Pre-Conference Workshops

Michael Bussieck
Steve Dirkse
Fred Fiand
Lutz Westermann
Outline

Part I: An Introduction to GAMS

Part II: Stochastic programming in GAMS

Part III: The GAMS Object-Oriented API's

Part IV: Code embedding in GAMS
An Introduction to GAMS: Agenda

- GAMS at a Glance
- Foundation and Design Principles
- GAMS – A simple Example
- Wrap-Up
Company

- Went commercial in 1987
- GAMS Development Corp. (USA), GAMS Software GmbH (Germany)
- 2016: New management team
- Software Tool Provider
GAMS at a Glance

- Pioneered Algebraic Modeling Languages
- Robust, scalable state-of-the-art algebraic modeling technology for complex, large-scale optimization
- Open architecture and uniform interface to all major commercial and academic solvers (30+ integrated)
- Evolution through more than 25 years of R&D and user feedback, maturity through experience and rigorous testing
**GAMS Users and Application Areas**

- **13,500+ licenses**
- **Users: 50% academic, 50% commercial or governmental**
- **Used in more than 120 countries**

### Broad Range of Application Areas

<table>
<thead>
<tr>
<th>Agricultural Economics</th>
<th>Applied General Equilibrium</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chemical Engineering</td>
<td>Economic Development</td>
</tr>
<tr>
<td>Econometrics</td>
<td>Energy</td>
</tr>
<tr>
<td>Environmental Economics</td>
<td>Engineering</td>
</tr>
<tr>
<td>Finance</td>
<td>Forestry</td>
</tr>
<tr>
<td>International Trade</td>
<td>Logistics</td>
</tr>
<tr>
<td>Macro Economics</td>
<td>Military</td>
</tr>
<tr>
<td>Management Science/OR</td>
<td>Mathematics</td>
</tr>
<tr>
<td>Micro Economics</td>
<td>Physics</td>
</tr>
</tbody>
</table>
Strong Development Environment

➢ Project management
➢ Editor / Syntax coloring / Spell checks
➢ Model Debugging / Profiling
➢ Solver selection / Option selection
➢ Data viewer
➢ GAMS Processes Control
Uniform System Documentation

GAMS Documentation Center

The GAMS Documentation Center provides you with the technical information on getting started, using and maintaining our GAMS (General Algebraic Modeling System) products.

- Release Notes - What's new in GAMS and all changes.
- Installation Notes - GAMS Installation guides on different operating systems.
- Licensing - GAMS Licensing.
- Tutorials and Examples - Step-by-step guides including how to use GAMS.

Model Libraries

This documentation guides GAMS User through several topics in GAMS system.

- **GAMS Language** - This part introduces the components of the GAMS language in an ordered way, interspersed with detailed examples that are often drawn from the model library. All models from the model library are enclosed in square parenthesis (for example, [TRANSPORT]).
  - **Introduction** - an introductory to GAMS User's Guide.
  - **GAMS Programs** - The structure of the GAMS language and its components
  - **Set Definition** - The declaration and initialization of sets, subsets, and domain checking.
  - **Dynamic Sets** - The membership assignment, the usage of dollar controls, and set operations.
  - **Sets as Sequences: Ordered Sets** - Special features used to deal with a set as if it were a sequence.
  - **Data Manipulations with Parameters** - The declaration and assignment of GAMS parameters.
  - **Data Entry: Parameters, Scalars and Tables** - Three basic forms of GAMS data types : Parameters, Scalars and Tables.
Simple Integration of GAMS Models

Object Oriented API’s

➢ Use GAMS for modeling and optimization tasks
➢ Connects GAMS to other environments
  ➢ Programming languages (.Net, C++, Java, Python)
  ➢ Applications (through Smart Links)
  ➢ (New) Embedded Code (Python)
➢ Communication through Memory or Files
Free Model Libraries

More than 1400 models!
Where to Find Help?

- **Documentation Center:**
  https://www.gams.com/latest/docs/

- **Free Model Libraries:**
  https://www.gams.com/latest/docs/modlibsindex.html

- **Mailing Lists, Newsletters, and Forum:**
  https://www.gams.com/community/newsletters-mailing-list/
  //forum.gamsworld.org/

- **YouTube Channel:**
  https://www.youtube.com/user/GAMSLessons

- **GAMS Support:** support@gams.com
Foundation of GAMS

- Powerful algebraic modeling language
- Open architecture with interfaces to other systems
- Independent layers
Powerful Declarative Language

- Similar to mathematical notation
- Easy to learn - few basic language elements: sets, parameters, variables, equations, models
- Model is executable (algebraic) description of the problem
- Lots of code optimization under the hood
Mix of Declarative and Imperative Elements

Control Flow Statements (e.g. loops, for, if,...), macros and functions

Advantages:
- Build complex problem algorithms within GAMS
- Simplified interaction with other systems:
  - Data exchange
  - GAMS process control
Foundation of GAMS

- Powerful algebraic modeling language
- Open architecture with interfaces to other systems
- Independent layers
Open Architecture

Designed to interact with other Systems

- Solver Interfaces: Proprietary or Open Source (COIN)
- Data Exchange and GAMS Control
  - ASCII or Binary (GDX)
  - OO-API (.Net, C++, Java, Python, file or memory)
- Smart Links to other applications (e.g. MS Excel, Databases, Matlab, R, ...)
- Code Embedding
Foundation of GAMS

- Powerful algebraic modeling language
- Open architecture with interfaces to other systems
- Independent layers
Separation of Model and Platform

Supported Platforms

➢ Move models between platforms with ease!
Separation of **Model and Solver**

One environment for a wide range of solvers

- All major commercial LP/MIP solver
- Open Source Solver (COIN)
- Also solver for NLP, MINLP, global, and stochastic optimization

➢ More than 30 Solvers integrated!
Separation of Model and Solver

Uniform interface to all major solvers
- Switching between solvers with one statement
- Unified Documentation
- Licensing (GAMS as a “license broker“)

Av. number of commercial solvers per license
- Academic clients: 2.9
- Commercial: 2.2
Separation of **Model** and **Data**

**Sets**
- i canning plants
- j markets

**Parameters**
- a(i) capacity of plant i in cases
- b(j) demand at market j in cases
- d(i,j) distance in thousands of miles
- c(i,j) transport cost in thousands of dollars per case

**Scalar** f

**Variables**
- x(i,j) shipment quantities in cases
- z total transportation costs in thousands of dollars

**Positive Variable** x

**Equations**
- cost define objective function
- supply(i) observe supply limit at plant i
- demand(j) satisfy demand at market j

Model transport /all/ ;

- Declarative Modeling
- Sparse Data Structures
- Various ways to exchange data
- ASCII
- Binary
Separation of Model and User Interface

No preference for a particular user interface

- Open architecture and interfaces to other systems
- OO-API’s for seamless integration
- Smart Links
- Mode of Operation
- Interactive or Embedded / Batch
- Local or Remote
Application – Cloud Computing

xyz Energy Company

Scenario Analysis in the Cloud
➢ Solve 1,000+ scenarios (MIPs, one hour) every week overnight
➢ Issues:
   ➢ Costs (Licensing)
   ➢ Automation / Security
Application – Cloud Computing

xyz – Energy Company

Implementation:
- Amazon Cloud: 1,000+ parallel machines (instances), Python, GAMS + OO Python API
- Automated setup, including
  - Starting instances
  - Prepare / Submit / Run GAMS jobs
  - Collect results
  - Stop instances
Application – Cloud Computing

Commercial Aspects

“Hardware” Amazon Cloud (1,000 instances):
Hardware Costs / run: $70!
(1,000 instances/run * $0.07 instance / hour)

Software Licensing:
➢ Gurobi and IBM offer per-usage license
➢ Client with strong preference for annual license fee, not a per-usage license
Application – Cloud Computing

45 Provided Model Instances

➢ Statistics:
  ➢ 163,608 – 1,959,550 rows
  ➢ 84,930 – 983,587 var. (32,240-258,796 dis.)
  ➢ 447,537 – 6,068,729 NZ
➢ Tests with CPLEX, SCIP, and CBC
➢ 60 minutes, gap max. 1%
➢ Manual option tuning for SCIP
  (thanks to Gerald Gamrath & Ambros Gleisner)
Results

- CPLEX: All instances solved to optimality
- SCIP:
  - Could solve all 45 instances
  - But: After 60 min. 2 instances with gap > 20%
- CBC:
  - Did also well
  - But: After 60 min. no solution for some instances (< 10%)
Application – Cloud Computing

Proposed Strategy

➢ Run all instances simultaneously with SCIP and CBC
  ➔ „hardware“ costs: $0.07 per instance hour

➢ After 60 minutes take the best solution

➢ If necessary solve „difficult“ model instances with CPLEX (outside the cloud)
Agenda

GAMS at a Glance

Foundation and Design Principle

GAMS – A simple Example

Wrap-Up
A Simple Transportation Problem

What does this example show?

- It gives a first glimpse of how a problem can be formulated in GAMS
- It shows some basics of data exchange with GAMS
- It shows how easy it is to change model type and, consequently, solver technology
A Simple **Transportation Problem**

Minimize Transportation cost
subject to Demand satisfaction at markets
Supply constraints

Supply (cases)

<table>
<thead>
<tr>
<th></th>
<th>Seattle</th>
<th>San Diego</th>
</tr>
</thead>
<tbody>
<tr>
<td>New York</td>
<td>350</td>
<td>600</td>
</tr>
<tr>
<td>Chicago</td>
<td>1.4</td>
<td>1.7</td>
</tr>
<tr>
<td>Topeka</td>
<td>1.8</td>
<td>2.5</td>
</tr>
</tbody>
</table>

Distance (thousand miles)

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

Demand (cases)

<table>
<thead>
<tr>
<th></th>
<th>Seattle</th>
<th>San Diego</th>
</tr>
</thead>
<tbody>
<tr>
<td>New York</td>
<td>275</td>
<td>325</td>
</tr>
<tr>
<td>Chicago</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>Topeka</td>
<td>300</td>
<td>325</td>
</tr>
</tbody>
</table>

Freight: $90 case / thousand miles
Model types in this example

LP
Determine minimum transportation cost.
Result: city to city shipment volumes.

MIP
Allows discrete decisions,
e.g. if we ship, then we ship at least 100 cases.

MINLP
Allows non-linearity,
e.g. a smooth decrease in unit cost when shipping volumes grows.
Mathematical Model Formulation

Indices: $i$ (Canning plants)  
         $j$ (Markets)  
Decision variables: $x_{ij}$ (Number of cases to ship)  
Data: $c_{ij}$ (Transport cost per case)  
      $a_i$ (Capacity in cases)  
      $b_i$ (Demand in cases)

\[
\begin{align*}
\text{min} & \quad \sum_i \sum_j c_{ij} \cdot x_{ij} \\
\text{subject to} & \quad \sum_j x_{ij} \leq a_i \quad \forall i \quad \text{(Shipments from each plant \( \leq \) supply capacity)} \\
& \quad \sum_i x_{ij} \geq b_j \quad \forall j \quad \text{(Shipments to each market \( \geq \) demand)} \\
& \quad x_{ij} \geq 0 \quad \forall i, j \quad \text{(Do not ship from market to plant)} \\
& \quad i, j \in \mathbb{N}
\end{align*}
\]
GAMS Algebra (declarative Model)

Sets
   i    canning plants
   j    markets

Parameters
   a(i)  capacity of plant i in cases
   b(j)  demand at market j in cases
   d(i,j) distance in thousands of miles
   c(i,j) transport cost in thousands of dollars per case

Scalar
   f   freight in dollars per case per thousand miles

Variables
   x(i,j)  shipment quantities in cases
   z       total transportation costs in thousands of dollars

Positive Variable
   x

Equations
   cost       define objective function
   supply(i)  observe supply limit at plant i
   demand(j)  satisfy demand at market j

\[
\begin{align*}
\text{cost} & \quad \text{sum}((i,j), c(i,j) \times x(i,j)) \\
\text{supply}(i) & \quad \text{sum}(j, x(i,j)) = a(i) \\
\text{demand}(j) & \quad \text{sum}(i, x(i,j)) = b(j)
\end{align*}
\]

Model transport /all/ ;

Model is executable description of the problem
Model types in this example

- **LP**: Determine minimum transportation cost. Result: city to city shipment volumes.
- **MIP**: Allows discrete decisions, e.g. if we ship, then we ship at least 100 cases.
- **MINLP**: Allows non-linearity, e.g. a smooth decrease in unit cost when shipping volumes grows.
**MIP Model: Minimum Shipment of 100 cases**

- **Shipment volume**: \( x \) (continuous variable)
- **Discrete decision**: \( \text{ship} \) (binary variable)

**Constraints:**

\[
x_{i,j} \geq 100 \cdot \text{ship}_{i,j} \quad \forall i,j \quad \text{(if ship=1, then ship at least 100)}
\]

\[
x_{i,j} \leq \text{bigM} \cdot \text{ship}_{i,j} \quad \forall i,j \quad \text{(if ship=0, then do not ship at all)}
\]

\[
\text{ship}_{i,j} \in \{0,1\}
\]
MIP Model: GAMS Syntax

```gams
* MIP
scalar minS minimum shipment / 100 /
bigM big M;
bigM = min(smax(i,a(i)), smax(j,b(j)));
binary variable ship(i,j) '1 if we ship from i to j, otherwise 0';

equation minship(i,j) minimum shipment
   maxship(i,j) maximum shipment;
minship(i,j).. x(i,j) =g= minS * ship(i,j);
maxship(i,j).. x(i,j) =l= bigM * ship(i,j);

Model transportMIP / transportLP, minship, maxship / ;
option optcr = 0;
Solve transportMIP using MIP minimizing z ;
rep(i,j,'MIP') = x.l(i,j);
display rep;
```
MIP Model: Results

**** REPORT SUMMARY :
  0 NONOPT
  0 INFEASIBLE
  0 UNBOUNDED

GAMS 24.8.5 r61358 Released May 10, 2017 WEX-WEI x86 64bit/MS Windows 08/07
General Algebraic Modeling System

Execution

---- 66 PARAMETER rep report parameter

LP MIP

seattle .new-york 50.000
seattle .chicago 300.000 300.000
san-diego.new-york 275.000 325.000
san-diego.topeka 275.000 275.000

EXECUTION TIME = 0.000 SECONDS

USER: Franz Nelissen
GAMS Software GmbH
Model types in this example

- **LP**: Determine minimum transportation cost. Result: city to city shipment volumes.
- **MIP**: Allows discrete decisions, e.g. if we ship, then we ship at least 100 cases.
- **MINLP**: Allows non-linearity, e.g. a smooth decrease in unit cost when shipping volumes grows.
MINLP: Cost Savings

The cost per case decreases with an increasing shipment volume.

Replace:

\[
\min \sum_i \sum_j c_{ij} \cdot x_{ij} \quad (\text{Minimize total transportation cost})
\]

With

\[
\min \sum_i \sum_j c_{ij} \cdot x_{ij}^{beta} \quad (\text{Minimize total transportation cost})
\]
* MINLP
Scalar beta / 0.95 /
Equation costnlp define non-linear objective function;
costnlp.. z =e= sum((i,j), c(i,j)*x(i,j)**beta) ;

Model transportMINLP / transportMIP - cost + costnlp /;

Solve transportMINLP using MINLP minimizing z ;

rep(i,j,'MINLP') = x.l(i,j);
display rep;
MINLP Model: Results

GAMS 24.8.5 r61358 Released May 10, 2017 WEX-WEI x86 64bit/MS Windows 08/07
General Algebraic Modeling System
Execution

--- 85 PARAMETER rep report parameter

<table>
<thead>
<tr>
<th>LP</th>
<th>MIP</th>
<th>MINLP</th>
</tr>
</thead>
<tbody>
<tr>
<td>seattle .new-york</td>
<td>50.000</td>
<td>50.000</td>
</tr>
<tr>
<td>seattle .chicago</td>
<td>300.000</td>
<td>300.000</td>
</tr>
<tr>
<td>san-diego .new-york</td>
<td>275.000</td>
<td>275.000</td>
</tr>
<tr>
<td>san-diego .topeka</td>
<td>275.000</td>
<td>275.000</td>
</tr>
</tbody>
</table>

LOWER LEVEL UPPER

VAR z

- INF 153.675 + INF

LOWER LEVEL UPPER

VAR z

- INF 115.438 + INF

Plane Index

<table>
<thead>
<tr>
<th>seattle</th>
<th>new-york</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>chicago</td>
<td></td>
<td>300.000</td>
</tr>
<tr>
<td>san-diego</td>
<td>new-york</td>
<td>275</td>
</tr>
<tr>
<td></td>
<td>topeka</td>
<td>275</td>
</tr>
</tbody>
</table>

Entry | Symbol | Type | Dim | Nr Elem |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>a</td>
<td>Par</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>b</td>
<td>Par</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>14</td>
<td>beta</td>
<td>Par</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>bigM</td>
<td>Par</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>c</td>
<td>Par</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>9</td>
<td>cost</td>
<td>Equ</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>19</td>
<td>costnlp</td>
<td>Equ</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>d</td>
<td>Par</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>
Agenda

- GAMS at a Glance
- Foundation and Design Principle
- GAMS – A simple Example
- Wrap-Up
Thank You

Meet us at the GAMS booth!