First Look at GAMS and GAMSIDE
An Introduction

Pei Huang¹

¹Department of Agricultural Economics
Texas A&M University

Based on materials written by Gillig & McCarl and improved upon by many previous lab instructors

Special thanks to Mario Andres Fernandez
Outline

1 Overview
   - Information
   - Topics

2 Introduction to GAMS/GAMSIDE
   - GAMS Language
   - GAMSIDE

3 Formulation of a Simple Problem
   - An Example
   - Model Specifications in GAMS
   - GAMS Output
   - GAMS Project
Info

- Office: AGLS 386
- Email: peihuang@tamu.edu or petephuang@gmail.com
- Office hour: “Open door policy” or by appointment
- Class time: 8:00am - 9:00am, Thursday. We may have more sessions in a week whenever Dr. McCarl is traveling. I will email you when there is a change.
- Total 10-12 sessions
Purpose: Connect lecture materials to practical programming in GAMS.

You will do Handson exercises, plus additional model exposure in lab but no assigned.

- 6 assignments
- Equally weighted
- Usually one week after assignment, by 5 pm
- Printed / electronic copy

For FULL CREDIT use good modeling practices

- Descriptive variable names ("demand" and "inputs" not "x" and "y")
- Use descriptive text
- Organize programs for a clear flow
- I need to see that you understand the problem, not just programming
Topics

- First Look at GAMS and GAMSIDE
- Formulation of A General Problem
- Model Inspection & Error Messages
- Power of GAMS
- Good Modeling Practices
Topics

- First Look at GAMS and GAMSIDE
- Formulation of A General Problem
- Model Inspection & Error Messages
- Power of GAMS
- Good Modeling Practices
Topics

- First Look at GAMS and GAMSIDE
- Formulation of A General Problem
- Model Inspection & Error Messages
- Power of GAMS
- Good Modeling Practices
Topics

- First Look at GAMS and GAMSIDE
- Formulation of A General Problem
- Model Inspection & Error Messages
- Power of GAMS
- Good Modeling Practices
Topics

- First Look at GAMS and GAMSIDE
- Formulation of A General Problem
- Model Inspection & Error Messages
- Power of GAMS
- Good Modeling Practices
Topics

- Conditionals
- GAMS Check: Pre/Post Solution Analysis
- Examining A Model for Flaws
- Report Writing
- Comparative Analysis: Multiple Submissions and Loops
- Non-linear Programming Problems
Topics

- Conditionals
- GAMS Check: Pre/Post Solution Analysis
- Examining A Model for Flaws
- Report Writing
- Comparative Analysis: Multiple Submissions and Loops
- Non-linear Programming Problems
Topics

- Conditionals
- GAMS Check: Pre/Post Solution Analysis
- Examining A Model for Flaws
- Report Writing
- Comparative Analysis: Multiple Submissions and Loops
- Non-linear Programming Problems
Topics

- Conditionals
- GAMS Check: Pre/Post Solution Analysis
- Examining A Model for Flaws
- Report Writing
- Comparative Analysis: Multiple Submissions and Loops
- Non-linear Programming Problems
Topics

- Conditionals
- GAMS Check: Pre/Post Solution Analysis
- Examining A Model for Flaws
- Report Writing
- Comparative Analysis: Multiple Submissions and Loops
- Non-linear Programming Problems
Topics

- Conditionals
- GAMS Check: Pre/Post Solution Analysis
- Examining A Model for Flaws
- Report Writing
- Comparative Analysis: Multiple Submissions and Loops
- Non-linear Programming Problems
What is GAMS?

- Generalized Algebraic Modeling System

- A programming language for setting up and solving mathematical programming optimization models

- All-In-One Package that allows one to
  - Specify the structure of an optimization model
  - Specify and calculate data that go into the model
  - Solve that model
  - Conduct report writing on a model
  - Perform a comparative static analysis
What is GAMS IDE?

- Integrated Development Environment

- A graphical interface to create, debug, edit and run GAMS files
- Download the free version from GAMS website
  http://www.gams.com/download/
Suppose we wish to solve the optimization problem

\[
\begin{align*}
\text{max} & \quad 3X_{\text{bread}} + 10X_{\text{meat}} + 18X_{\text{wine}} & \quad \text{Objective} \\
\text{s.t.} & \quad 1.5X_{\text{bread}} + 6X_{\text{meat}} + 8X_{\text{wine}} \leq 20 & \quad \text{Money} \\
& \quad 5X_{\text{bread}} + 10X_{\text{meat}} + 30X_{\text{wine}} \leq 60 & \quad \text{Time} \\
& \quad X_{\text{bread}} \geq 0.8 & \quad \text{Sustenance} \\
& \quad X_{\text{bread}}, X_{\text{meat}}, X_{\text{wine}} \geq 0 & 
\end{align*}
\]

- **Variables:** \(X_{\text{bread}}, X_{\text{meat}}, X_{\text{wine}}\)
- **Equations**
  - **Objective**
  - **Constraints:** Money, Time, Sustenance
Using GAMS

```
* A Simple Example for GAMS Class, FALL 09

Variable
  utility this is the objective you want to maximize;

Positive Variables
  bread lbs of bread to be consumed
  meat lbs of meat to be consumed
  wine gallons of wine to be consumed;

Equations
  objective the objective function that defines utility
  money the daily budget
  time the time constraint
  sustenance the minimum level of bread consumption;

objective.. utility =e= 3*bread + 10*meat + 18*wine;

money..
  1.5*bread + 6*meat + 8*wine =l= 20;

time..
  5*bread + 10*meat + 30*wine =l= 60;

sustenance..
  bread =g= 0.8;

Model Consumption /All/;
Solve Consumption using LP maximizing utility;
```
Using GAMS

- Variable specifications
- Objective function

Example GAMS code:

```
Variable
  utility this is the objective you want to maximize;

Positive Variables
  bread  lbs of bread to be consumed
  meat  lbs of meat to be consumed
  wine  gallons of wine to be consumed;

Equations

  objective the objective function that defines utility
  money  the daily budget
  time  the time constraint
  sustenance the minimum level of bread consumption;

objective..  
  utility =e= 3*bread + 10*meat + 18*wine;

money..  
  1.5*bread + 6*meat + 8*wine =l= 20;

time..  
  5*bread + 10*meat + 30*wine =l= 60;

sustenance..  
  bread =g= 0.8;

Model Consumption /All/;
Solve Consumption using LP maximizing utility;
```
Using GAMS

- Variable specifications
- Objective function
- Considered variables

Example code:

Variable
utility this is the objective you want to maximize;

Positive Variables
bread lbs of bread to be consumed
meat lbs of meat to be consumed
wine gallons of wine to be consumed;

Equations
objective the objective function that defines utility
money the daily budget
time the time constraint
sustenance the minimum level of bread consumption;

objective..
utility =e= 3*bread + 10*meat + 18*wine;

money..
1.5*bread + 6*meat + 8*wine =l= 20;
time..
5*bread + 10*meat + 30*wine =l= 60;
sustenance..
bread =g= 0.8;

Model Consumption /All/;
Solve Consumption using LP maximizing utility;
Using GAMS

- Variable specifications
  - Objective function
  - Considered variables
- Equation specifications
  - Declaration
Using GAMS

- **Variable specifications**
  - Objective function
  - Considered variables

- **Equation specifications**
  - Declaration
  - Algebraic structure specification
Using GAMS

- Variable specifications
  - Objective function
  - Considered variables
- Equation specifications
  - Declaration
  - Algebraic structure specification
- Model statement
Using GAMS

- Variable specifications
  - Objective function
  - Considered variables
- Equation specifications
  - Declaration
  - Algebraic structure specification
- Model statement
- Solve statement

Example GAMS code:

```
* A Simple Example for GAMS Class, FALL 09

Variable
  utility this is the objective you want to maximize;

Positive Variables
  bread lbs of bread to be consumed
  meat lbs of meat to be consumed
  wine gallons of wine to be consumed;

Equations
  objective the objective function that defines utility
  money the daily budget
  time the time constraint
  sustenance the minimum level of bread consumption;

objective ..
  utility =e= 3*bread + 10*meat + 18*wine;

money..
  1.5*bread + 6*meat + 8*wine =l= 20;

time..
  5*bread + 10*meat + 30*wine =l= 60;

sustenance..
  bread =g= 0.8;

Model Consumption /all/;
Solve Consumption using LP maximizing utility;
```
Variable Specification

Types of variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Unrestricted variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive Variable</td>
<td>Nonnegative</td>
</tr>
<tr>
<td>Binary Variable</td>
<td>0 or 1</td>
</tr>
<tr>
<td>Integer Variable</td>
<td>0, 1, 2, ..., 9999</td>
</tr>
</tbody>
</table>

Example.gms

```gams
* A Simple Example for GAMS Class, FALL 09

Variable
utility  this is the objective you want to maximize;

Positive Variables
bread  lbs of bread to be consumed
meat   lbs of meat to be consumed
wine   gallons of wine to be consumed

Equations
objective  the objective function that defines utility
money     the daily budget
sustenance the minimum level of bread consumption;

objective.. utility =e= 3*bread + 10*meat + 18*wine;
money..
   1.5*bread + 6*meat + 8*wine =l= 20;
time..
   5*bread + 10*meat + 30*wine =l= 60;

Consumption /All/;
Solve Consumption using LP maximizing utility;
```

13:30 | Insert
Variable Specification

- **Types of variables**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Unrestricted variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive Variable</td>
<td>Nonnegative</td>
</tr>
<tr>
<td>Binary Variable</td>
<td>0 or 1</td>
</tr>
<tr>
<td>Integer Variable</td>
<td>0, 1, 2, ..., 9999</td>
</tr>
</tbody>
</table>

- **GAMS requires variables in each problem to be identified.**
Variable Specification

- **Types of variables**
  - | Variable | Unrestricted variables |
  - | Positive Variable | Nonnegative |
  - | Binary Variable | 0 or 1 |
  - | Integer Variable | 0, 1, 2, ..., 9999 |

- GAMS requires variables in each problem to be identified.

- The objective function together has to be defined as a variable.
Variable Specification

- **Types of variables**
  - | Variable | Unrestricted variables |
  - | Positive Variable | Nonnegative |
  - | Binary Variable | 0 or 1 |
  - | Integer Variable | 0, 1, 2, ..., 9999 |

- GAMS requires variables in each problem to be identified.
- The objective function together has to be defined as a variable.
- Variable names can be up to 64 characters and can have explanatory text.

```plaintext
Variable
utility this is the objective you want to maximize;
Positive Variables
bread lbs of bread to be consumed
meat lbs of meat to be consumed
wine gallons of wine to be consumed;

Equations
objective the objective function that defines utility
money the daily budget
time the time constraint
sustenance the minimum level of bread consumption;

objective.
utility =e= 3*bread + 10*meat + 18*wine;

money.. 1.5*bread + 6*meat + 8*wine =l= 20;
time.. 5*bread + 10*meat + 30*wine =l= 60;
sustenance.. bread =g= 0.8;
```

Example 0: gms
Equation Specification

```
Variable
utility this is the objective you want to maximize;

Positive Variables
bread  lbs of bread to be consumed
meat   lbs of meat to be consumed
wine  gallons of wine to be consumed;

Equations
objective  the objective function that defines utility
money   the daily budget
time   the time constraint
sustenance the minimum level of bread consumption;

objective.. utility =e= 3*bread + 10*meat + 18*wine;
money..   1.5*bread + 6*meat + 8*wine =l= 20;
time..    5*bread + 10*meat + 30*wine =l= 60;
sustenance.. bread =g= 0.8;
```

Model Consumption /All/;
Solve Consumption using LP maximizing utility;
```
Equation Specification

- Name each equation in the model.

```
Variable
  utility   this is the objective you want to maximize;

Positive Variables
  bread     lbs of bread to be consumed
  meat      lbs of meat to be consumed
  wine      gallons of wine to be consumed

Equations
  objective  the objective function that defines utility
  money      the daily budget
  time       the time constraint
  sustenance the minimum level of bread consumption;

  objective..
    utility =e= 3*bread + 10*meat + 18*wine;
  money..
    1.5*bread + 6*meat + 8*wine =l= 20;
  time..
    5*bread + 10*meat + 30*wine =l= 60;
  sustenance..
    bread =g= 0.8;

Model Consumption /All/;
Solve Consumption using LP maximizing utility;
```
Equation Specification

- Name each equation in the model.
- Specify the algebraic structure.
Equation Specification

- Name each equation in the model.
- Specify the algebraic structure.
- Two dots following each equation name
Equation Specification

- Name each equation in the model.
- Specify the algebraic structure.
- Two dots following each equation name.
- The specific forms of constraints:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Constraint forms</th>
</tr>
</thead>
<tbody>
<tr>
<td>=e=</td>
<td>equal</td>
</tr>
<tr>
<td>=l=</td>
<td>less than or equal</td>
</tr>
<tr>
<td>=g=</td>
<td>greater than or equal</td>
</tr>
</tbody>
</table>
Model Statement

- Name the model
- Specify equations that will be included in the model in slashes / /

```gams
Variable
utility this is the objective you want to maximize;
Positive Variables
bread lbs of bread to be consumed
meat lbs of meat to be consumed
wine gallons of wine to be consumed;
Equations
objective the objective function that defines utility
money the daily budget
time the time constraint
sustenance the minimum level of bread consumption;

objective..
    utility =e= 3*bread + 10*meat + 18*wine;

money..
    1.5*bread + 6*meat + 8*wine =l= 20;

time..
    5*bread + 10*meat + 30*wine =l= 60;

sustenance..
    bread =g= 0.8;

Model Consumption /All/;
Solve Consumption using LP maximizing utility;
```
Model Statement

- Name the model
- Specify equations that will be included in the model in slashes `/ /`

Examples

```gams
Model Consumption1 /All/;
Model Consumption2 /objective, money, time/;
```
Solve Statement

**Examples**

**SOLVE** Consumption **USING**

**LP Maximizing** Utility;

**SOLVE** Consumption **USING**

**LP Minimizing** Disutility;

**SOLVE** Consumption **USING**

**MIP Maximizing** Utility;

**SOLVE** Consumption **USING**

**NLP Maximizing** Utility;
Heads Up for Semi-colons

- Each declaration must terminate with a “;”.
- Statements may be several lines long or may contain several elements.
- Omission would lead to (many) Syntax Errors!
GAMS Output

Optimal solution found.
Objective :  40.640000

<table>
<thead>
<tr>
<th>LOWER</th>
<th>LEVEL</th>
<th>UPPER</th>
<th>MARGINAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>EQU objective</td>
<td>.</td>
<td>.</td>
<td>1.000</td>
</tr>
<tr>
<td>EQU money</td>
<td>-INF</td>
<td>20.000</td>
<td>20.000</td>
</tr>
<tr>
<td>EQU time</td>
<td>-INF</td>
<td>60.000</td>
<td>60.000</td>
</tr>
<tr>
<td>EQU sustenance</td>
<td>0.800</td>
<td>0.800</td>
<td>+INF</td>
</tr>
</tbody>
</table>

Objective
- the objective function that defines utility
Money
- the daily budget
Time
- the time constraint
Sustenance
- the minimum level of bread consumption

<table>
<thead>
<tr>
<th>LOWER</th>
<th>LEVEL</th>
<th>UPPER</th>
<th>MARGINAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAR utility</td>
<td>-INF</td>
<td>40.640</td>
<td>+INF</td>
</tr>
<tr>
<td>VAR bread</td>
<td>.</td>
<td>0.800</td>
<td>+INF</td>
</tr>
<tr>
<td>VAR meat</td>
<td>.</td>
<td>1.160</td>
<td>+INF</td>
</tr>
<tr>
<td>VAR wine</td>
<td>.</td>
<td>1.480</td>
<td>+INF</td>
</tr>
</tbody>
</table>
GAMS Output

Output navigation

Objective: 40.640000

- **EQU objective**: 1.000
- **EQU money**: -INF 20.000 20.000 1.200
- **EQU time**: -INF 60.000 60.000 0.280
- **EQU sustenance**: 0.800 0.800 +INF -0.200

**Objective**: the objective function that defines utility
**Money**: the daily budget
**Time**: the time constraint
**Sustenance**: the minimum level of bread consumption

---

**VAR utility**: -INF 40.640 +INF .
**VAR bread**: . 0.800 +INF .
**VAR meat**: . 1.160 +INF .
**VAR wine**: . 1.480 +INF .
GAMS Output

- Output navigation
- Objective function solution
GAMS Output

- Output navigation
- Objective function solution
- Equation solutions
GAMS Output

- Output navigation
- Objective function solution
- Equation solutions
- Variable solutions
GAMS Project

- \*.gpr: GAMS project file
- \*.gms: GAMS file where to write program codes BEFORE the model runs
- \*.lst: GAMS output file generated AFTER the model runs
- \*.log: GAMS log file generated WHILE the model runs
GAMS Project

- ***.gpr**: GAMS project file
- ***.gms**: GAMS file where to write program codes BEFORE the model runs
- ***.lst**: GAMS output file generated AFTER the model runs
- ***.log**: GAMS log file generated WHILE the model runs
Heads On 1

Solve the feeding problem using GAMS

\[
\begin{align*}
\text{min} & \quad 3X_{\text{corn}} + 4X_{\text{hay}} + 5X_{\text{soybeans}} & \quad \text{Cost/lb} \\
\text{s.t.} & \quad 0.01X_{\text{corn}} + 0.07X_{\text{hay}} + 0.03X_{\text{soybeans}} \geq 0.05 & \quad \text{Protein} \\
& \quad 0.02X_{\text{corn}} + 0.14X_{\text{hay}} + 0.1X_{\text{soybeans}} \geq 0.09 & \quad \text{Vitamin A} \\
& \quad X_{\text{corn}} + X_{\text{hay}} + X_{\text{soybeans}} = 1 & \quad \text{Unit} \\
& \quad X_{\text{corn}}, X_{\text{hay}}, X_{\text{soybeans}} \geq 0 & \quad \text{Non-negative}
\end{align*}
\]

1. Set up the model in GAMS and solve it.
2. Solve the model when the corn price increases to $6?
3. Solve the model when the minimum requirement of vitamin A is set to zero?
Requirements

- Create HandsOn1.gms with “* your name” on the first line of the code.
- A good programming habit: make comments in your code.
- Turn in an electronic copy of *..gms file for Part 1 via email.
- Turn in a hard copy of *.lst file (from Solution Report to the end) for ALL three parts.
- You can work as a group with no more than three people.
- The hands on is due by 5:00pm, September 12.