GAMS

Software Tutorial

Pete Steacy PSteacy@gams.com

GAMS Development Corp.

www.gams.com

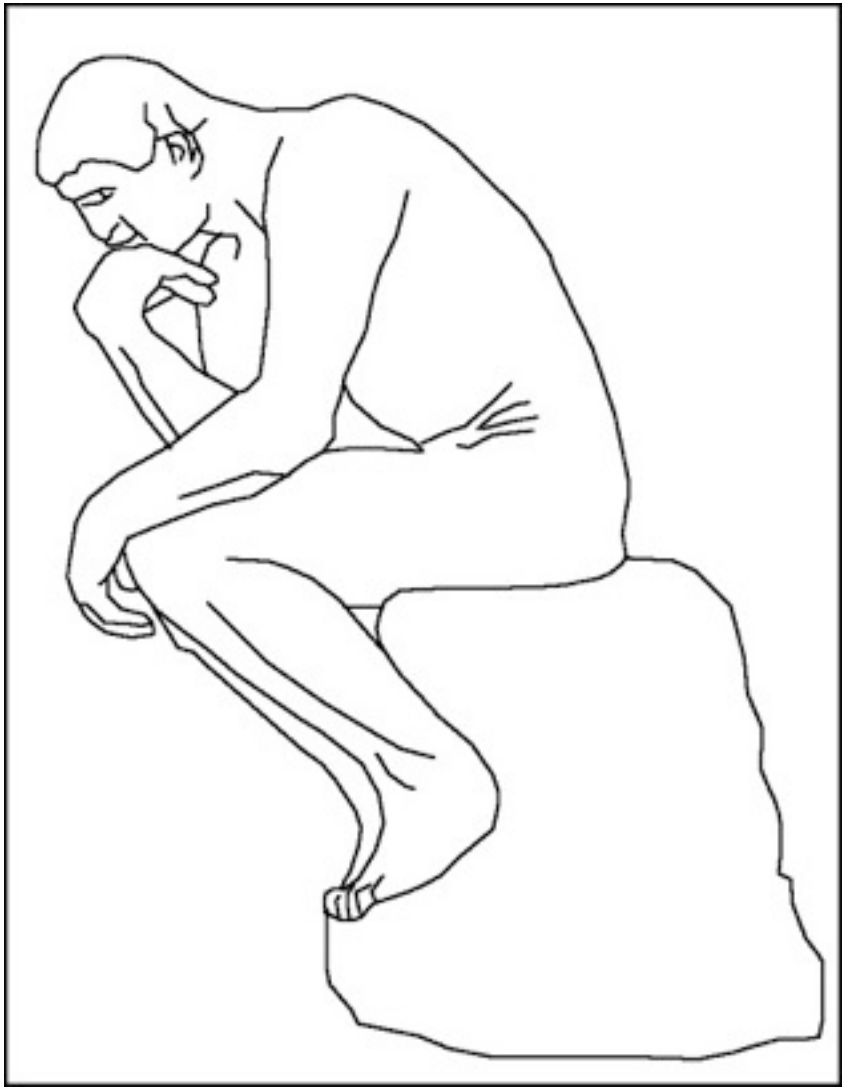
Huntington Beach, CA April 2012

Agenda

- What is GAMS?
- What is new?

Agenda

- What is GAMS?
  - GAMS at a Glance
  - A simple Example
  - Interfacing with other Applications

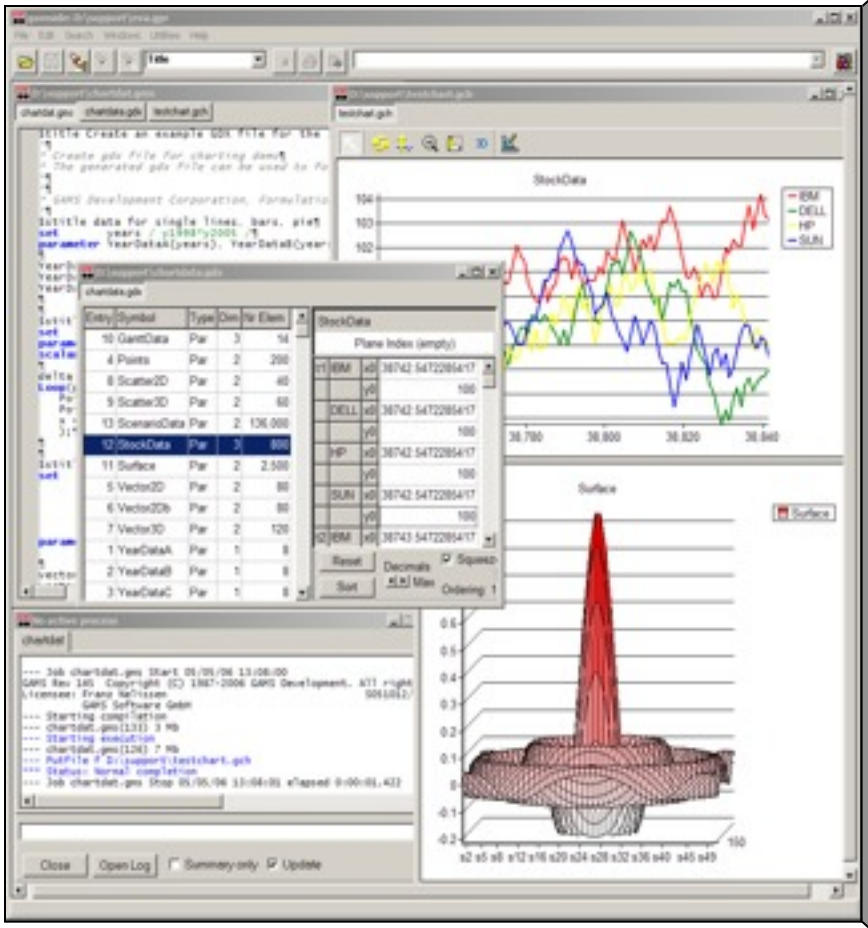


What does this modeler have to think about?

- Application
- Mathematics
- Computer Code

GAMS eases the transitions between these domains.

GAMS Development / Software at a Glance



- Roots: World Bank, 1976
- Went commercial in 1987
- GAMS Development Corp. (US)
- GAMS Software GmbH (Europe)
- Technical tool provider (Software)
- Broad academic & commercial user community and network
  - GAMS is used in more than 120 countries
  - Half of licenses commercially used

Broad Academic and Commercial Network

Downloads (March 2012)

Download GAMS Distribution 23.8.1 - March 17, 2012

Windows

Windows 32 bit — Windows 7, Windows Vista, Windows XP, Windows Server 2008, Windows Server 2003, and compatible on AMD- or Intel-based (x86_32) architectures

Windows 64 bit — Windows 7 x64, Windows Vista x64, Windows Server 2008 x64, Windows Server 2003 x64, and compatible on AMD- or Intel-based (x64_64) architectures

Unix

AIX — AIX 5.3 or higher, PowerPC chip, 64 bit (ppc_64)

Linux 32 bit — ASMD- or Intel-based 32-bit Linux systems. The software was built with the GNU Compiler Collection (GCC) toolset, ver 4.4 or higher.

Linux 64 bit — ASMD- or Intel-based 64-bit Linux systems (x86_64). The software was built with the GNU Compiler Collection (GCC) toolset, ver 4.4 or higher.

Mac OS X Intel 32 bit — Macintosh Intel-based systems (x86_32) built on Darwin 10.6 (Snow Leopard). Please note that this is a Mac OS X Terminal application and must be started/executed using the command line interface. Additional Information

Mac OS X Intel 64 bit — Macintosh Intel-based systems (x64_64) built on Darwin 10.6 (Snow Leopard). Please note that this is a Mac OS X Terminal application and must be started/executed using the command line interface. Additional Information

Solaris SPARC 32 bit — Solaris 2.8 or higher on SUN Sparc (sparc_32). Missing Fortran Run Time Environment!

Solaris SPARC 64 bit — Solaris 2.8 or higher on SUN Sparc (sparc_64)

Solaris x64 64 bit — Solaris 10 or higher on AMD- or Intel-based 64-bit (x64_64)

Wine

Linux Wine (beta) — ASMD- or Intel-based Linux systems. The software uses the Windows 32bit GAMS build and Wine. No separate Wine installation is required. For more info please visit this page.

Please also visit the information about the distribution history, changes, and incremental updates. For older distributions please follow this link. There are some mailing lists, which you about forthcoming releases, provide additional information, and are useful for questions about GAMS and modeling issues.

Amazon CloudFront Download Usage Report »		$67.04
United States		
$0.120 per GB - first 10 TB / month data transfer out	197.126 GB	23.66
$0.0100 per 10,000 HTTPS Requests	3 Requests	0.01
$0.0075 per 10,000 HTTP Requests	52,154 Requests	0.04
		23.71
Europe		
$0.120 per GB - first 10 TB / month data transfer out	212.982 GB	25.56
$0.0120 per 10,000 HTTPS Requests	1 Request	0.01
$0.0090 per 10,000 HTTP Requests	16,456 Requests	0.01
		25.58
Asia Pacific (Tokyo) Region		
$0.201 per GB - first 10 TB / month data transfer out (includes consumption tax).	23.800 GB	4.78
$0.0095 per 10,000 HTTP Requests (includes consumption tax).	4,676 Requests	0.01
		4.79
Asia Pacific (Singapore) Region		
$0.190 per GB - first 10 TB / month data transfer out	39.512 GB	7.51
$0.012 per 10,000 HTTPS Requests	1 Request	0.01
$0.0090 per 10,000 HTTP Requests	18,087 Requests	0.02
		7.54
South America		
$0.250 per GB - first 10 TB / month data transfer out	21.656 GB	5.41
$0.0160 per 10,000 HTTP Requests	1,535 Requests	0.01
		5.42

Total: 495 GB ~ 5,500 monthly downloads

Agenda

- What is GAMS?
- GAMS at a Glance
- A simple Example
- Interfacing with other Applications

GAMS at a Glance

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

GAMS at a Glance: Balanced mix..

Balanced mix of declarative and procedural elements

- Algebra (Expressions): model equations
- Relational Algebra (SQL) for data manipulation

Mathematical Notation → Notation in GAMS

$$\sum_{\substack{c,p:\\(c,p)\in\mathcal{N}}} tcost \cdot dist(c,p) \cdot x_p^c \rightarrow \min$$

$$\sum_{\substack{c,p:\\(c,p)\in\mathcal{N}}} x_p^c \leq sup(c) \quad \forall c$$

$$\sum_{\substack{c,p:\\(c,p)\in\mathcal{N}}} x_p^c \geq dem(p) \quad \forall p$$

$$x_p^c \geq 0 \quad \forall c,p : (c,p) \in \mathcal{N}$$

C:\work\gdxrw\trnsport.gms

trnsport.gms | trnsport.lst

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

cost ..        z  =e=  sum((i,j), c(i,j)*x(i,j)) ;

supply(i) ..   sum(j, x(i,j))  =l=  a(i) ;

demand(j) ..   sum(i, x(i,j))  =g=  b(j) ;

Model transport /all/ ;

Solve transport using lp minimizing z ;

Display x.l, x.m ;
```

GAMS at a Glance: Balanced Mix..

Balanced mix of declarative and procedural elements

- For

```
scalar scen;
for(scen=1 to 10 by 0.5,
   f       = 10*scen;
   c(i,j) = f * d(i,j) / 1000;
   Solve transport using lp minimizing z;
   Display z.l;);
```

- Loop/If

```
loop(h,
   if(work(h),
      pay(i,h) = 0.6*pay(i,h);
   else
      pay(i,h) = 1.5*pay(i,h);
   );
);
```

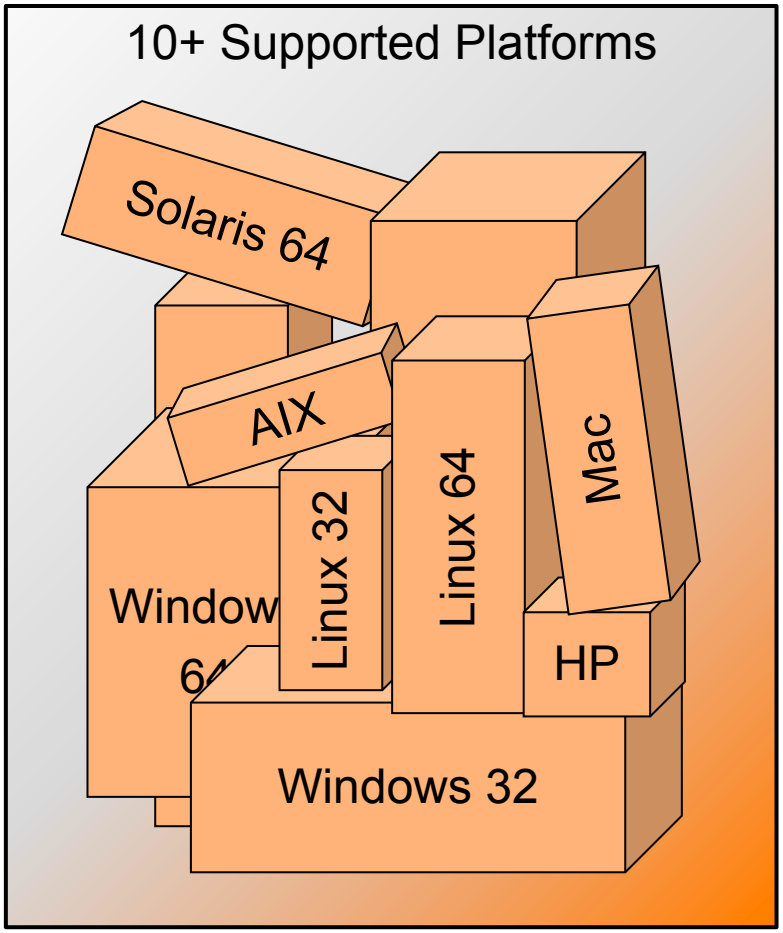
- While

```
Scalar scen /1/;
while(scen<=10,
   f       = 10*scen;
   c(i,j) = f * d(i,j) / 1000;
   Solve transport using lp minimizing z;
   scen = scen + 0.5;
);
```

- Macros
- User defined function libraries

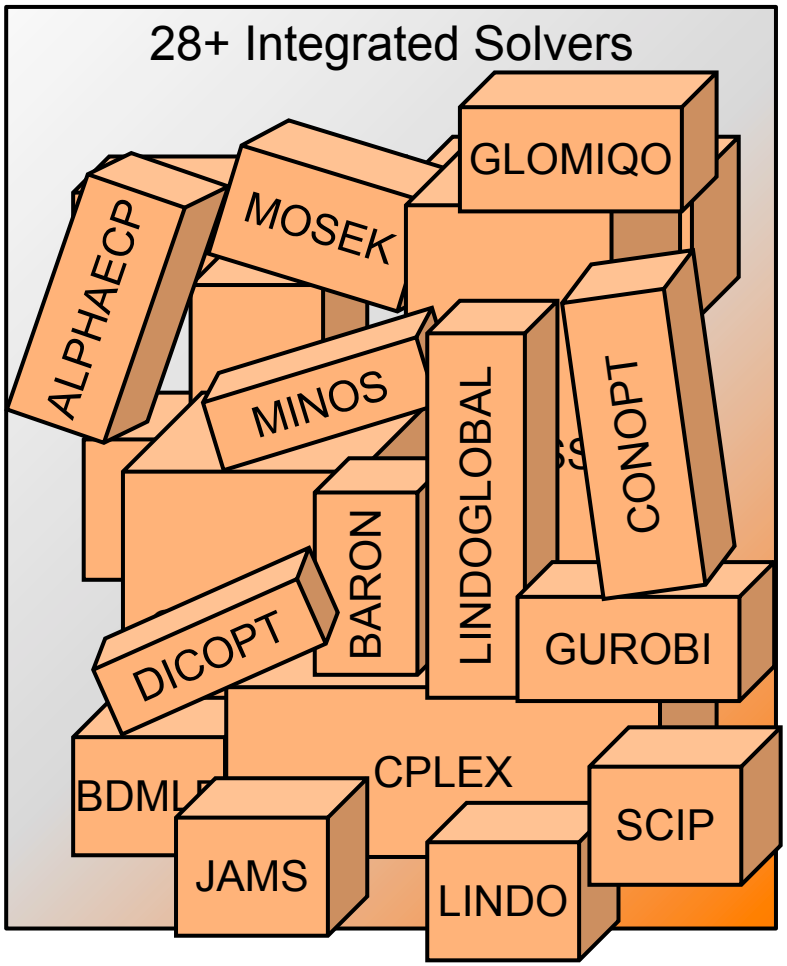
GAMS at a Glance

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



GAMS at a Glance

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



GAMS at a Glance

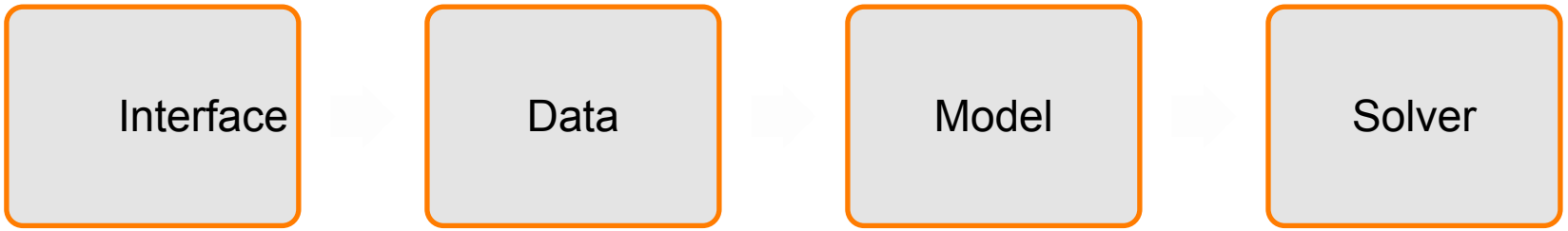
- Balanced mix of declarative and procedural elements
- Platform independence
- Hassle-free switch of solution methods
- **Open architecture and interfaces to other systems**
- Independent Layers

- ASCII
- Gams Data eXchange (Binary)
  - MS Excel, Access
  - Databases
  - ….
- API's
- Component Libraries
- .NET Integration

GAMS at a Glance

Independence of

- Model and data
- Model and solution methods (solver)
- Model and operating system
- Model and user interface



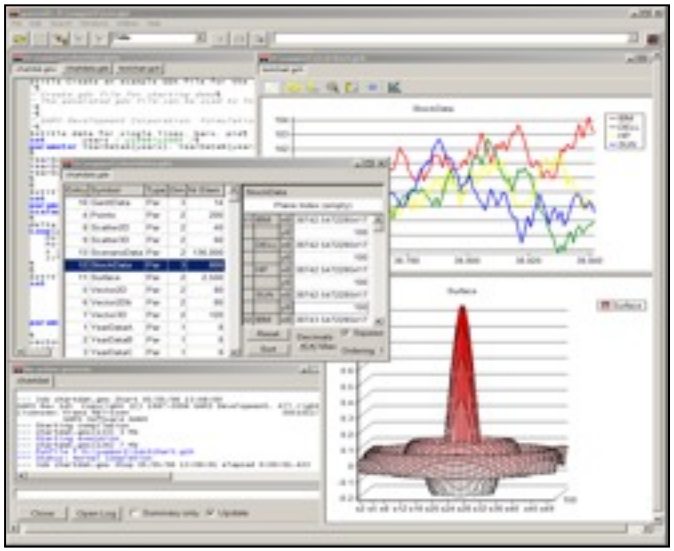
→Models benefit from

- Advancing hardware
- Enhanced / new solver technology
- Improved / upcoming interfaces to other systems

GAMS at a Glance: Development Environm.

GAMS IDE

- 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



Agenda

- What is new?
  - GAMS System
  - Platforms
  - Solvers
  - Interfaces
  - Stochastic Programming

What is new: GAMS System

- Support for user-defined
  - Macros
  - Function libraries
  - External equations
- Asynchronous execution
- Extended Mathematical Programming (EMP)
- More and further details: http://www.gams.com/docs/release/release.htm

What is new: Asynchronous Execution

- `$Call /Execute /put_utility 'Exec':`
  Start jobs from GAMS

- Three ways to start a job asynchronously,
  - at compile time (CT):
    - `$Call.ASync …`
  - at execution time (ET):
    - `Execute.Async '…';`
    - `put_utility 'Exec.ASync' / '…';`

What is new: Platforms

- Support for MAC OS X
- Cross- platform licenses
- Wine (Linux, Mac)

What is new: Solvers

- GloMIQO: Branch-and-bound global optimization for mixed-integer quadratic models
- Lindo: Global and stochastic optimization
- Gather-Update-Solve-Scatter
- *(Stochastic) EMP*

What is new: Gather-Update-Solve-Scatter

Setting	Solve time (secs)
Solvelink=0 (default)	40.297
Solvelink=%Solvelink.LoadLibrary%	03.625
GUSS	00.797

- Updates model data instead of matrix coefficients/rhs
- Hot start (keeps the model hot inside the solver and uses solver's best update mechanism)
- Saves model generation and solver setup time
- Model unchanged from scenario to scenario
- A priori knowledge of all scenario data

What is new: Solving Data Related Models

The common way:

```
Set s / s1*s10 /
Parameter
     A_s(s,i,j) Scenario data
     xlo_s(s,j) Scenario lower bound for variable x
     xl_s(s,j)  Scenario solution for x.l
     em_s(s,i)  Scenario solution for e.m;
Loop(s,
  A(i,j) = A_s(s,i,j);
  x.lo(j)= xlo_s(s,j);
  solve mymodel min z using lp;
  xl_s(s,j) = x.l(j);
  em_s(s,i) = e.m(i);
);
```

- GAMS generates model and writes it to hd
- GAMS writes database to scratch files on hd
- GAMS calls solver and vacates memory
- After solver is done: GAMS restarts and swaps database

What is new: Solving Data Related Models

The new way:

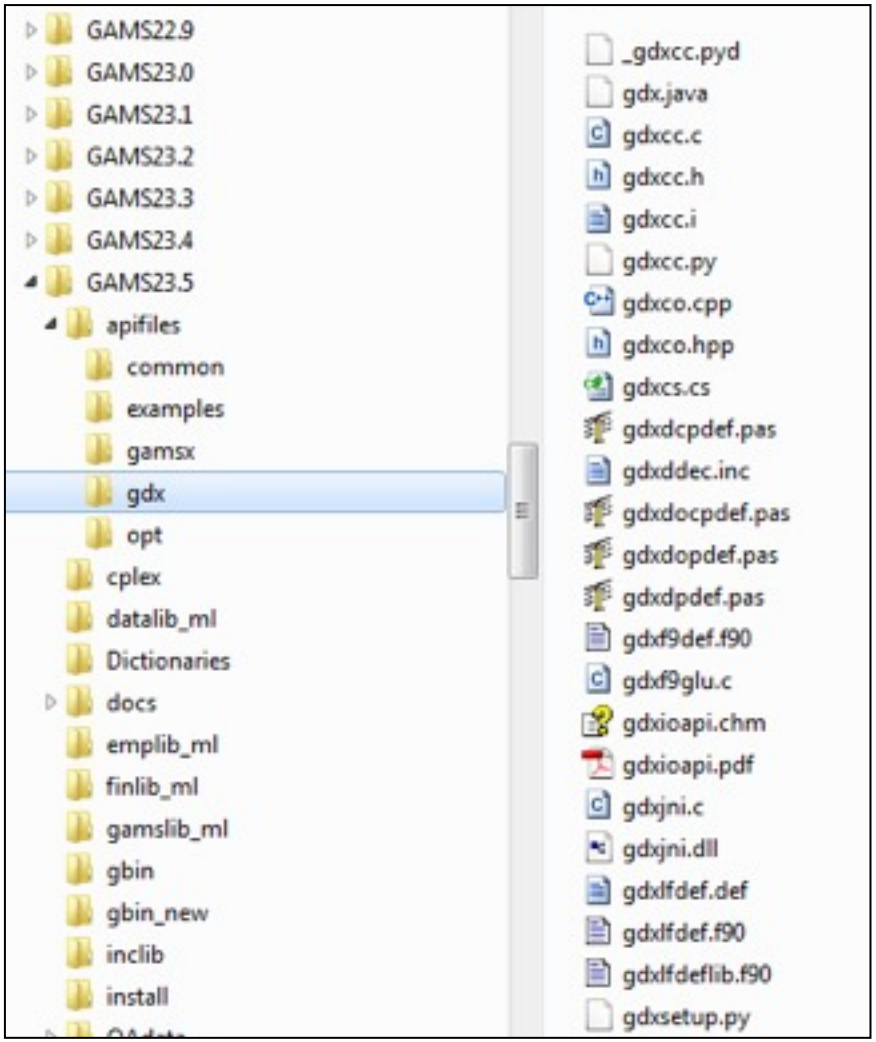
```
Set s / s1*s10 /
Parameter
     A_s(s,i,j) Scenario data
     xlo_s(s,j) Scenario lower bound for variable x
     xl_s(s,j)  Scenario solution for x.l
     em_s(s,i)  Scenario solution for e.m;
Set dict / s.   scenario. ''
           A.   param.    A_s
           x.   lower.    xlo_s
           x.   level.    xl_s
           e.   marginal. em_s   /;
solve mymodel min z using lp scenario dict;
```

What is new: Interfaces

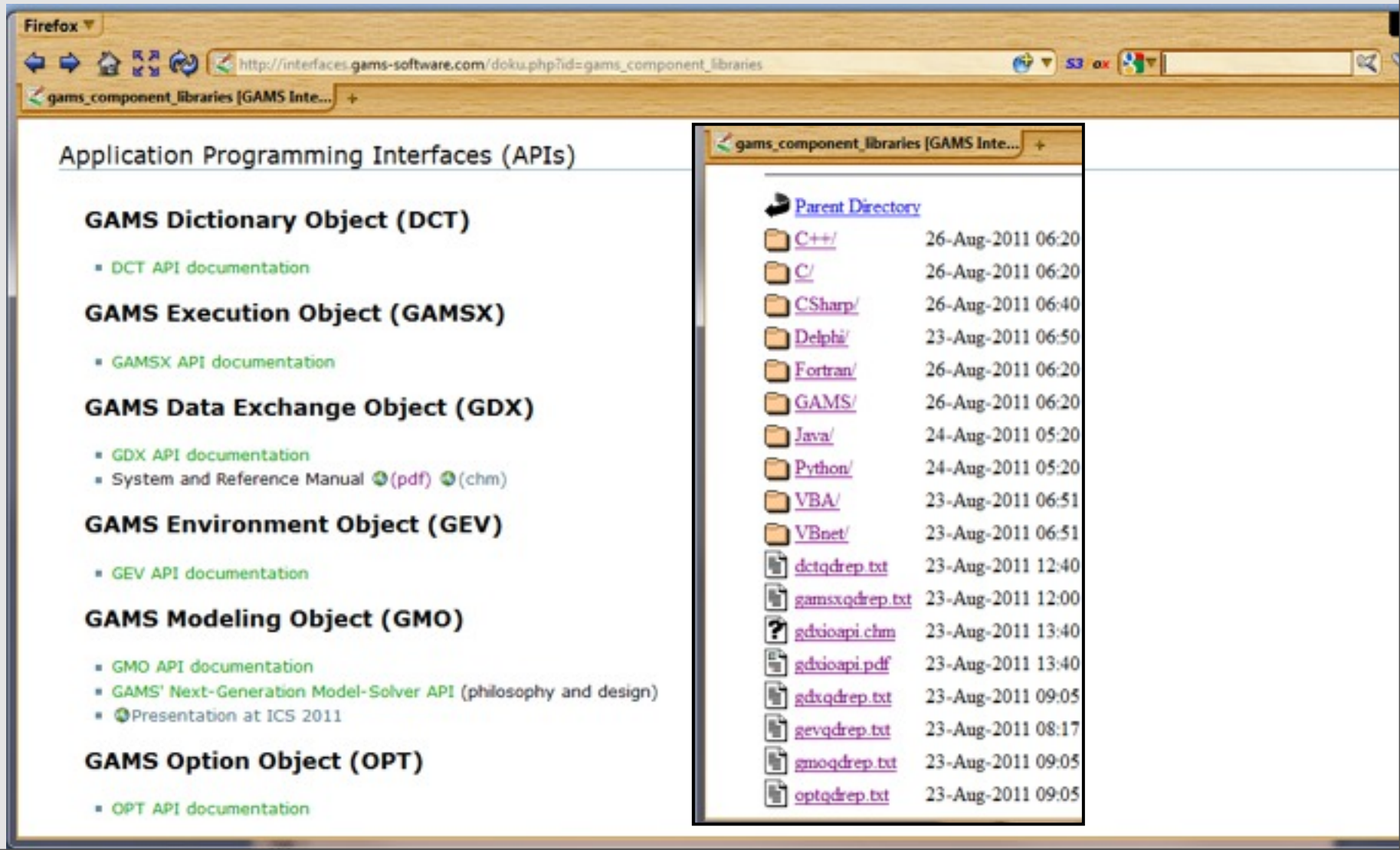
- API's for various programming languages (C, Fortran, Delphi)
- Component libraries
- Better integration into Python

What is new: Distributed GAMS APIs

- Component Libraries
  - GAMS
  - GDX
  - Option
- Supported languages
  - C, C++, C#
  - Delphi
  - Fortran
  - Java
  - VBA, VB.Net
  - Python
- Examples/Documentation



What is new: GAMS Component Libraries



What is new: Calling GAMS from Python

```
if __name__ == "__main__":

    numberParams = len(sys.argv)
    if numberParams != 2 :
        print "Usage:", sys.argv[0], "sysDir"
        os._exit(1)

    gdxHandle   = new_gdxHandle_tp()
    optHandle   = new_optHandle_tp()
    gamsxHandle = new_gamsxHandle_tp()

    sysDir = sys.argv[1]
    print sys.argv[0], "using GAMS system directory:", sys.argv[1]

    assert gamsxCreateD(gamsxHandle, sysDir, GMS_SSSIZE)[0]
    assert gdxCreateD  (gdxHandle,   sysDir, GMS_SSSIZE)[0]
    assert optCreateD  (optHandle,   sysDir, GMS_SSSIZE)[0]
```

```
from gdxcc import *
from gamsxcc import *
from optcc import *
import sys
import os
```

```
    status = writeModelData(gdxHandle, "demanddata.gdx")
    if not status:
        print("Model data not written")
        terminate(gdxHandle, gamsxHandle, optHandle)
```

Creating Input for GAMS Model

```
    status = callGams(gamsxHandle, optHandle, sysDir)
    if not status:
        print("Call to GAMS failed")
        terminate(gdxHandle, gamsxHandle, optHandle)
```

Callout to GAMS

```
    status = readSolutionData(gdxHandle, "results.gdx")
    if not status:
        print("Could not read solution back")
```

Reading Solution from GAMS Model

```
    terminate(gdxHandle, gamsxHandle, optHandle)
```

What is new: .Net Integration

Transport8.cs | Transport7.cs | Transport6.cs | Transport5.cs | Transport4.cs | Transport3.cs | Transport2.cs | Transport1.cs*

TransportSeq.Transport1 — Main(string[] args)

```
using System;
using System.Collections.Generic;
using System.Text;
using GAMS;

namespace TransportSeq
{
    class Transport1
    {
        static void Main(string[] args)
        {
            GAMSWorkspace ws = new GAMSWorkspace();
            ws.GamsLib("trnsport");
            using (GAMSJob t1 = ws.AddJobFromFile("trnsport.gms"))
            {
                t1.Run();
                Console.WriteLine("Ran with Default:");
                foreach (GAMSVariableRecord rec in t1.OutDB.GetVariable("x"))
                    Console.WriteLine("x(" + rec.Keys[0] + "," + rec.Keys[1] + "): level=" + rec.Level + " marginal=" + rec.Marginal);

                using (GAMSOptions opt = ws.AddOptions())
                {
                    opt.AllModelTypes = "xpress";
                    t1.Run(opt);
                }
                Console.WriteLine("Run with XPRESS:");
                foreach (GAMSVariableRecord rec in t1.OutDB.GetVariable("x"))
                    Console.WriteLine("x(" + rec.Keys[0] + "," + rec.Keys[1] + "): level=" + rec.Level + " marginal=" + rec.Marginal);

            }
        }
    }
}
```

Agenda

- What is new?
  - GAMS System
  - Platforms
  - Solvers
  - Interfaces
  - Stochastic Programming

Stochastic Programming Claims and '*Facts*'

- Lots of application areas (Finance, Energy, Telecommunication)
- Mature field (Dantzig '55)
- Variety of SP problem classes with specialized solution algorithms (e.g. Bender's Decomposition)

- Small fraction compared to deterministic mathematical programming (NEOS)
- Few commercially supported solvers for SP
- Various frustrations with industrial SP projects

Extended Mathematical Programming

- Embedded Complementarity Systems
- Disjunctive Programs
- Bilevel Programs
- Extended Nonlinear Programs
- Stochastic Programming
- …

- ➢ Breakouts of traditional MP classes
- ➢ No conventional syntax
- ➢ Limited support with common model representation
- ➢ Incomplete/experimental solution approaches
- ➢ Lack of reliable/any software

Thank You !

USA

GAMS Development Corp.
1217 Potomac Street, NW
Washington, DC 20007
USA

Phone: +1 202 342 0180
Fax: +1 202 342 0181

http://www.gams.com
sales@gams.com
support@gams.com

Europe

GAMS Software GmbH
Eupener Str. 135-137
50933 Cologne
Germany

Phone: +49 221 949 9170
Fax: +49 221 949 9171

http://www.gams.com
info@gams.de
support@gams.com