Lecture 5 Good modeling

Zidong Mark Wang
2015 Fall

Based on material written by Gillig and McCarl; Improved upon by many previous lab instructors; Special thanks to Pei Huang.
Why bother?

- How easy is it to reuse or modify a model at a later time for you?
- How easy can a colleague work with your code?
- A user-friendly model should include the following features:
  - Using longer names or descriptions
  - Including comments on nature and source of data
  - Including as much raw data as possible as opposed to externally calculated data
  - Less * as a set speciation for input data
  - Using sets to aid in readability
  - A readable format
An Aside: Subset and universal set

• One can define subsets containing part of the elements of another set using a set statement. The general format is

```
SET subsetname(setname) optional explanatory text
    / Elementname1 optional explanatory text
    Elementname2 optional explanatory text/;
```

• where `subsetname` is the name of the subset and `setname` is the name of the “upper” set.
1. Allow items to be treated simultaneously in some places, but separately elsewhere.
2. Allow small to large modeling
3. Conditional statements (discuss later)

```
SETS
ALLI       ALL BUDGET ITEMS
    /Corn, Soybeans
    Cropland, Pasture
    Labor, Water
    Nitrogen, Potassium
    TranCost
/

Primary(ALLI) Primary products
    /Corn, Soybeans /
Input(ALLI) National inputs
    /Nitrogen, Potassium /
LandType(ALLI) Land types
    /Cropland, Pasture /
Resource(ALLI) Resource use
    /Labor, Water /
TCost(ALLI) Transportation cost
    /TranCost /

Sets
allsupplies   / seattle, san-diego, houston, boston, miami/
allmarkets   / new-york, chicago, topeka, las-vegas, knoxville,
               dallas, austin/;

Sets
supplies(allsupplies) /seattle, san-diego/
markets(allmarket)     /new-york, chicago, topeka/;

* after test run with small subsets, use the following statements to
* expand to large model with full sets

supplies(allsupplies) = yes;
markets(allmarket)    = yes;
```
Universal set *

• Set references may be **indefinite allowing any entries** at all by referring to the universal set.

```plaintext
parameter x(*) parameter defined on university set;

x("aa")=5;
x("April")=3;
x("empty")=100;
```

---

15 PARAMETER x parameter defined on university set

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>aa</td>
<td>5.000</td>
<td>April</td>
<td>3.000</td>
</tr>
<tr>
<td>empty</td>
<td>100.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

• One application of the universal set is quick report writing to make summary tables (will discuss in the following section).
Naming Conventions

- GAMS allows 63 character long names and 200 characters of explanatory text to define Sets, parameters, tables, scalars, variables, equations, models.

- It is wise to make GAMS code to be self documenting by using descriptive character names and make sure that there is no item that goes undefined.

- Enter units, sources, and descriptions. Check for completeness with $ONSYMLIST.

- Check for unused items that are already defined with rf=list.ref in the dialogue box.
The symbol listing in the listing file. Default in GAMS is

```gams
$ONSYMLIST
SETS
   Ingredient       Feeding ingredients / Corn, Hay, Wheat /
   Nutrient        Formula characteristics / Calories, Protein / ;

Symbol Listing

FUNCTIONS

SETS
Ingredient       Feeding ingredients
Nutrient        Formula characteristics

PARAMETERS
Cost             The ingredient cost per unit
MinNutrient      The minimum nutrient content requirement
NutrientContent  Nutrient content in ingredients

VARIABLES
Quantity         Each ingredient's quantity used
TotCost          The total feeding cost

EQUATIONS
NutrientReq      Minimum nutrient requirement
TotalDietCost    Minimize total diet cost
TotalVolume      Total volume of diet

MODELS
DietProblem
```

The symbol listing in the listing file. Default in GAMS is

```gams
$OFFSYMLIST
```
The reference list

<table>
<thead>
<tr>
<th>Id</th>
<th>Type</th>
<th>Domain</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Parameter</td>
<td>(i)</td>
<td>capacity of plant i in cases</td>
</tr>
<tr>
<td>2</td>
<td>Parameter</td>
<td>(j)</td>
<td>demand at market j in cases</td>
</tr>
<tr>
<td>3</td>
<td>Parameter</td>
<td>(i, j)</td>
<td>transport cost in thousands of dollars per case</td>
</tr>
<tr>
<td>4</td>
<td>Equation</td>
<td></td>
<td>define objective function</td>
</tr>
<tr>
<td>5</td>
<td>Parameter</td>
<td>(i, j)</td>
<td>distance in thousands of miles</td>
</tr>
<tr>
<td>6</td>
<td>Equation</td>
<td>(j)</td>
<td>satisfy demand at market j</td>
</tr>
<tr>
<td>7</td>
<td>Parameter</td>
<td></td>
<td>freight in dollars per case per thousand miles</td>
</tr>
<tr>
<td>8</td>
<td>Set</td>
<td>(^)</td>
<td>canning plants</td>
</tr>
<tr>
<td>9</td>
<td>Set</td>
<td>(^)</td>
<td>markets</td>
</tr>
<tr>
<td>10</td>
<td>Equation</td>
<td>(i)</td>
<td>observe supply limit at plant i</td>
</tr>
<tr>
<td>11</td>
<td>Model</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Variable</td>
<td>(i, j)</td>
<td>shipment quantities in cases</td>
</tr>
<tr>
<td>13</td>
<td>Variable</td>
<td></td>
<td>total transportation costs in thousands of dollars</td>
</tr>
</tbody>
</table>
Using longer names

\[ \text{cs} \ldots \quad z = \text{SUM}(i, j), c(i, j) \times x(i, j) \; ; \]

\[ \text{su}(i) \ldots \quad \text{SUM}(j, x(i, j)) = \text{L} = a(i) \; ; \]

\[ \text{de}(j) \ldots \quad \text{SUM}(i, x(i, j)) = \text{G} = b(j) \; ; \]

Same algebras but different names

\[ \text{Costsum} \ldots \]
\[ \quad \text{TotalCost} = \text{SUM}((\text{Source, Destination}), \]
\[ \quad \text{Trancost}(\text{Source, Destination}) \times \text{Transport}(\text{Source, Destination})) ; \]

\[ \text{Supplybal(\text{Source})} \ldots \]
\[ \quad \text{SUM}(\text{Destination, Transport(\text{Source, Destination}))} \]
\[ \quad = \text{L} = \text{Supply(\text{Source})} ; \]

\[ \text{Demandbal(\text{Destination})} \ldots \]
\[ \quad \text{SUM}(\text{Source, Transport(\text{Source, Destination}))} \]
\[ \quad = \text{G} = \text{Need(\text{Destination})} ; \]
Questions often asked when looking at a set of data are:

- Where did the data come from?
- What characteristics such as units, and year of applicability do those data possess?

It is nice to go beyond the GAMS 80 character description by putting several lines of description identifying what document a data set is from including sources, page numbers, table number, years, units, etc.
Raw vs. Calculated Data

Modelers often face two choices with respect to data.

- Enter raw data into GAMS and transform it to the extent needed inside GAMS
- Externally process data entering the final results in GAMS (e.g. from a spreadsheet where the data are previously manipulated)

**Recommendation:** Put data in as close to the form as it was collected into GAMS and then manipulate the data in GAMS code

**Justification:**
(1) Over time spreadsheets change or get lost.
(2) Keep a record of what you did.
Instead of directly entering the transportation cost that was previously calculated in the spreadsheet in GAMS using `TABLE` statement, one should enter a raw data in GAMS and then let GAMS do a calculation.

```
TABLE TranCost(Source, Destination)  Transport cost in dollars per case

  "New York"  Chicago  Topeka
Seattle       250.000  178.000  187.000
San Diego     250.000  187.000  151.000
```

**TABLE**  Distance(Source, Destination)  Distance in thousands of miles

<table>
<thead>
<tr>
<th></th>
<th>&quot;New York&quot;</th>
<th>Chicago</th>
<th>Topeka</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seattle</td>
<td>2.5</td>
<td>1.7</td>
<td>1.8</td>
</tr>
<tr>
<td>&quot;San Diego&quot;</td>
<td>2.5</td>
<td>1.8</td>
<td>1.4</td>
</tr>
</tbody>
</table>

**SCALAR**

| PrMileCst | Freight cost in $ per case per 1000 miles /90/ |
| LoadCost  | Freight loading cost in $ per case /25/ |

**PARAMETER**

\[
\text{TranCost(Source, Destination)} \quad \text{Transport cost in dollars per case;}
\]
\[
\text{Trancost(Source, Destination)} \quad = \text{Loadcost} + \text{PrMileCst} \times \text{Distance(Source, Destination)};
\]
Cautions about Calculation

- **Dynamic**: calculations repeated every time the model is generated. Only calculations in the model .. statements are dynamic.

```
Obj..
ObjF =E= \text{Sum}(\text{Crop}, (\text{Price}(\text{Crop}) \times \text{Yield}(\text{Crop})-\text{Cost}(\text{Crop})) \times \text{Acres}(\text{Crop})) ;
```

- **Static**: calculations executed once only at the place the GAMS instruction appears in the code.

```
\text{PARAMETER}
\text{Revenue}(\text{Crop}) \quad \text{revenues by crop} ;
\text{Revenue} (\text{Crop}) = \text{Price}(\text{Crop}) \times \text{Yield}(\text{Crop})-\text{Cost}(\text{Crop}) ;
```
Repeated Static: calculations within a GAMS flow control structure (e.g. loop) which are executed repeatedly but are static within the control structure.

```
LOOP ( LANDCHANGE,
    LAND = LAND * (1 + VALUE ( LANDCHANGE ) / 100. ) ;
```
Dynamic vs. Static Calculation

The data on revenue is previously calculated using the PARAMETER statement.

```plaintext
SET Crop Crop production /corn/ ;
PARAMETER
Price(Crop) Price in dollars per bushel
Yield(Crop) Yield in bushels per acre
Cost(Crop) Cost in dollars per bushel
Revenue(Crop) Revenue in dollars per bushel;

Price(Crop) = 2.00;
Yield(Crop) = 100;
Cost(Crop) = 50;
Revenue (Crop) = Price(Crop) * Yield(Crop) - Cost(Crop);

EQUATIONS
Obj objective function
Land Land available ;

VARIABLES Profit Objective function;
POSITIVE VARIABLES Acres(Crop) Cropped Acres ;

Obj.. Profit =E= SUM(Crop, Revenue(Crop) * Acres(Crop));
Land.. SUM(Crop, Acres(Crop)) =L= 100;

MODEL FarmProfit /ALL/
SOLVE FarmProfit USING LP MAXIMIZING Profit;
```

Then this revenue is used in the OBJ.. equation.
No * In input data Set Specification

```
TABLE Miscdata(*,RawMaterial) miscellaneous production data

scrap new
max-stock  400  275
store-cost  .5  2
endinv-value  15  25  ;
```

Profitacct.. profit =E=

```
SUM(quarter,
    SUM(products, expectprof(products,quarter)
        *production(products,quarter))
   - SUM(RawMaterial, miscdata("store-cost",RawMaterial)
        *openstock(RawMaterial,quarter)))
+ SUM(RawMaterial, miscdata("endinv-valu",RawMaterial)
        *openstock(RawMaterial,"winter"));
```

* in the first index position of MiscData. GAMS allows anything in that position suppressing “domain” checking. Suppose we mistyped endinv-value as endinv-valu, then GAMS code would compile and execute w/o a GAMS error but the result would be wrong.
No * In input data Set Specification

Here if we replace * set with \textbf{InputItem} set,

\begin{verbatim}
TABLE Miscdata(InputItem,RawMaterial) miscellaneous production data
    scrap new
max-stock  400  275
store-cost  .5  2
endinv-value 15  25

Profitacct.. profit =E=
56  SUM(quarter,
57      SUM(products, expectprof(products,quarter)
58          *production(products,quarter))
59        - SUM(RawMaterial, miscdata("store-cost",RawMaterial)
60            *openstock(RawMaterial,quarter)))
61        + SUM(RawMaterial, miscdata("endinv-valu",RawMaterial)
62            $170
*****
***** 170 Domain violation for element
63        *openstock(RawMaterial,"winter");
\end{verbatim}
The solution from solving the model with mistyping on "endinv-value" as "endinv-valu"

<table>
<thead>
<tr>
<th>S O L V E</th>
<th>S U M M A R Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODEL</td>
<td>robert</td>
</tr>
<tr>
<td>TYPE</td>
<td>LP</td>
</tr>
<tr>
<td>SOLVER</td>
<td>BDMLP</td>
</tr>
<tr>
<td>OBJECTIVE</td>
<td>profit</td>
</tr>
<tr>
<td>DIRECTION</td>
<td>MAXIMIZE</td>
</tr>
<tr>
<td>FROM LINE</td>
<td>72</td>
</tr>
<tr>
<td>SOLVER STATUS</td>
<td>1 NORMAL COMPLETION</td>
</tr>
<tr>
<td>MODEL STATUS</td>
<td>1 OPTIMAL</td>
</tr>
<tr>
<td>OBJECTIVE VALUE</td>
<td>6741.6667</td>
</tr>
</tbody>
</table>

The solution from solving the model with correction on "endinv-value"

<table>
<thead>
<tr>
<th>S O L V E</th>
<th>S U M M A R Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODEL</td>
<td>robert</td>
</tr>
<tr>
<td>TYPE</td>
<td>LP</td>
</tr>
<tr>
<td>SOLVER</td>
<td>BDMLP</td>
</tr>
<tr>
<td>OBJECTIVE</td>
<td>profit</td>
</tr>
<tr>
<td>DIRECTION</td>
<td>MAXIMIZE</td>
</tr>
<tr>
<td>FROM LINE</td>
<td>72</td>
</tr>
<tr>
<td>SOLVER STATUS</td>
<td>1 NORMAL COMPLETION</td>
</tr>
<tr>
<td>MODEL STATUS</td>
<td>1 OPTIMAL</td>
</tr>
<tr>
<td>OBJECTIVE VALUE</td>
<td>11025.0000</td>
</tr>
</tbody>
</table>
Improve Readability

Format the code for readability using **spacing** and **indents**.

- Align item names, descriptions, and definitions
- Indent in sums, loops, and ifs to delineate terms
- Use blank lines to set things off
- Do not split variables between lines in equations, but rather keep them together with all their index positions
Do you prefer this?

**Variables**
- production(products, Quarters) production and sales
- openstock(rawmaterial, Quarters) opening stocks, profit

**Positive variables**
- production, openstock

**Equations**
- capacity(quarter) capacity constraint, stockbalan(rawmaterial, Quarters) stock balance,
- profitacct profit definition
- capacity(quarter). sum(products, production(products, quarter))
- l = mxcapacity
- stockbalan(rawmaterial, Quarters+1) openstock(rawmaterial, Quarters+1) = openstock(rawmaterial, Quarters) -
- sum(products, usage(rawmaterial, products) *production(products, Quarters));
- profitacct. profit = sum(quarter, sum(products, expectprof(products, quarter) *production(products, quarter)) - sum(rawmaterial, miscdata("store-cost", rawmaterial) *openstock(rawmaterial, quarter)) +
- sum(rawmaterial, miscdata("endinv-value", rawmaterial) * openstock(rawmaterial, "winter"));
- openstock.up(rawmaterial, "spring") = miscdata("max-stock", rawmaterial);
Equations

\[ \text{capacity(quarter)} \]
\[ \text{stockbalan(rawmaterial, Quarters)} \]
\[ \text{profitacct} \]
\[ \text{capacity constraint} \]
\[ \text{stock balance} \]
\[ \text{profit definition} \]

\[ \text{capacity(quarter)}. . . \]
\[ \text{sum(products, production(products, quarter))} \]
\[ = \text{L= mxcapacity} \]
\[ \text{stockbalan(rawmaterial, Quarters+1)}. . . \]
\[ \text{openstock(rawmaterial, Quarters+1)} \]
\[ = \text{E= openstock(rawmaterial, Quarters)} \]
\[ - \text{sum(products, usage(rawmaterial, products)} \]
\[ * \text{production(products, Quarters)} \] ;

\[ \text{profitacct}. . . \]
\[ \text{profit} \]
\[ = \text{E= } \]
\[ \text{sum(quarter, sum(products, expectprof(products, quarter)} \]
\[ * \text{production(products, quarter)} \] )
\[ - \text{sum(rawmaterial, miscdata("store-cost", rawmaterial)} \]
\[ * \text{openstock(rawmaterial, quarter)} \] ) )
\[ + \text{sum(rawmaterial, miscdata("endinv-value", rawmaterial)} \]
\[ * \text{openstock(rawmaterial, "winter") } \] ;
Questions?