Pre Conference Workshop

Lutz Westermann
Clemens Westphal

GAMS Software GmbH
GAMS Development Corporation

LWestermann@gams.com
CWestphal@gams.com

www.gams.com

Rotterdam, September 3, 2013
Outline

- **GAMS**
  - GAMS at a Glance
  - Simple Example
  - GAMS/Base

- **Features you might not know about**
  - Syntax
  - Data Import/Export
  - Advanced Use of GAMS Solver Links
  - Extending the GAMS Syntax
  - Other Tools
# Outline

<table>
<thead>
<tr>
<th>• GAMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>– GAMS at a Glance</td>
</tr>
<tr>
<td>– Simple Example</td>
</tr>
<tr>
<td>– GAMS/Base</td>
</tr>
<tr>
<td>• Features you might not know about</td>
</tr>
<tr>
<td>– Syntax</td>
</tr>
<tr>
<td>– Data Import/Export</td>
</tr>
<tr>
<td>– Advanced Use of GAMS Solver Links</td>
</tr>
<tr>
<td>– Extending the GAMS Syntax</td>
</tr>
<tr>
<td>– Other Tools</td>
</tr>
</tbody>
</table>
GAMS at a Glance

Algebraic Modeling System

• Facilitates to formulate mathematical optimization problems similar to algebraic notation
  ➡ Simplified model building

• Provides links to appropriate state-of-the-art external algorithms
  ➡ Efficient solution process
GAMS at a Glance

**General Algebraic Modeling System**

- Roots: World Bank, 1976
- Went commercial in 1987
- GAMS Development Corp.
- GAMS Software GmbH
- Broad academic & commercial user community and network
GAMS’ Fundamental concepts

- Platform independence

- Hassle-free switch of solution methods

- Open architecture and interfaces to other systems

- Balanced mix of declarative and procedural elements
GAMS’ Fundamental concepts

- Platform independence
- Hassle-free switch of solution methods
- Open architecture and interfaces to other systems
- Balanced mix of declarative and procedural elements

30+ Integrated Solvers
GAMS’ Fundamental concepts

- Platform independence
- Hassle-free switch of solution methods
- **Open architecture and interfaces to other systems**
- Balanced mix of declarative and procedural elements

Binary Data Exchange

- Fast exchange of data
- Syntactical check on data before model starts
- Data Exchange at any stage (Compile and Run-time)
- Platform Independent
- Direct GDX interfaces and general API
- Scenario Management Support
- Full Support of Batch Runs
GAMS’ Fundamental concepts

- Platform independence
- Hassle-free switch of solution methods
- Open architecture and interfaces to other systems
- Balanced mix of declarative and procedural elements

Declaration of..
- Sets
- Parameters
- Variables
- Equations
- Models
- …

Procedural Elements like…
- loops
- if-then-else
- …
Outline

<table>
<thead>
<tr>
<th>• GAMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>– GAMS at a Glance</td>
</tr>
<tr>
<td>– Simple Example</td>
</tr>
<tr>
<td>– GAMS/Base</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>• Features you might not know about</th>
</tr>
</thead>
<tbody>
<tr>
<td>– Syntax</td>
</tr>
<tr>
<td>– Data Import/Export</td>
</tr>
<tr>
<td>– Advanced Use of GAMS Solver Links</td>
</tr>
<tr>
<td>– Extending the GAMS Syntax</td>
</tr>
<tr>
<td>– Other Tools</td>
</tr>
</tbody>
</table>
A Transportation Model

Seattle (350)
San Diego (600)
Chicago (300)
Topeka (275)
New York (325)
A Transportation Model

Minimize Transportation cost
subject to Demand satisfaction at markets
Supply constraints
Model Formulation

Indices: $i$ (Canning plants)  
               $j$ (Markets)

Decision variables: $x_{ij}$ (Number of cases to ship)

Parameter: $c_{ij}$ (Transport cost per case)

$$\min \sum_i \sum_j c_{ij} \cdot x_{ij}$$ (Minimize total transportation cost)

subject to

$$\sum_j x_{ij} \leq sup_i \quad \forall i$$ (Shipments from each plant $\leq$ supply capacity)

$$\sum_i x_{ij} \geq dem_j \quad \forall j$$ (Shipments to each market $\geq$ demand)

$x_{ij} \geq 0 \quad \forall i, j$

$i, j \in \mathbb{N}$
Variables
\[ x(i,j) \] shipment quantities in cases
\[ z \] total transportation costs in thousands of dollars;

Positive Variable \( x \);

Equations
\[ \text{cost} \] define objective function
\[ \text{supply}(i) \] observe supply limit at plant \( i \)
\[ \text{demand}(j) \] satisfy demand at market \( j \);

\[ \text{cost} \ldots \quad z = \text{sum}(i,j), c(i,j) \times x(i,j) \]
\[ \text{supply}(i) \ldots \quad \text{sum}(j, x(i,j)) = a(i) \]
\[ \text{demand}(j) \ldots \quad \text{sum}(i, x(i,j)) = b(j) \]

Model transport /all/ ;
Outline

<table>
<thead>
<tr>
<th>GAMS</th>
<th>Features you might not know about</th>
</tr>
</thead>
<tbody>
<tr>
<td>• GAMS at a Glance</td>
<td>• Syntax</td>
</tr>
<tr>
<td>• Simple Example</td>
<td>• Data Import/Export</td>
</tr>
<tr>
<td>• GAMS/Base</td>
<td>• Advanced Use of GAMS Solver Links</td>
</tr>
<tr>
<td></td>
<td>• Extending the GAMS Syntax</td>
</tr>
<tr>
<td></td>
<td>• Other Tools</td>
</tr>
</tbody>
</table>
GAMS at a Glance

The GAMS/BASE Module

- Compiler and Execution System
- GAMS IDE (Windows)
- Documentation + Model libraries
- GDX Utilities
- Free Solvers/Solver Links
GAMS at a Glance

The GAMS/BASE Module

• Compiler and Execution System

• GAMS IDE (Windows)

• Documentation + Model libraries

• GDX Utilities

• Free Solvers/Solver Links
Integrated Development Environment

- Project management
- Editor / Syntax coloring / Spell checking
- Launching and monitoring of (multiple) GAMS processes
- Listing file / Tree view / Syntax-error navigation
- Solver selection / Option selection
- GDX viewer
  - Data cube
  - Data export (e.g. to MS Excel)
  - Charting facilities
- Model libraries
- Documentation
- Diff for GDX and Text
GAMS at a Glance

The GAMS/BASE Module

- Compiler and Execution System
- GAMS IDE (Windows)
- Documentation + Model libraries
- GDX Utilities
- Free Solvers/Solver Links
Documentation

• **Distributed Documentation**
  – GAMS Users Guide
  – Expanded GAMS Users Guide (McCarl)
  – Solver Manuals
  – GAMS Utility Manuals

• **Wikis**
  – Support Wiki  [http://support.gams-software.com](http://support.gams-software.com)
  – Interfaces Wiki  [http://interfaces.gams-software.com](http://interfaces.gams-software.com)
Documentation

- **Groups**
  - User Group [http://www.gams.com/maillist/gams_l.htm](http://www.gams.com/maillist/gams_l.htm)
  - Google Group [http://groups.google.de/group/gamsworld](http://groups.google.de/group/gamsworld)

- **Newsletter**
  - Release List

- **Search all GAMS Websites** [http://www.gams.com/search.htm](http://www.gams.com/search.htm)
Distributed Model Libraries

- **GAMS Model Library**
  - Example and user-contributed models
  - Very often used as templates
  - Tests for
    - Solver robustness and correctness
    - Backward compatibility

- **GAMS Test Library**
  - Transparent and reproducible Quality Assurance Tests
  - Tests for
    - Solver correctness
    - Special functions
    - GAMS utilities
Distributed Model Libraries

- **GAMS Data Utilities Library**
  - Demonstration of the various utilities interfacing GAMS with other applications
  - E.g. gdxxrw, mdb2gms, sql2gms

- **GAMS EMP Library**
  - Examples for the use of Extended Mathematical Programming

- **Practical Financial Optimization Models**
  Models of the book

  “PRACTICAL FINANCIAL OPTIMIZATION – A Library of GAMS Models”

  by Consiglio, Nielsen and Zenios
GAMS at a Glance

The GAMS/BASE Module

- Compiler and Execution System
- GAMS IDE (Windows)
- Documentation + Model libraries
- GDX Utilities
- Free Solvers/Solver Links
Gams Data eXchange

Binary Data Exchange

- Fast exchange of data
- Syntactical check on data before model starts
- Data Exchange at any stage (Compile and Run-time)
- Platform Independent
- Direct GDX interfaces and general API
- Scenario Management Support
- Full Support of Batch Runs

GDX Tools

- Invert
- IDE
- GDX Viewer
- GDXrank
- GDX2HAR/HAR2GDX
- GDXmerge
- GDXdump
- GDXcopy
- GDXAPI
- GAMS
- GDX2XLS
- GDXxrw
- MDB2GMS
- GDXdiff
- GDXcopy
GAMS at a Glance

The GAMS/BASE Module

- Compiler and Execution System
- GAMS IDE (Windows)
- Documentation + Model libraries
- GDX Utilities
- Free Solvers/Solver Links
GAMS at a Glance

The GAMS/BASE Module

Free Solvers:

- Convert
- EMP/JAMS, DE, NLPEC
- BENCH, EXAMINER, GAMSCHK
- BDMLP, LS, and MILES
- KESTREL (Remote Solver Execution on NEOS Servers)
- COIN-OR: Cbc, IpOpt, BonMin, Couenne, ...
- Soplex, Scip (academic only)
- All other solvers in limited versions
GAMS/Kestrel

- Remote Solver Execution on NEOS Servers
- From within your usual GAMS modeling environment
- Receiving results that can be processed as with any local solver

```gams
Model transport /all/;

Option lp=kestrel;
transport.optfile=1;

$onecho > kestrel.opt
ketrel_solver xpress
neos_server www.neos-server.org
socket_timeout 10
$offecho

Solve transport using lp minimizing z;
```

--- Executing KESTREL: elapsed 0:00:00.005
Connecting to: http://www.neos-server.org:3332

NEOS job#=956988, pass=LXBsrGJe
Check the following URL for progress report:
http://www.neos-server.org/neos/cgi-bin/nph-neos-solver.cgi?admin=results&jobnumber=956988&pass=LXBsrGJe

FICO-Xpress 24.1.2 r40979 Released Jun 16, 2013 LEG x86_64/Linux
Xpress Optimizer 24.01
Xpress Optimizer 64-bit v24.01.04 (Hyper capacity)
Coin-OR

An initiative to spur the development of open-source software for the OR community

http://www.coin-or.org/

• A repository of currently ~50 open-source projects
  – Solvers
  – Interfaces
  – Tools

• An active OR community
  – Mailing lists
  – Wikis
The Coin-OR / GAMSLinks Project

https://projects.coin-or.org/GAMSLinks

Goals

• Easy access to COIN-OR solvers via GAMS
• Broadening the audience of COIN-OR
• Broadening the audience of GAMS
• Help developers to connect their solvers to GAMS
• Provide access to GAMS benchmarking and quality assurance tools
The Coin-OR / GAMSLinks Project

GAMS interfaces to open-source Solvers

• COIN-OR Linear Programming (CLP) and Branch and Cut (CBC)
  – LP and MIP solver from J. Forrest

• COIN-OR Open Solver Interface (OSI)
  – Bare bone LP/MIP solver links to Cplex, Gurobi, Mosek, Soplex and Xpress

• Interior Point Optimizer (IPOPT)
  – Large scale NLP solver from A. Wächter
GAMS interfaces to open-source Solvers

- Solving Constraint Integer Programs (SCIP)
  - MIP/MINLP solver developed at Zuse Institute Berlin, TU Darmstadt and FAU Erlangen/Nürnberg

- Basic Open-source Nonlinear Mixed Integer programming (BONMIN)
  - Branch and Cut based MINLP solver from P. Bonami et.al.

- Convex Over and Under Envelopes for Nonlinear Estimation (COUENNE)
  - Branch and Bound MINLP solver from P. Belotti
Outline

<table>
<thead>
<tr>
<th>• GAMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>– GAMS at a Glance</td>
</tr>
<tr>
<td>– Simple Example</td>
</tr>
<tr>
<td>– GAMS/Base</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>• Features you might not know about</th>
</tr>
</thead>
<tbody>
<tr>
<td>– Syntax</td>
</tr>
<tr>
<td>– Data Import/Export</td>
</tr>
<tr>
<td>– Advanced Use of GAMS Solver Links</td>
</tr>
<tr>
<td>– Extending the GAMS Syntax</td>
</tr>
<tr>
<td>– Other Tools</td>
</tr>
</tbody>
</table>
Then ...

In Table 17.1 we list sizes and attributes of representative models that are “large” in the sense that they are near the limit of what is practical on a personal computer, along with the model generation time (GAMS) and solution time (solver), both in minutes. These examples were run on an 8 MHz AT with an 80287 coprocessor and 640K of RAM. The times shown are to give you a rough idea of what is possible: these are not precisely controlled benchmarks, and we have a host of performance improvements in mind for the near future.

Table 17.1: Problem Characteristics

<table>
<thead>
<tr>
<th>Name</th>
<th>Number of Rows</th>
<th>Number of Columns</th>
<th>Number of Nonzeros</th>
<th>Generation Time (^a)</th>
<th>Solution Time (^a)</th>
<th>Iterations</th>
<th>Solver</th>
</tr>
</thead>
<tbody>
<tr>
<td>DINAMICO</td>
<td>318</td>
<td>425</td>
<td>4156</td>
<td>3.0</td>
<td>30.1</td>
<td>628</td>
<td>MINOS</td>
</tr>
<tr>
<td>SARF</td>
<td>532</td>
<td>542</td>
<td>3949</td>
<td>37.7</td>
<td>115.8</td>
<td>2775</td>
<td>MINOS</td>
</tr>
<tr>
<td>FERTD (^b)</td>
<td>458</td>
<td>2968</td>
<td>7252</td>
<td>11.4</td>
<td>28.3</td>
<td>1368</td>
<td>ZOOM</td>
</tr>
<tr>
<td>CAMCGB (^c)</td>
<td>243</td>
<td>280</td>
<td>1356</td>
<td>0.8</td>
<td>7.0</td>
<td>189</td>
<td>MINOS</td>
</tr>
<tr>
<td>GANGES (^d)</td>
<td>274</td>
<td>357</td>
<td>1405</td>
<td>1.8</td>
<td>7.3</td>
<td>187</td>
<td>MINOS</td>
</tr>
<tr>
<td>YEMCGBM (^e)</td>
<td>168</td>
<td>258</td>
<td>953</td>
<td>0.9</td>
<td>7.6</td>
<td>600</td>
<td>ZOOM</td>
</tr>
<tr>
<td>EGTYP (^f)</td>
<td>281</td>
<td>618</td>
<td>3168</td>
<td>4.0</td>
<td>25.3</td>
<td>1551</td>
<td>ZOOM</td>
</tr>
</tbody>
</table>

\(^a\) Measured in minutes.
\(^b\) The problem is too big for MINOS, ZOOM was used instead.
\(^c\) A nonlinear problem. 63\% of the non-zeroes are nonlinear.
\(^d\) A nonlinear problem. 58\% of the non-zeroes are nonlinear.
\(^e\) A mixed binary problem, with 55 binary variables (solved with a relative termination criterion of 10\%).
\(^f\) A linear problem, solved using XMP which is contained within ZOOM.
... and now

<table>
<thead>
<tr>
<th>Type</th>
<th>s in 1988</th>
<th>s in 2013</th>
<th>Improvement Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>camcge NLP</td>
<td>468</td>
<td>0.031</td>
<td>15097</td>
</tr>
<tr>
<td>dinamico LP</td>
<td>1986</td>
<td>0.125</td>
<td>15888</td>
</tr>
<tr>
<td>egypt* LP</td>
<td>1758</td>
<td>0.015</td>
<td>117200</td>
</tr>
<tr>
<td>fertd* MIP</td>
<td>2382</td>
<td>0.062</td>
<td>38419</td>
</tr>
<tr>
<td>ganges NLP</td>
<td>546</td>
<td>0.109</td>
<td>5009</td>
</tr>
<tr>
<td>sarf LP</td>
<td>9210</td>
<td>0.139</td>
<td>66259</td>
</tr>
<tr>
<td>yemcem* MIP</td>
<td>510</td>
<td>0.140</td>
<td>3643</td>
</tr>
</tbody>
</table>

* 1988 solver ZOOM, 2008 solver CPLEX 11.0.1
Improvements on all Frontiers

- **Solver Technology**
  - Updates for existing solver
  - New solvers

- **Productivity Tools**
  - Databases, spreadsheets
  - Specialized visualization tools
  - IDE improvements
  - Grid computing

- **Interfaces**
  - Gams Data eXchange
  - Using GAMS from other applications
Outline

• GAMS
  – GAMS at a Glance
  – Simple Example
  – GAMS/Base

• Features you might not know about
  – Syntax
  – Data Import/Export
  – Advanced Use of GAMS Solver Links
  – Extending the GAMS Syntax
  – Other Tools
Basic Set Declaration

Set  i  / i1, i2, i3  /
Set  j  / j1 * j3  /
Set  k  / k3 * k1  /
Set  l(i,j) / i1.j1, i1.j2, i1.j3
     / i2.j1, i2.j2, i2.j3
     / i3.j1, i3.j2, i3.j3  /
Set  m(i,j) / (i1*i3).(j1*j3)  /
Set  n(i,j) / #i.#j  /;
Matching Operator

Set  i / t1.s3,t2.s4,t3.s5 /; (Product Operator)

  can be written as

Set  i / t1*t6:s3*s5 /; (Matching Operator)

• Example “Count Tuples”:

  Sets  h /h1*h24/, d /d1*d365/, t /t1*t8760/
  dh (d,h)    /#d.#h/
  tdh (t,d,h) /#t:#dh/;

  \[ t1.d1.h1, t2.d1.h2, \ldots, \\[ t25.d2.h1, t26.d2.h2, \ldots \]
Matching Operator in Option Statements

Set  i /i1*i2/,  j /j3*j5/
     k /k1*k5/,  cnt /c1*c100/
 ijk(i,j,k),  x(I,j,k,cnt);

Option  ijk(i:j,k),  x(ijk:cnt);

rijk:  i1.j3.k1, i1.j3.k2, …,
       i2.j4.k1, i2.j4.k2, …,
       i2.j4.k5

xhr:  i1.j3.k1.c1, i1.j3.k2.c2, …,
      i2.j4.k1.c5, i2.j4.k2.c6, …,
      i2.j4.k5.c10
Enhanced Data Statements

- Allow initial values for equations and variables
- Follow the syntax for list and table data statement for parameters by adding an additional dimension to specify the specific data attribute

Variable table \( x(i,j) \) initial values

<table>
<thead>
<tr>
<th></th>
<th>l</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>seattle. new-york</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>seattle. Chicago</td>
<td>300</td>
<td></td>
</tr>
<tr>
<td>seattle. topeka</td>
<td></td>
<td>0.36</td>
</tr>
<tr>
<td>san-diego. new-york</td>
<td>275</td>
<td></td>
</tr>
<tr>
<td>san-diego. Topeka</td>
<td>275</td>
<td></td>
</tr>
<tr>
<td>san-diego. chicago</td>
<td>0.009</td>
<td></td>
</tr>
</tbody>
</table>
The GAMS Macro Facility

• Basic Definition
  - $macro name macro body
  - $macro name(arg1,...) macro body with tokens arg1,...

• Multi-Argument Example
  $macro ratio(a,b) a/b
  \[ z = ratio(x_1,x_2); \]
  \[ z = x_1/x_2; \]

• Macros within Macros
  $macro product(a,b) a*b
  $macro addup(i,x,z) sum(i,product(x(i),z))
  \[ z = addup(j,a1,x1); \]
  \[ z = sum(j,a1(j)*x1); \]
The GAMS Macro Facility (contd.)

- Careful expansion (&)
  
  \$macro f(i) sum(j, x(i,j))
  \$macro equ(q) equation equ_&q; equ_&q.. q =e= 0;
  equ(f(i))
  \rightarrow equation equ_f(i); equ_f(i).. sum(j, x(i,j)) =e= 0;

- Removing outer set of quotes (&&)
  
  \$macro d(q) display &&q;
  d(""here it is", i, k')
  \rightarrow display "here it is", i, k;

  \$macro dd(q) &&q)
  z=dd('sum(j,a1(j)');
  \rightarrow z=sum(j,a1(j));
## Outline

- **GAMS**
  - GAMS at a Glance
  - Simple Example
  - GAMS/Base

- **Features you might not know about**
  - Syntax
  - Data Import/Export
  - Advanced Use of GAMS Solver Links
  - Extending the GAMS Syntax
  - Other Tools
GDXXRW

- Read and write Excel spreadsheet data
- Can read multiple ranges in a spreadsheet and write the data to a GDX file
- Can read from a GDX file and write the data to different ranges in a spreadsheet
- Examples in the GAMS Data Library

Hands-On
GDXMRW

- Import/export data between GAMS and MATLAB
- Call GAMS models from MATLAB
- Get results back in MATLAB
- Gives MATLAB users the ability to use all the optimization capabilities of GAMS
- Allows visualization of GAMS models directly within MATLAB
- More Information:
GDXRRW

• GDXRRW bridges the gap between R and GAMS (import/export data between GAMS and R)

• Fits into the ecosystem of existing GDX utilities

• Presents data in a natural form for R users

• More information:  
  http://support.gams.com/doku.php?id=gdxrrw:interfacing_gams_and_r

Source: http://blog.modelworks.ch
Load from GDX

Compile Time:
$gdxIn transSol.gdx  // open file for reading
$load               // list file content
$load i             // load symbol i
$load jj=j           // load symbol j as jj
$loadDC a b         // load a & b domain controlled
$load[DC]M k        // load symbol k, merge content
$load[DC]R l        // load symbol l, replace content
$gdxIn              // close open file

Execution Time:
execute_load 'transSol.gdx' a;

put_utility 'gdxin' / 'transSol.gdx';
execute_load b;
## Outline

<table>
<thead>
<tr>
<th><strong>GAMS</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>– GAMS at a Glance</td>
</tr>
<tr>
<td>– Simple Example</td>
</tr>
<tr>
<td>– GAMS/Base</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Features you might not know about</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>– Syntax</td>
</tr>
<tr>
<td>– Data Import/Export</td>
</tr>
<tr>
<td>– Advanced Use of GAMS Solver Links</td>
</tr>
<tr>
<td>– Extending the GAMS Syntax</td>
</tr>
<tr>
<td>– Other Tools</td>
</tr>
</tbody>
</table>
Model transport /all/ ;
Option solvelink = { %Solvelink.ChainScript%,
                        %Solvelink.CallScript%,
                        %Solvelink.CallModule%,
                        %Solvelink.AsyncGrid%,
                        %Solvelink.AsyncSimulate%,
                        %Solvelink.LoadLibrary%};

solve transport using lp minimizing z;

• ChainScript [0]: Solver process, GAMS vacates memory
  + Maximum memory available to solver
  + protection against solver failure (hostile link)
  - swap to disk
Solvelink Option – cont.

• Call{Script [1]/Module [2]}: Solver process, GAMS stays live
  + protection against solver failure (*hostile* link)
  + no swap of GAMS database
  - file based model communication

• LoadLibrary [5]: Solver DLL in GAMS process
  + fast memory based model communication
  + update of model object inside the solver (hot start)
  - not supported by all solvers
Solving Scenarios

transport.gms (LP) solved 500 times with CPLEX:

```plaintext
Loop(s,
    d(i,j) = dd(s,i,j);
    f = ff(s);
    solve transport using lp minimizing z;
    rep(s) = transport.objval;
);
```

<table>
<thead>
<tr>
<th>Setting</th>
<th>Solve time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solvelink=%Solvelink.ChainScript%</td>
<td>52.221</td>
</tr>
<tr>
<td>Solvelink=%Solvelink.CallModule%</td>
<td>37.366</td>
</tr>
<tr>
<td>Solvelink=%Solvelink.LoadLibrary%</td>
<td>03.252</td>
</tr>
</tbody>
</table>
Gather-Update-Solve-Scatter (GUSS)

<table>
<thead>
<tr>
<th>Setting</th>
<th>Solve time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solvelink=%Solvelink.ChainScript%</td>
<td>52.221</td>
</tr>
<tr>
<td>Solvelink=%Solvelink.CallModule%</td>
<td>37.366</td>
</tr>
<tr>
<td>Solvelink=%Solvelink.LoadLibrary%</td>
<td>03.252</td>
</tr>
<tr>
<td>GUSS</td>
<td>01.046</td>
</tr>
</tbody>
</table>

- Update model data instead of matrix coefficients/rhs
- Hot start (keep the model hot inside the solver and use solver’s best update mechanism)
- Save model generation and solver setup time
- Model rim unchanged from scenario to scenario
- Apriori knowledge of all scenario data

53
GUSS

- Dynamic model – rolling horizon

- Example:
  - Combined Heat and Power Planning with Heat Storage. All data known apriori but heat storage level
  - Can’t use GUSS
  - Implement GUSS in programming language
    - Identify some parameters as “modifiable” parameters
    - Implement rolling horizon in programming language
Tracing Solve Process

Solver Options (e.g. Cplex, Gurobi, Xpress):

- **MipTrace**
  - Writes a file that records the best integer and best bound values every `miptracenode` nodes and at `miptracetime`-second intervals

- **MipTraceNode**
  - Specifies the node interval between entries to the `MipTrace` file [Default: 100]

- **MipTraceTime**
  - Specifies the time interval, in seconds, between entries to the `MipTrace` file [Default: 5]
Generates a Trace file during solve:

* miptrace file gurobi.trc: ID = Gurobi
* fields are lineNum, seriesID, node, seconds, bestFound, bestBound
1, S, 0, 0, -0, 30
2, N, 100, 0.113, 21, 27
3, N, 200, 0.169, 21, 27
4, N, 300, 0.212, 21, 27
5, N, 400, 0.255, 21, 26
6, N, 500, 0.31, 21, 26
7, E, 683, 0.668, 21, 23
* miptrace file gurobi.trc closed

- Common format among all solvers that support this option
- Available with: ANTIGONE, BONMIN, CBC, CPLEX, COUENNE, GloMIQO, Gurobi, SBB, SCIP, Sulum, Xpress (Partly using different option names)
Solution Pool

- Several solver links can write out alternative solutions to GDX: AlphaECP, ANTIGONE, BARON, CBC, CPLEX, GloMIQO, Gurobi, SCIP, Xpress
- BARON, CPLEX, and Xpress also offer functionality to explicitly search for alternative solutions
- See GAMS Model Library model solnpool

```plaintext
---- 142 PARAMETER xcostX cost structure by solution

<table>
<thead>
<tr>
<th></th>
<th>totcost</th>
<th>tcost</th>
<th>fcost</th>
</tr>
</thead>
<tbody>
<tr>
<td>file1</td>
<td>499.000</td>
<td>219.000</td>
<td>280.000</td>
</tr>
<tr>
<td>file2</td>
<td>512.000</td>
<td>212.000</td>
<td>300.000</td>
</tr>
<tr>
<td>file3</td>
<td>985.000</td>
<td>355.000</td>
<td>630.000</td>
</tr>
</tbody>
</table>
```
Outline

<table>
<thead>
<tr>
<th>GAMS</th>
<th>Features you might not know about</th>
</tr>
</thead>
<tbody>
<tr>
<td>- GAMS at a Glance</td>
<td>- Syntax</td>
</tr>
<tr>
<td>- Simple Example</td>
<td>- Data Import/Export</td>
</tr>
<tr>
<td>- GAMS/Base</td>
<td>- Advanced Use of GAMS Solver Links</td>
</tr>
<tr>
<td></td>
<td>- Extending the GAMS Syntax</td>
</tr>
<tr>
<td></td>
<td>- Other Tools</td>
</tr>
</tbody>
</table>
Function Libraries

- Allows users to import functions from an external library into a GAMS model.
- Imported functions can be used in the same way as intrinsic GAMS functions.
- Some function libraries are included in the GAMS distribution.
- Users can create their own libraries using an open programming interface (simple examples written in C, Delphi and Fortran come with every GAMS system).
- To make a library available call
  \$FuncLibIn <IntLibName> <ExtLibName>
- Declare functions similar to sets, parameters, ...,
  : Function <IntFuncName> /<IntLibName>.<FuncName>/;
Function Libraries – Included Examples

• FITfclib
  – FITPACK from P. Dierckx
  – One and two dimensional spline interpolation
• LSAdclib
  – Use sampling routines from Lindo inside GAMS
  – Requires GAMS/Lindo license (or runs in limited demo mode)
• PWPcclib
  – Piecewise polynomial function evaluation
• STOdclib
  – Random deviates, probability density functions, cumulative density functions and inverse cumulative density functions
  – E.g., ChiSquare, Gumbel, Logistic, Rayleigh, …
• TRIcclib, TRIdclib, TRIfclib
  – Simple examples compiled and as source code written in C, Delphi and Fortran respectively
Function Libraries – Interface

- int LibInit(
  abcRec_t *abc,    // in  handle
  const int version, // in  library version
  char *msg)         // out message

- int <FUNCTIONNAME>(
  abcRec_t *abc,    // in  handle
  const int DR,     // in  derivative request
  const int args,   // in  number of arguments
  const double x[], // in  arguments
  double *f,        // out function value
  double g[],       // out gradient
  double h[],       // out hessian
  void *cb,         // in  error callback
  void *usermem)    // in  user memory for error callback
Stochastic Programming in GAMS

- The Extended Mathematical Programming (EMP) framework is used to replace parameters in the model by random variables

- Support for Multi-stage recourse problems and chance constraint models

- Easy to add uncertainty to existing deterministic models, to either use specialized algorithms or create Deterministic Equivalent (new free solver DE)
With new modeling and solution concepts do not:
• overload existing GAMS notation right away!
• attempt to build new solvers right away!

But:
• Use existing language features to specify additional model features, structure, and semantics
• Express extended model in symbolic (source) form and apply existing modeling/solution technology
• Package new tools with the production system

→ Extended Mathematical Programming (EMP)
JAMS: a GAMS EMP Solver

- EMP Information
- Original Model
  - Translation
  - Mapping Solution into original space
  - Reformulated Model
    - Solving using established Algorithms
    - Solution
  - Viewable
Simple Example: Newsboy (NB) Problem

- A newsboy faces an uncertain demand for newspapers
- He can buy newspapers for fixed costs per unit
- He can sell newspapers for a fixed price
- For hold units he has to pay a disposal fee
- He has to satisfy his customers demand or has to pay a penalty

Decisions to make:
- How much newspaper should he buy “here and now” (without knowing the outcome of the uncertain demand)? → *First-stage decision*
- How many customers are lost after the outcome becomes known? → *Second-stage or recourse decision*
- Recourse decisions can be seen as
  - penalties for bad first-stage decisions
  - variables to keep the problem feasible

Hands-On ➔ nbsimple.gms
Random Variables

Discrete Distribution

Normal Distribution

Poisson Distribution

Exponential Distribution
Random Variables (RV) \[\text{randVar}\]

- Defines both discrete and parametric random variables:

  \[
  \text{randVar } \text{rv } \text{discrete } \text{prob } \text{val} \{\text{prob } \text{val}\}
  \]

- The distribution of discrete random variables is defined by pairs of the probability \(\text{prob}\) of an outcome and the corresponding realization \(\text{val}\):

  \[
  \text{randVar } \text{rv } \text{distr } \text{par} \{\text{par}\}
  \]

- The name of the parametric distribution is defined by \(\text{distr, par}\) defines a parameter of the distribution
Joint Random Variables

Demand
- Prob: 0.2, d: 40
- Prob: 0.7, d: 45
- Prob: 0.1, d: 50

Price
- Prob: 0.2, p: 55
- Prob: 0.7, p: 60
- Prob: 0.1, p: 65

Demand / Price
- Prob: 0.04, d: 40 / p: 55
- Prob: 0.14, d: 40 / p: 60
- Prob: 0.02, d: 40 / p: 65
- Prob: 0.14, d: 45 / p: 55
- Prob: 0.49, d: 45 / p: 60
- Prob: 0.07, d: 45 / p: 65
- Prob: 0.02, d: 50 / p: 55
- Prob: 0.07, d: 50 / p: 60
- Prob: 0.01, d: 50 / p: 65

vs.

Demand / Price
- Prob: 0.2, d: 40
- Prob: 0.7, d: 45
- Prob: 0.1, d: 50

Price
- Prob: 0.2, p: 55
- Prob: 0.7, p: 60
- Prob: 0.1, p: 65
Joint RVs [jRandVar]

• Defines discrete random variables and their joint distribution:

\[ j\text{RandVar \ rv \ rv \ {rv} \ prob \ val \ val \ {val}} \]
\{prob \ val \ val \ {val}\}\)

• At least two discrete random variables \( r\text{v} \) are defined and the outcome of those is coupled
• The probability of the outcomes is defined by \( \text{prob} \) and the corresponding realization for each random variable by \( \text{val} \)
Stages

Stage 1: Decision X

Observation of d

Stage 2: Decisions S, I, L

...
Stages [stage]

• Defines the stage of random variables (rv), equations (equ) and variables (var):

  stage stageNo rv | equ | var {rv | equ | var}

• StageNo defines the stage number
• The default StageNo for the objective variable and objective equation is the highest stage mentioned
• The default StageNo for all the other random variables, equations and variables not mentioned is 1
Chance Constraints

OBJ.. Z =e= X1 + X2;
E1.. om1*X1 + X2 =g= 7;
E2.. om2*X1 + 3*X2 =g= 12;
Model sc / all /;
solve sc min z use lp;

chance E1 0.6
chance E2 0.6
Chance Constraints

3 out of 4 must be true
\[0.75 \geq 0.6\]
- \(1 \cdot X_1 + X_2 \geq 7\)
- \(2 \cdot X_1 + X_2 \geq 7\)
- \(3 \cdot X_1 + X_2 \geq 7\)
- \(4 \cdot X_1 + X_2 \geq 7\)

2 out of 3 must be true
\[0.66 \geq 0.6\]
- \(1 \cdot X_1 + 3 \cdot X_2 \geq 12\)
- \(2 \cdot X_1 + 3 \cdot X_2 \geq 12\)
- \(3 \cdot X_1 + 3 \cdot X_2 \geq 12\)
Chance Constraints \([\text{chance}]\)

- Defines individual or joint chance constraints (CC):

\[
\text{chance equ \{equ\} [holds] minRatio [weight|varName]}
\]

- Individual CC: A single constraint \(\text{equ}\) has to hold for a certain ratio \((0 \leq \text{minRatio} \leq 1)\) of the possible outcomes
- Joint CC: A set of constraints \(\text{equ}\) has to hold for a certain ratio \((0 \leq \text{minRatio} \leq 1)\) of the possible outcomes
- If \text{weight} is defined, the violation of a CC gets penalized in the objective \((\text{weight} \ \text{violationRatio})\)
- If \text{varName} is defined the violation get multiplied by this existing variable
## Outline

- **GAMS**
  - GAMS at a Glance
  - Simple Example
  - GAMS/Base

- **Features you might not know about**
  - Syntax
  - Data Import/Export
  - Advanced Use of GAMS Solver Links
  - Extending the GAMS Syntax
  - Other Tools
Matrix Utilities

- **INVERT**
  - Calculates the inverse of a matrix

- **CHOLESKY**
  - Computes the Cholesky factors of a symmetric positive-definite matrix

- **EIGENVALUE**
  - Computes the eigenvalues of a symmetric matrix

- **EIGENVECTOR**
  - Computes the eigenvalues and eigenvectors of a symmetric matrix
Check for GAMS Updates

```
C:\Program Files (x86)\GAMS23.6

C:\t m p>chk4upd
*** Processing GAMS system directory C:\PROGRA~2\GAMS23.6
*** Reading license file C:\PROGRA~2\GAMS23.6\gamslice.txt
*** You are running the x86/MS Windows version of GAMS
*** You could also use the x86_64 MS Windows version
*** The version of your GAMS system is 23.6.5
*** There is an update available for this system
*** You can download and use the most recent version of GAMS (23.7.3)
*** Please visit http://www.gams.com/download/
C:\t m p>
```