, it was CPLEX then for a while then it was changed to the free solver SOPLEX. Now it uses CPLEX if a license is present and SOPLEX otherwise.

Also while SCIP is free to those doing research or education others need a commercial license. that GAMS now has available.

3 Are you tired of fighting with "Domain Errors"

Sometimes modelers wish to draw elements form one set that are in another but the two sets are independent in GAMS and thus one cannot simply index both sets (yes sameas can be used but here we discuss another way of doing this).

GAMS contains the $onuni commands that stops GAMS from doing domain checking and $offuni that restarts it. Sometimes this may be useful. However note that maintaining domain checking helps avoid serious errors and should generally occur. In particular, suppose we read data that pertains to two independent but existing sets. In that case one could go back and define either a superset or make the sets in question subsets of the universal set but the latter may require one to restart the whole process and a quicker and dirtier solution offered by the $onuni $offuni syntax may be desirable. Consider the example below

```
set fruit   / apple, pear /
    veggie  / carrot, pea /
    produce / #fruit, #veggie /;
parameter   produceCalories(produce) per 100g
                 / apple 52, pear 57, carrot 41, pea 81 /
            fruitcal(fruit) calories per 100g of fruit
            vegcal(veggie) calories per 100g;
$onuni
     fruitcal(fruit) = produceCalories(fruit);
     vegcal(veggie)  = produceCalories(veggie);
$offuni
display fruitcal, vegcal;
```

Note in this case we are trying to pull the data on fruit calories from the producecalories parameter and ditto for the vegetable calories items. However, while the set produce is the union of the entries for fruit and vegetables that the fruit and veggie sets are **not subsets**. Thus the commands

```
            fruitcal(fruit) = produceCalories(fruit);
            vegcal(veggie)  = produceCalories(veggie);
```

would give domain errors. But adding $onuni before these commands suspends domain checking allowing the commands to work. Subsequent use of $offuni reinstates the domain checking. It is recommended that you only do this in special cases and for a few statements as possible so insert the $offuni as soon as possible.

4 Including a whole set when defining another

The commands appearing above

```
set fruit   / apple, pear /
    veggie  / carrot, pea /
    produce / #fruit, #veggie /;
```

contain a syntax worth mentioning. In particular in the third command the entries `#fruit`, `#veggie` cause the entire contents of the sets fruit and veggie to be defined within the produce set.

5 BASIC and Advanced courses offered soon

I will be teaching

- Basic to Advanced GAMS class Aug 7, 2017- Aug 11, 2017 (5 days) in the Colorado mountains at Frisco (near Breckenridge). The course spans from Basic topics to an Advanced GAMS class. Details are found at http://www.gams.com/courses/mccarl_combined.pdf
- Basic GAMS class Aug 7, 2017- Aug 9, 2017 (3 days) in the Colorado mountains at Frisco (near Breckenridge). The course starts assuming no GAMS background. Details are given at http://www.gams.com/courses/mccarl_basic.pdf.

- Advanced GAMS class Aug 9, 2017- Aug 11, 2017 (3 days) in the Colorado mountains at Frisco (near Breckenridge). The course is for users who have a GAMS background. Details are found at http://www.gams.com/courses/mccarl_advanced.pdf .

Further information and other courses are listed on http://www.gams.com/courses.htm . Note I also give custom courses for individual groups a couple of times a year.

6 Unsubscribe or subscribe to future issues of this newsletter

Please unsubscribe through the web form available at:
http://app.streamsend.com/public/XLmY/5eq/subscribe

Those who wish to subscribe to future issues can do this through the newsletter section of http://www.gams.com/maillist/index.htm.

This newsletter is not a product of GAMS Corporation although it is distributed with their cooperation.

July 3, 2017