Bruce McCarl's GAMS Newsletter Number 45
June 2020

This Newsletter addresses the following

1 Notable developments in GAMS releases 30-31

Since the last newsletter GAMS has released versions 30 and 31 with release notes being at https://www.gams.com/latest/docs/RN_MAIN.html. When I peruse these notes, things that strike of broad interest are:

1.1 GAMS system

1.1.1 Domain for variables

GAMS extended the syntax of the **model statement** to allow one to **restrict the domain of variables** used in the model at one spot. To do this one needs to define a set with same dimensions as the variable and then add a reference to the variable and its domain definition in the model statement. For example, in a GMS file for a variable TRAN(supply,demand,mode) one would define a set for the applicable domain like allowtran(supply,demand,mode) then in the model statement add the domain definition as follows

```
model mytran /all,TRAN(allowtran)/;
```

This is the same as imposing the conditional TRAN((supply,demand,mode)$allowtran(supply,demand,mode) everywhere the TRAN variable is used.

I do find the implementation that mandates the domain inclusions in the model statement a little awkward and I believe this will create long model statements.

1.1.2 License file location

GAMS has done a good job of locating the license file in a more generic location. For years, the license file has had to be placed in the GAMS system directory. However, now there is a more generic place so that you do not have to copy it or point the IDE or GAMS to a different license location every time a new update comes out. The places where GAMS will search for the license can be are revealed by running

```
gamsinst  -listdirs
```

and on my windows 10 machine these are

```
C:\Users\mccarl\Documents\GAMS
C:\Users\mccarl\AppData\Roaming\GAMS
C:\ProgramData\GAMS
C:\GAMS\win64\31.1\data
C:\GAMS\win64\31.1\data\GAMS
C:\GAMS\win64\31.1
```

Here the first three are persistent and do not vary with GAMS installation version. I also urge GAMS to include the location C:\GAMS or C:\GAMS\win64 as that is where I think of as the GAMS installation location on a Windows machine.

You can also continue to use the **license** command line parameter or IDE option to point to a specific license file.

1.1.3 Generic yaml GAMS command line, and options customization procedure

For years, I have been frustrated with customization of GAMS. GAMS has long allowed p one to customize things like the default solver, LIMROW, LIMCOL, SCALEOPT, OPTCR and SOLPRINT by manipulating a file in the systems directory only to have the customization left behind when I got an update that used a new location for the systems directory. However, the new releases allow use of a more generic place and procedure. In particular, GAMS now reads a configuration file called gamsconfig.yaml where one can specify default values, command line parameters, environment variables and option as detailed in **GAMS Configuration in YAML Format**.

This can be used to implement user-specific default option settings and if located in a space other than the system directory are shared with future GAMS updates. GAMS also plans to use that in the future to "modernize" some of the defaults while also providing a backwards compatibility pathway.

The places where GAMS will look for the gamsconfig.yaml are also revealed by running

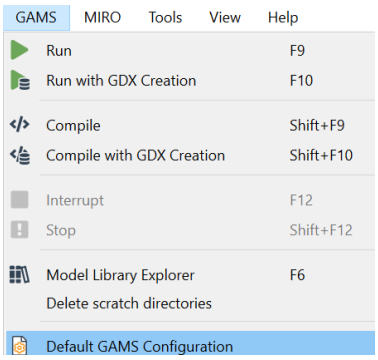
```
Gamsinst  -listdirs
```

And on my Windows 10 machine are

```
C:\GAMS\win64\31.1
C:\ProgramData\GAMS
C:\Users\mccarl\AppData\Local\GAMS
C:\Users\mccarl\Documents\GAMS
```

where the last three do not vary with the current system directory. Again, I would have also liked to see the search include the location C:\GAMS or C:\GAMS\win64 as that is where I think of as the GAMS installation location.

I did note in looking at this that this introduces an apparently widely used yaml approach but in terms of GAMS this introduces a fourth format with which things like LIMROW and LP and SOLPRINT can be specified (their values can be changed through the command line, option statement, model attribute and now the yaml file). Fortunately, GAMS introduced a yaml file editor in STUDIO that knows the yaml format and the allowable settings so you do not have to learn all the nuances of yet another way of doing this. The customization editor is accessed by opening a yaml file or through the GAMS menu choice under Default GAMS Configuration as below.



On my machine a newly created yaml file is saved in C:\Users\mccarl\Documents\GAMS.

Note that the exact settings that will be used in a run are subject to a precedence order as follows:

1. Specification within a GMS file via the latest Option or model attribute setting command
2. Specification on the command line in the STUDIO/IDE command line box.

3. Specifications in the IDE ***Execute*** box (this is reached through File|Options|Execute).
4. Specifications in other IDE dialogs, e.g. Output (this is reached through File|Options|...).
5. Entries in the GAMS configuration file gamsconfig.yaml.
6. Entries in the GAMS system parameter file (gmsprmNT.txt on windows in the GAMS system directory i.e. 31.1).

1.1.4 Data encryption

GAMS has introduced new commands **encryptKey** and **decryptKey** that result in the encryption and decryption of files based on a user specified key. This allows developers to create **encrypted input files** that can be used under any GAMS license with the right key.

1.1.5 Loading from GDX where domains don't match

GAMS has provided an alternative version of **$load** that is called **$loadfiltered**, that operates essentially the same but ignores set domain violations. That means, when items are loaded, **domain violations** are not flagged as compilation errors. Rather items violating the domain definition are ignored. This is the opposite of **$loadDC**. All other features are the same as discussed under **$load**.

1.2 Solvers

Many solvers were updated with new libraries. In terms of large changes that I see, the SCIP solver had many features added and a new solver (SHOT) was added that solves MINLP problems.

1.3 GAMS IDE and GDXVIEWER are being phased out

GAMS intends to phase out the IDE and GDXVIEWER. GAMS switched the default application for GMS files on Windows from the IDE to **GAMS Studio**. They plan to drop updating for these packages in the very near future if not already due to compilers becoming obsolete. Also in their last release notes state they will drop the GAMS IDE and GDXVIEWER with an upcoming major release although I have been told this will not be in the near future. STUDIO will replace the IDE. Capabilities in STUDIO and a new standalone will replace GDXVIEWER.

1.4 Larger DEMO limits and DEMO licenses

GAMS has changed the capability of system operation in free demonstration mode in two fundamental ways.

- The size of models that can be solved with a free, demo license has been expanded. Now GAMS will generate linear (LP, RMIP, MIP) models that contain up to 2000 constraints or 2000 variables and other model types up to 1000 constraints or 1000 variables. No other limits are applied (e.g. no limits on non-zeros or discrete variables). However, some solvers impose **additional restrictions** (e.g., the global solvers ANTIGONE, BARON, and LindoGlobal).

- GAMS now requires Demo users to have obtained a free, demo license. The licenses are time-limited for 12 months and can be directly generated by the user through the GAMS download page. The user is also required to use a recent GAMS version (no older than 18 months). Users are to comply with the restriction that the demo license is for demonstration and evaluation but not for commercial and production work. The **license documentation** contains details.

1.5 Free community license

GAMS now provides a free **community license** for users on request. This community license allows generating even larger models: up to 5000 constraints and variables for LP, MIP, and RMIP models. For all other model types, the model cannot be larger than 2500 variables or 2500 constraints. In addition, limits that are more stringent are enforced by some solvers, the license works for 12 months, and the GAMS system used cannot be older than 18 months. The license again is for demonstration and evaluation but not for commercial and production work. For details, see GAMS End User License Agreement.

1.6 GAMS STUDIO

Studio has been augmented in a number of ways. One recent augmentation is the inclusion of a procedure that identifies any differences between two GDX files. In particular, they added a **GDX Diff dialog** . This is opened by choosing Tools then GDX Diff from the menu. It can also be opened through the STUDIO Explorer when one right clicks on a GDX file then chooses the procedure.

1.7 GAMS MIRO

GAMS MIRO has been officially released. It provides a deployment environment for GAMS models. It is downloaded through https://www.gams.com/miro/download.html. MIRO is open source and can work on a standalone basis. However, use jointly with GAMS requires a proprietary connector. Use of it connected to your local GAMS does not require a separate license for academics but does for commercial users. There is also a cloud version that has a separate license arrangement.

2 A new web page

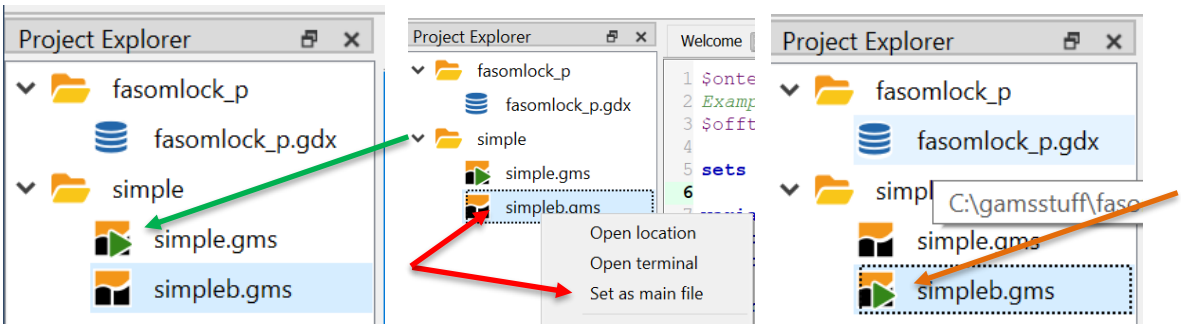
GAMS launched a new version of their website this week. They did this to simplify the website and make it easier to maintain and less vulnerable to attacks. Visit www.gams.com to see the new one. I note I cannot find certain things either easily or in a consistent fashion. For example, the MIRO documentation and download is under products while the counterparts for the GAMS system are under the main menu. Also, contributed documentation and software were moved to https://forum.gamsworld.org/. Finally I can't find any mention of; a) reference to or signup for GAMS-List; b) cases; c) presentations; and d) mention of the GAMS library.

3 Some issues I have had with STUDIO

Given the GAMS is planning to phase out the IDE plus the emerging capabilities of STUDIO, I have begun to use STUDIO. In doing this, I ran into a couple of difficulties that have work arounds that I believe are worth sharing.

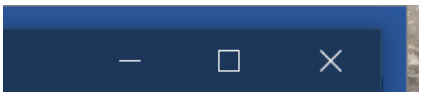
First, I have had trouble with the concept of a main file as opposed the IDE project. Namely, suppose you have a bunch of GMS files in a directory and you are working on one then go to run it but find STUDIO runs another file. Why? This is a characteristic of STUDIO in that it always solves what is called the main file (see the explanation at https://www.gams.com/latest/docs/T_STUDIO.html#STUDIO_PROJ_VS_GROUPS).

Under STUDIO, when you are working in a group of files STUDIO will have one, typically the first one you worked with in that area, marked as the main file (the one with the green triangle in the left screen shot below). In that case, if you want to solve another file in that group you must redefine what STUDIO considers to be the main file. To do this you right click on the desired file in that group and choose **set as main**. This moves the green triangle redefining the main file and then this will be the file run in future executions.



Second, on several occasions, I had STUDIO fail in the middle of a session just closing down and then have had it refuse to restart. I do not know why this happened and have been unable to reproduce the exact situation leading to this. Nevertheless, for this newsletter and future use I wanted to cover a general way to get STUDIO back to working (At the time I gave up and reverted to the IDE). Thus I asked one of the GAMS developers how to get it restarted and received the following. *The most general approach would be to start Studio from the command line with the --ignore-settings flag or with --reset-settings (the first will ignore your last settings for just one session; the latter will delete them completely).* On my windows machine I the found I could run either through the MSDOS window in the IDE or with "windows run" the command C:\GAMS\win64\31.1\studio\studio.exe --ignore-settings

Third, again on several occasions, I have had STUDIO place its window partially off the screen losing access to the top bar with the items clipped out below becoming invisible – off the screen.



Further, the screen was totally filled up and I could not reach the sides or bottoms of the STUDIO window. Thus, I could not use the mouse to move or resize the window. This left me with the case where I could not do much and I finally had to go into the task manager and kill STUDIO. I also recall that upon STUDIO reopening that I ran into cases where the problem persisted and I just gave up going back to the IDE. I also had a conversation with another user who faced the same issue. Again for this newsletter I wanted to know the work around for this so asked a GAMS developer and got the response *If Studio is focused, you could press WindowsKey+Arrows to move it to a particular spot on your monitor. Alt+F4 will close Studio. Also, you could press Ctrl+F2 to reset the view of Studio to its default (that should change the size and positions of Studio and its widgets, but not close any files)*. I have not tried this as I have not had the problem lately but felt it worthwhile to write down.

4 Dealing with models that are unbounded

Optimization problems occasionally yield unbounded solutions. To find the cause I recommend that one modify the model and solve it again in a wat that allows one to gain information on the unboundedness cause. I do this by imposing unrealistically large bounds. Solvers indicate unboundedness when they find a variable that is attractive to make larger, but find that the variable may be increased without limit. In GAMS some solvers return information identifying the particular variable at issue but there may be mulitiple unbounded variables and generally only information on one is reported.

Furthermore, the LST file does not generally give additional information on the cause of the unboundedness and pre-solves rarely find such problems. Commonly, the solution report contains an instance where a particular item is tagged as unbounded (with the marker UNBND), but there will also be other variables marked as non-optimal (NOPT) which may or may not be unbounded.

Finally, note that use of GAMSCHK ANALYSIS with correction of all identified problems does not fix the issue and models still can be unbounded. Thus, most modelers will occasionally contend with models that are unbounded and will need to discover what is causing that condition.

4.1 What causes an unbounded model

Causes of unbounded models are not always easily identified. Solvers may report a particular variable as unbounded when in reality a different variable or interactions between variables are the real cause. Consider the following example:

$$\begin{array}{llllll} \text{Max} & 3X_1 & -X_2 & +X_3 & & \\ \text{s.t.} & X_1 & -X_2 & & = & 0 \\ & & & X_3 & \leq & 20 \\ & X_1 & , X_2 & , X_3 & \geq & 0 \end{array}$$

Here the unboundedness is caused by the interrelationship between X_1 and X_2 and the solvers would generally identify X_2. However, there may be several potential explanations as to why this

unboundedness is present. The profit contribution from X_1 and X_2 may not be volume independent and some form of diminishing revenue or increasing cost as sales increase may be omitted. Second, there may be omitted constraints on X_1 or X_2. Third, there may be multiple errors involving the above cases. A run with CPLEX resulted in the marking of X_2 as the unbounded item. This may or may not be a proper identification of the problem-causing mistake. The mistake may be on the X_1 side and we do not see anything about that in the output.

The general point here usually unboundedness occurs because of the interaction of multiple variables and constraints, not just the one variable that the solver happens to mark. In a more complex model, potentially a set of 50 variables and constraints could be involved. Thus, we need to find the involved set of variables and equations and then look for the root cause of the unboundedness. How then does one go about discovering this? Model modifications can help.

4.2 Finding causes of unboundedness -- basic theory

The obvious solution to an unbounded model is to add bounds. Thus, we bound variables we think are potentially unbounded so they are less than or equal to some very large number like 10^{10}. The consequent model will be bounded, but the solution may have some quite large valued variables. In this case, we will bound the X_1 and X_3 variables since both contribute revenue to the objective function.

$$
\begin{array}{llllll}
\text{Max} & 3X_1 & -X_2 & +X_3 & & \\
\text{s.t.} & X_1 & -X_2 & & = & 0 \\
 & & & X_3 & \leq & 20 \\
 & X_1 & & & \leq & 1000000000 \\
 & & & X_3 & \leq & 1000000000 \\
 & X_1 & , X_2 & , X_3 & \geq & 0
\end{array}
$$

Note we are making the problem "artificially" bounded. If it is truly unbounded, then we should expect that the solution would show X_1 and X_2 taking on large values, which are far larger than any anticipated "realistic" value. However, when unboundedness is not present the large upper bound constraints should be redundant with no effect on the solution (although in solvers using dual simplex they can slow down convergence substantially). The resultant solution is

```
Objective : 20000000040.000000

                  LOWER      LEVEL          UPPER    MARGINAL
---- EQU obj        .          .              .        1.0000
---- EQU r1      -INF          .              .        1.0000
---- EQU r2      -INF       20.0000        20.0000     2.0000

                  LOWER      LEVEL            UPPER      MARGINAL
---- VAR objmax  -INF     2.000000E+10        +INF         .
---- VAR x1        .      1.000000E+10   1.000000E+10     2.0000
```

```
---- VAR x2          .       1.000000E+10       +INF          .
---- VAR x3          .         20.0000      1.000000E+10     .
```

This solution signals what is wrong through the variable levels. The levels for both X_1 and X_2 are distorted while X_3 is unaffected. Thus, the modeler would receive signals that the unboundedness involves the interaction of the X_1 and X_2. In turn, one would examine these variables and any binding equations relating them to fix the unboundedness.

The above material indicates a way of finding the cause of unboundedness. Namely, set up the model with large bounds present, solve it and look for distorted (large) levels to find the causal set of variables and equations. One word of caution, this will always identify some of the unboundedness causes, but in the face of a non-unique primal solution caused by degeneracy or alternative optimals may not reveal them all. Thus, multiple applications of the procedure may be needed.

4.3 Details on large bound approach to resolving unboundedness

The following gives the steps for finding unboundedness causes.

Step 1	Identify the relevant variables for which artificially large bounds need to be added.
Step 2	Add bounds to those variables.
Step 3	Solve the model.
Step 4	Examine the model solution. When variable and equation solution levels are found which are excessively large, identify those as the variables and equations to be examined for the cause of infeasibility.
Step 5	Fix the model and repeat the process if needed.

There are several questions inherent in the above procedure. In particular, which items need bounds? What type of bounds should be entered? How does one find an excessively large level? Each is discussed below.

4.3.1 Where do we add large bounds?

The places where adding bounds may be in order can be determined in several ways. One could look at the model solution and just add bounds on the variables marked by the solver as unbounded or non-optimal. However, while this rather readily points to proper places in the example model, it does not always do such in other models. An approach I recommend is to add bounds to all potentially unbounded variables.

Linear programming models are unbounded when the solver finds the objective function can be improved by altering the value of a variable, but finds that variable is not limited by a constraint. Thus, to identify all potentially unbounded variables then one has to find all variables that contribute to the objective function, but are not directly bounded. Such cases in a maximization context involve

a) Non-negative variables with positive objective coefficients and no upper bound
b) Non-positive variables with negative objective coefficients and no lower bound
c) Unrestricted or free variables with positive objective coefficients and no upper bound
d) Unrestricted or free variables with negative objective coefficients and no lower bound

These cases identify a larger than necessary set since the restrictions imposed by the constraint set are not considered. However, tests that are more complex would be needed to factor in those constraints. The ADVISORY and NONOPT procedures in GAMSCHK have been written to create a list of all occurrences of these cases. Use of ADVISORY does not require the model be solved or while NONOPT in IDENTIFY mode only works after a model solve.

GAMS permits an alternative technique for bounding the problem. Namely, one can provide a large upper bound on the variable to be maximized or if the problem is a minimization problem, a large negative lower bound.

4.3.2 How do I find distorted levels?

The next question involves finding the distorted levels. The simple aspect of this is that one can simply review the output and look for variable levels with large exponents. The more complex aspect is that in a model with thousands of variables and equations this information can be hard to find. The GAMSCHK NONOPT procedure has been written to help in this quest. All items with levels in an optimal solution that are larger in absolute value than 10 to a filter value, are output as potential causes of the unboundedness.

4.3.3 Comparing the bounding techniques

As mentioned above there are two bounding approaches that can be used:

- Bound multiple individual variables which contribute to the desirability of the objective function (for example, those that are profitable in a maximization problem)
- Bound the single variable which is being optimized in the problem (i.e. if your solve statement says maximizing Z then impose the bound Z.up=10000000000;).

There can be substantial differences in information generated by these two techniques. The most distinguishing characteristics involve simplicity of use and completeness of information.

- **Simplicity of use -** When one simply bounds the variable being optimized one adds a single bound without having to think through which variables are desirable to the objective function and then add multiple bound statements on those.
- **Completeness of information** Simplicity has its costs, as information content in the solution is generally less under the simpler bound technique. Namely, when an unbounded model is solved and there is more than one set of variables causing the

unboundedness, the use of the single bound will only reveal one unbounded case at a time.

4.4 NLPs, MIPs and unboundedness

When dealing with unbounded cases in mixed integer programs or nonlinear programs the approach is essentially that above. However, in the nonlinear programming case one also has to consider has two additional issues: objective function form and solver numerical properties.

- In terms of objective function form, nonlinear programming theory requires a concave objective function for the attainment of global optimality in maximization problems and a convex objective function in the case of minimization problems. When a nonlinear programming model is judged unbounded, then one should investigate the objective function convexity/concavity characteristics.
- When a nonlinear programming model is unbounded, one can be running into numerical problems. In particular, issues such scaling, starting points, tolerances and other numerical issues can be the problem. The bounding technique above has been shown to work well for NLPs in the authors work. However, on occasions models have been found to be suffering from numerical problems, which needed to be resolved before proceeding.

5 Online Basic, Advanced and Basic-to-Advanced courses offered soon

This year I will again be teaching my family of GAMS courses for basic and advanced users. Given the COVID pandemic and accompanying travel restrictions, I am going online. I have tried to set up a schedule that accommodates multiple time zones with a bias toward the eastern US and Europe (class will meet from 8am to 1 pm Denver time). These courses will be offered in late July and I encourage interested parties to enroll. Furthermore

- I will be using ZOOM to deliver the classes and distributing PDF and GMS file materials so that all overheads and class examples that are used in the zoom presentation are also available to each of the students.
- I will record the ZOOM sessions and will make them available through the cloud for I believe as long as 30 days after the class is over.
- **Schedule -** To accommodate varying time zones including a number in Europe I have shortened the class day and extended the number of days on which there will be classes by one day. In particular the

    - Basic class will be taught on July 27, 28, 29 and 30 from 8 AM until about 1 PM US Mountain (Denver) time. (full schedule is here and registration details including fees here)

    - Advanced class will be taught on July 29, 30, 31 and August 3 again from 8 AM until about 1 PM US Mountain (Denver) time. (Full schedule is here and registration details here).

    - Combined class will be taught on July 27 - 31 and then on August 3. (full schedule is here and registration details here)

- I also will make available optional discussion times both scheduled and negotiated before and after class to accommodate people in different time zones.

Further information and other courses are listed on https://www.gams.com/courses/ . Note I also have given custom courses for individual groups.

6 Unsubscribe or subscribe to future issues of this newsletter

Please unsubscribe through the web form available at: https://gams.us18.list-manage.com/unsubscribe?u=f1497c76718404eae593beb11&id=45ccea2ee0

Those who wish to subscribe to future issues can do this through the newsletter section of https://www.gams.com/newsletter/signup/.

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

June 23, 2020