An Introduction

Lutz Westermann, GAMS Software GmbH

GAMS

Company Background

Roots: World Bank, 1976

Went commercial in 1987

GAMS Development Corporation (Washington)

GAMS Software GmbH (Cologne, Braunschweig) 1996

Tool Provider: **G**eneral **A**lgebraic **M**odeling **S**ystem

GAMS

Agenda

What is GAMS?

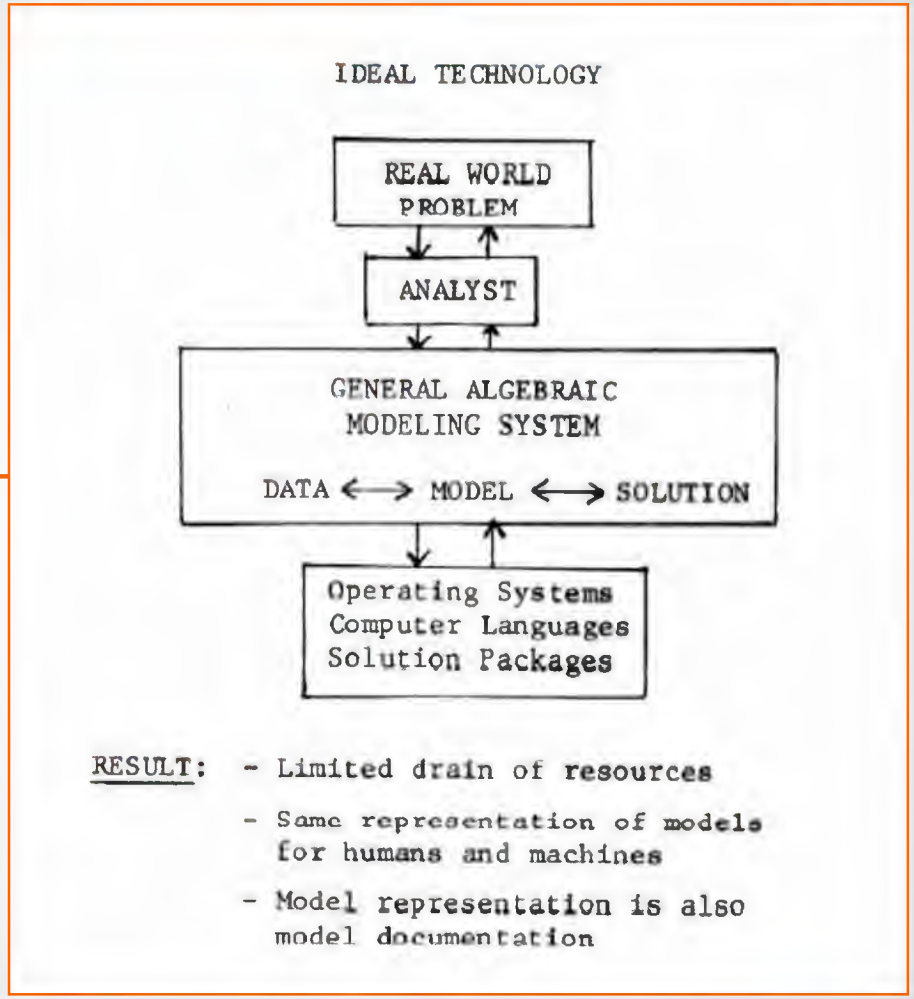
- Algebraic Modeling Languages – A Success Story
- GAMS – Highlights and Design Principles
- Striving for Innovation and Compatibility
  - New Modeling and Solution Concepts
  - Software Quality Assurance

A Simple Example

GAMS

1976 - A World Bank Slide

The Vision



GAMS

2012 INFORMS Impact Prize

36 Years later

Originators of Algebraic Modeling Languages

AIMMS 3 | AMPL | MAXIMAL SOFTWARE | GAMS | LINDO SYSTEMS

GAMS

Algebraic Modeling Languages (AML)

1

- High-level **computer programming languages**
  - Formulation of **mathematical optimization problems**
  - **Notation similar to algebraic notation**

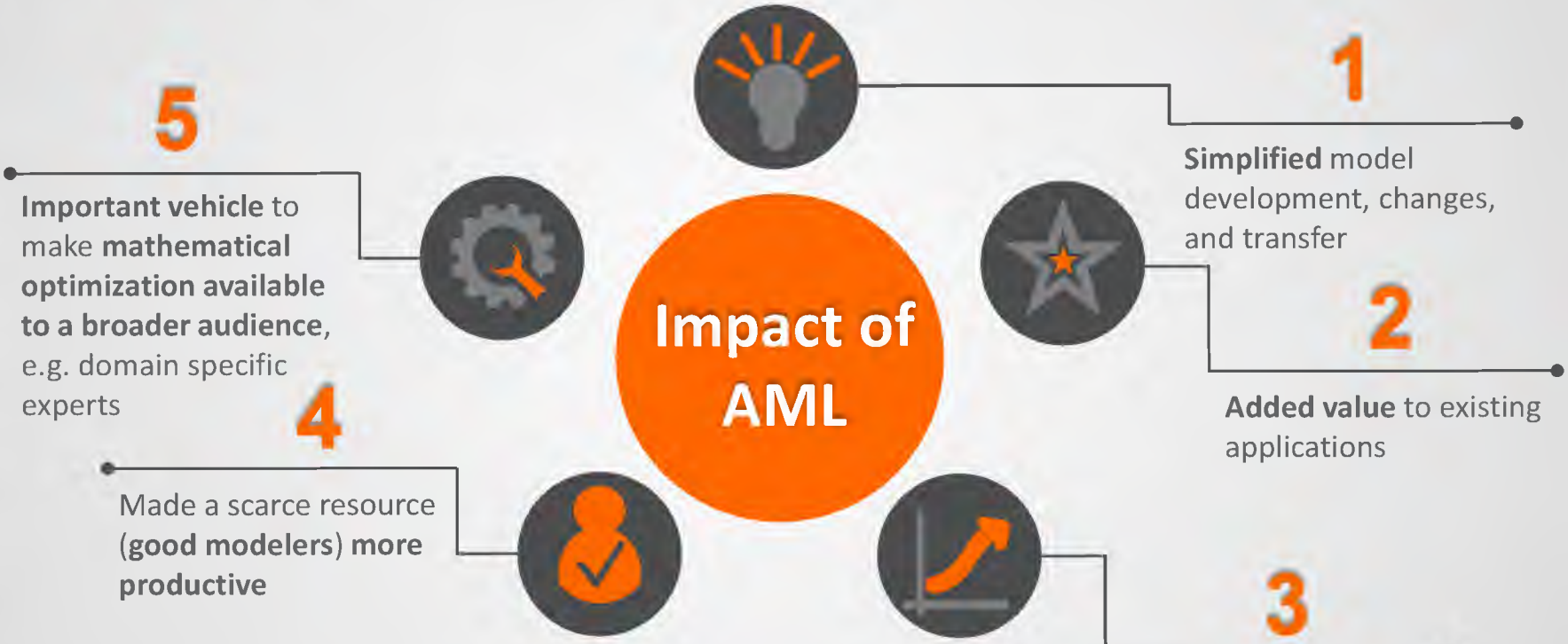
2

- **Do not solve problems directly**, but offer links to state-of-the-art algorithms ("solver-links")

Source: http://en.wikipedia.org/wiki/Algebraic_modeling_language

GAMS

Impact of Algebraic Modeling Languages



What does a modeler have to think about?

1. Problem
2. Mathematics
3. Programming
4. Performance
5. Scalability
6. Connectivity
7. Deployment
8. Maintenance (Life Cycle)
9. ...

Why is GAMS a tool for him?

GAMS

Broad User Community and Network

GAMS used in more than 120 countries

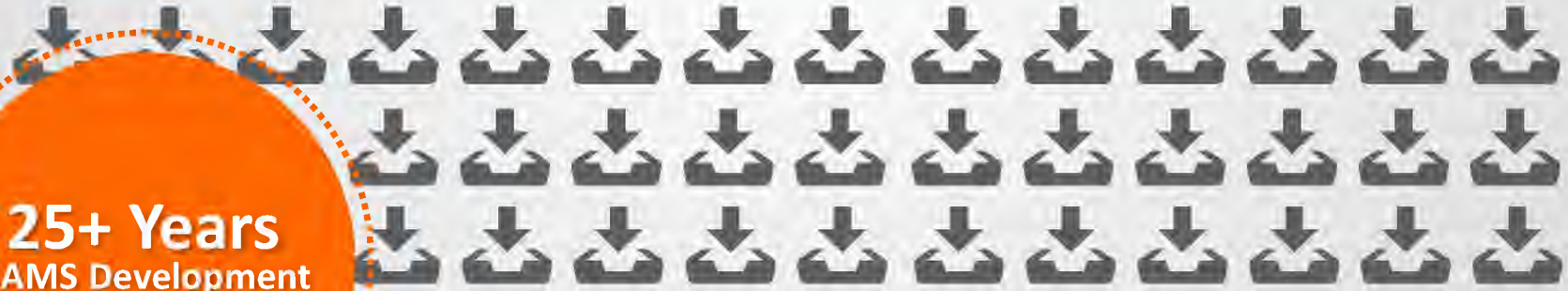
25+ Years
GAMS Development

GAMS

Broad User Community and Network

More than 10,000 licenses

50% academic users, 50% commercial users



25+ Years
GAMS Development

6,000+ monthly downloads of the free system

GAMS

Broad Range of Application Areas

Agricultural Economics	Applied General Equilibrium
Chemical Engineering	Economic Development
Econometrics	Energy
Environmental Economics	Engineering
Finance	Forestry
International Trade	Logistics
Macro Economics	Military
Management Science/OR	Mathematics
Micro Economics	Physics

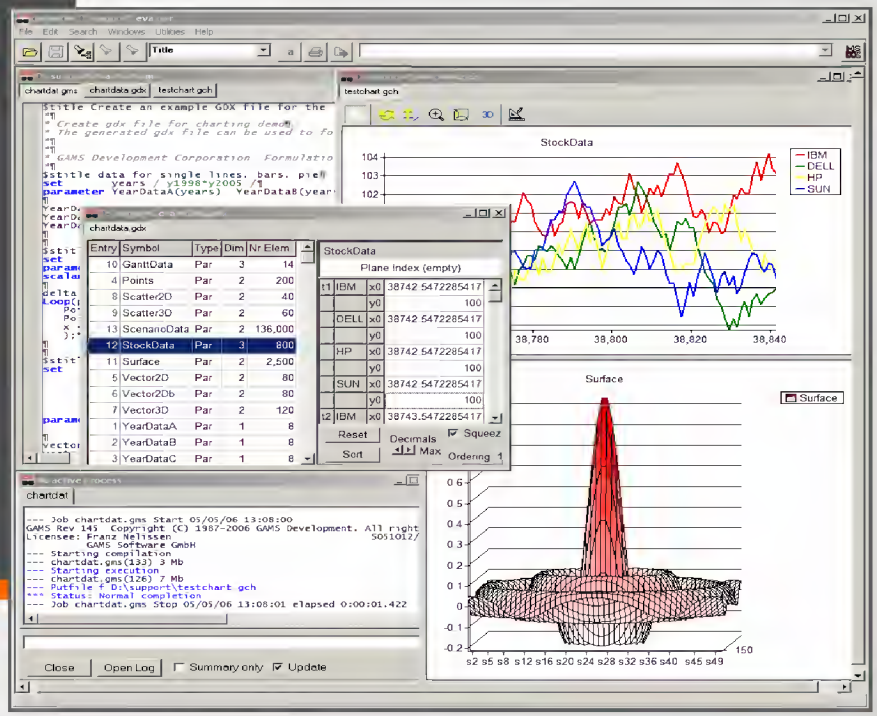
25+ Years
GAMS Development

GAMS

Strong Development Environment

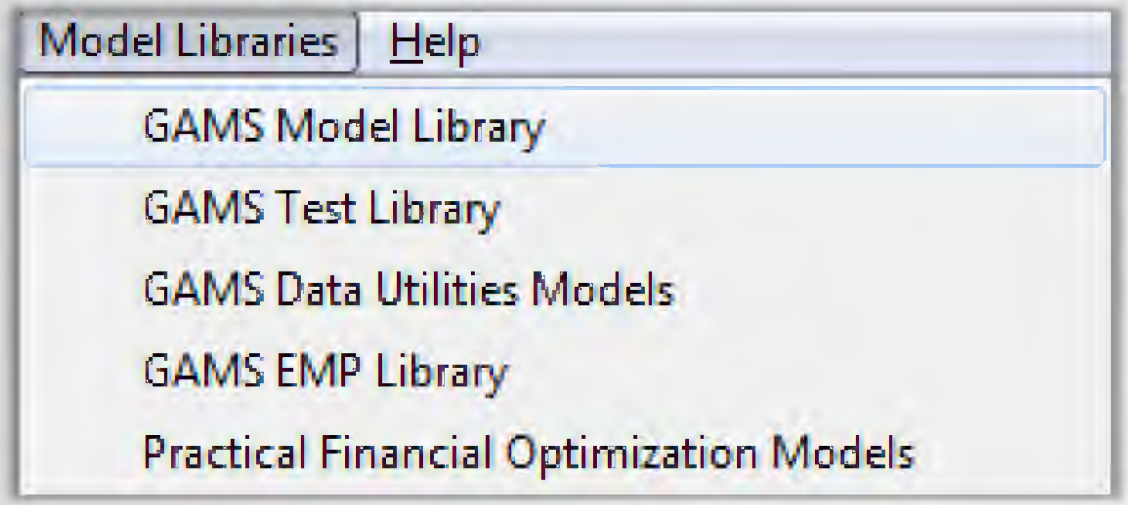
GAMS IDE

- Project management
- Editor / Syntax coloring / Spell checks
- Listing file / Tree view / Syntax-error navigation
- Model Debugging / Profiling
- Solver selection / Option selection
- Data viewer (GDX)
  - Export
  - Charting
- GAMS Processes Control
- Model Libraries

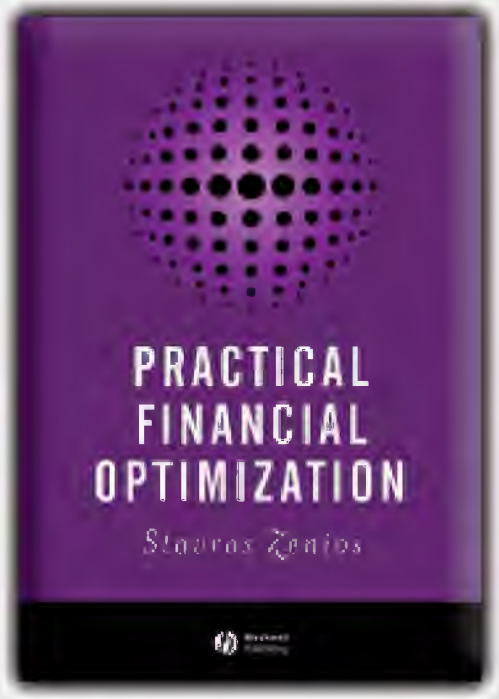


GAMS

Free Model Libraries



More than 1250 models!



GAMS

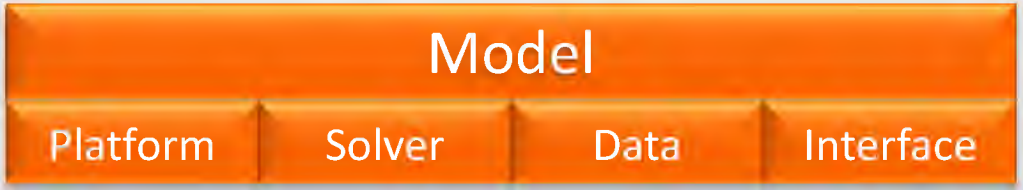
Design Principles

1

- Simple modeling language with a balanced mix of declarative and procedural elements

2

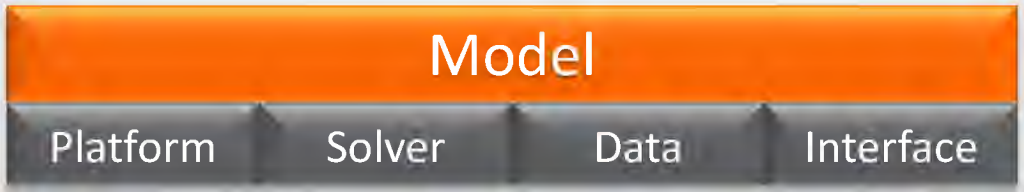
- Open architecture and interfaces to other systems, independent layers



GAMS

Simple Declarative Language

1. • Few basic language elements: sets, parameters, variables, equations, models
2. • Language similar to mathematical notation
3. • Easy to learn
4. • Model is executable description of the problem
5. • Lot‘s of code optimization under the hood



GAMS

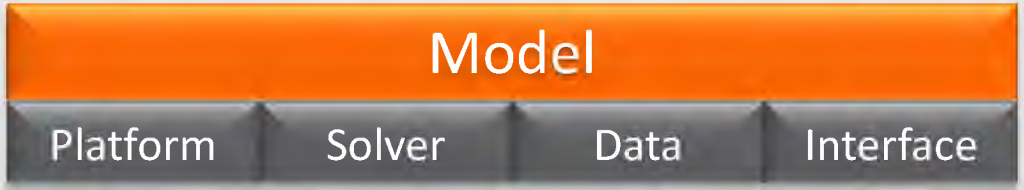
Mix of Declarative and Procedural Elements

Procedural elements like loops, for, if, macros and functions

Allow to build complex problem algorithms within GAMS

Interaction with other systems:

- Job control
- Data exchange



GAMS

Independence of Model and Operating System

Platforms supported by GAMS:

➡ Models can be moved between platforms with ease!

GAMS

Independence of Model and Solver

One environment for a wide range of model types and solvers

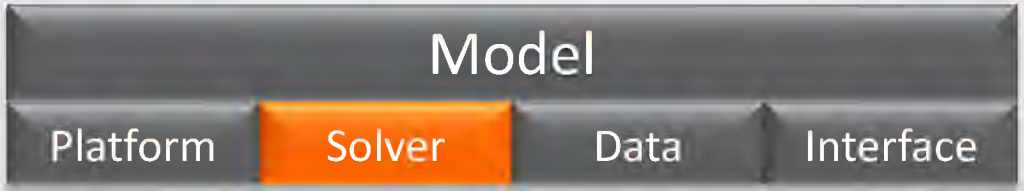
All major commercial LP/MIP solver

Open Source Solver (COIN)

Also solver for NLP, MINLP, global, and stochastic optimization

FICO · Gurobi Optimization · IBM · mosek · Sulum · COIN-OR

Switching between solvers with one line of code!

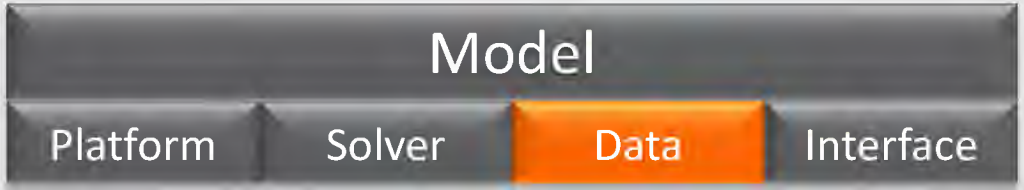


GAMS

Independence of Model and Data

- Declarative Modeling
- ASCII: Initial model development

- GDX: Data layer ("contract") between GAMS and applications
  - Platform independent
  - No license required
  - Direct GDX interfaces and general API
  - …



GAMS

Independence of Model and User Interface

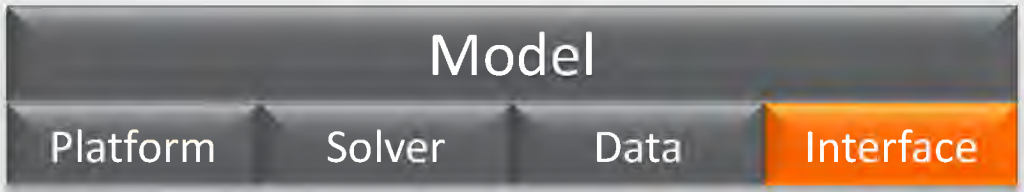
API's

- *Low Level*
- **Object Oriented**: .Net, Java, Python
- No modeling capability: Model is written in GAMS
- Wrapper class that encapsulates a GAMS model

Simple Encapsulation of a GAMS Model

Very simple interface:

- Properties to **communicate input data and results**
- Properties to **change options** like the solver to use
- Run() method to **run the model**



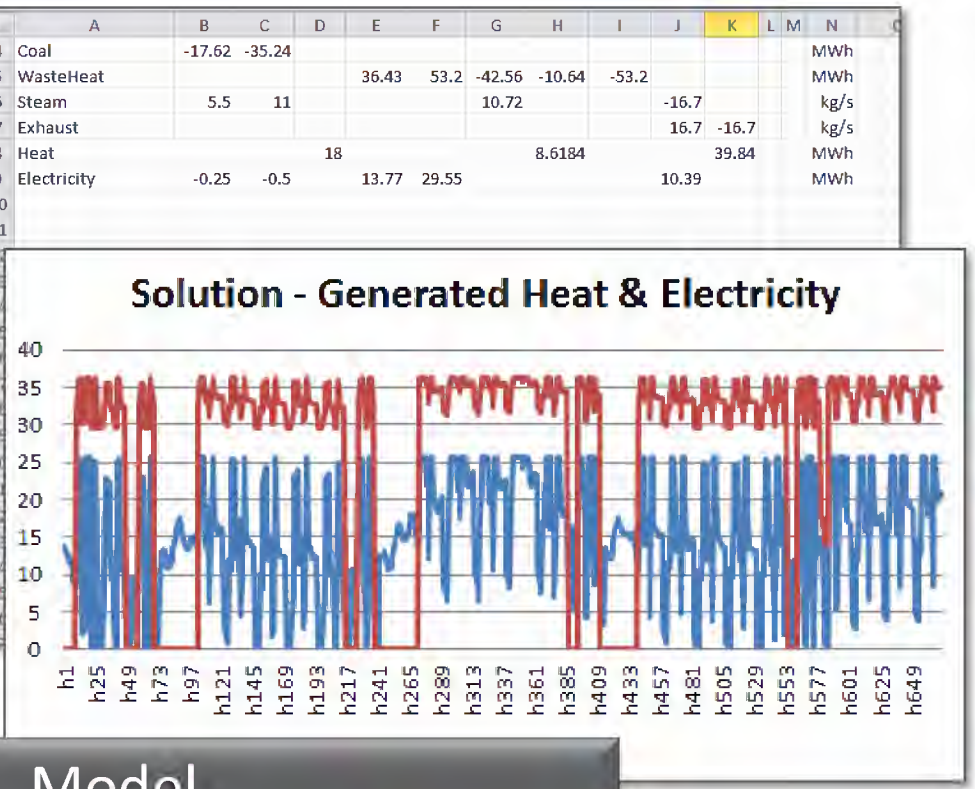
GAMS

Smart Links to other Applications

- User keeps working in his productive tool environment
- Application accesses all optimization capabilities of GAMS through API
- Visualization and analysis of model data and results in the application

MS Excel

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
4	Coal	-17.62	-35.24											MWh
5	WasteHeat				36.43	53.2	-42.56	-10.64	-53.2					MWh
6	Steam	5.5	11				10.72			-16.7				kg/s
7	Exhaust									16.7	-16.7			kg/s
8	Heat			18				8.6184			39.84			MWh
9	Electricity	-0.25	-0.5		13.77	29.55				10.39				MWh



Model

Platform | Solver | Data | Interface

GAMS

Smart Links to other Applications

- User keeps working in his productive tool environment
- Application accesses all optimization capabilities of GAMS through API
- Visualization and analysis of model data and results in the application

MatLab

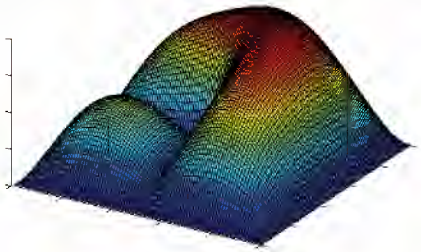


Figure 1: US dollar short rate scenarios

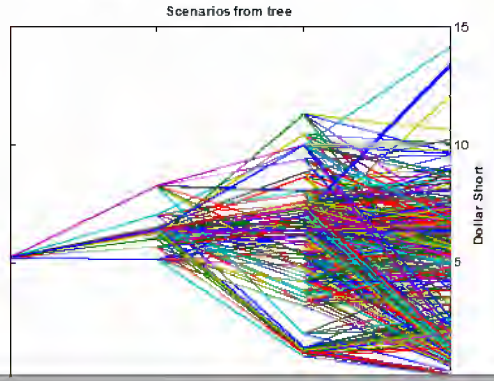
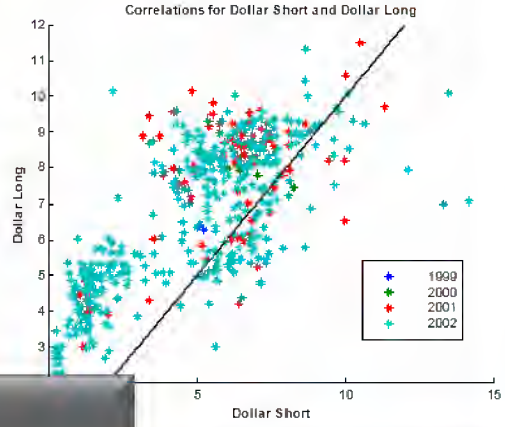


Figure 2: Short vs. long rates



Model

Platform | Solver | Data | Interface

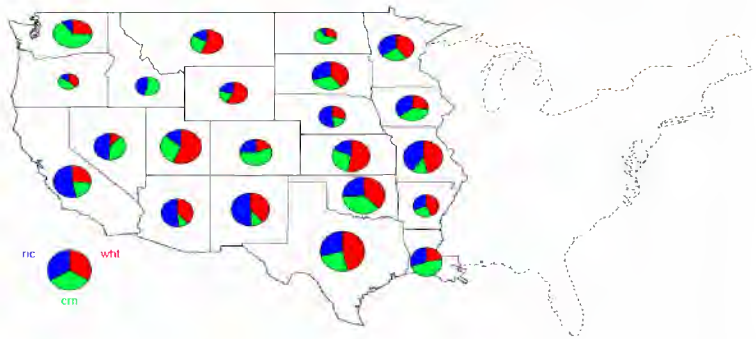
GAMS

Smart Links to other Applications

- User keeps working in his productive tool environment
- Application accesses all optimization capabilities of GAMS through API
- Visualization and analysis of model data and results in the application

R

```
v <- rgdx(fnSol, list(name = "tour", form = "sparse"))
nxt <- v$val[, 2]
# compute the sequence of cities, based on nxt
solSeq <- NA * c(1:n + 1)
k <- 1
for (j in c(1:n)
    solSeq[j] <-
    k <- nxt[k]
}
solSeq[n + 1] <-
if (k != 1) stop(
loc <- cmdscale(
rx <- range(x <-
ry <- range(y <-
tspres <- loc[sol
s <- seq(n)
```



Model

Platform | Solver | Data | Interface

GAMS

Striving for Innovation and Compatibility

Models must benefit from:

- Advancing hardware / New Platforms
- Enhanced / new solver and solution technology
- Improved / upcoming interfaces to other systems
- **New Modeling Concepts**

Protect investments of Users

- Life time of a model: 15+ years
- New maintainer, platform, solver, user interface
- Backward Compatibility
- **Software Quality Assurance**

GAMS

New Modeling and Solution Concepts

Examples:

- Disjunctive Programs
- Bilevel Programs
- Extended Nonlinear Programs
- Stochastic Programming
- …

Issues:

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

GAMS is conservative when it comes to syntax extensions

GAMS

The "GAMS" – Approach

Extended Mathematical Programming

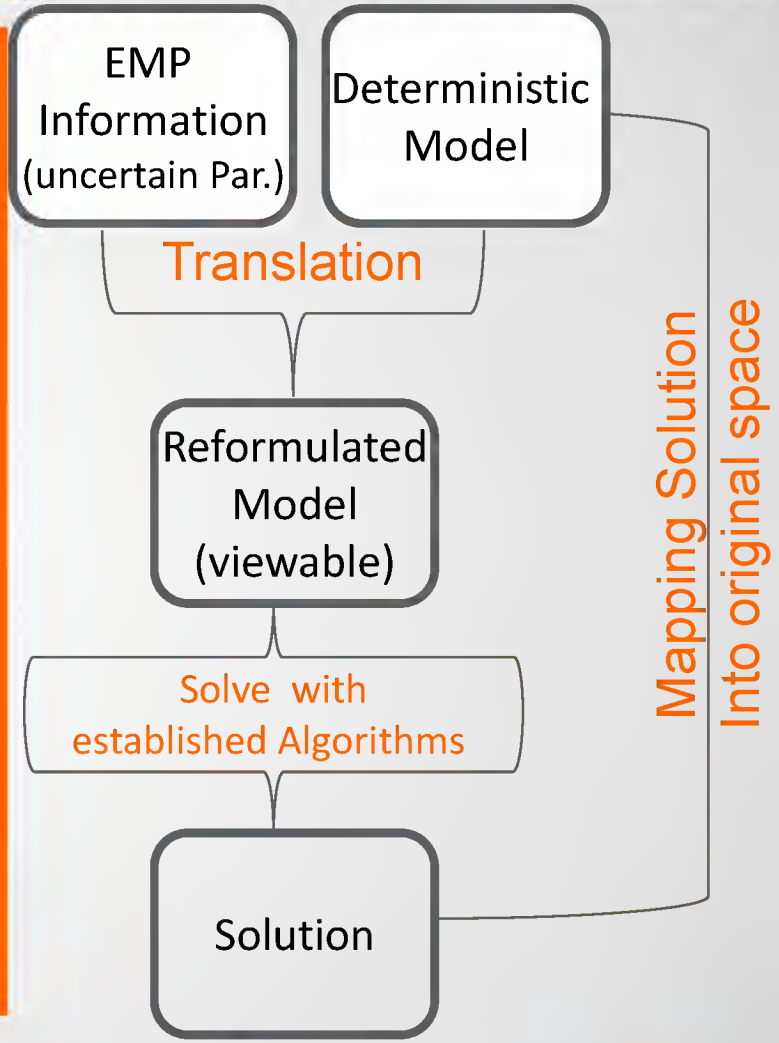
- **Experimental framework** for **automated** mathematical programming **reformulations**
- Keep the language simple: Do **not overload** existing GAMS notation
- Use **existing language features** to specify additional model features, structure, and semantics
- Express **extended model information** in **symbolic (source) form** and apply existing modeling/solution technology
- Package new tools with the production system

GAMS

Stochastic Programming in GAMS

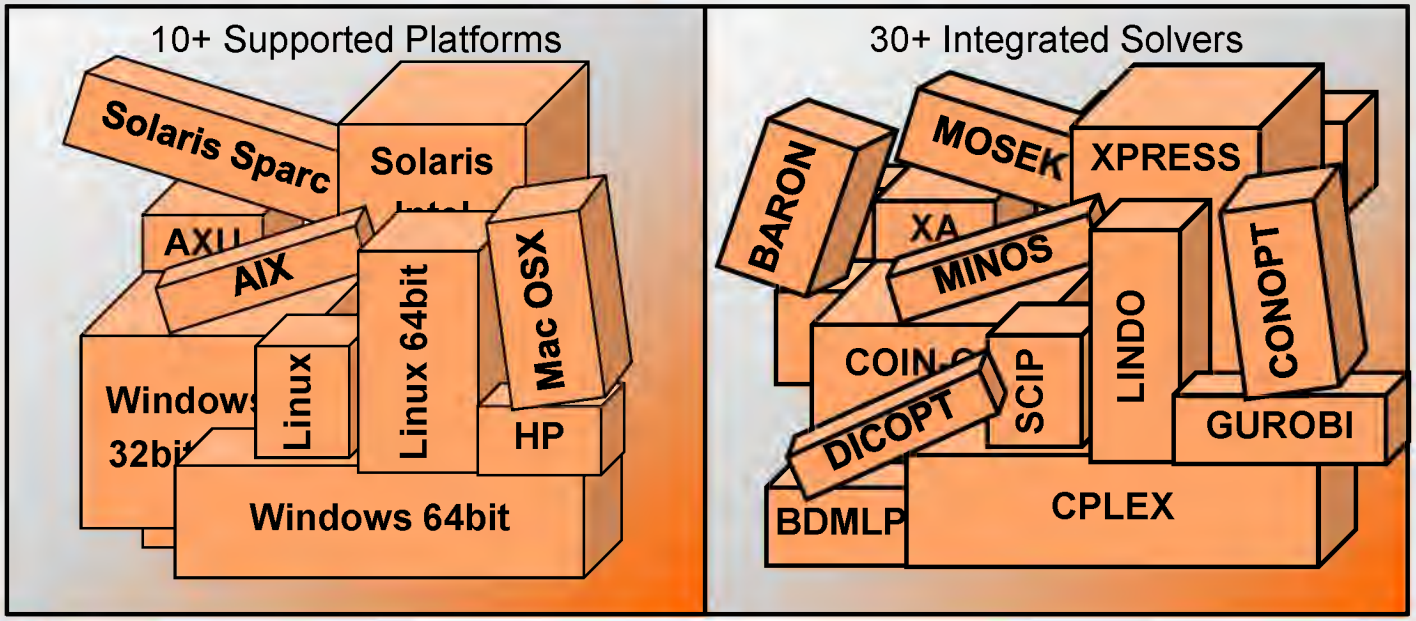
EMP/SP

- Simple interface to add uncertainty to existing deterministic models
- (EMP) Keywords to describe uncertainty include: discrete and parametric random variables, stages, chance constraints, Value at Risk, ...
- Available solution methods:
  - Automatic generation of **D**eterministic **E**quivalent (can be solved with any solver)
  - Specialized commercial algorithms (DECIS, LINDO)



GAMS

Software Quality Assurance



Perspectives:

- What is the impact of new features?
- What is the impact of updated or new solvers?
- Is the new distribution backward compatible?
- …

GAMS

Quality Test Models Library

- Tests to verify proper behavior of the system
- More than 600 quality test models, each containing numerous pass/fail tests
- Automatically executed every night for all solver combinations: → 12,000 runs / platform (all tests)
- Automatically generated test summaries with different level of information
- Assurance about the basic functionality of the software!

Latest GAMS System Builds and Test Results — Sunday 23Mar14 04:52 (UTC)

[Latest Builds | Alpha Builds | Beta Builds | Nightly Builds | Glossary] — Comments?

NOTE: The (nightly) alpha builds are internal development versions of the GAMS system. They may have known bugs, unfinished features, beta versions of third-party software, or may not function at all! Not for production use!

nightly a	System	Libraries	Build	Rev	Status and Time (UTC)		Initial Tests		Full Tests	
Friday	lnx	Download	24.3.0	45152	Test done	22Mar2014 12:42:54	805 runs 0 failures (q=0,s=0)	Report	11780 runs 0 failures (q=0,s=0)	Report
Saturday	leg	Download	24.3.0	45152	Test done	22Mar2014 22:03:49	804 runs 0 failures (q=0,s=0)	Report	*2198 runs 0 failures (q=0,s=0)*	Report
Friday	vs8	Download	24.3.0	45152	Test started	22Mar2014 01:12:52	947 runs 0 failures (q=0,s=0)	Report	results pending	
Friday	wei	Download	24.3.0	45152	Test started	22Mar2014 03:47:14	944 runs 1 failures (q=1,s=0)	Report	results pending	

GAMS

Why GAMS?

- Experience of 25+ years
- Broad user community from different areas
- Lots of model templates
- Strong development interface

- Consistent implemenation of design principles
  - Simple, but powerful modeling language
  - Independent layers
  - Open architecture: Designed to interact with other applications

- Open for new developments
- Protecting investments of users

Agenda

What is GAMS?

A Simple Example

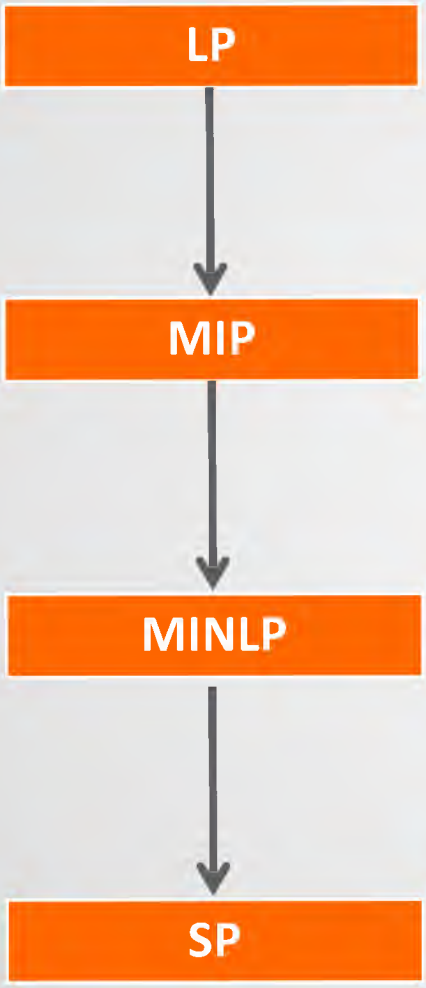
GAMS

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 how easy it is to change model type and, consequently, solver technology

Model types in this example



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

GAMS

A Simple Transportation Problem

Canning Plants (supply) → shipments (Number of cases) → **Markets** (demand)

Seattle (350)

San Diego (600)

Topeka (275)

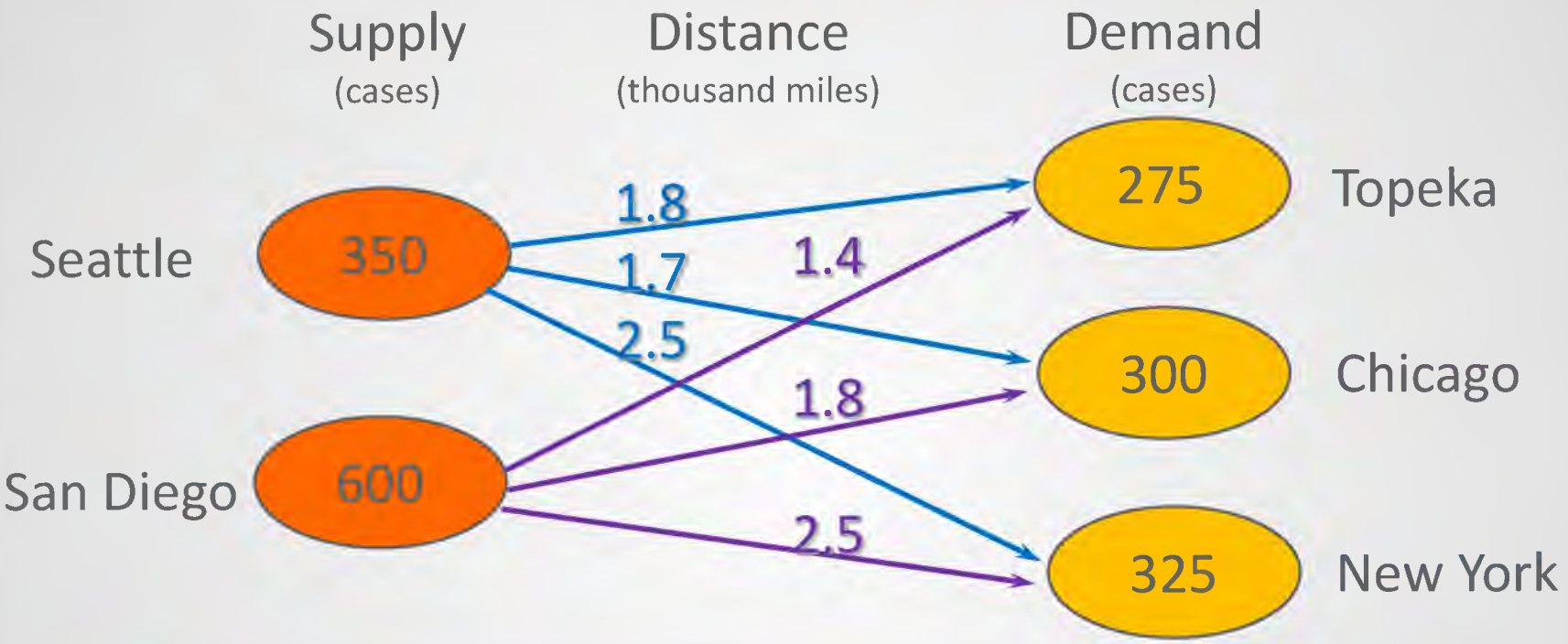
Chicago (300)

New York (325)

Freight: $90 case / thousand miles

GAMS

A Transportation Model



Minimize	Transportation cost
subject to	Demand satisfaction at markets
	Supply constraints

GAMS

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)

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

subject to

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

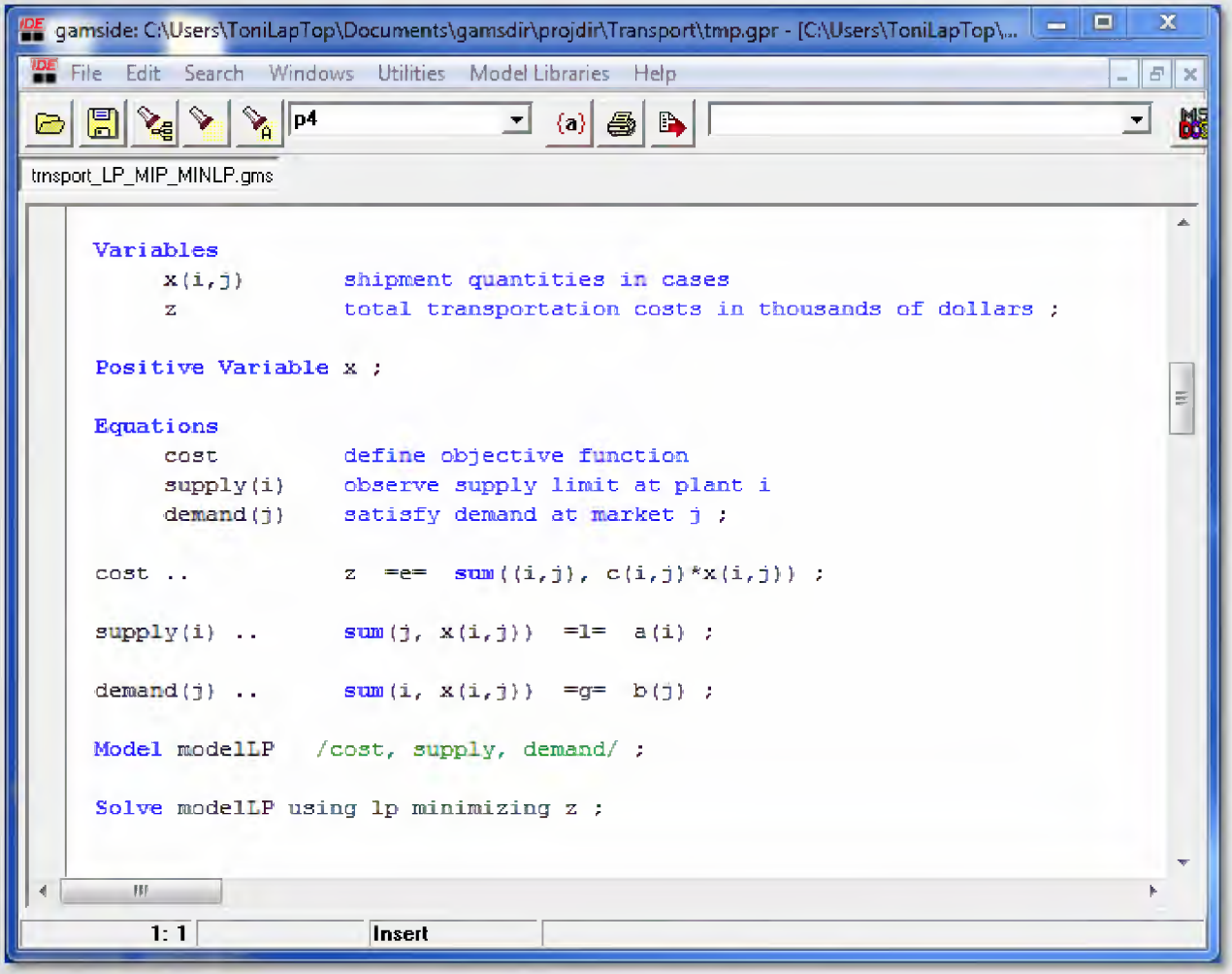
$\sum_i x_{ij} \geq b_j \quad \forall\, j$ (Shipments to each market $\geq$ demand)

$x_{ij} \geq 0 \quad \forall i,j$ (Do not ship from market to plant)

$i, j \in \mathbb{N}$

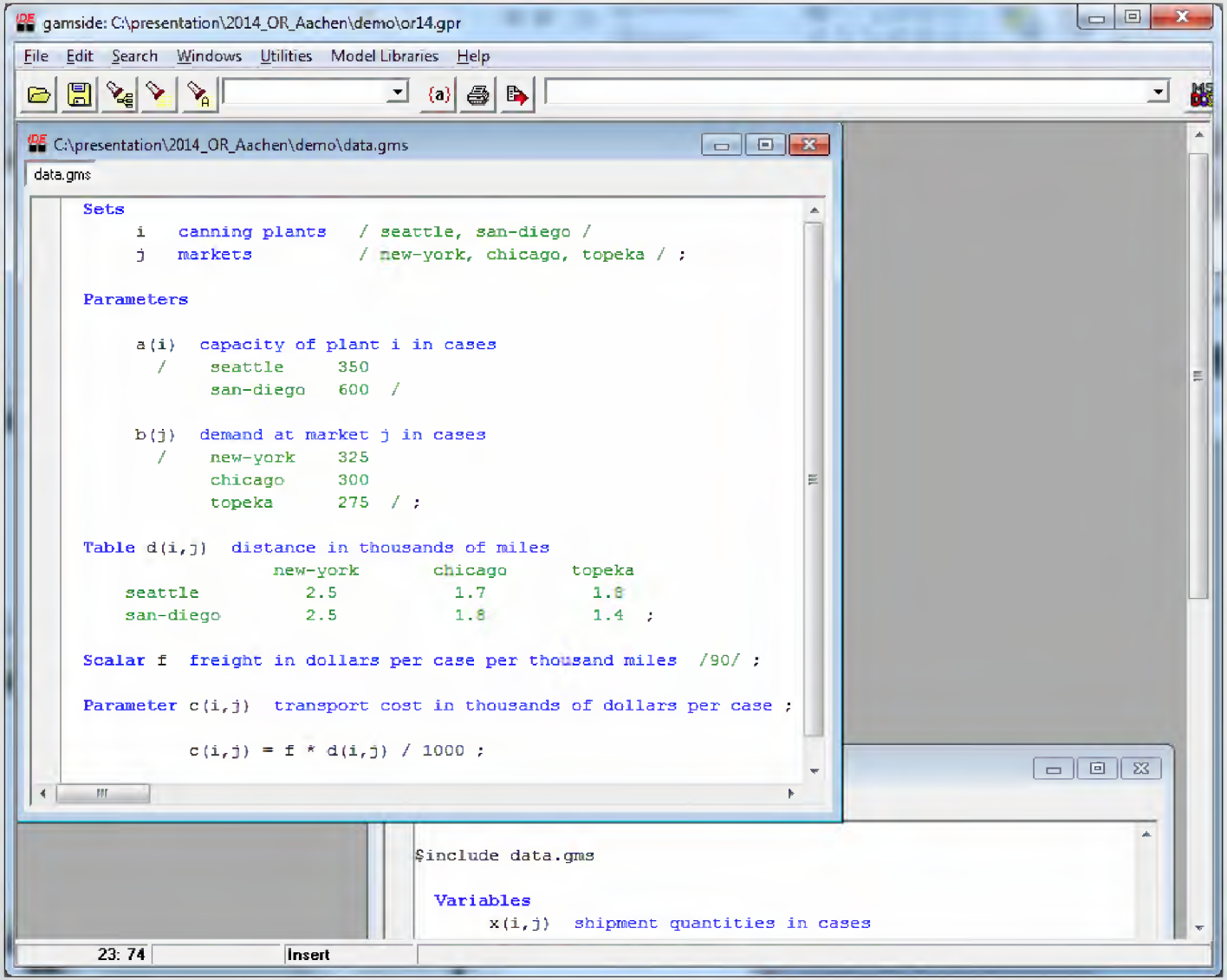
GAMS

GAMS Algebra (LP Model)

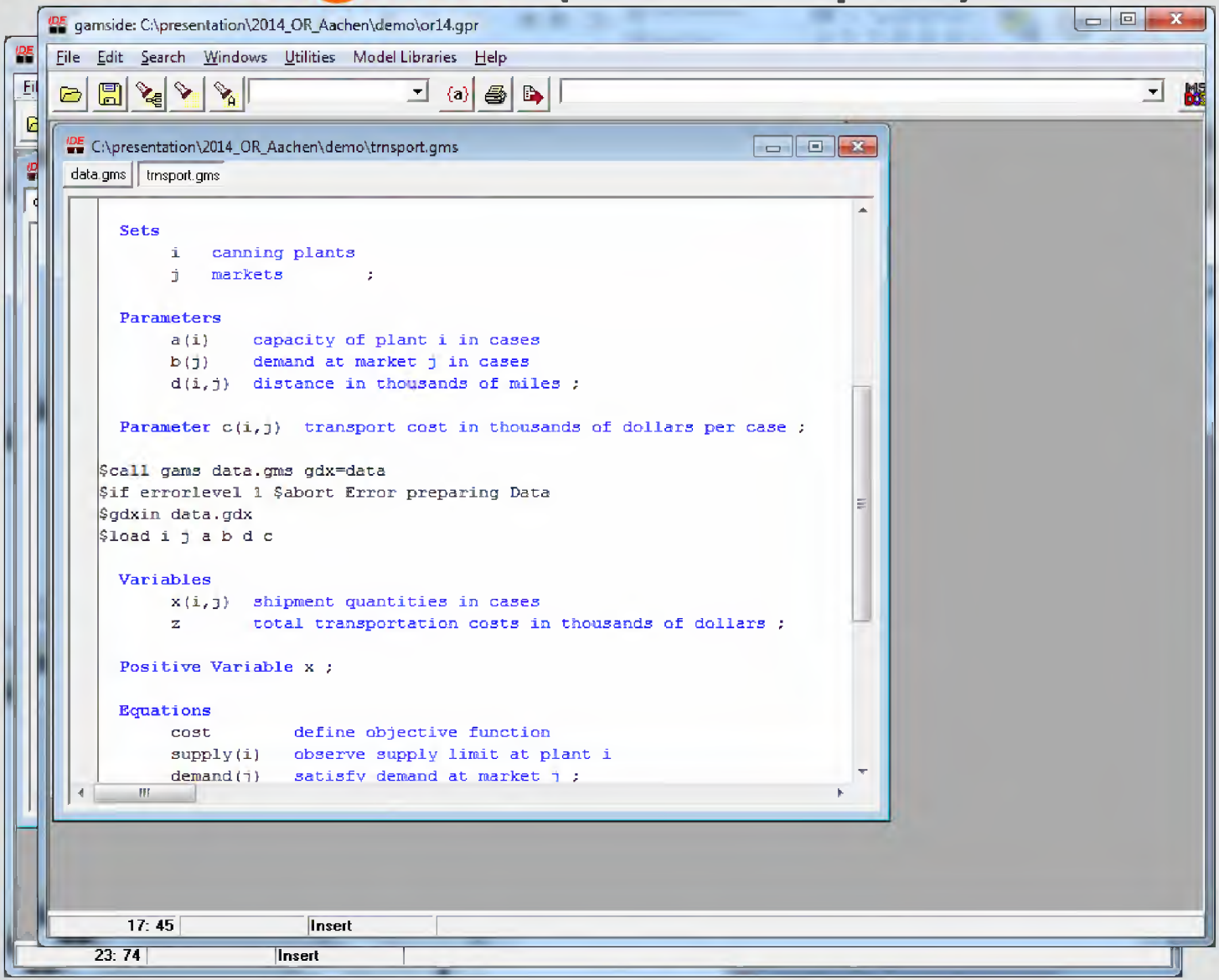


Hands-On

GAMS Algebra (Data Input)

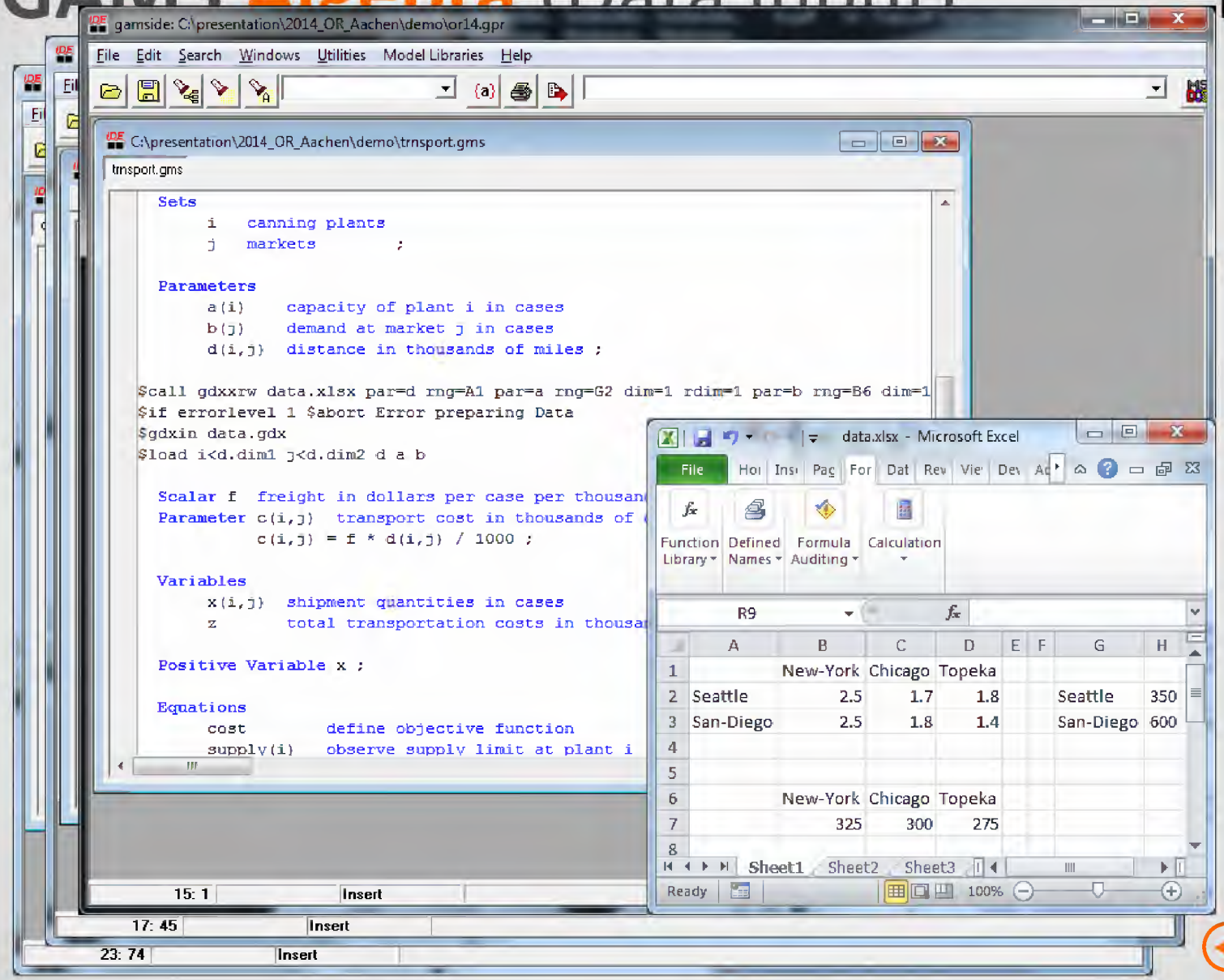


GAMS Algebra (Data Input)



GAMS

GAMS Algebra (Data Input)



GAMS

Solution to LP model

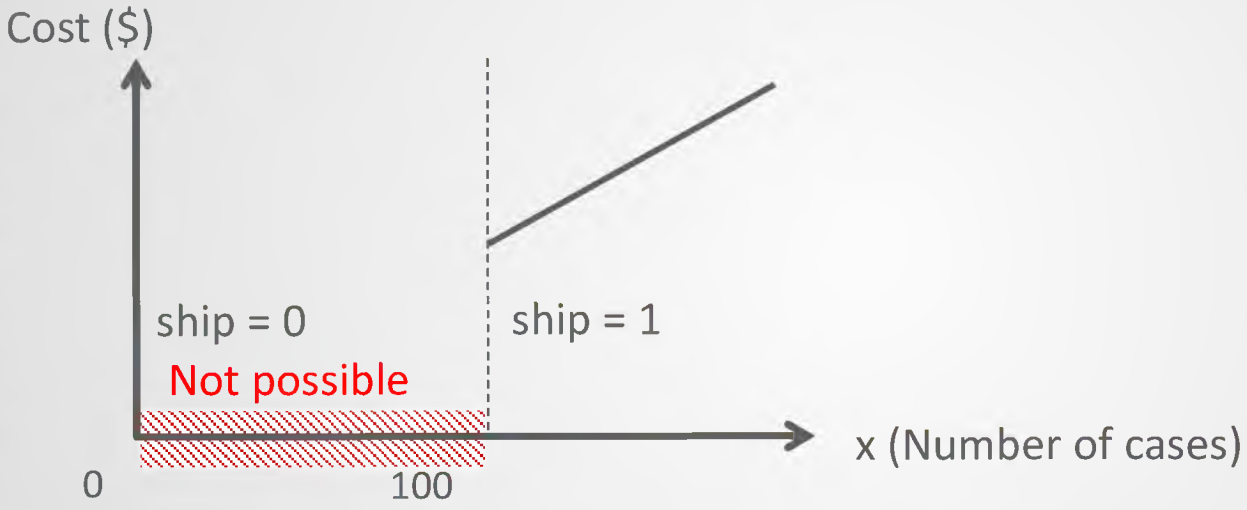
Canning Plants (supply) → shipments (Number of cases) → **Markets** (demand)

Freight: $90 case / thousand miles

Total cost: $153,675

Minimum shipment of 100 cases

- Shipment volume: **x** (continuous variable)
- Discrete decision: **ship** (binary variable)



add constraints:

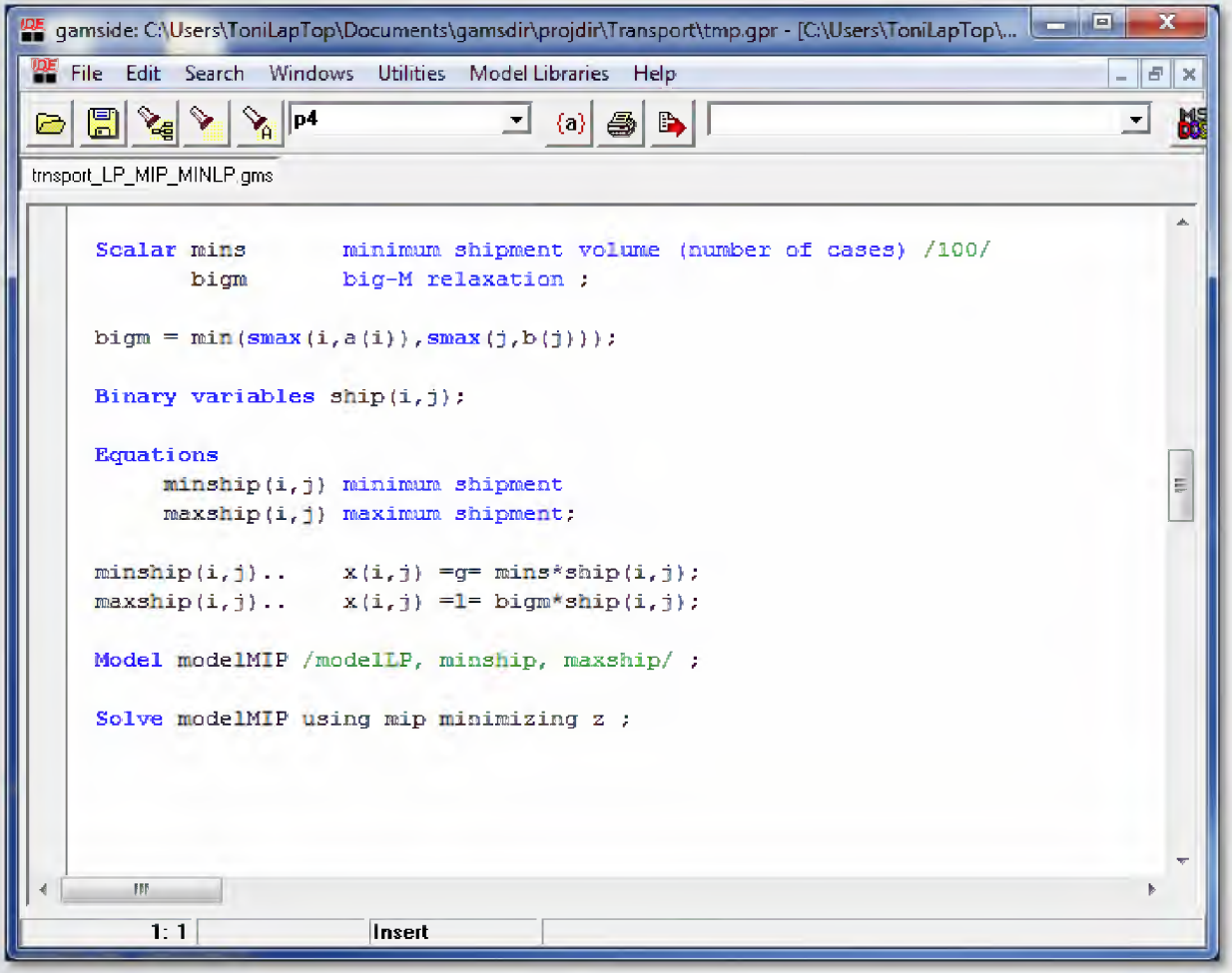
$x_{i,j} \geq 100 \cdot ship_{i,j} \quad \forall\, i,j$ (if ship=1, then ship at least 100)

$x_{i,j} \leq bigM \cdot ship_{i,j} \quad \forall\, i,j$ (if ship=0, then do not ship at all)

$ship_{i,j} \in \{0,1\}$

Hands-On

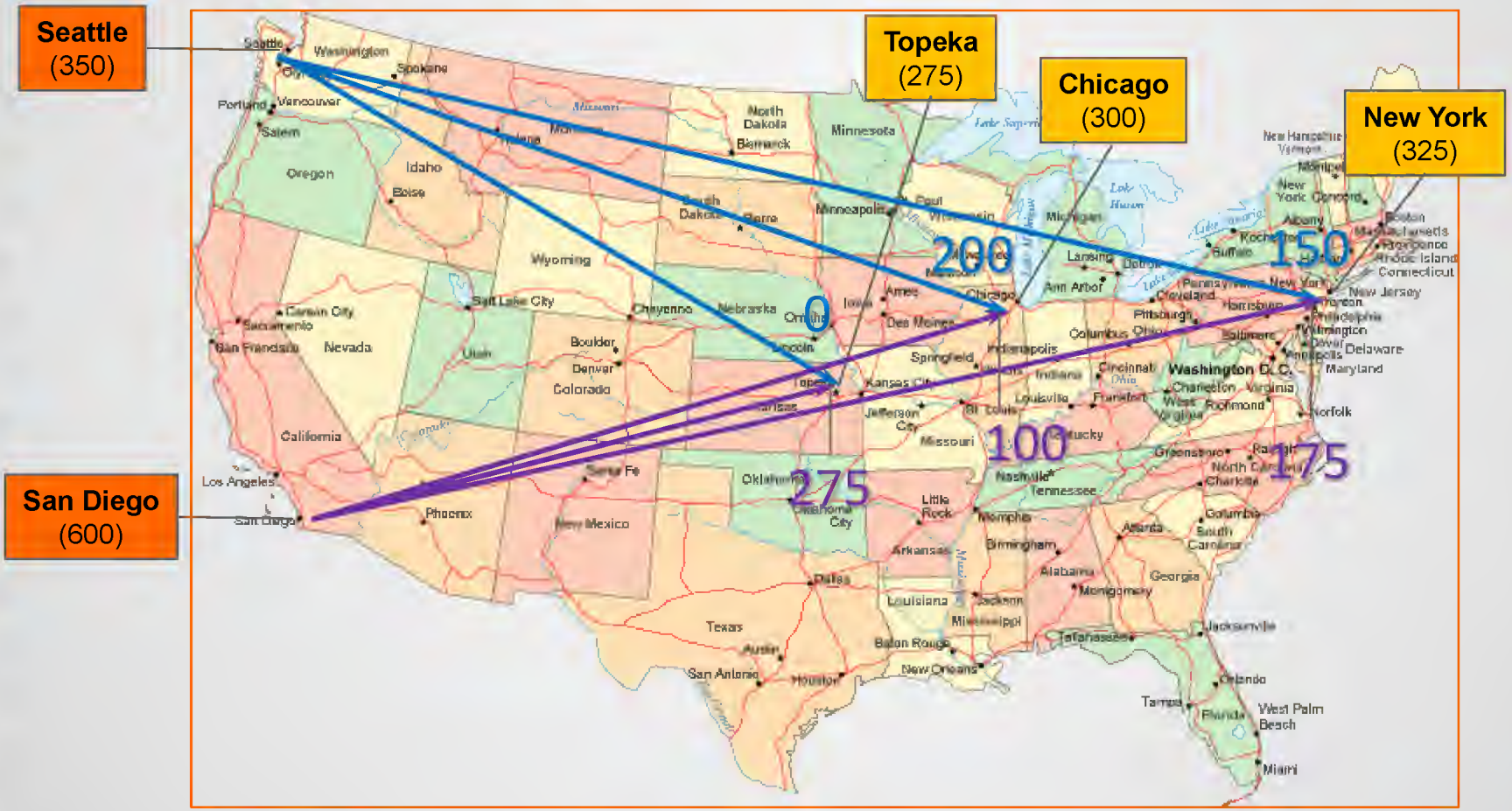
GAMS Algebra (MIP Model)



GAMS

Solution to MIP model

Canning Plants (supply) → shipments (Number of cases) → **Markets** (demand)

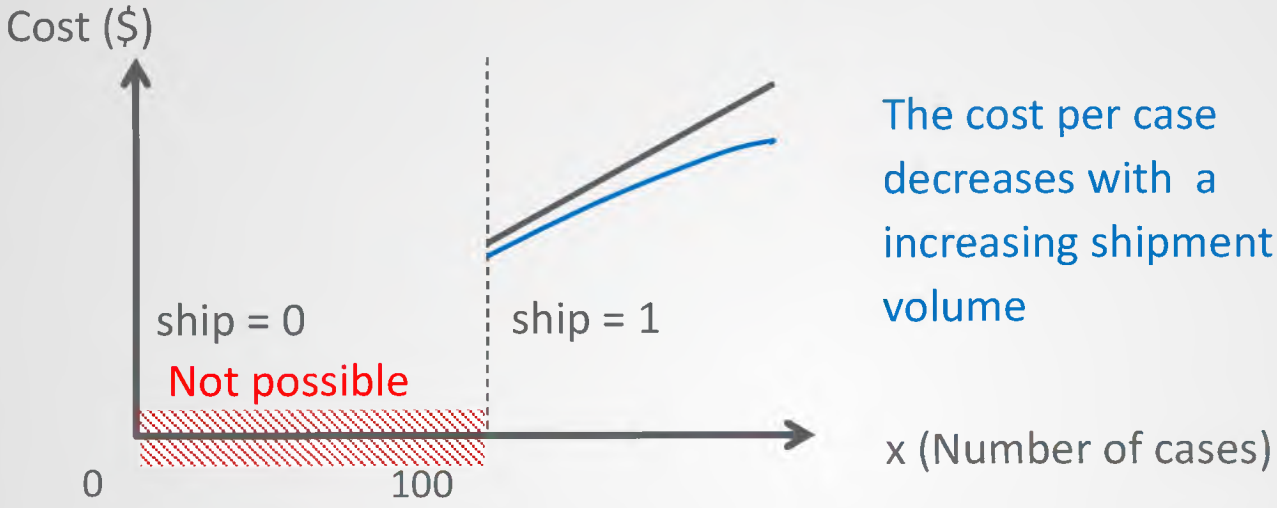


Freight: $90 case / thousand miles

Total cost: $153,675

GAMS

Cost Savings (non-linear)



The cost per case decreases with a increasing shipment volume

Replace:

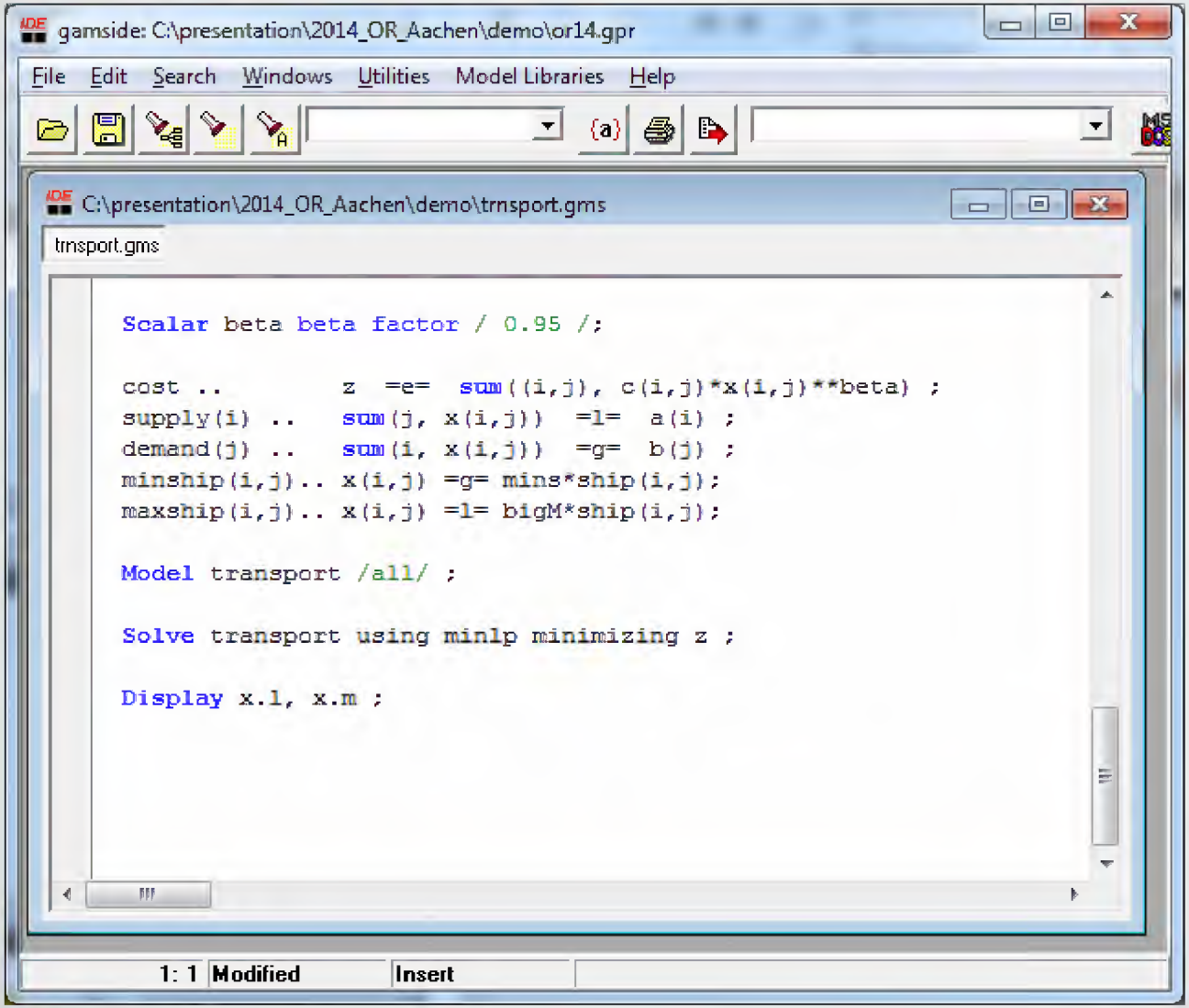
min $\sum_i \sum_j c_{ij} \cdot x_{ij}$ (Minimize total transportation cost)

With

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

Hands-On

GAMS Algebra (MINLP Model)



GAMS

Solution to MINLP model

Canning Plants (supply) → shipments (Number of cases) → **Markets** (demand)

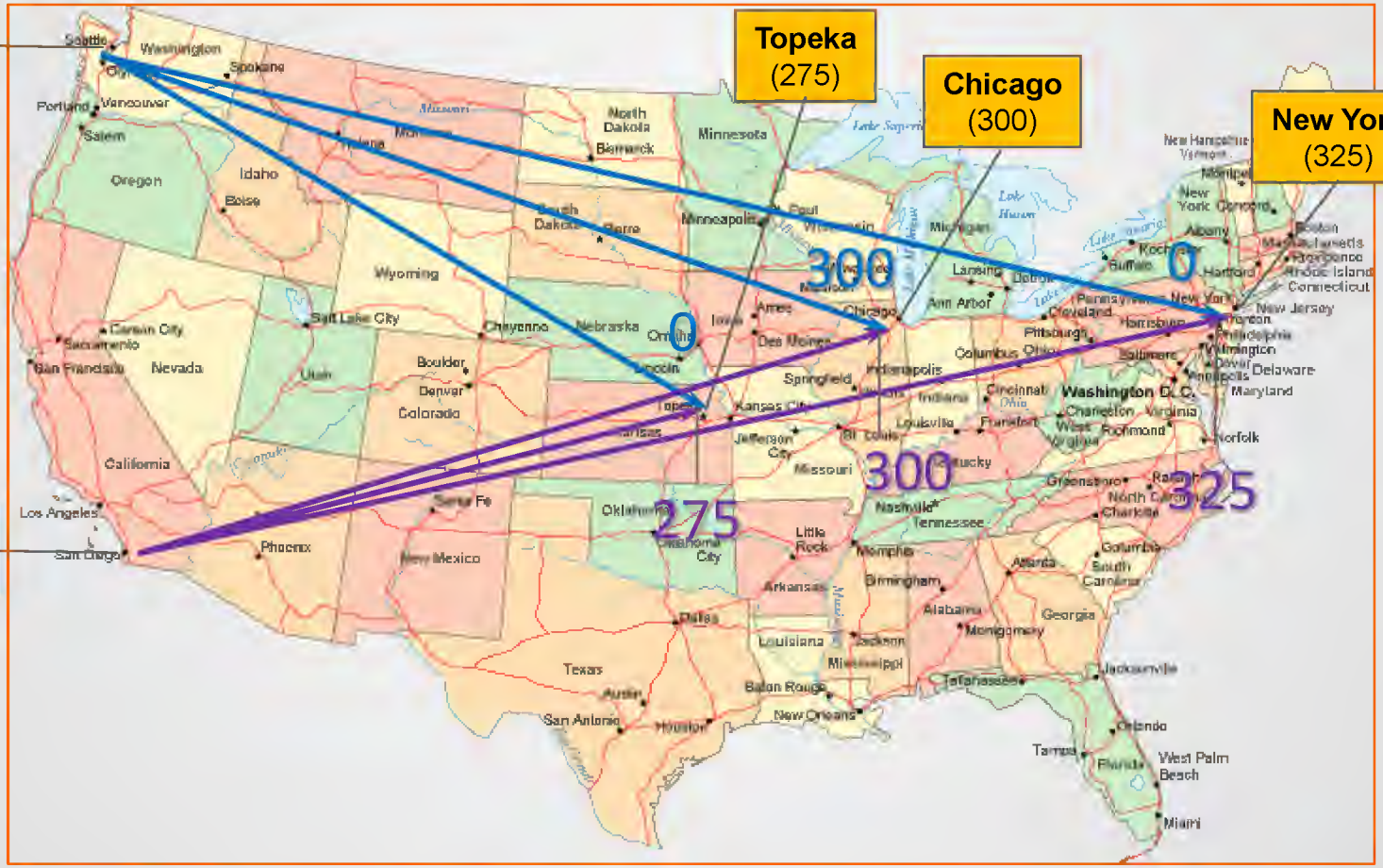
Seattle (350)

San Diego (600)

Topeka (275)

Chicago (300)

New York (325)



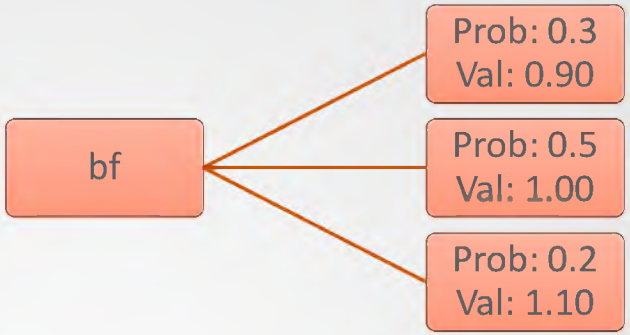
Freight: ~$90 case / thousand miles

Total cost: $115,438

GAMS

Uncertain Demand

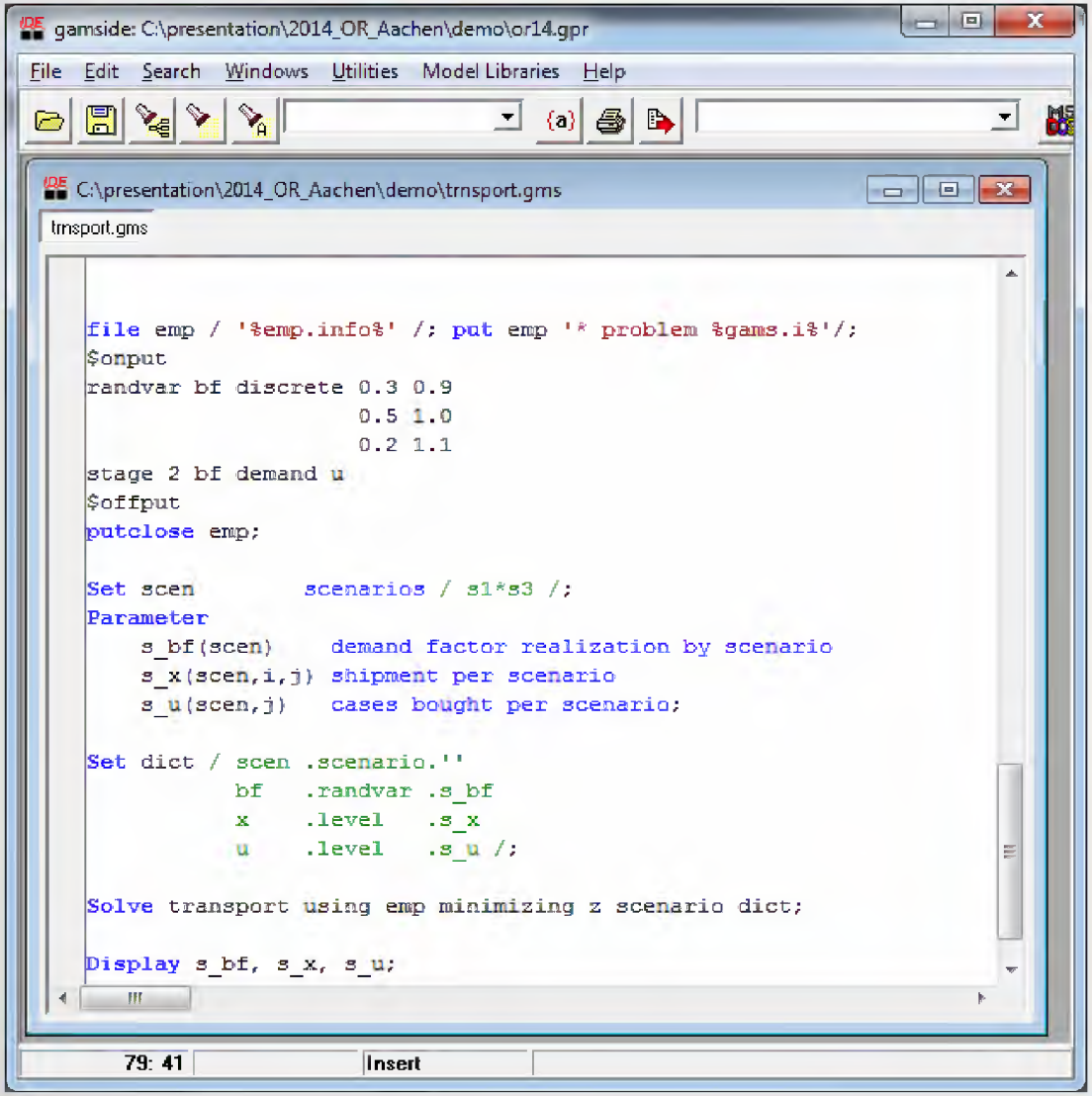
- Uncertain demand factor *bf*



- Decisions to make:
  - How many units should be shipped "here and now" (without knowing the outcome of the uncertain demand)?
    → *First-stage decision*
  - What can be done after the outcome becomes known and we did not ship enough?
    → *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

GAMS Algebra (SP Model)



GAMS

Solution to SP model

Canning Plants (supply) → shipments (Number of cases) → **Markets** (demand)

Freight: $90 case / thousand miles

Total cost: $158,588

GAMS

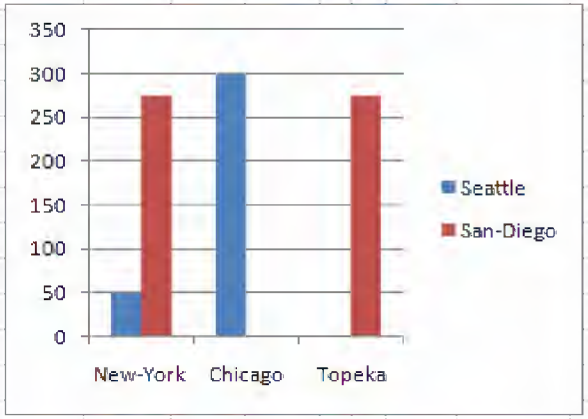
Stochastic Programming in GAMS

- 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)
- More information: http://www.gams.com/dd/docs/solvers/empsp.pdf

52

Outlook: Running GAMS in an Application

	A	B	C	D	E	F
1	**Distance**	New-York	Chicago	Topeka		Supply
2	Seattle	2.5	1.7	1.8		350
3	San-Diego	2.5	1.8	1.4		600
4						
5	Demand	325	300	275		
6						
7	Freight cost	90				
8						
9						
10	**SHIPMENT**	New-York	Chicago	Topeka		
11	Seattle	50	300	0		
12	San-Diego	275	0	275		



Clear Solution

GAMS Directory:	c:\program files\GAMS23.2\
Working Directory:	c:\tmp\
Solver:	CPLEX

SOLVE LP

SOLVE MIP

```
Solver: CPLEX
Equations: 6  Variables: 7
Model Status: 1 Optimal
Solver Status: 1 Normal Completion
Iterations: 4  Solve Time: 0.00
---
Objective Value: 153.675
```

Hands-On

53

Thank You

Europe
GAMS Software GmbH
P.O. Box 40 59
50216 Frechen, Germany
Phone: +49 221 949 9170
Fax: +49 221 949 9171
info@gams.de

USA
GAMS Development Corp. 1217 Potomac Street, NW Washington, DC 20007
USA
Phone: +1 202 342 0180
Fax: +1 202 342 0181
sales@gams.com