# Introducing GAMSsm:<br>A Scenario Manager for Numerical Models

Thomas F. Rutherford

Department of Economics

University of Colorado

rutherford@colorado.edu

October, 2001

**Abstract**

*GAMS*SM is the "GAMS Scenario Manager," a computer program which provides an intuitive and simple Windows-based interface to economic models and other mathematical programs implemented with GAMS. I have developed the program myself, and it is distributed as a third-party product, independent of GAMS Development Corporation. This tool helps GAMS model builders more easily communicate model results to clients who are non-modellers. The program permits persons who are unfamiliar with GAMS syntax and programming tools (such as text editors) to perform scenario analysis, produce tables and figures and transfer model results into spreadsheets and reports. This program can dramatically simplify the distribution of GAMS models. Any GAMS model can be set up using a *licensed* copy of *GAMS*SM to produce a *scenario file.* A scenario file can be distributed and accessed using the *unlicensed* version of *GAMS*SM in the same way that a document distilled to PDF with the Acrobat Writer can be examined using the Acrobat Reader.

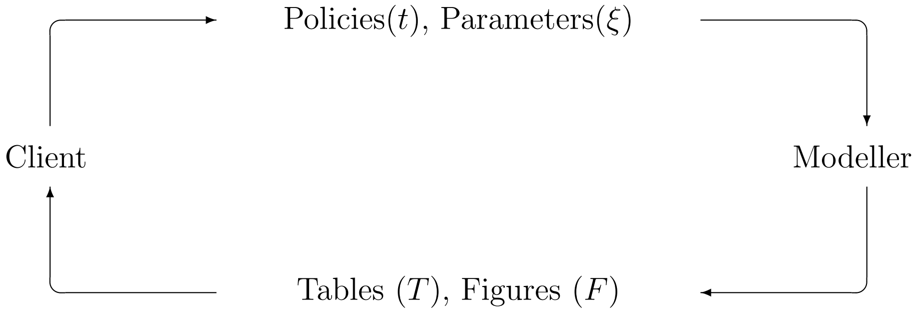
# 1 Motivation

GAMS is a modeling language designed for use by specialized analysts in a wide range of disciplines. The name GAMS stands for Generalized Algebraic Modeling System. GAMS was invented by an applied physicist, Alex Meeraus, for use by economists at the World Bank during the 1970s. The fundamental strngth of GAMS lies in the ease with which mathematically-defined models can be formulated and solved. GAMS itself does not provide model solutions; these tasks are delegated to a large number of *subsystems*, specialized algorithms for particular classes of mathematical programs. GAMS provides an intermediate set of services by translating an analyst's description of a mathematical model into a format understood by an appropriate solver.

GAMS has been designed primarily to meet the needs of the modeller rather than the needs of the modeller's client. A client for a mathematical model typically has data, a conceptual framework and a set of policy-related questions. The client contracts the modeller to formulate and analyze a model (or several models) which address the policy questions in which she is interested. The deliverables for a modeling contract typically include an operational GAMS program and a written report summarizing the mathematical details of the model and the model results. Results are typically in the form of tables or figures which compare model outcomes for alternative policies and a common set of assumptions. It would also be common practice to deliver a sensitivity analysis of model results for a particular policy, under a range of alternative parameter assumptions. For simplicity, we can name these "policy" and "sensitivity" analyses.

It has been my experiece that clients are often disappointed with the amount of specialized knowledge required to assess a model. A written report provides only a glimpse into what has been learned in the total modelling exercise. If the modeller has good writing skills but poor programming skills, the client may very easily be mislead about the true nature of what can be learned on the basis of the calculations. On the other hand, if the modeller is a good programmer but a poor writer, the client may have a very informative model but at the same time have no idea of the policy implications of the results.

A model-based analysis often involves iterations between the client and the modeller in which the specific set of policy instruments and underlying parameter assumptions are defined and evluated. We can represent this with the following diagram:



In this diagram, the client suggests a set of policy inputs (or experiments) which may be conducted with the model, $t$. She also may suggest a set of assumptions for behavioral parameters (e.g., elasticities of substitution or long-run growth rates, represented by $\xi$).

The modeller takes these suggestions, revises her GAMS program accordingly, and then generates a set of tables and figures, $T(t, \xi)$ and $F(t, \xi)$.

The GAMS Scenario Manager, *GAMS*SM, provides a tool through which the dialogue between the client and the modeller can be partially automated. Using *GAMS*SM, a non-programmer can develop a clear understanding of model results without developing a close familiarity with GAMS syntax or other programming tools. In short, *GAMS*SM permits *users* of GAMS models to avoid learning how to program in GAMS. Instead, the client may use the scenario manager to query the model and compare tables and graphs of results under alternative assumptions regarding the policy instruments, t, and underlying structural parameter or elasticities, $\xi$.

*GAMS*SM enters the modeling process at a point where a model has been implemented but is not thoroughly analyzed. The modeller defines the set of all possible policy parameters, T, the set of all model assumptions, $X$, and the set of model outputs to be returned, $Y$. A scenario file sets up a link between a given model program and a user-friendly program. A scenario file is defined by a subset of policy parameters $t \in T$, a subset of the parameters $\xi \in X$, and a set of output vectors for producing tables or figures.[1]

The *GAMS*SM program may be operated on PCs running Windows versions 98 and later. The program is intended to provide modelers with a simple way to share model insights with non-modelers. Three objectives guided the system design: (i) All that a model-user should need to know in order to use *GAMS*SM is how to navigate a Windows menu; (ii) All that a model-developer should need to know in order to use *GAMS*SM is how to program in GAMS; and (iii) *GAMS*SM should be small, easy to install and

[1] A complex model may involve a sufficiently large number of policy and structural parameters that several scenario files may be required to capture all of the tables and figures used in an academic paper based on the model.

royalty-free.

*GAMS*SM is inspired in part by the Adobe Acrobat programs which have revolutionized the distribution of electronic documents. *GAMS*SM aims to provide a similar function for model-based policy analysis. The unlicensed version of *GAMS*SM permits users to both read and *rewrite* scenario files. Working with the *GAMS*SM interface, a user may define a set of cases, tables and figures to illustrate a particular perspective on model results. The resulting analysis can be saved as a new scenario file for distribution to other policy-oriented researchers working with the same model. The system is intended to facilitate dialogue between researchers. One scenario file, with a particular set of tables and figures, may provide a very different perspective on the implications of a given policy than a second scenario file based on the same model.

The motivation for *GAMS*SM came primarily from my own frustrations in sharing models with students, colleagues and clients. The communication of policy insights from GAMS models seems to be continually limited by my (in)ability to teach programming techniques to non-computer oriented people. It is my experience that the world is full of clear thinkers who understand model results and policy issues but have no aptitude for computing. *GAMS*SM is designed with these people in mind.

## 2 Functionality

Using *GAMS*SM a user can specify model inputs (define cases) and then compare model outputs (tables and figures) across alternative sets of scenarios. The program is intended to help policy analysts use model results to write papers and reports. With this objective, *GAMS*SM provides a simple means of transfering data from figures or tables into worksheets using tab-delimited data and the Windows clipboard.

The *GAMS*SM main menu has six tabs, each of which, following conventional Windows program design, may be accessed either with a mouse click or a hotkey shortcut (alt-*underline letter*).

**File** is used to open an existing scenario file, save the current scenario file, solve unsolved cases, install a new scenario file, or exit the program. A scenario file may be saved under the same name or a different name. The *exact* solution of unsolved cases requires a GAMS system and the associated model but does *not* require a licensed version of *GAMS*SM. The generation of a *new* scenario file requires GAMS, the associated model, and a licensed version of *GAMS*SM.

**Case** is used to produce an index of defined cases, define a new case, or to edit or delete an existing case. A case is defined as a set of assumptions regarding model inputs. A scenario file may have as many as 10 inputs. A given scenario file may contain thousands of model solutions, but only 25 cases may be named at a time.

**Table** is used to produce an index of defined tables, define a new table, or to edit or delete an existing table. Tables may be constructed using one model output and up to ten cases (one case per column), or tables may be defined for several outputs from a single case.

**Figure** is used to produce an index of defined figures, define a new figure, or to edit or delete an existing figure. Figures may be constructed using one model output and up to five cases (one case per series), or figures may be defined for several model outputs from a single case.

**View** provides a single page overview of model definitions (if provided), a link to a external documentation file (if provided), and it provides navigation shortcuts. External documentation may be provided in any portable format, such as PDF or DOC. The navigation shortcuts vary depending on context. next moves to the next table/figure/index, previous moves to the previous item and index moves up to a tabular index of defined tables or figures. The functions provided by this menu item may also be performed using the keyboard (up/down/left/right). When viewing an index page, the "?" key provides a shortcut to the model definitions dialogue.

**Help** provides an overview of keystoke shortcuts, a short introduction to the *GAMS*SMinterface and an "About box" which provides program license information.

The program provides three different navigation options. The first interface is through the program menu. Menu tabs may be selected using either the mouse or pressing Alt-letter where letter indicates the underlined letter on a given menu item.

The second interface is provided through the mouse, which can be clicked on various images on the screen to navigate between, for example, the table index, tables or scenario definitions. A dashed, light-grey rectangle on the screen identifies a navigation button. One mouse click should turn on the navigation control (turning an item grey), at which point a second click changes the program context.

A third user interface is provided through the arrow keys (up/down and left/right). A short description of available keystroke navigation options is provided under Help/keystrokes.

# 3 A Guided Tour

The easiest way to learn to use *GAMS*SM is simply to try it out. It should take a user familiar with Windows applications only a few minutes to get a feeling for how the program works. The present section provides some tips for a reader who is running *GAMS*SM for the first time.

The scenario file I am using to illustrate the model interface, `ppf.scn`, was created as a teaching tool for international trade theory. This model investigates factors which determine the shape of the production possibility frontier (PPF) for a two-sector economy and the responsiveness of the PPF to changes in technology and factor endowments. In this economy, the two produced goods are $X$ and $Y$, and the factors of production include labor ($L$) and capital ($K$). The PPF is defined by efficiency allocations of labor and capital between sectors $X$ and $Y$ subject to aggregate factor supply constraints and given production functions:

$$L_x + L_y = \overline{L}$$

$$K_x + K_y = \overline{K}$$

$$X = \phi_x \left(\alpha_x L_x^\rho + (1-\alpha_x)K_x^\rho\right)^{\beta_x/\rho}$$

$$Y = \phi_y \left(\alpha_y L_y^\rho + (1-\alpha_y)K_y^\rho\right)^{\beta_y/\rho}$$

In this system of equations $\overline{L}$ and $\overline{K}$ are aggregate factor endowments, $\phi_x$ and $\phi_y$ are productivity parameters, $\rho$ is an elasticity parameter, and $\beta_x$ and $\beta_y$ determine returns to scale in both sectors. We will assume constant or decreasing returns to scale, so $\beta_x \leq 1$ and $\beta_y \leq 1$.

The *GAMS*SM program file (`gamssm.exe`) and the scenario file for this example (`ppf.scn`) are available on the *GAMS*SM home page:

`http://debreu.colorado.edu/gamssm`.

Copy both files to a new directory, start the program, and then enter File/open to load the PPF scenario file. When this file is loaded, the program will initially present the Case Index page shown in Figure 1. Each row in this table represents a named scenario. The columns in the table, other than the first and last, correspond to model inputs. The first column in the case index lists input identifiers.

For this model, the inputs which chararcterize the *benchmark economy* include:

`X_0` the base year output share for good $X$ (default valual 0.5, option 0.25)

`sigma` the elasticity of substitution between labor and capital (default value 1, optional value 0.5)

`beta_x` the elasticity of scale in sector $X$,

`beta_y` the elasticity of scale in sector $Y$,

`lvs_x` the benchmark labor share of value-added in sector $X$ (default value 0.75, optional value 0.25)

`lvs_y` the benchmark labor share of value-added in sector Y (default value 0.25, optional value 0.50)

These inputs determine model parameters $\overline{L}$, $\overline{K}$, $\alpha_x$, $\alpha_y$, $\phi_x$ and $\phi_y$, consistent with the assumption that the aggregate value of output in the observed benchmark equilibrium is 1. Given these parameters, it is then possible to numerically determine the shape of the benchmark PPF which is returned in `ppfbmk`. Any two scenarios having identical values for these six inputs will share the same benchmark PPF.

Four additional inputs determine the shock which is applied to the model:

`prd_x` the X-sector productivity index in the counter-factual scenario, defined as a proportaional scaling of $\phi_x$. Default value for $PRD_X$ is 1, and the optional value, 1.1, represents a 10% improvement)

`prd_y` the Y-sector productivity level in the counter-factual scenario, defined as a proportaional scaling of $\phi_y$. Default value for $PRD_Y$ is 1 and the optional value, 1.1, represents a 10% improvement)

`e_labor` the labor endowment level in the counter-factual scenario (default value 1, and optional value 2 represents a doubling of the workforce)

`e_capital` the labor endowment level in the counter-factual scenario (default value 1, and optional value 2 represents a doubling of the capital stock)

Model output `ppfscn` describes the shape of the PPF following the exogenous shock.

The final column of the Case Index in Figure 1 indicates the solution status to the corresponding case. When the scenario file is initially loaded, all of the cases have been solved, so each row is labeled as “Solved”.

It is a simple matter to modify a case which appears in this table. This can be done either by using the menu command (Case/edit), the mouse (click twice on a case identifier in the first column of the table), or with the keyboard, using the ↑ or ↓ keys and then pressing Enter.

When the case editor is loaded, a dialogue such as that shown in Figure 2 appears on the screen. In this dialogue there is a “combo box” presented for each model input. In the language of Windows interfaces, a "combo box" describes an input form in which a user may either choose from a list or enter a value manually. When the case edit dialogue loads, the input assumptions for the selected case are highlighted in the combo boxes.

Assumed values for an input may either be selected from the list of alternative values, or they may be entered from the keyboard. If the user only selects input values from the displayed lists, then the resulting case is drawn from the database of model solutions. If, however, alternative values are entered manually for one or more inputs, then the case may not refer to a model solution in the database, and the case solutions may appear on the case index as "Approximated" rather than "Solved".

Click on the "OK" button to exit the case editor and return to the Case Index. Once the Case Index is displayed, hit the → key to move to the Figure Index which appears as in Figure 3. (Alternatively, the Figure Index can be recalled by entering menu command View/next or Figure/index.)

The Figure Index indicates the names of figures which have been defined in the first column, the figure "item" in the second column, and the figure series in the subsequent columns. A figure item may either be a model output or a case. For example, in Figure 3, if the figure item is a model output, then the figure series correspond to cases. If the figure item is a case, then the figure series correspond to model output. In Figure 3 the row labeled `Labor` examines two outputs from the Labor10 case, `ppfbmk` and `ppfscn`. All other figures examine a single model output across two or more scenarios.

Use the ↓ key to move down the Figure Index, stopping on the record which describes the $X0$ figure, then press Enter to display the highlighted figure. The displayed figure should appear as in Figure 4. When a figure is displayed, it is possible to click on the figure title at the top in order to load the Figure Edit dialogue. It is also possible to load the Case Edit dialogue for any of the figure series by clicking on a scenario key at bottom of the figure.

The menu command Figure/export can be used to transfer data for the currently-viewed figure into the Windows clipboard as a tab-delimited table. This is illustrated in Figure 5. You may then open Excel or Corel Presentations in a new window and use the PASTE command in Excel or Presentations to transfer the figure data into a new or existing worksheet. The transferred data appears as a highlighted range after it is pasted in the worksheet, and it is then possible to use the chart wizard to reproduce the *GAMS*SM figure.

The cases, figures and tables which are found in `ppf.scn` were defined at the time the scenario file was generated. Subject to a limit of 25 cases in total, it is possible to add cases to a scenario file using the Case/new menu command. This command produces a dialogue requesting a name for the new case and the case on which to initially base the input assumptions. This first new case dialogue is shown in Figure 6. After entering OK, the next dialogue is the case editor, as shown in Figure 2.

The introduction of a new figure requires the menu command Figure/new. This invokes an initial dialogue querying for a figure identifier and the figure content, either a single output with series representing alternative cases or a single case with series representing alternative outputs. Figure 7 displays this dialogue. After having established the figure content, the next dialogue, Figure 8, queries which item to plot. This involves selecting from the set of outputs or from the set of cases, depending on the desired figure content. After establishing the figure identifier and content, the figure series editor dialogue appears, as shown in Figure 9. In this dialogue, use a mouse to highlight an item from the left list, then hit the Add button to place the item into the figure series list on the right. The Del button is used to remove a highlighted series from the figure list. The ordering of series in the figure follows the sequence in which they are added.

When defining a table or figure with several outputs from a *single case* (i.e., when the output item is a case), the selection list on the left of Figure 7 is updated after one item is added to reflect the domain of the selected item. Tables and figures may be defined for multiple outputs from a single case only when those outputs are defined over the same domain. For example, suppose that model outputs consist of `w(j)`, `x(i)`, `y(j)` and `z(i)`. A table with two items from a single case may then include either `w` and `y` or `x` and `z`, but not, for example, `w` and `x` or `w` and `z`. In the table definition dialogue, all four outputs would be in the selection list initially, but once `y` had been added to the table, `x` and `z` would disappear from the selection list because they are defined over a different domain.

The `ppf.scn` file contains several tables as well as figures, and these can be accessed through the Table Index. The arrow keys are the easiest way to navigate between tables and figures. Viewing a figure or table, an ↑ keystroke moves up to the associated (figure or table) index. A → keystroke then moves to the next index, and a ↓ keystroke moves down the rows of the next index.

## 4 Installation and Invocation

The *GAMS*SM download files can be found at the *GAMS*SM home page:

`http://debreu.colorado.edu/gamssm`.

The *GAMS*SM distribution currently contains only a few sample models, but more models will be added during the next few months. The home page will also provide a revision history and announcements about ongoing program development.

*GAMS*SM is distributed in a Windows setup program, `gamssm_setup.exe`, which is roughly 1.2 megabytes in size. Right click on the link to this file and choose "Save As." This program provides a link to *GAMS*SM from the Windows Program menu. It can be convenient to install a desktop shortcut to *GAMS*SM. To do this, right-click on the *GAMS*SM menu.

If you install *GAMS*SM on the DOS path, it can be invoke from the command line using the syntax:

```
gamssm [filename.extension] [/b]
```

With no argument, the program loads and the user must enter the File/open statement to load a scenario file. When an argument is provided, the file extension determines whether a scenario file is to be loaded or a installed. Three extensions are recognized:

`.scn` loads a given scenario file

`.def` indicates a scenario definition file. The program reads the definition file, calls GAMS to solve all specified cases and writes out a scenario file with the same file name prefix and a `.scn` extension. *This invocation requires a licensed copy of GAMS*SM. The `/b` (batch mode) flag provides the option of model installation using an unattended batch program. The _ flag causes the program to skip all dialogues requiring user input.

`.ses` reloads the most recent *GAMS*SM session. A session file is simply a flat text file in which the first line defines the currently edited scenario file.

# A Model Solutions and Model Approximations

A scenario file contains a database of possibly thousands of model solutions from which a user may define cases. Let $k = \{1, \ldots, N\}$ index the set of solutions in a scenario file, let $i = \{1, \ldots, n\}$ denote the set of model *inputs*, and let $j = \{1, \ldots, m\}$ denote the set of model *outputs*. Each model input, $x_i$, is a scalar number or a text string. Each model output, $y_j$, is a vector. When producing a table comparing output $y_j$ across several cases, the rows of the table correspond to elements of the output vector and the row text identifiers are set labels from the underlying GAMS program.

The scenario file provides a database of solution values which can be thought of as though they were obtained through evaluation of an implicit function:

$$\overline{y}_{jk} = f_{jk}(\overline{x}_{1k}, \ldots, \overline{x}_{nk})$$

If a user wishes to define a scenario drawn from the existing solution database, she simply selects input values for a particular solution $k$. These presolved cases are selected by choosing input values from a list of optional values defined for each input in the case definition dialogue (Case/new or Case/edit).

If, however, a user wishes to define a scenario with values for one or more inputs which are not provided in the options list, *GAMS*SM then attempts to provide a *linear approximation* of the corresponding model solution. This feature permits users to define and investigate new cases, even when they do not have a GAMS system on their machine.

A linear approximation is provided only when the input vector for the new case, $x_i$, lies within the convex hull of the existing solutions; i.e., when there exist values $\theta_k$ such that:

$$x_i = \sum_k \theta_k \overline{x}_{ik}, \quad \theta_k \geq 0, \quad \sum_k \theta_k = 1$$

When such an approximation is possible, the values of $\theta$ are chosen using a linear programming module linked in *GAMS*SM to minimize $\sum_k \theta_k |x_i - \overline{x}_{ik}|$. Having found $\theta$, solution values for a new case are based on the linear approximation:

$$y_j \approx \sum_k \theta_k y_{jk}$$

If a new case cannot be approximated because $x$ lies outside the convex hull of the existing solved cases, then the new case is labeled "unsolved," and although it may be added to tables or figures, it will not be displayed.

Whenever new cases are defined, they are introduced in the solution database as linear approximations. If a GAMS system and the scenario file's associated model file are present on the user system, the approximate solutions of any unsolved cases may be replaced by *exact* solutions with the File/solve command. Calls to GAMS are possible even with the unlicensed version of *GAMS*SM. The only restriction on the unlicensed program is that the File/install command is unavailable – new scenario files may not be produced "from scratch."

## B Instructions for GAMS Programmers

This appendix is provided for GAMS programmers who wish to produce a *GAMS*SM scenario file from a GAMS model.

A *GAMS*SM scenario (`.scn`) file is a binary file which cannot be viewed or edited by other programs. A scenario file is based on a *scenario definition file*, a `.def` file, and an associated GAMS model. A `.def` file is a text file which is produced like a GAMS program using a text editor. Here is the scenario definition file used to generate `ppf.scn`:

```
modelfile=ppfmodel

input=x0        default=0.5 option=0.25
input=sigma     default=1   option=0.5
input=beta_x    default=0.5 option=1
input=beta_y    default=0.5 option=1
input=lvs_x     default=0.75 option=0.25
input=lvs_y     default=0.25 option=0.50
input=e_labor   default=1   option=2
input=e_capital default=1   option=2
input=prd_x     default=1  option=1.1
input=prd_y     default=1  option=1.1

output=summary(sumset)
output=ppfbmk(plt) label=pltlbl
output=ppfscn(plt) label=pltlbl

case=ref
case=x25 x0=0.25
case=x50 x0=0.5
case=lvsx25 lvs_x=0.25
case=lvsx50 lvs_x=0.50
case=lvsx75 lvs_x=0.75
case=els1   beta_x=1    beta_y=1
```

```
case=els50  beta_x=0.5  beta_y=0.5
case=els25  beta_x=0.25 beta_y=0.25
case=sigma1  sigma=1
case=sigma05 sigma=0.5
case=labor10   e_labor=1.1
case=capital10 e_capital=1.1
case=2xlabor   e_labor=2
case=2xcapital e_capital=2

figure=x0       output=ppfbmk case=x25 case=x50
figure=labor    case=2xlabor output=ppfbmk  output=ppfscn
figure=els      output=ppfbmk case=els1 case=els50 case=els25
figure=sigma    output=ppfbmk case=sigma1 case=sigma05
figure=capital  output=ppfscn case=ref case=capital10 case=2xcapital

table=capital   output=summary case=ref case=capital10 case=2xcapital
table=labor     output=summary case=ref case=labor10 case=2xlabor
table=els       output=summary case=els1 case=els50 case=els25
table=sigma     output=summary case=sigma1 case=sigma05


docfile=ppf.pdf

definitions=begin
The Production Possibility Frontier

Input parameters which characterize technology and endowments in the
base year include:

        x0      Base year output of good X

        sigma   Elasticity of substitution K versus L

        beta_x  Elasticity of scale in sector X
        beta_y  Elasticity of scale in sector Y

        lvs_x   Labor value share in sector X
        lvs_x   Labor value share in sector Y

Input parameters which characterize a counterfactual shock to the
economy:

        e_labor       Labor endowment index
        e_capital     Capital endowment index
        prd_x          X sector productivity index
        prd_y          Y sector productivity index

definitions=end
```

The keywords which appear at the start of records in a scenario definition file must appear in the following sequence:

| | |
|---|---|
| `modelfile` | identifies the GAMS program file for the underlying model |
| `input` | defines a model input |
| `output` | defines a model output |
| `case` | defines a case to be initially defined in the scenario file |
| `table` | defines a table to be added to the scenario |
| `figure` | defines a figure to be added to the scenario |
| `docfile` | defines an external file which provides a detailed description of the model features (typically a PDF file) |
| `definitions` | prsents a page of definitions for model inputs and outputs, as provided by the View/definitions command. |

## B.1 Inputs

Keywords in a scenario definition file are always followed by "=" signs. The `input` keyword is followed by a text string identifier with 10 or fewer characters. This identifier must correspond to a `$setglobal` environment variable in the GAMS model. The second field in an `input` definition record must have the form `default=`*value*. Other optional values may appear after the default value, in fields of the form `option=`*value*. There is no fixed upper bound on the number of explicit optional values which may be defined for a single input, but some care is required to avoid generating too many implicit solutions. When a `.def` file is processed, an enumeration of *all* input combinations is produced, and each case is solved by a call to GAMS. If, for example, a model has 10 inputs and each input has a default value and two other optional values, then installation of the model involves the repeated solution of the model for $3^{10} = 59049$ cases. Even if the model were of modest size and an average case solved in 60 seconds, it would take nearly 6 weeks of computing to enumerate all the solutions; and if each solution generated 500 real values, the resulting scenario file would exceed 120MB!

Inputs appear as `$setglobal` environment variables in the GAMS model file. *GAMS*SM does not scan the model file to be sure that all inputs define in the `.def` file are used in the model file – this is the modeler's responsibility.

It is the design philosophy of *GAMS*SM that any model file used in *GAMS*SM should operate in "stand-alone" mode. The way to do this is to include statements at the top of the GAMS file of the form:

`$if not setglobal` *input* `$setglobal` *input default_value*

The following lines appear at the top of the `ppfmodel.gms` program which is used to produce `ppf.scn`:

```
$if not setglobal x0            $setglobal x0 0.5
$if not setglobal sigma         $setglobal sigma 1
$if not setglobal beta_x        $setglobal beta_x 0.5
$if not setglobal beta_y        $setglobal beta_y 0.5
$if not setglobal lvs_x         $setglobal lvs_x  0.75
$if not setglobal lvs_y         $setglobal lvs_y  0.25
$if not setglobal sigma_x       $setglobal sigma_x 1
$if not setglobal sigma_y       $setglobal sigma_y 1
$if not setglobal prd_x         $setglobal prd_x 1
$if not setglobal prd_y         $setglobal prd_y 1
$if not setglobal e_labor       $setglobal e_labor 1
$if not setglobal e_capital     $setglobal e_capital 1
```

In GAMS `$setglobal` environment variables are handled as straight string substitutions, and they may appear in parameter statements, assignments, or even as file names in `$include` statements. For example, in the PPF model the following assignments use the environment variables `prd_x`, `prd_y`, `e_labor` and `e_capital` to adjust factor endowments and productivity levels for the calculation of the counter factual equilibria:

```
        sl = %e_labor%;
        sk = %e_capital%;
        prd("x") = %prd_x%;
        prd("y") = %prd_y%;
```

A single scenario file may identify at most 10 inputs. A model file may, however, include any number of `$setglobal` environment variables, and *multiple scenario files may be used with a single model file.* This means that one scenario file for a given model may be defined to explore a particular set of inputs while a second scenario file may investigate a different set of inputs. In this way, the restriction of 10 inputs per scenario file does not limit the range of sensitivity analyses which can be performed with a given model.

## B.2 Outputs

The syntax for an output record in a scenario definition file is:

`output=`*identifier*(*domain*)    [`label=`*subset*]

The output identifier must be a single dimensional parameter from the GAMS program file, and the domain must refer to a static set from the GAMS program. The label subset is required only for outputs which are to be included in *GAMS*SM figures. The subset labels are used both to annotate the X-axis in *GAMS*SM figures and also to define which elements of the output are printed when the output is added to a *GAMS*SM Table. For example, if we have set definitions in the GAMS program of the form:

```
set     tt /2000*2100/,
        tl(tt) /2000,2020,2040,2060,2080,2100/;
```

Then the `consum` output (listed above) produces *GAMS*SM figures with 101 data points and 6 label ticks on the X axis. If `consum` is displayed in a *GAMS*SM table, that table will have 6 rows.

The maximum number of rows in a *GAMS*SM table is 25, so this represents an upper bound on the cardinality of the sets associated with model outputs. When a label is unspecified, the output domain may contain no more than 25 elements. When a label is specified, the domain may have any number of elements, but the label subset may have at most 25 elements.

## B.3 Cases, Tables and Figures

The `case=`, `table=` and `figure=` statements are provided so that a scenario file can be generated with a specific set of identifiers. The `case=` field defines a case identifier, and it is followed by the input assumptions to be associated with that case. The values assigned to individual inputs in a case statement need not correspond to values from the inputs' options lists.

All case statements should appear before tables or figures are defined. Table and figure records are of two varieties. They either define an output as the second field followed by one or more cases, or they define a case as the second field followed by one or more outputs. A table may have as many as 10 columns, and a figure may have at most 5 series.

During the early stages of model development with *GAMS*SM it is helpful to omit option lists for all the inputs and use the case, table and figure statements within the scenario definition file to produce a scenario file with a specific set of tables and graphs and avoid the time required to enumerate a large number of cases.