Introduction to GAMS

A programming language designed for formulation of mathematical programs

GAMS Background

- Born at the World Bank
- Designed as a tool for the applied economist
- Best for math programming, but a full-featured high-level programming language
GAMS – Organization of a Program

- A GAMS program is a series of statements separated by semi-colons
- Organization can be random as long as declarations of symbols precede use
- It is better to adopt some scheme for organizing your programs

Organizing Your Programs

- Some things that make sense are
  - Keep model (equation definitions) together
  - Organize data associated with submodels together
  - Use comments to help the reader figure out your program
GAMS Object Types

- Objects – Things you can name (Symbols and Identifiers)
  - Sets
  - Parameters (including Scalars and Tables)
  - Variables
  - Equations
  - Models
  - (A few other obscure things that we will not use)

Structure of a Data-Type Declaration

- Data-type keyword (e.g., Parameter or Variable)
- Identifier (e.g., A or X)
- Domain list (e.g., (I) or (I,J))
- Text (e.g., Supply at source I or Amount shipped from I to J in cases)
Naming Conventions

- Length of Symbol and Identifier names ≤ ten characters
- Some characters cannot appear as part of a name: !^@#$%^&*()-=+[]}{":"-,<.>/?
- Legal characters include alphabet, numbers (not as first character), and underscore (_)
- Some words are reserved (e.g., SUM)

Labels – Elements in Sets

- Length of labels ≤ ten characters
- Legal characters include alphabet, numbers, “+”, “-”, and “_”
Text – Comments

- Comments associated with identifiers and labels
- Limited to 80 characters or less on a single line
- May not contain ‘,’ ‘,’ ‘.’ or ‘/’

Numbers

- Decimal points are optional
- Blanks may not be embedded
- Signs (positive or negative) may be used
- Scientific notation (e.g., 2E3=2x10^3) is OK
- Special symbols for infinity (INF), undefined (UNDF), not available (NA), and “epsilon” (EPS)
Delimiters – Separators

- Statements are ended with semi-colons
- Entries in data list are delimited by “,” or a new line
- Data lists are terminated by “/”

Comments

- Text is associated with a symbol – comments are not
- A line beginning with a “*” is a comment
- Anything between “$ontext” and “$offtext” is a comment
- Comments are not processed by GAMS
GAMS Data-Types – Sets

- A set is a **collection of elements** useful for subscripting:
  - Parameters
  - Variables
  - Equations
  - (NOT models)

Sets (cont’d.)

- Specify sets via lists
  - SET clowns / bozo, flippo, ronald / ;

- Or using the * operator
  - SET years / 1999*2001 /
  - (This specifies the list 1999, 2000, 2001.)
  - SET fiscalyr / FY97*FY99, FY00 /
  - (This specifies the list FY97, FY98, FY99, FY00.)
Sets (cont’d.)

- GAMS is quite literal, and the elements should be thought of as character strings

- The following sets are quite different, having only one element in common
  - SET numbers / 1*10 / ;
  - (1,2,3,4,5,6,7,8,9,10)
  - SET umbernays / 01*10 / ;
  - (01,02,03,04,05,06,07,08,09,10)

Sets (cont’d.)

- To refer to a set by more than one name, we use the ALIAS command:
  - SET locations / loc8*loc12 / ;
  - ALIAS (locations, places) ;

- Both sets locations and places refer to the same list of elements
  - Note that they contain loc8,loc9,loc10,loc11,loc12.
Sets (cont’d.)

- Subsets are easily accommodated as seen in the example:

  SET american / hotdogs,mom/applepie /
  usfood(american) / hotdogs,applepie / ;

Sets (cont’d.)

- Text can be associated with the set name or with elements in the set:

  SET dogs A set of dog breeds
  / great_dane A big dog
  chihuaua A small dog
  collie A medium dog / ;
Sets (cont’d.)

Set elements do not have values – they are not numerical even if they look like they are!

- SET years / 1980*1990 / ;
- The first element in this set is 1980, but it has no numerical value
- To refer to the place of occurrence (i.e., first) in the set years of 1980 a special conversion is needed

Sets (cont’d.)

The ORD function is used to refer to the ordinality of a set element

- PARAMETER beta(years) ;
  beta(years) = 0.95**(ORD(years)-1) ;
  Results in beta(‘1980’)=1, beta(‘1981’)=0.95, beta(‘1982’)=.9025, etc.
Sets (cont’d.)

- In order to use ORD, the set must be “static” and “ordered”
  - “Static” means the membership of the set does not change during program execution
  - “Ordered” means that the order of the elements in the initialization list must be the same as the order in which GAMS first sees the elements

Example:
- SET t / 2*5 /
- s / 1*4 /
- r / 2*5,1 /
- p / a,b,2,4 /
- q / 2,4,1,c /

The sets t and r are ordered, while s is not.

Are either of p or q ordered?
Sets (cont’d.)

- Example:
  - SET t / 2*5 /
  - s / 1*4 /
  - r / 2*5,1 /
  - p / a,b,2,4 /
  - q / 2,4,1,c /

- The key to understanding is the “unique element list” which for this example is:
  - 2,3,4,5,1,a,b,c

Sets (cont’d.)

- GAMS also supports familiar operations with sets – union, complement and intersection:
  - SET clothes / shirt,shoes,pants,socks,dress,
  -    hat,tie,scarf,cufflinks /
  - top(clothes) Coverings for the top half
  -    / shirt,hat,tie,scarf,dress /
  - bot(clothes) Coverings for the bottom half
  -    / shoes,pants,socks,dress /
  - both(clothes),neither(clothes),either(clothes);
Sets (cont’d.)

Ⅰ The set both is the intersection of top and bot:
Ⅰ both(clothes) = top(clothes)*bot(clothes) ;

Ⅰ The set neither is the complement of both:
Ⅰ neither(clothes) = yes ;
Ⅰ neither(clothes) = neither(clothes) – bot(clothes)
Ⅰ – top(clothes) ;

Sets (cont’d.)

Ⅰ The set either is the union of top and bottom:
Ⅰ either(clothes) = top(clothes)+bottom(clothes) ;
GAMS Data – Parameters

- GAMS allows data to be entered in natural form and then manipulated
- Parameters come in three flavors – PARAMETERS, SCALARS, and TABLES

GAMS Data – SCALARS

- Scalars are parameters that are not indexed by a set – they have no domain lists
- Examples of scalar declarations:
  - SCALAR rho /-0.3/;
  - SCALAR pi a trig constant /3.14159/;
  - SCALAR a10 /7/, b_prime /9/;
GAMS Data – PARAMETERS

- Parameters are indexed by sets (i.e., they have domain lists):

  - SET rows / r1*r3 /, cols / c1*c3 /;
  - PARAMETER
    - data(rows) / r1 2, r2 3, r3 5 /;
    - matrix(rows,cols) / r1.c1 3, r2.c2 3, r3.c2 3 /;

- Indices precede values, and commas separate entries. Default values are zero.

```
data  matrix
r1  2  r1  3  0  0  
r2  3  r2  0  3  0  
r3  5  r3  0  3  0
```
GAMS Data – PARAMETERS

- Two more efficient ways to enter matrix:

  ```
  TABLE matrix(rows,cols)  Example table input
  c1       c2       c3
  r1      3
  r2                   3
  r3                   3 ;

  PARAMETER matrix(rows,cols) / r1.c1 = 3
  r2*r3.c2 = 3 / ;
  ```

- When indexed by more than three sets, indices continue to be separated by “.”:

  ```
  PARAMETER cube(rows,rows,cols) 3D parameter
  / r1.r1.c1 1, r2.r1.c2 2, r1.r2.c2 3 / ;

  TABLE cube(rows,rows,cols)
  r1.c1           r2.c2        r1.c2
  r1      1                 3
  r2                                         2 ;
  ```
GAMS Data – PARAMETERS

| Or, … |

| TABLE cube(rows,rows,cols) |
|                                     |
| r1.r1   1 | c1   c2 |
| r1.r2   3 |         |
| r2.r1   2 |         |

Purdue University Ag. Econ. 652
Lecture 3 33