

GAMS



GAMS – Features you might not know about

Alex Meeraus

ameeraus@gams.com

Michael Bussieck

mbussieck@gams.com

Jan-Hendrik Jagla

jhjagla@gams.com

Franz Nelissen

fnelissen@gams.com

Lutz Westermann

lwestermann@gams.com

GAMS Software GmbH

www.gams.de

GAMS Development Corp.

www.gams.com



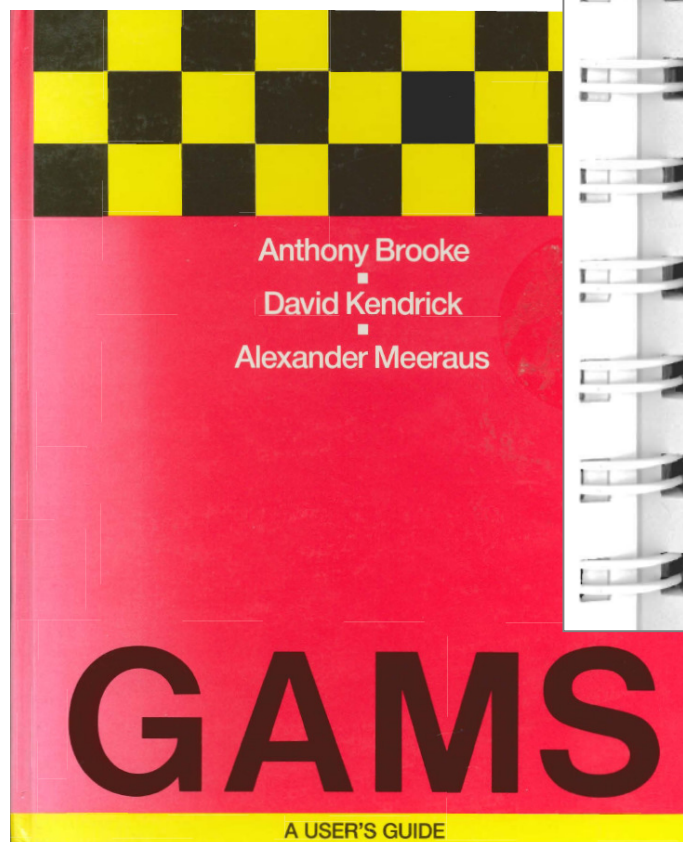
GAMS



Bonn, July 6, 2009



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

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

^aMeasured in minutes.

^bThe problem is too big for MINOS. ZOOM was used instead.

^cA nonlinear problem. 63% of the non-zeroes are nonlinear.

^dA nonlinear problem. 58% of the non-zeroes are nonlinear.

^eA mixed binary problem, with 55 binary variables (solved with a relative termination criterion of 10%).

^fA linear problem, solved using XMP which is contained within ZOOM.

GAMS Users Guide (1988)



... and now

	Type	s in 1988	s in 2008	Improvement Factor
camcge	NLP	468	0.031	15097
dinamico	LP	1986	0.125	15888
egypt*	LP	1758	0.015	117200
fertd*	MIP	2382	0.062	38419
ganges	NLP	546	0.109	5009
sarf	LP	9210	0.139	66259
yemcem*	MIP	510	0.140	3643

* 1988 solver ZOOM, 2008 solver CPLEX 11.0.1



Improvements on all Frontiers

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

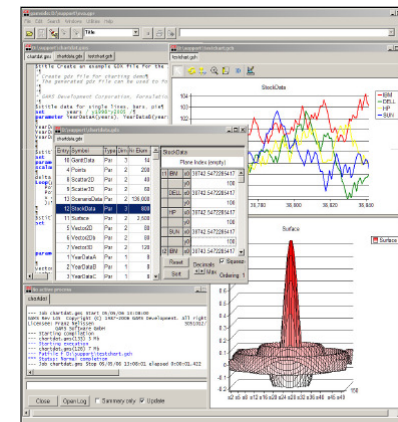
- **Productivity Tools**
 - Databases, spreadsheets
 - Specialized visualization tools
 - IDE improvements
 - Grid computing

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



The GAMS/BASE Module

- Compiler and Execution System
- Model libraries + Documentation
- GAMS IDE (Windows)
- GDX Utilities
- Solvers:
 - Convert (convert model to different formats)
 - BENCH, EMP, EXAMINER, GAMSCHK, LOGMIP, and NLPEC
 - BDMLP, LS, and MILES
 - COIN-OR
 - Cbc, IpOpt, BonMin, Couenne
 - Glpk, Scip (academic only)





Distributed Model Libraries

- **GAMS Model Library**

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

Model Libraries	Help
GAMS Model Library	
GAMS Test Library	
GAMS Data Utilities Models	
GAMS EMP Library	
Practical Financial Optimization Models	

- **GAMS Test Library**

- Transparent and reproducible Quality Assurance
- Test for
 - Solver correctness
 - Special functions
 - GAMS utilities



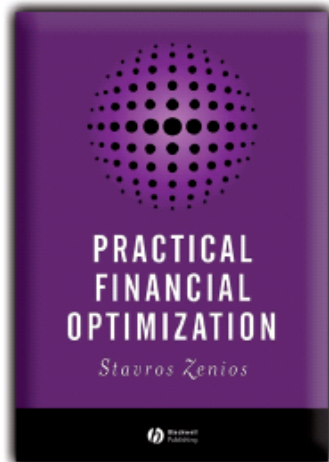
Distributed Model Libraries

- **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



GAMS/Gurobi MIP Solver



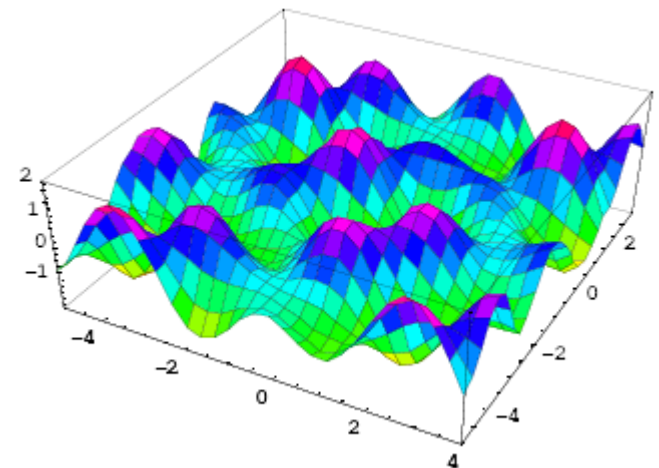
- A new MIP Solver with a Pedigree:
Zonghao Gu, Edward Rothberg, and Robert Bixby
former members of CPLEX' R&D team
- The Gurobi MIP solver includes shared memory parallelism, capable of simultaneously exploiting any number of processors and cores per processor. The implementation is deterministic.
- No extra cost for parallel option
- MIP optimality and feasibility benchmarks from H.Mittelmann are available





Global Optimization of Non-Convex MINLPs

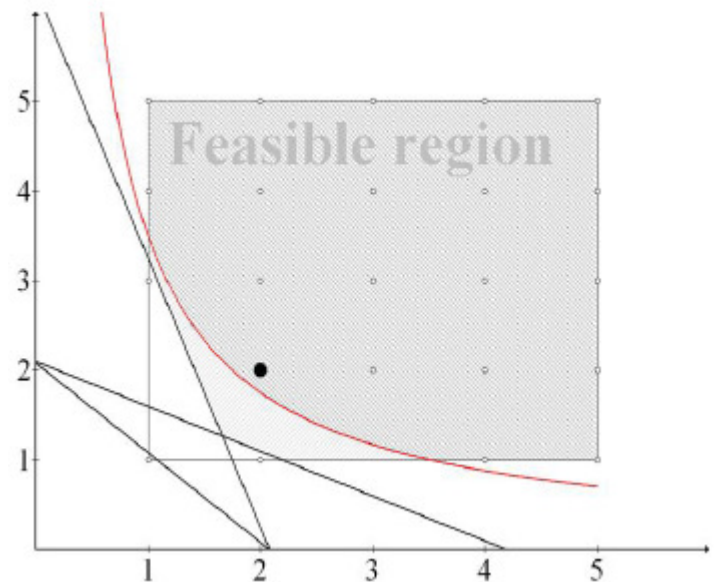
- Non-linear, non-convex optimization problems
 - Multiple local optima
 - Disconnected feasible region
- Global Optimization Solvers
 - Deterministic global solution/bounds
 - Relative/absolute gap similar to MIP
 - Multi-start/Search algorithms
- Baron, CoinCouenne, LindoGlobal
- LGO, MSNLP, OQNLP





AlphaECP MINLP Solver

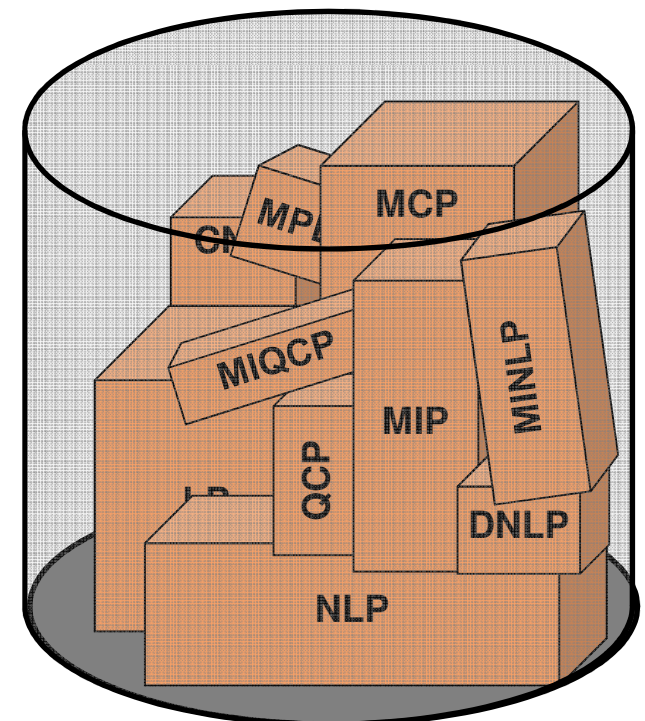
- Based on Kelley's method for convex NLP (1960)
- Developed by researchers at Åbo University, Finland
- Works with all GAMS/MIP solvers
- No NLP solver required (few NL evaluations)
- Works well on large scale non-convex MIQCP problems





New solution concepts

- Extended Nonlinear Programs
 - Embedded Complementarity Systems
 - Bilevel Programs
 - Disjunctive Programs
 - ...
- Do not fit into traditional MP classes
 - Limited support with common model representation
 - Incomplete/experimental solution approaches
 - Lack of reliable/any software
- ➔ Translation service





Extended Mathematical Programming - EMP

- Uses existing language features to specify additional model features
 - Expresses extended model in symbolic form and passes it to existing solution methods via embedded GAMS calls
 - Reads solution back into original space
 - Facilitates to write out the reformulated model
-
- Automates error-prone and time-consuming manual algebra (re)writing
 - Offers solutions through established and powerful solution engines
 - Provides nonstandard model information to solver developers → new algorithms/software?



Logical Mixed Integer Programming (LogMIP)

- Developed by Aldo Vecchietti and Ignacio E. Grossmann

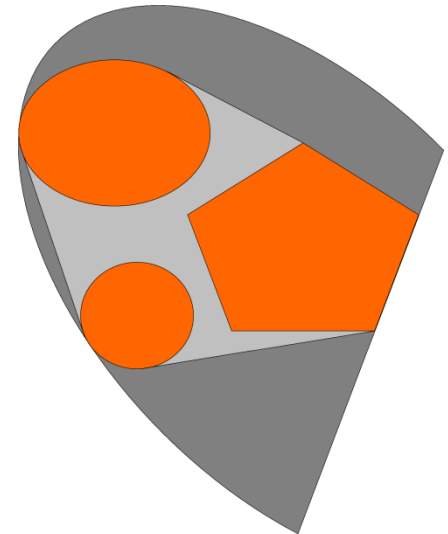
<http://www.logmip.ceride.gov.ar/>

- Add-on modeling framework to formulate models with disjunction and logic propositions
- Language Compiler
- Solution algorithms and techniques for solving linear and nonlinear disjunctive programming problems
- Add-on tool as of GAMS Distribution 21.7 (April 2005)



LogMIP “Solver”

- Reformulation
 - Convex Hull (only linear)
 - Big M (only linear)
 - Require MIP solver
- Logic based Method
 - Logic-based Outer Approximation
 - Solves a series of NLP and MIP sub-problems
 - Requires NLP and MIP solver





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 ~30 open-source projects
 - Solvers
 - Interfaces
 - Tools
- An active OR community
 - Mailing lists
 - Google group
 - Wikis



The Coin-OR / GAMSLinks Project

<https://projects.coin-or.org/GAMSlinks>

Stefan Vigerske (Humboldt-University Berlin)



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**)
 - state of the art LP and MIP solver from J. Forrest
- Gnu Linear Programming Kit (**GLPK**)
 - LP and MIP solver from A. Makhorin
- Interior Point Optimizer (**IPOPT**)
 - large scale NLP solver from A. Wächter
- Solving Constraint Integer Programs (**SCIP**)
 - LP/MIP solver developed at Zuse Institute Berlin (ZIB)





The Coin-OR / GAMSLinks Project

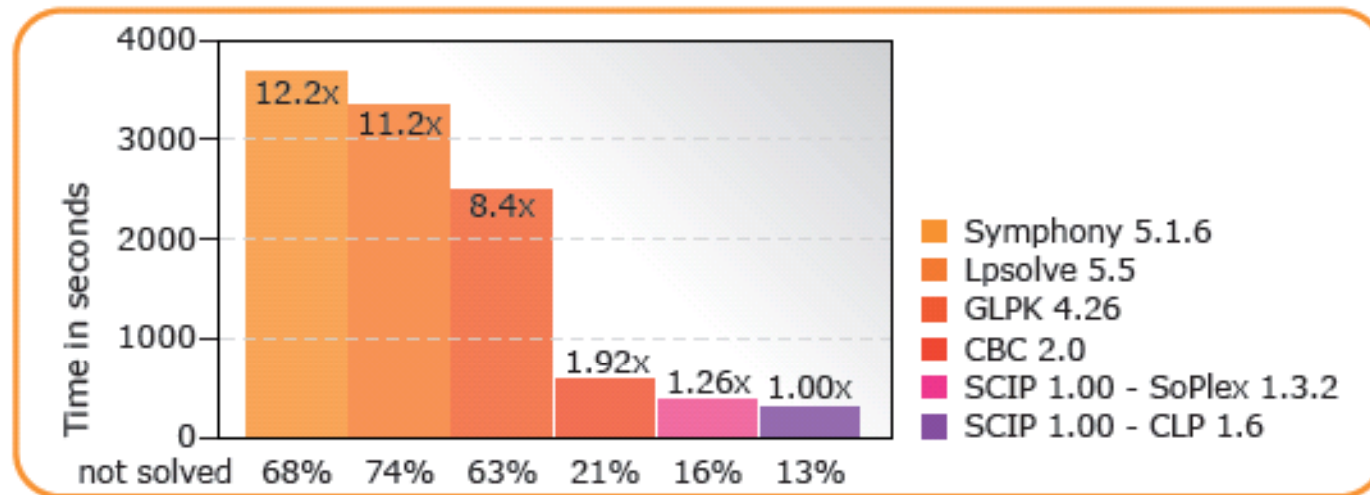
GAMS interfaces to open-source Solvers

- 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
- Lagrangian Global Optimizer (**LaGO**)
 - Convexification and Branch and Cut based MINLP solver from I. Nowak and S. Vigerske





The Coin-OR / GAMSLinks Project



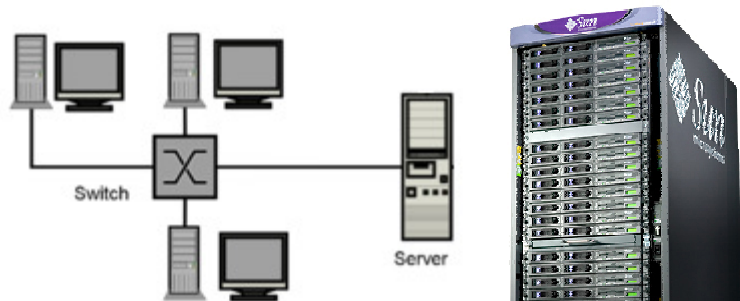
Performance Benchmark of MIP codes free for academic use by H. Mittelmann. Solution times are geometric means where unsolved instances were assigned a 2 hours solution time (time limit). Details at scip.zib.de

GAMS QA and testing supports maturing of COIN-OR solvers!

Coin-OR solvers enable GAMS to offer dependable free solvers!

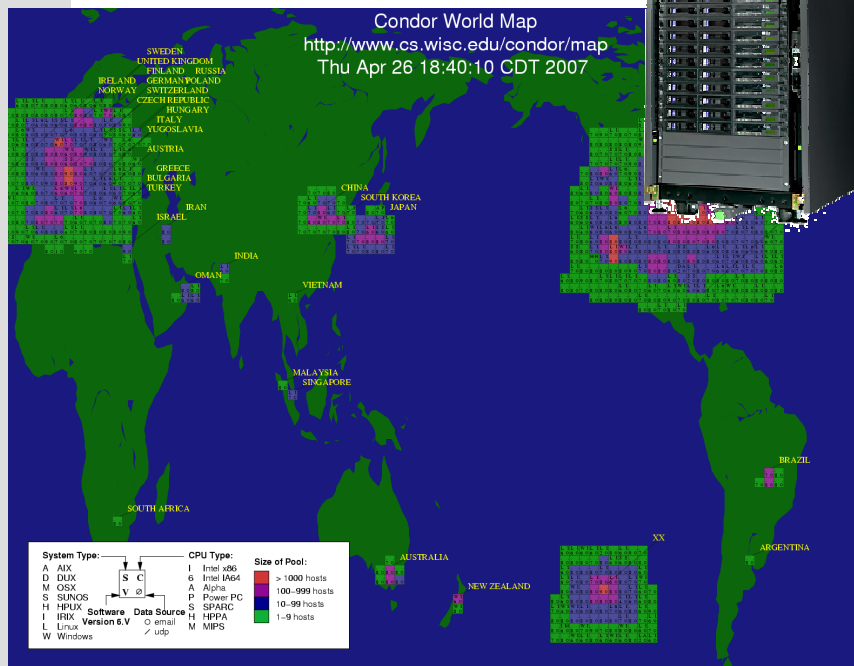


What is Grid Computing?



A pool of connected computers managed and available as a common computing resource

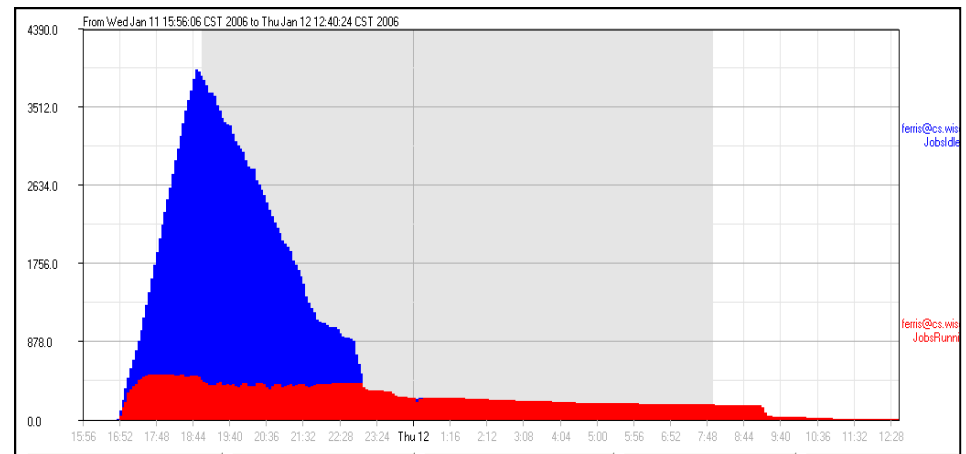
- Effective sharing of CPU power
- Massive parallel task execution
- Scheduler handles management tasks
- E.g. Condor, Sun N6 Grid Engine, Globus
- Can be rented or owned in common





GAMS & Grid Computing

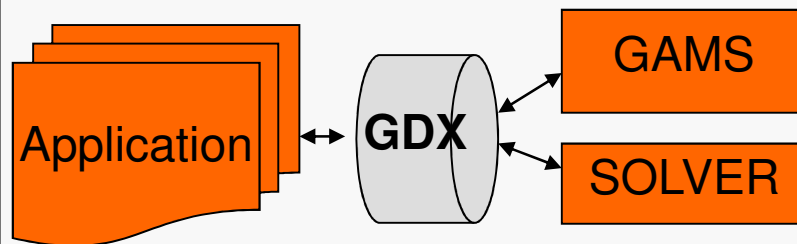
- **Scalable:**
 - support of massive grids, **but also**
 - multi-cpu / multiple cores desktop machines
 - “1 CPU - Grid”
- Platform **independent**
- Only **minor changes** to model required
- **Separation** of model and solution method
→ Model stays **maintainable**





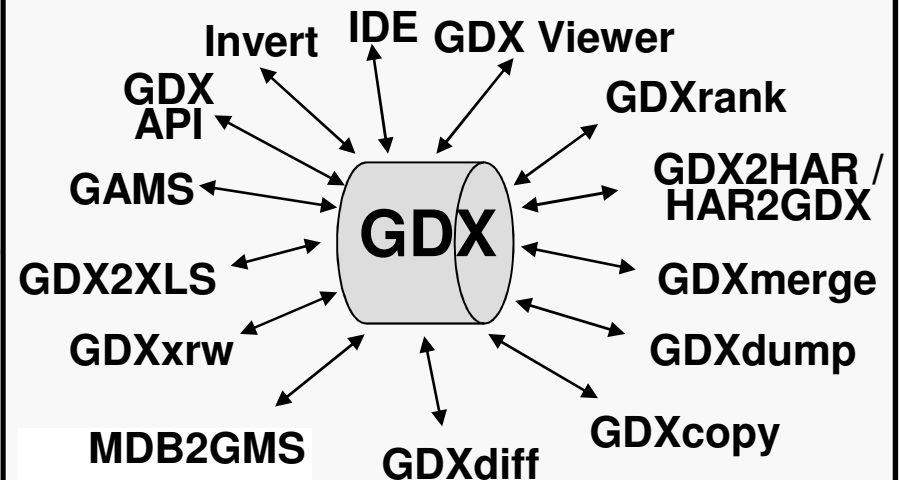
Gams Data eXchange

Binary Data Exchange



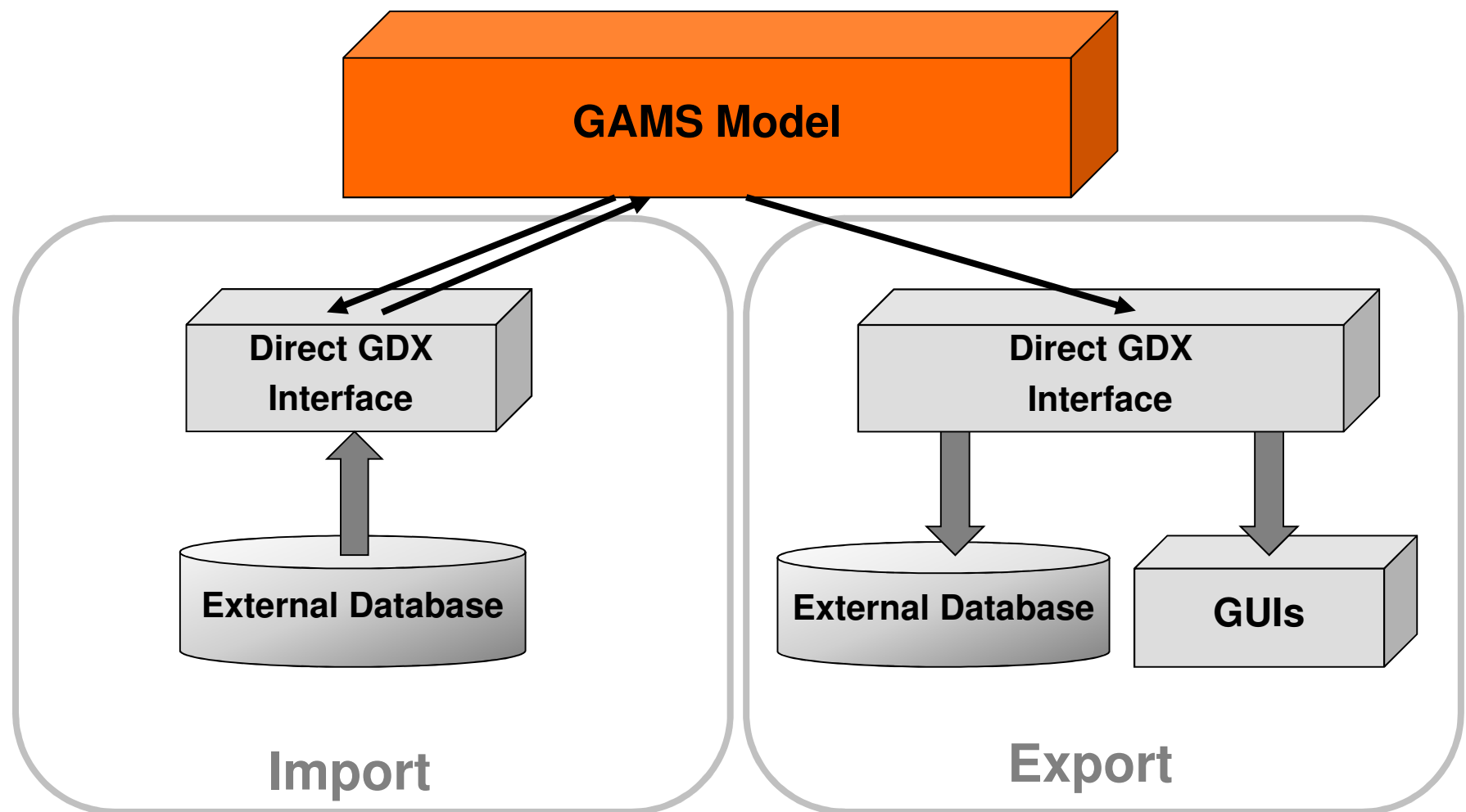
- Fast exchange of data
- Syntactical check on data before model starts
- Data Exchange at any stage (Compile and Run-time)
- Platform Independent
- Direct GDX interfaces and general API
- Scenario Management Support
- Full Support of Batch Runs

GDX Tools



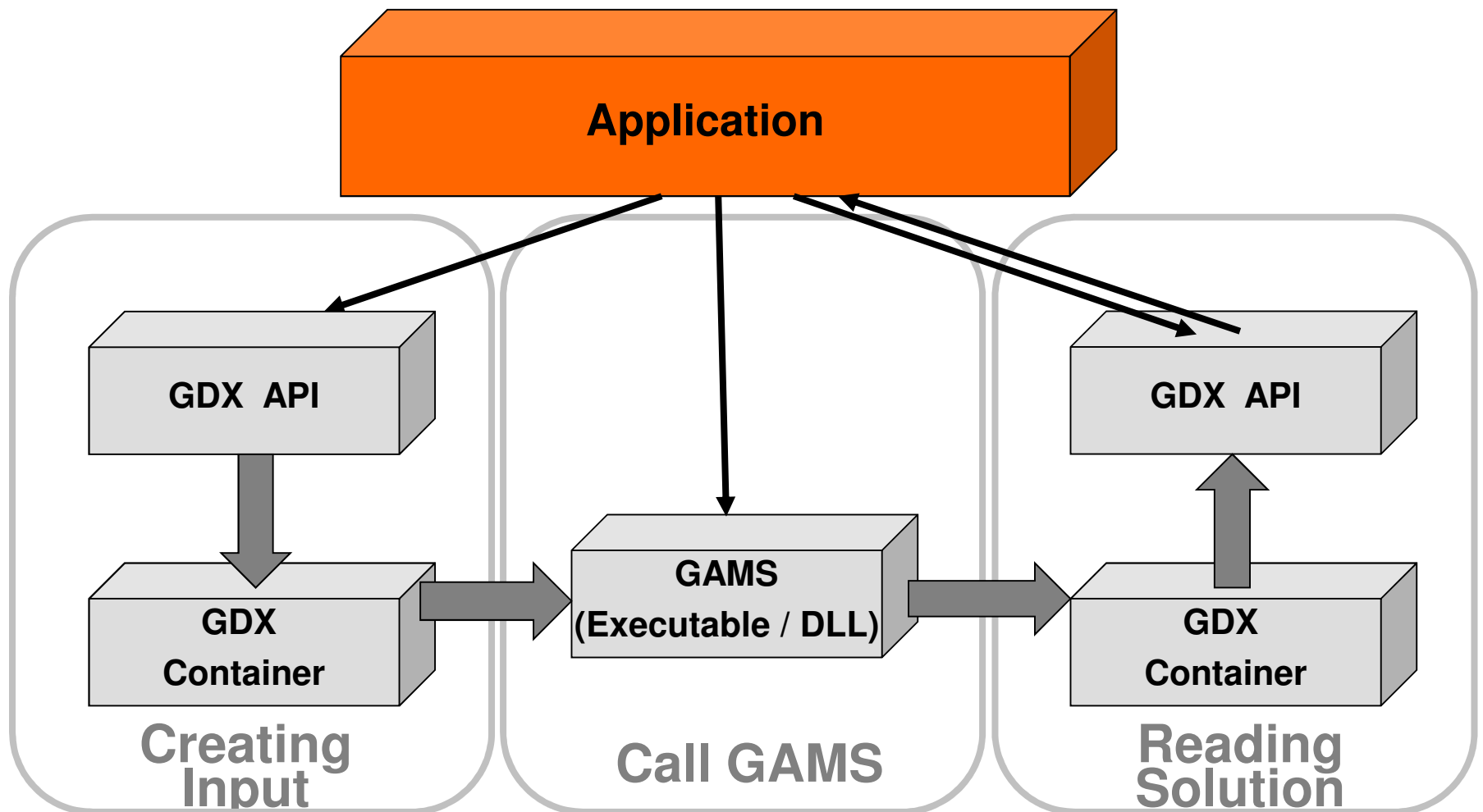


GAMS in Control





Application in Control





Calling GAMS from your Application

Creating Input for GAMS Model

Callout to GAMS

Reading Solution from GAMS Model

- Data handling using GDX API

```
PGDX.gdxDataWriteStrStart('Demand', 'Demand data', 1, gms_dt_par, 0)
Values[vallevel] := 324.0; Indx[1] := 'New-York';
PGDX.gdxDataWriteStr(Indx, Values);
```

- GAMS option settings using Option API

```
POPT.Input := fnModel;
POPT.LogOption := 2;
```

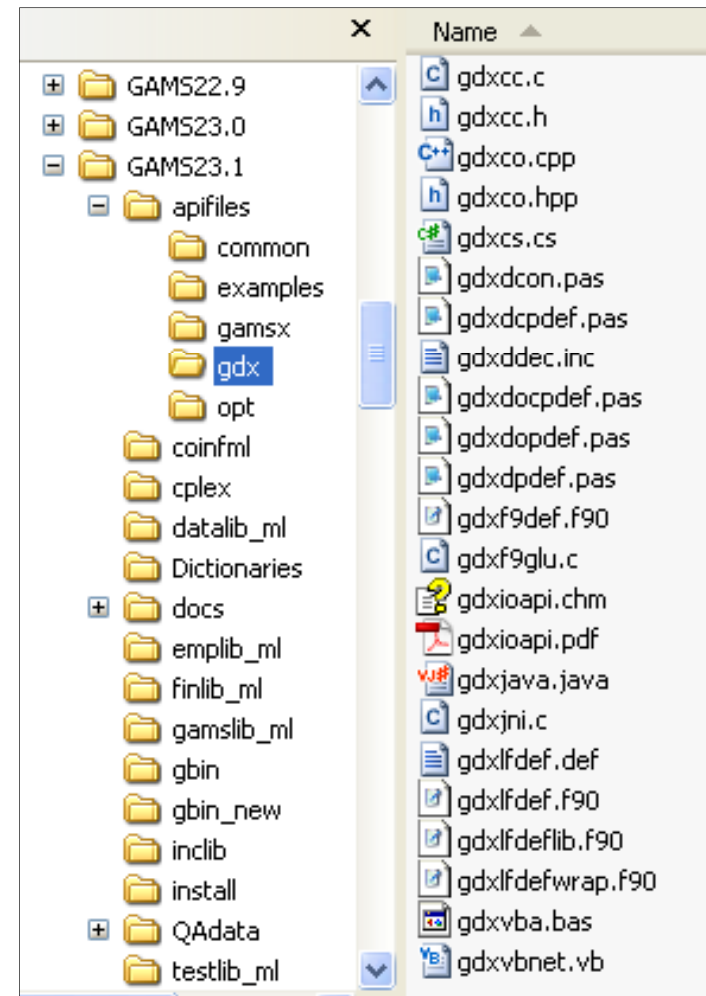
- Starting GAMS using GAMS API

```
ErrNr := PGMS.RunExec(Msg, 1);
```



Distributed APIs

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





Solver Integration

```
solve mymodel using minlp min obj;  
mymodel.solveLink = {ChainScript, CallScript,  
    CallModule, AsyncGrid, AsyncSimulate, LoadLibrary};
```

- ChainScript: Solver process, GAMS vacates memory
 - + Maximum memory available to solver
 - + protection against solver failure (*hostile* link)
 - swap to disk
- Call{Script/Module}: Solver process, GAMS stays live
 - + protection against solver failure (*hostile* link)
 - + no swap of GAMS database
 - file based model communication



Solver Integration – cont.

- LoadLibrary: Solver DLL in GAMS process
 - + fast memory based model communication
 - + update of model object inside the solver (hot start)
 - not (yet) supported by all solvers
- trnsport.gms (LP) solved 500 times with CPLEX:

```
set ss /s1*s500/; loop(ss,
    solve transport minimizing z using lp);
```

- | | | |
|----------------|------------------|--|
| – ChainScript: | 33.04 s (28.9s)* | |
| – CallModule: | 13.78 s (12.7s) | |
| – LoadLibrary: | 2.37 s (2.0s) | |
| – Hot Start: | 0.37 s (0.4s) | |

Cplex simplex
time: 0.2 s

* without Virus Scanner



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/;
```

→ $t1.d1.h1, t1.d1.h2, \dots,$
 $t25.d2.h1, t26.d2.h2, \dots$



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);
```

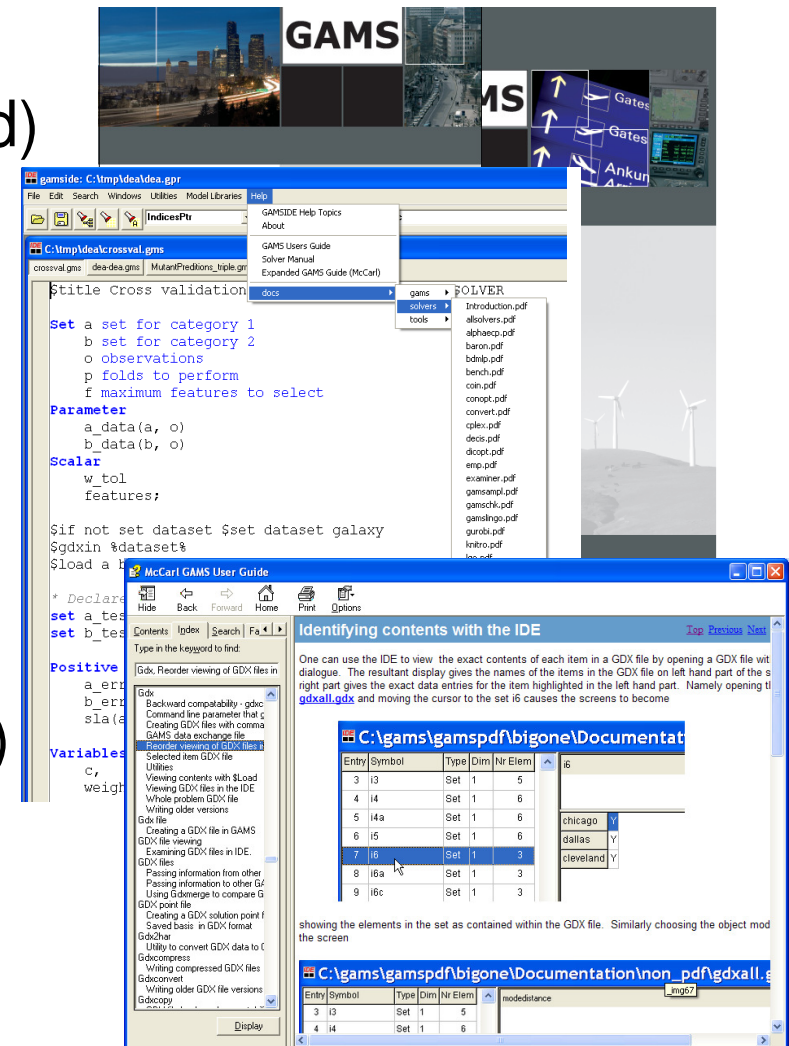
```
➔ ijk:  i1.j3.k1, i1.j3.k2, ...,  
        i2.j4.k1, i2.j4.k2, ...,  
        i2.j4.k5
```

```
➔ x:    i1.j3.k1.c1, i1.j3.k2.c2, ...,  
        i2.j4.k1.c5, i2.j4.k2.c6, ...,  
        i2.j4.k5.c10
```



Documentation and Information Sources

- *A User's Guide* and *The Solver Manual* (pdf and printed)
- Expanded Users Guide (McCarl) (pdf and chm)
- GAMS Web sites and Support wiki
- **support/info@gams.com**
- User forum
 - GAMS-L (email list, 1000+ members)
 - gamsworld (Google group, 300+ members)





Semidefinite Programming

- (New) class of efficiently solvable MP problems
- Solution Technology versus Modeling Paradigm similar to SOCP
- For example, used in convex relaxation of 0/1 MIQCP, requires non-trivial reformulations
- Experimental GAMS Interface
 - Problem exchange via SDPA sparse format (not via *solve*)
 - SDP problems solved with COIN-OR's **csdp**
 - GAMS used as a *matrix generator*
 - Examples in GAMS Model Library
 - SDP Convexifications of the Generalized Quadratic Assignment Problem (**gqapsdp**)
 - Goemans/Williamson Randomized Approximation Algorithm for MaxCut (**maxcut**)
 - Solving the Transportation LP Problem using SDP (**trnsspd**)



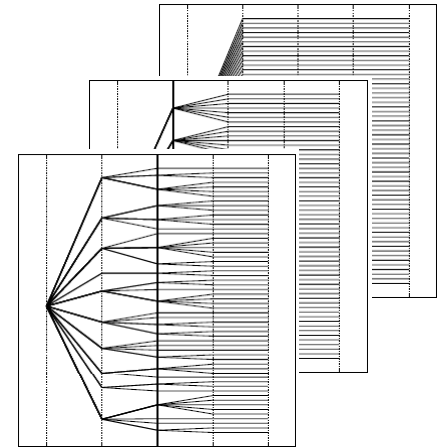
Matrix Utilities

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



SCENRED/SCENRED2

- SCENRED provides a collection of software routines dealing with scenario tree manipulation algorithms in stochastic programming (SCENario REDuction)
- New in SCENRED2
 - Tree construction: SCENRED2 can create trees from collections of independent scenarios
 - Visualization. SCENRED2 contains new options to create input files for GNUPLOT
 - Improved metrics. Tree reduction can now be carried out w.r.t. the Fortet-Mourier metric, instead of the upper bounds given by the Monge-Kantorovich metric.





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(x1,x2);
```

➔ `z = x1/x2;`

- 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(i)*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))
```

➔ `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')
```

➔ `display "here it is", i, k;`

```
$macro dd(q) &&q)
z=dd('sum(j, a1(j)');
```

➔ `z=sum(j, a1(j));`



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*

		l	m	
seattle.	new-york	50		
seattle.	Chicago	300		
seattle.	topeka		0.36	
san-diego.	new-york	275		
san-diego.	Topeka	275		
san-diego.	chicago		0.009	;



Contacting GAMS

Europe

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

Phone: +49 221 949 9170

Fax: +49 221 949 9171

<http://www.gams.de>

info@gams.de

support@gams-software.com

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