

GAMS – An Introduction

Get ready to learn the basics of GAMS

Frederik Fiand & Lutz Westermann

GAMS Software GmbH

Agenda

GAMS at a Glance

GAMS - Hands On Examples

Outlook 1 - Deployment of GAMS models

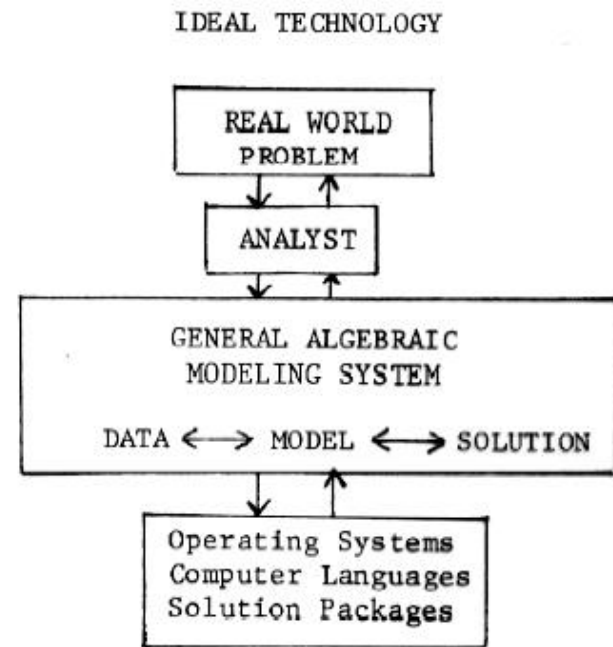
Outlook 2 - Solving Multiple Models Efficiently

GAMS at a Glance

1976 - A World Bank Slides

The Vision

GAMS came into being!



- RESULT:
- Limited drain of resources
 - Same representation of models for humans and machines
 - Model representation is also model documentation

The aim of this system is to provide one representation of a model which is easily understood by both humans and machines.

[J. Bischoff, A. Meeraus, On the Development of a General Algebraic Modeling System in a Strategic Planning Environment. *Mathematical Programming Study* 20 (**1982**) 1-29.]

Self-documenting model. A GAMS model is a machine-executable documentation of an optimization problem.

[M. Bussieck & A. Meeraus, Algebraic Modeling for IP and MIP (GAMS), *Annals of Operations Research* 149(1): *History of Integer Programming: Distinguished Personal Notes and Reminiscences*, Guest Editors: Kurt Spielberg and Monique Guignard-Spielberg, February, **2007**, pp. 49-56]

Company

- Roots: World Bank, 1976
- Went commercial in 1987
- Locations
 - GAMS Development Corporation (Washington)
 - GAMS Software GmbH (Germany)
- Product
 - The **G**eneral **A**lgebraic **M**odeling **S**ystem

What did this give us?

Simplified model development & maintenance

Increased productivity tremendously

Made mathematical optimization available to a broader audience (domain experts)

➤ 2012 INFORMS Impact Prize

Broad User Community and Network

14,000+ licenses

Users: 50% academic, 50% commercial

GAMS used in more than 120 countries

Uniform interface to ~40 solvers



30+ Years
GAMS Development

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

Declarative and Procedural Language Elements

Declarative elements

- Similar to mathematical notation
- Easy to learn - few basic language elements: sets, parameters, variables, equations, models
- Model is executable (algebraic) description of the problem

Procedural elements

- Control Flow Statements (e.g. loops, for, if,...),
- Build complex problem algorithms within GAMS
- Simplified interaction with other systems
 - Data exchange
 - APIs

Cross Platform GUI – GAMS Studio

- Open source Qt project (Mac/Linux/Win)
 - Published on GitHub under GPL
- Released in beta status
- Group Explorer
- Editor / Syntax coloring / Spell checks

- Tree view / Syntax-error navigation
- Option Editor
- Integrated Help
- Model Debugging & Profiling
- Solver selection & setup
- Data viewer
- GAMS Processes Control

The screenshot displays the GAMS Studio interface with the following components:

- Option Editor:** Shows 'SolveLink' set to 5 and 'LP' set to 'Xpress'.
- Explorer:** A tree view on the left showing a project structure with folders like 'bendersMPI', 'spBendersSeq', and 'tnsport', and files like 'tnsport.gms'.
- Editor:** The central window showing the GAMS code for 'tnsport.gms'. The code includes parameters for freight cost, shipment quantities, and total transportation costs, along with equations for the objective function and supply/demand constraints.
- Output:** The rightmost window showing the solver's output. It reports 12 non-zero elements, 0 entities, and 0 sets. The final objective value is 153.675, and the solution is found after 3 simplex iterations.

At the bottom, a status bar indicates the file path 'C:\Users\User\Documents\GAMSStudio\workspace\tnsport.gms', line 67, and encoding 'UTF-8'.

Independence of Model and Operating System

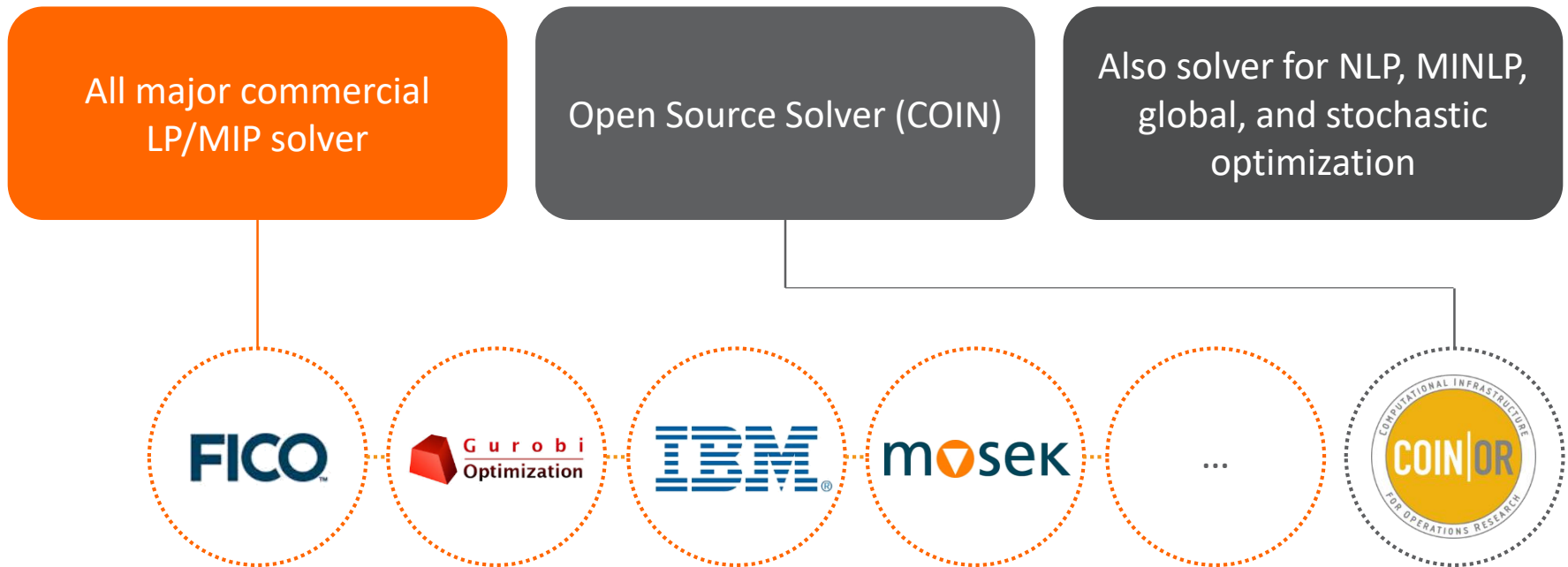


Platforms supported by GAMS:

➡ Models can be moved between platforms with ease!

Independence of Model and Solver

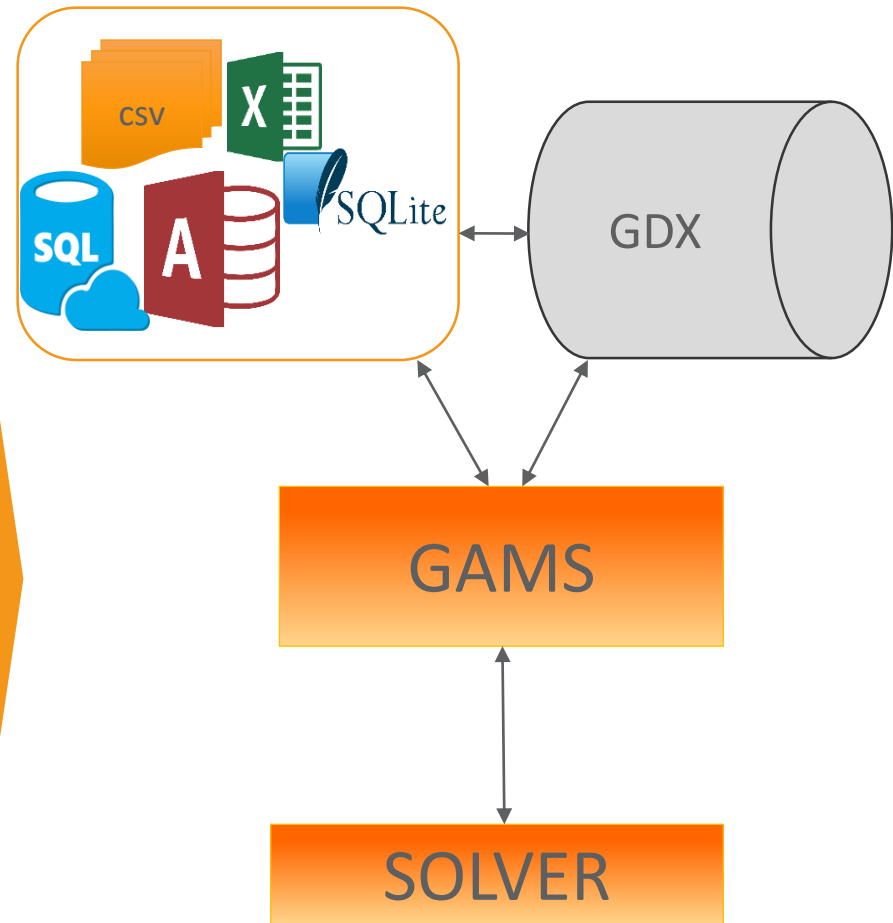
One environment for a wide range of model types and solvers



➔ Switching between solvers with one line of code!

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
 - ...



Independence of Model and User Interface

API's

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



Model Libraries Help

- GAMS Model Library
- GAMS Test Library
- GAMS Data Utilities Models
- GAMS EMP Library
- GAMS API Library
- Practical Financial Optimization Models
- Nonlinear Optimization Applications (N. Andrei)

➤ **More than 1400 models!** ◀ 18 ▶

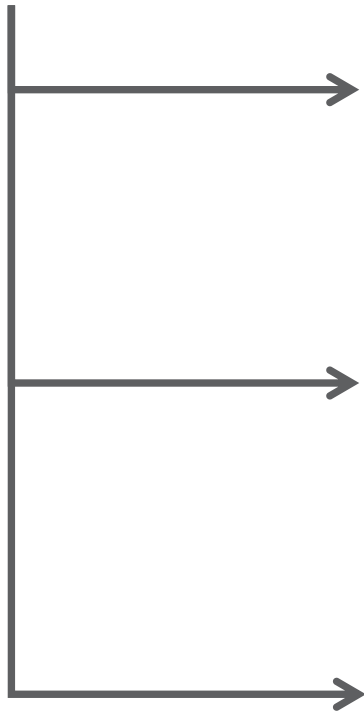
Why GAMS?

- Experience of 30+ years
- Broad user community from different areas
- Lots of model templates
- Strong development interface
- Consistent implementation 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

GAMS – Hands On Examples

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

LP

- Determine minimum transportation cost
- Result: city to city shipment volumes

MIP

- Discrete decisions
- E.g.: Ship at least 100 cases

MINLP

- Non-linearity
- E.g.: Decrease in unit cost with growing volumes

SP

- Uncertainty
- E.g.: Uncertain demand

A Simple Transportation Problem

Canning Plants (supply)

shipments

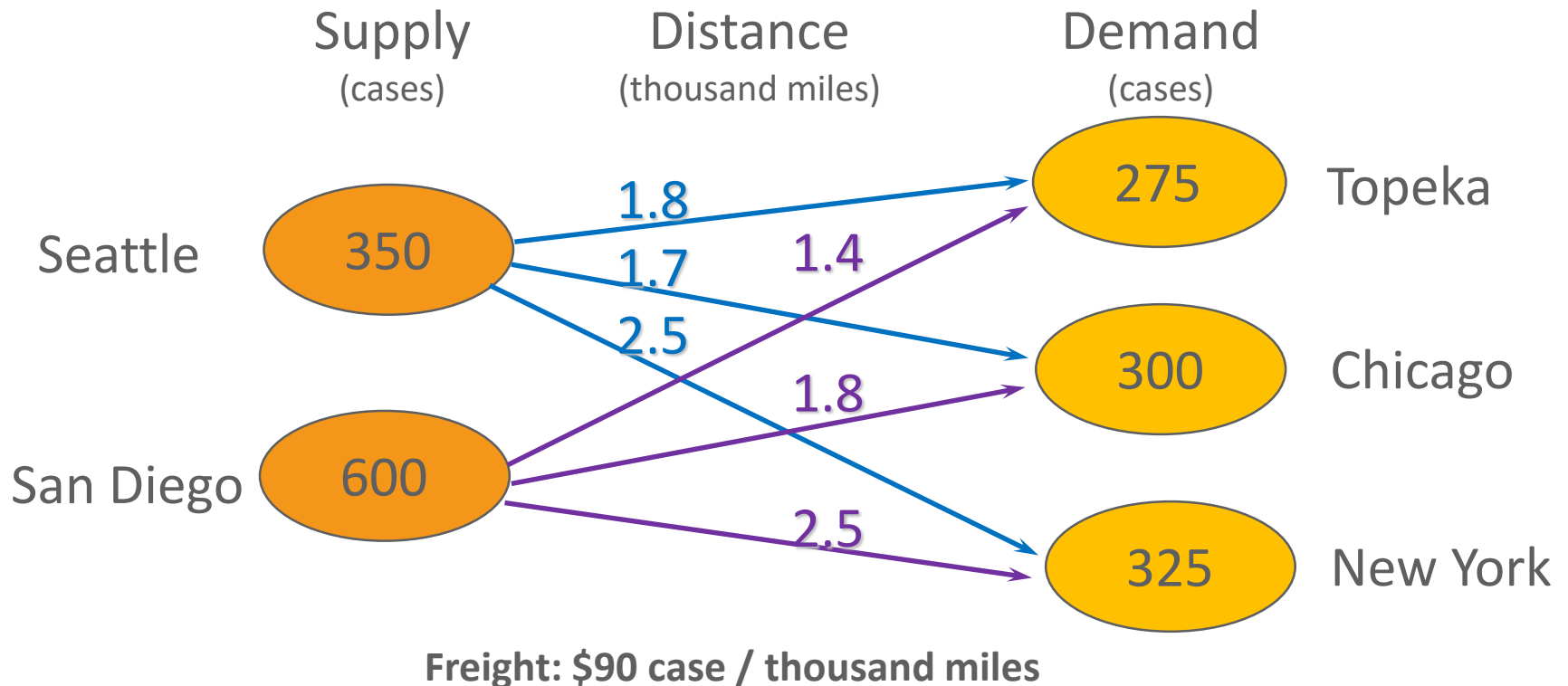
(Number of cases)

Markets (demand)



Freight: \$90 case / thousand miles

A Simple Transportation Problem



Minimize
subject to

Transportation cost
Demand satisfaction at markets
Supply constraints

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_j (Demand in cases)

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

subject to

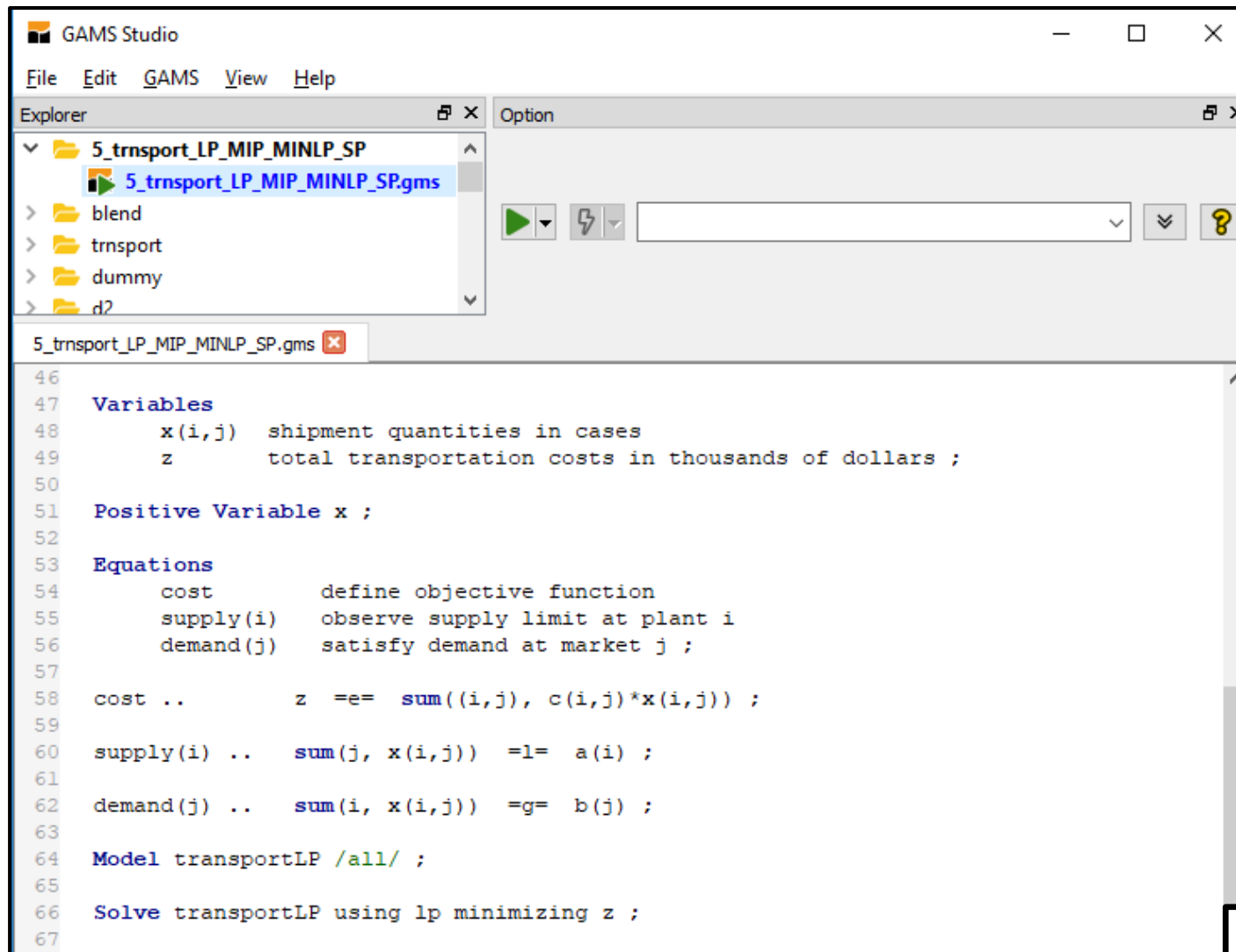
$\sum_j x_{ij} \leq a_i \quad \forall i$ (Shipments from each plant \leq supply capacity)

$\sum_i x_{ij} \geq b_j \quad \forall j$ (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 Syntax (LP Model)

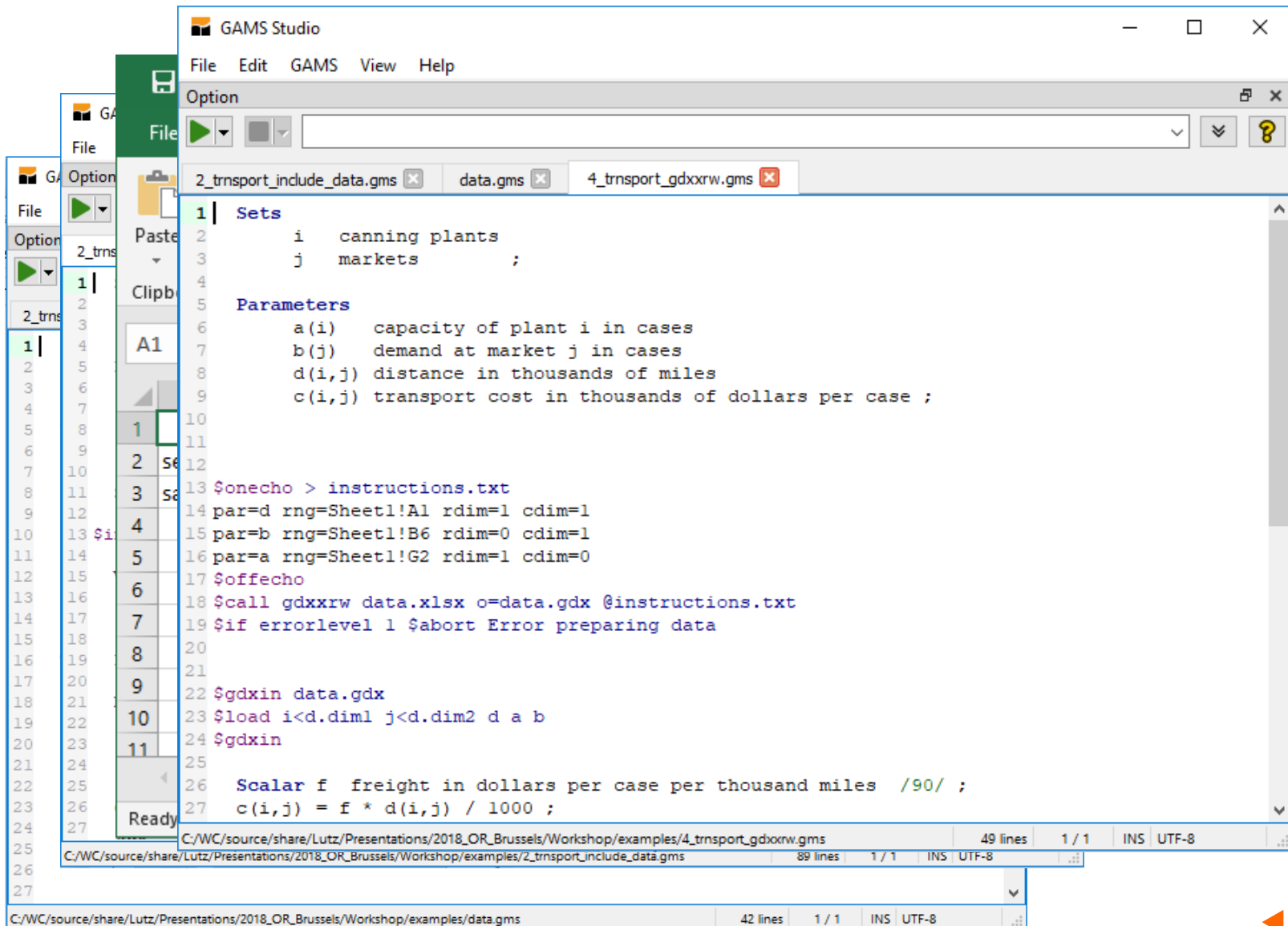


The screenshot shows the GAMS Studio application window. The title bar reads "GAMS Studio". The menu bar includes "File", "Edit", "GAMS", "View", and "Help". The "Explorer" pane on the left shows a project structure with a folder "5_trnsport_LP_MIP_MINLP_SP" containing a file "5_trnsport_LP_MIP_MINLP_SP.gms". Below the Explorer is a tab for the selected file. The "Option" pane on the right contains execution controls. The main editor displays the following GAMS code:

```
46
47 Variables
48     x(i,j)  shipment quantities in cases
49     z        total transportation costs in thousands of dollars ;
50
51 Positive Variable x ;
52
53 Equations
54     cost      define objective function
55     supply(i) observe supply limit at plant i
56     demand(j) satisfy demand at market j ;
57
58 cost ..      z  =e= sum((i,j), c(i,j)*x(i,j)) ;
59
60 supply(i) ..  sum(j, x(i,j))  =l= a(i) ;
61
62 demand(j) ..  sum(i, x(i,j))  =g= b(j) ;
63
64 Model transportLP /all/ ;
65
66 Solve transportLP using lp minimizing z ;
67
```

Hands-On

GAMS Syntax (Data Input)



```
1 | Sets
2     i  canning plants
3     j  markets      ;
4
5 | Parameters
6     a(i)  capacity of plant i in cases
7     b(j)  demand at market j in cases
8     d(i,j) distance in thousands of miles
9     c(i,j) transport cost in thousands of dollars per case ;
10
11
12
13 $onecho > instructions.txt
14 par=d rng=Sheet1!A1 rdim=1 cdim=1
15 par=b rng=Sheet1!B6 rdim=0 cdim=1
16 par=a rng=Sheet1!G2 rdim=1 cdim=0
17 $offecho
18 $call gdxrw data.xlsx o=data.gdx @instructions.txt
19 $if errorlevel 1 $abort Error preparing data
20
21
22 $gdxin data.gdx
23 $load i<d.dim1 j<d.dim2 d a b
24 $gdxin
25
26 Scalar f  freight in dollars per case per thousand miles  /90/ ;
27 c(i,j) = f * d(i,j) / 1000 ;
```

The screenshot shows the GAMS Studio interface with three open files: 2_trnsport_include_data.gms, data.gms, and 4_trnsport_gdxxrw.gms. The main editor displays the code for 4_trnsport_gdxxrw.gms. The code defines sets 'i' (canning plants) and 'j' (markets), and parameters 'a(i)' (capacity), 'b(j)' (demand), 'd(i,j)' (distance), and 'c(i,j)' (transport cost). It then uses the \$onecho/\$offecho block to execute a program that loads data from 'data.xlsx' into 'data.gdx'. Finally, it uses \$gdxin to load the data into the model, defining a scalar 'f' for freight cost and calculating 'c(i,j)'.

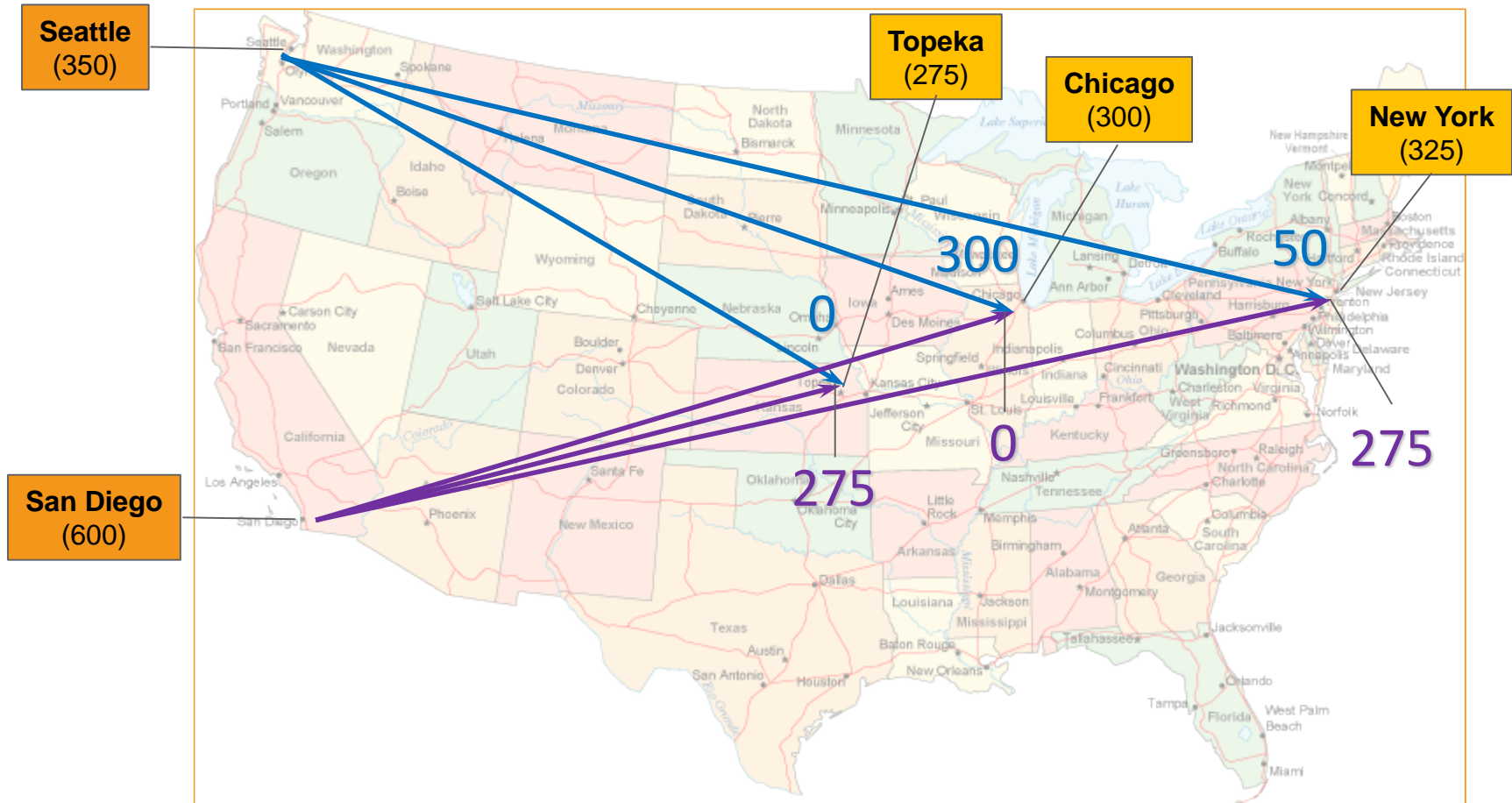
Solution to LP model

Canning Plants (supply)

shipments

(Number of cases)

Markets (demand)



Freight: \$90 case / thousand miles

Total cost: \$153,675

LP

- Determine minimum transportation cost
- Result: city to city shipment volumes

MIP

- Discrete decisions
- E.g.: Ship at least 100 cases

MINLP

- Non-linearity
- E.g.: Decrease in unit cost with growing volumes

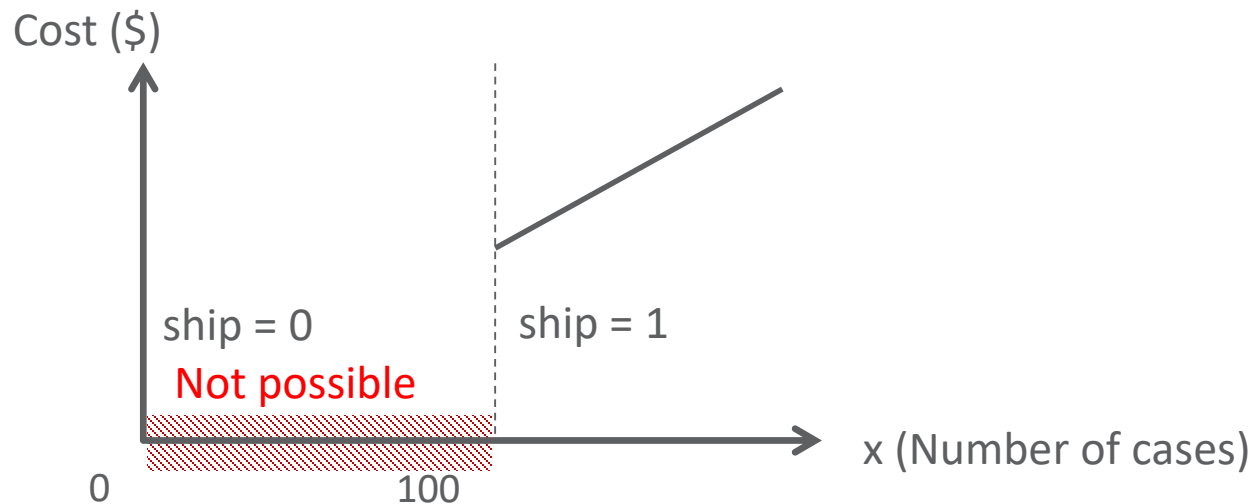
SP

- Uncertainty
- E.g.: Uncertain demand

MIP Model:

Minimum Shipment of 100 cases

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



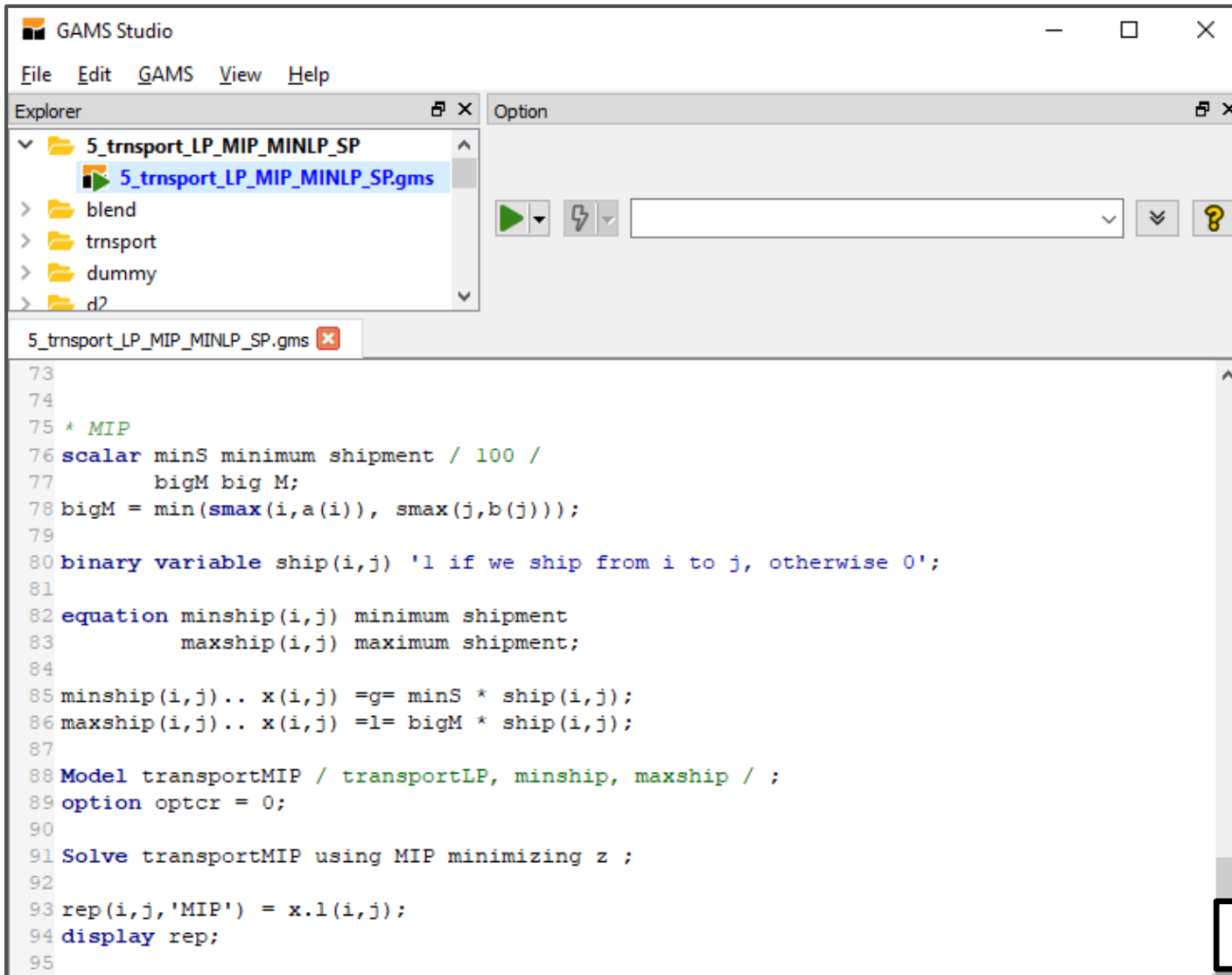
add constraints:

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

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

$$\text{ship}_{i,j} \in \{0,1\}$$

MIP Model: GAMS Syntax



The screenshot shows the GAMS Studio interface. The Explorer panel on the left lists the project structure: 5_trnsport_LP_MIP_MINLP_SP, 5_trnsport_LP_MIP_MINLP_SP.gms, blend, trnsport, dummy, and d2. The Option panel on the right contains a green play button, a lightning bolt icon, a dropdown menu, and a help icon. The main code editor displays the following GAMS code:

```
73
74
75 * MIP
76 scalar minS minimum shipment / 100 /
77      bigM big M;
78 bigM = min(smax(i,a(i)), smax(j,b(j)));
79
80 binary variable ship(i,j) '1 if we ship from i to j, otherwise 0';
81
82 equation minship(i,j) minimum shipment
83      maxship(i,j) maximum shipment;
84
85 minship(i,j).. x(i,j) =g= minS * ship(i,j);
86 maxship(i,j).. x(i,j) =l= bigM * ship(i,j);
87
88 Model transportMIP / transportLP, minship, maxship / ;
89 option optcr = 0;
90
91 Solve transportMIP using MIP minimizing z ;
92
93 rep(i,j,'MIP') = x.l(i,j);
94 display rep;
95
```

Hands-On

MIP Model: Results

GAMS Studio

File Edit GAMS View Help

Explorer

- 5_trnsport_LP_MIP_MINLP_SP
 - 5_trnsport_LP_MIP_MINLP_SP.gdx
 - 5_trnsport_LP_MIP_MINLP_SP.lst
 - 5_trnsport_LP_MIP_MINLP_SP.gms
- blend
- transport

Option

5_trnsport_LP_MIP_MINLP_SP.gms 5_trnsport_LP_MIP_MINLP_SP.lst 5_trnsport_LP_MIP_MINLP_SP.gdx

```

481 seattle .chicago      .      1.0000      1.0000      EPS
482 seattle .topeka        .      1.0000      1.0000      EPS
483 san-diego.new-york     .      1.0000      1.0000      EPS
484 san-diego.chicago     .      1.0000      1.0000      EPS
485 san-diego.topeka      .      1.0000      1.0000      EPS
486
487
488 **** REPORT SUMMARY :      0      NONOPT
489                          0      INFEASIBLE
490                          0      UNBOUNDED
491 GAMS 25.2.0 r67480 ALFA Released 2Aug18 WEX-
492 A Transportation Problem (TRANSPORT,SEQ=1)
493 E x e c u t i o n
494
495
496 ----      94 PARAMETER rep      report parameter
497
498
499
500 seattle .new-york      50.000
501 seattle .chicago     300.000      300.000
502 san-diego.new-york    275.000      325.000
503 san-diego.topeka     275.000      275.000
504
505
506
507 EXECUTION TIME      =      0.000 SECONDS
    
```

C:\WC\source\share\Lutz\Presentations\2018_OR_Brussels\Workshop\examples\5_trnsport_LP_MIP_MINLP_SP.lst

GAMS Studio

File Edit GAMS View Help

Explorer

- 5_trnsport_LP_MIP_MINLP_SP
 - 5_trnsport_LP_MIP_MINLP_SP.gdx
 - 5_trnsport_LP_MIP_MINLP_SP.lst
 - 5_trnsport_LP_MIP_MINLP_SP.gms
- blend
- transport

Option

5_trnsport_LP_MIP_MINLP_SP.gms 5_trnsport_LP_MIP_MINLP_SP.lst 5_trnsport_LP_MIP_MINLP_SP.gdx

Entry	Name	Type	Dim	Records	Text	i	j	LP	Value
3	a	Parameter	1	2	capacity of plant i in c...	seattle	new-york	LP	50
4	b	Parameter	1	3	demand at market j in ...	seattle	chicago	LP	300
15	bigM	Parameter	0	1	big M	san-diego	new-york	LP	275
7	c	Parameter	2	6	transport cost in thous...	san-diego	topeka	LP	275
10	cost	Equation	0	1	define objective functi...	seattle	chicago	MIP	300
5	d	Parameter	2	6	distance in thousands ...	san-diego	new-york	MIP	325
12	demand	Equation	1	3	satisfy demand at mar...	san-diego	topeka	MIP	275
6	f	Parameter	0	1	freight in dollars per ca...				
1	i	Set	1	2	canning plants				
2	j	Set	1	3	markets				
18	maxship	Equation	2	6	maximum shipment				
14	minS	Parameter	0	1	minimum shipment				
17	minship	Equation	2	6	minimum shipment				
13	rep	Parameter	3	7	report parameter				
16	ship	Variable	2	6	1 if we ship from i to j, ...				
11	supply	Equation	1	2	observe supply limit at...				
8	x	Variable	2	6	shipment quantities in ...				
9	z	Variable	0	1	total transportation co...				

Symbol Search: ☐ All Columns ☐ Squeeze Defaults

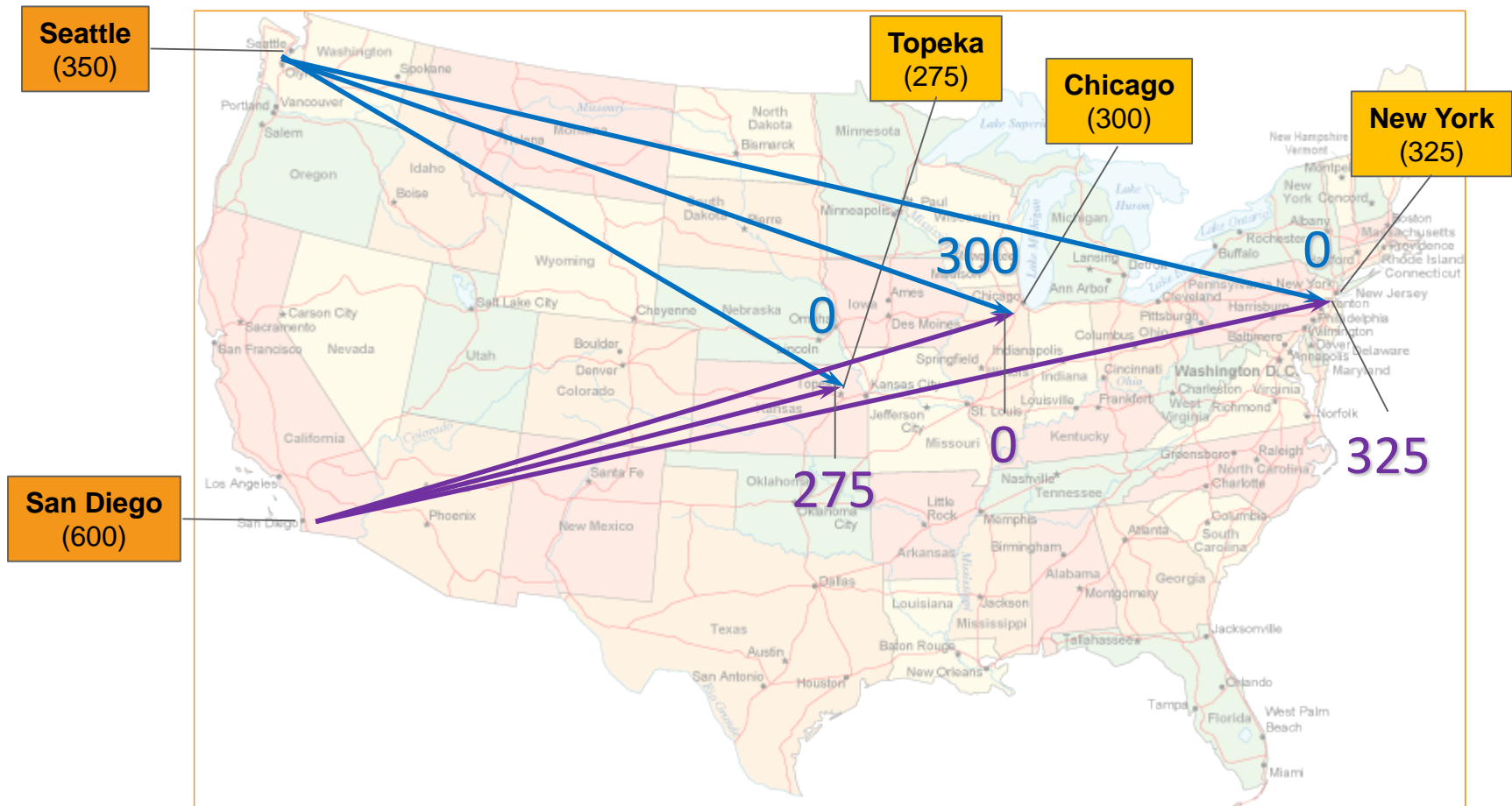
C:\WC\source\share\Lutz\Presentations\2018_OR_Brussels\Workshop\examples\5_trnsport_LP_MIP_MINLP_SP.gdx

MIP Model: Solution

Canning Plants (supply)

shipments
(Number of cases)

Markets (demand)



Freight: \$90 case / thousand miles

Total cost: \$153,675

LP

- Determine minimum transportation cost
- Result: city to city shipment volumes

MIP

- Discrete decisions
- E.g.: Ship at least 100 cases

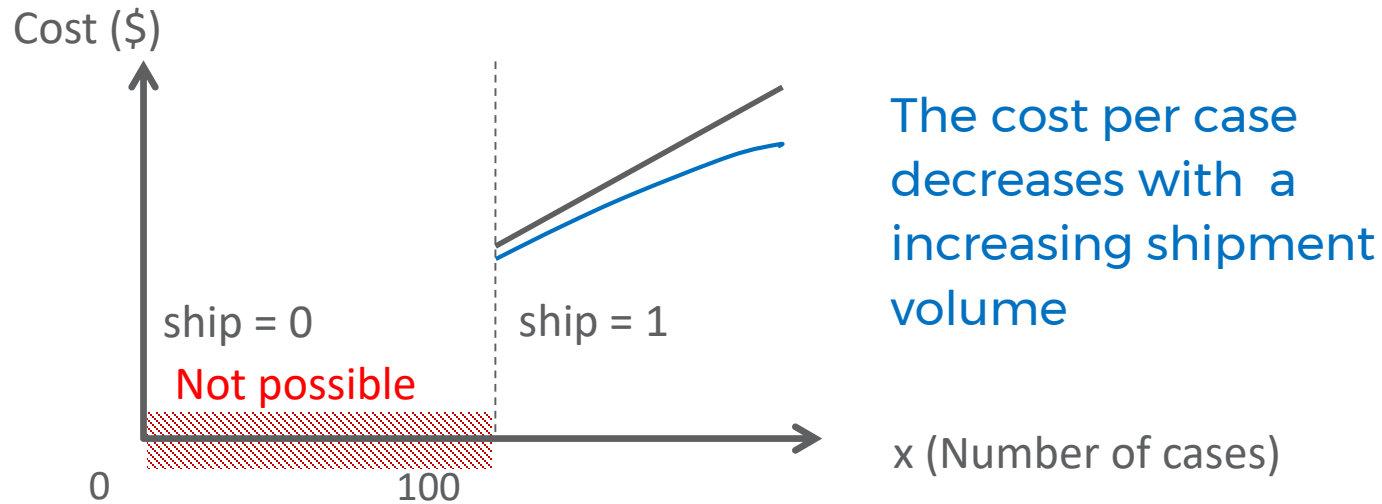
MINLP

- Non-linearity
- E.g.: Decrease in unit cost with growing volumes

SP

- Uncertainty
- E.g.: Uncertain demand

MINLP: Cost Savings



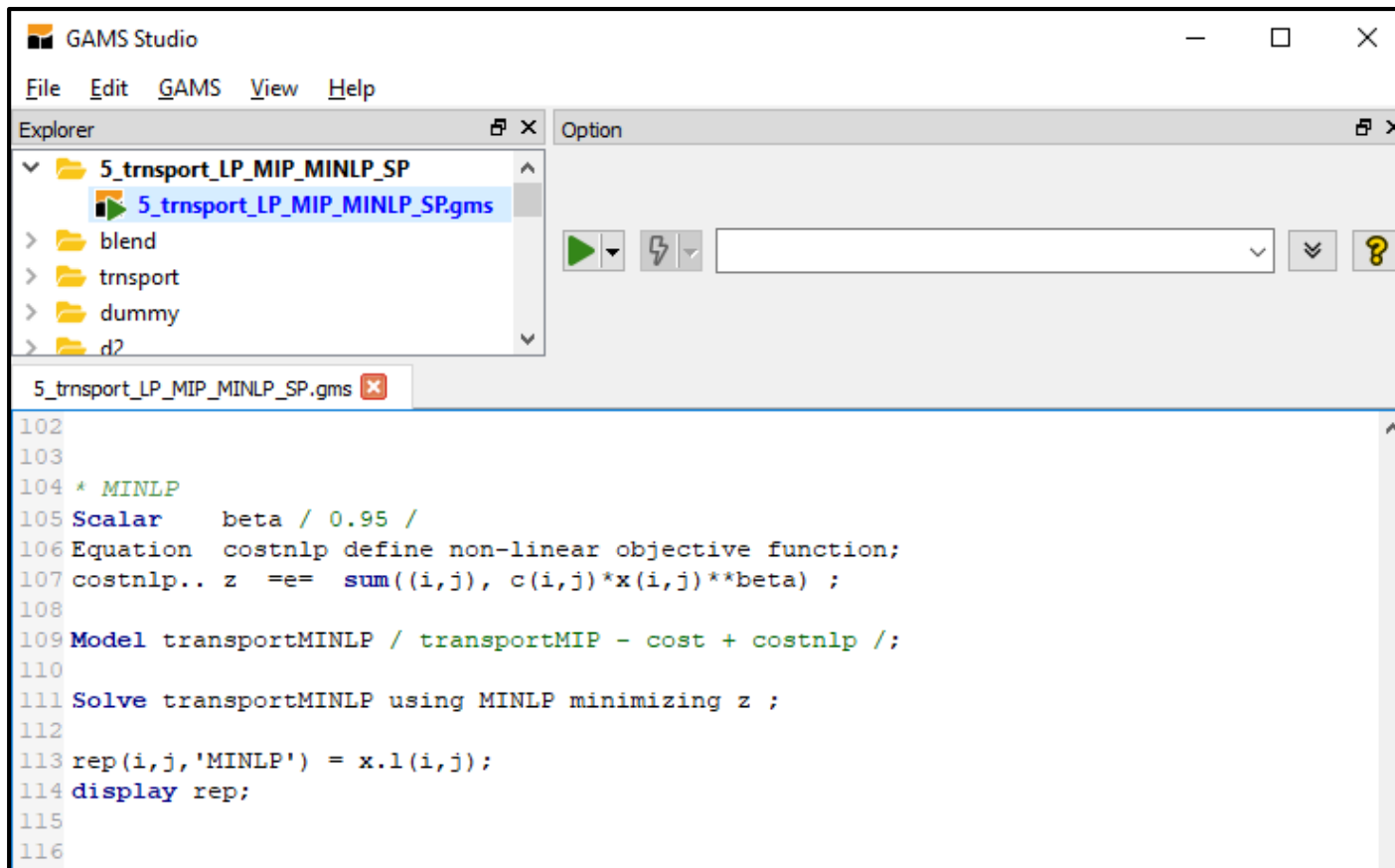
Replace:

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

With

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

MINLP Model: GAMS Syntax



The screenshot displays the GAMS Studio environment. The 'Explorer' pane on the left shows a project structure with folders 'blend', 'transport', 'dummy', and 'd2', and a file '5_transport_LP_MIP_MINLP_SP.gms' selected. The 'Option' pane on the right contains a dropdown menu and a search icon. The main code editor shows the following GAMS code:

```
102
103
104 * MINLP
105 Scalar    beta / 0.95 /
106 Equation  costnlp define non-linear objective function;
107 costnlp.. z  =e= sum((i,j), c(i,j)*x(i,j)**beta) ;
108
109 Model transportMINLP / transportMIP - cost + costnlp /;
110
111 Solve transportMINLP using MINLP minimizing z ;
112
113 rep(i,j,'MINLP') = x.l(i,j);
114 display rep;
115
116
```

Hands-On

MINLP Model: Results

GAMS Studio

File Edit GAMS View Help

Explorer

- 5_trnsport_LP_MIP_MINLP_SP
 - 5_trnsport_LP_MIP_MINLP_SP.gdx
 - 5_trnsport_LP_MIP_MINLP_SP.lst
 - 5_trnsport_LP_MIP_MINLP_SP.gms
- blend
- transport

Option

5_trnsport_LP_MIP_MINLP_SP.gms 5_trnsport_LP_MIP_MINLP_SP.lst 5_trnsport_LP_MIP_MINLP_SP.gdx

```

773 seattle .topeka . . 1.0000 EPS
774 san-diego.new-york . . 1.0000
775 san-diego.chicago . .
776 san-diego.topeka . . 1.0000
777
778
779 **** REPORT SUMMARY :      0      NOOPT
780                          0      INFEASIBLE
781                          0      UNBOUNDED
782                          0      ERRORS
783 GAMS 25.2.0 r67480 ALFA Released 2Aug18 WEX-VS8 x86 32b
784 A Transportation Problem (TRANSPORT,SEQ=1)
785 E x e c u t i o n
786
787
788 ---- 113 PARAMETER rep report parameter
789
790                      LP      MIP      MINLP
791
792 seattle .new-york      50.000
793 seattle .chicago      300.000      300.000      300.000
794 san-diego.new-york      275.000      325.000      325.000
795 san-diego.topeka      275.000      275.000      275.000
796
797
798
799 EXECUTION TIME      =      0.000 SECONDS      3 MB 25
    
```

C:\WC\source\share\Lutz\Presentations\2018_OR_Brussels\Workshop\examples\5_trnsport_LP_MIP_MINLP_SP.lst

GAMS Studio

File Edit GAMS View Help

Explorer

- 5_trnsport_LP_MIP_MINLP_SP
 - 5_trnsport_LP_MIP_MINLP_SP.gdx
 - 5_trnsport_LP_MIP_MINLP_SP.lst
 - 5_trnsport_LP_MIP_MINLP_SP.gms
- blend
- transport

Option

5_trnsport_LP_MIP_MINLP_SP.gms 5_trnsport_LP_MIP_MINLP_SP.lst 5_trnsport_LP_MIP_MINLP_SP.gdx

Entry	Name	Type	Dim	Records	Text
3	a	Parameter	1	2	capacity of plant i i...
4	b	Parameter	1	3	demand at market j...
19	beta	Parameter	0	1	
15	bigM	Parameter	0	1	big M
7	c	Parameter	2	6	transport cost in th...
10	cost	Equation	0	1	define objective fu...
20	costnlp	Equation	0	1	define non-linear o...
5	d	Parameter	2	6	distance in thousan...
12	demand	Equation	1	3	satisfy demand at ...
6	f	Parameter	0	1	freight in dollars pe...
1	i	Set	1	2	canning plants
2	j	Set	1	3	markets
18	maxship	Equation	2	6	maximum shipment
14	minS	Parameter	0	1	minimum shipment
17	minship	Equation	2	6	minimum shipment
13	rep	Parameter	3	10	report parameter
16	ship	Variable	2	6	1 if we ship from i t...
11	supply	Equation	1	2	observe supply limi...

i	j		Value
seattle	new-york	LP	50
seattle	chicago	LP	300
san-diego	new-york	LP	275
san-diego	topeka	LP	275
seattle	chicago	MINLP	300
san-diego	new-york	MINLP	325
san-diego	topeka	MINLP	275
seattle	chicago	MIP	300
san-diego	new-york	MIP	325
san-diego	topeka	MIP	275

Symbol Search: ☐ All Columns ☐ Squeeze Defaults

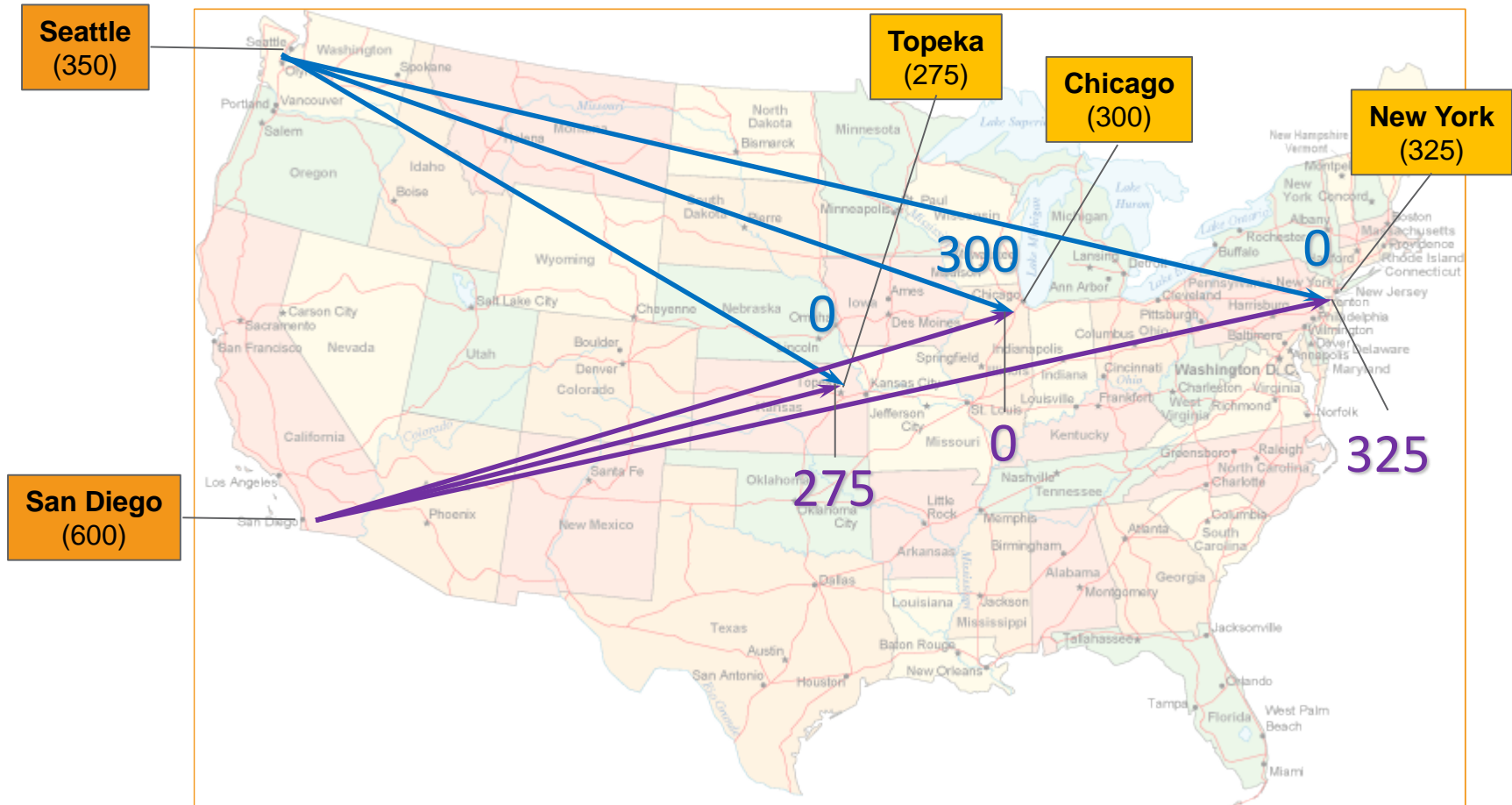
C:\WC\source\share\Lutz\Presentations\2018_OR_Brussels\Workshop\examples\5_trnsport_LP_MIP_MINLP_SP.gdx

MINLP Model: **Solution**

Canning Plants (supply)

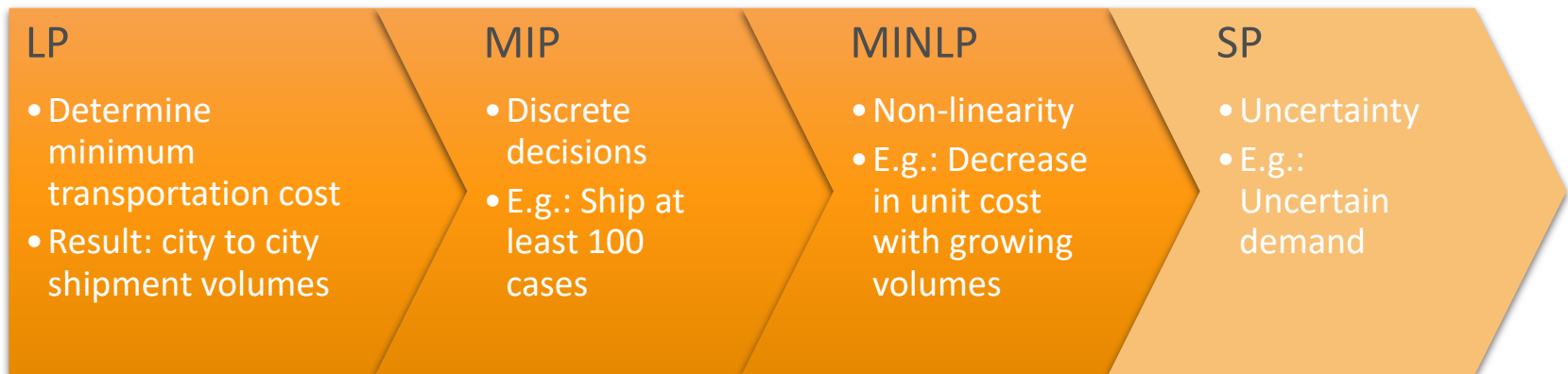
shipments
(Number of cases)

Markets (demand)



Freight: \$90 case / thousand miles

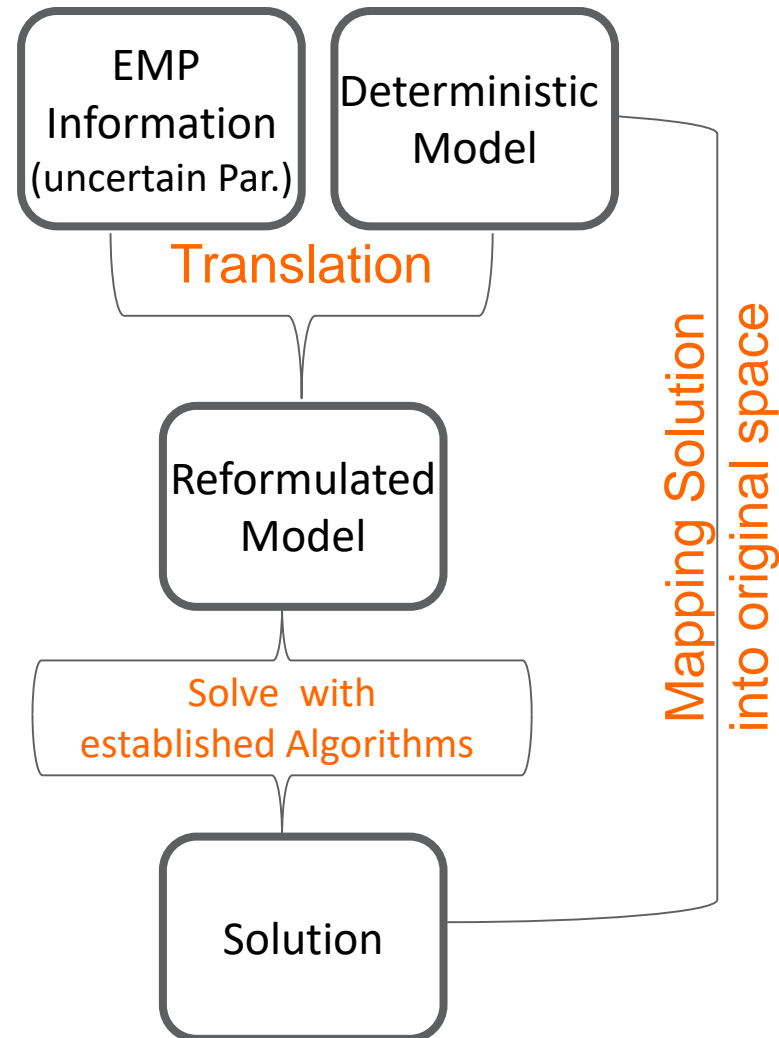
Total cost: \$115,438



Stochastic Programming in GAMS

EMP/SP

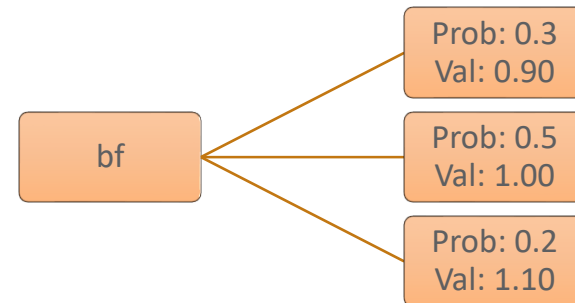
- Simple interface to add uncertainty to existing deterministic models
- (EMP) Keywords to describe uncertainty include: discrete and parametric random variables, stages, chance constraints, Value at Risk, ...
- Available solution methods:
 - Automatic generation of **D**eterministic **E**quivalent (can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



Transport Example – Uncertain Demand

b(j): demand at market j in cases	
new-york	325
chicago	300
topeka	275

Uncertain
demand factor bf



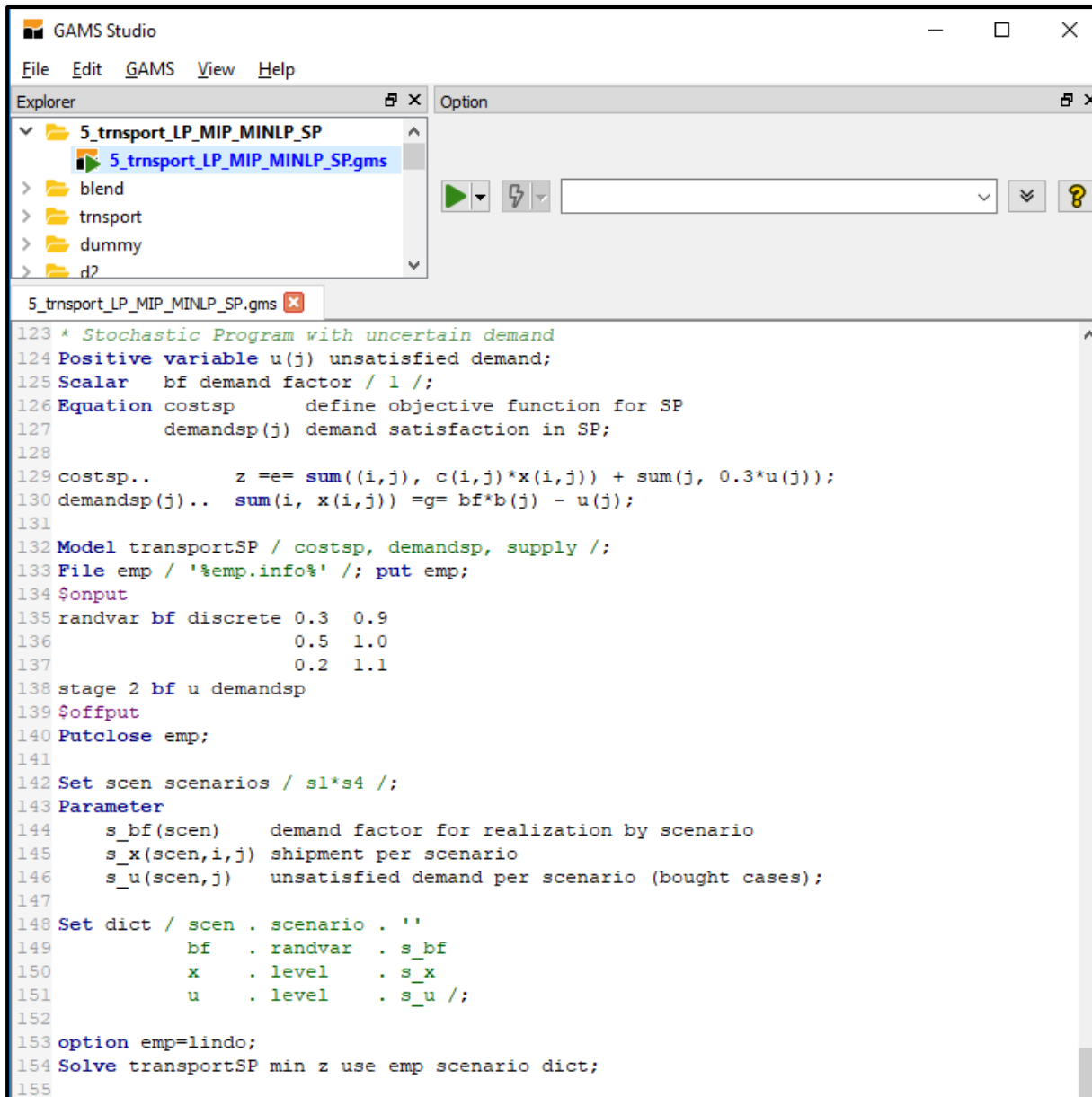
Decisions to make

- First-stage decision: How many units should be shipped “here and now” (without knowing the outcome)
- Second-stage (recourse) decision:
 - How can the model react if we do not ship enough?
 - Penalties for “bad” first-stage decisions, e.g. buy additional cases $u(j)$ at the demand location:

```

costsp .. z =e= sum((i,j), c(i,j)*x(i,j)) + sum(j, 0.3*u(j));
demandsp(j) .. sum(i, x(i,j)) =g= bf*b(j) - u(j) ;
  
```

Uncertain Demand: GAMS Algebra



The screenshot shows the GAMS Studio interface. The Explorer pane on the left displays a project structure with folders 'blend', 'transport', 'dummy', and 'd2', and a file '5_transport_LP_MIP_MINLP_SP.gms'. The main editor pane shows the GAMS code for a stochastic program with uncertain demand. The code defines variables, sets, and equations, and includes a scenario dictionary for solving the model using the LINDO solver.

```
123 * Stochastic Program with uncertain demand
124 Positive variable u(j) unsatisfied demand;
125 Scalar bf demand factor / 1 /;
126 Equation costsp define objective function for SP
127 demandsp(j) demand satisfaction in SP;
128
129 costsp.. z =e= sum((i,j), c(i,j)*x(i,j)) + sum(j, 0.3*u(j));
130 demandsp(j).. sum(i, x(i,j)) =g= bf*b(j) - u(j);
131
132 Model transportSP / costsp, demandsp, supply /;
133 File emp / '%emp.info%' /; put emp;
134 $onput
135 randvar bf discrete 0.3 0.9
136 0.5 1.0
137 0.2 1.1
138 stage 2 bf u demandsp
139 $offput
140 Putclose emp;
141
142 Set scen scenarios / s1*s4 /;
143 Parameter
144 s_bf(scen) demand factor for realization by scenario
145 s_x(scen,i,j) shipment per scenario
146 s_u(scen,j) unsatisfied demand per scenario (bought cases);
147
148 Set dict / scen . scenario . ''
149 bf . randvar . s_bf
150 x . level . s_x
151 u . level . s_u /;
152
153 option emp=lindo;
154 Solve transportSP min z use emp scenario dict;
155
```

Hands-On

Uncertain Demand: Results

GAMS Studio

File Edit GAMS View Help

Explorer

- 5_transport_LP_MIP_MINLP_SP
 - 5_transport_LP_MIP_MINLP_SP.gdx
 - 5_transport_LP_MIP_MINLP_SP.lst
 - 5_transport_LP_MIP_MINLP_SP.gms
- blend
- transport

Option

5_transport_LP_MIP_MINLP_SP.gms 5_transport_LP_MIP_MINLP_SP.lst 5_transport_LP_MIP_MINLP_SP.gdx

Column

- x
- z
- ship
- Model Statistics SOLVE transportMIP ...
- Solution Report SOLVE transportMIP...
- Solve EQU
- Solve VAR
- Execution
- Display
- Equation Listing SOLVE transportMIN...
- Equation
- Column Listing SOLVE transportML...
- Column
- Model Statistics SOLVE transportMIN...
- Solution Report SOLVE transportMIN...
- Solve EQU
- Solve VAR
- Execution
- Display
- Equation Listing SOLVE transportSP ...
- Equation
- Column Listing SOLVE transportSP ...
- Column
- Model Statistics SOLVE transportSP U...
- Solution Report SOLVE transportSP ...
- Solve EQU
- Solve VAR
- Execution
- Display
- s_bf

1031 ---- 156 PARAMETER s_bf demand factor for realization by scenario

1032

1033 s1 0.900, s2 1.000, s3 1.100

1034

1035

1036 ---- 156 PARAMETER s_b demand per scenario

1037

1038 new-york chicago topeka

1039

1040 s1 292.500 270.000 247.500

1041 s2 325.000 300.000 275.000

1042 s3 357.500 330.000 302.500

1043

1044

1045 ---- 156 PARAMETER s_x shipment per scenario

1046

1047 new-york chicago topeka

1048

1049 s1.seattle 50.000 300.000

1050 s1.san-diego 242.500 275.000

1051 s2.seattle 50.000 300.000

1052 s2.san-diego 242.500 275.000

1053 s3.seattle 50.000 300.000

1054 s3.san-diego 242.500 275.000

1055

1056

1057 ---- 156 PARAMETER s_u unsatisfied demand per scenario (bought cases)

1058

1059 new-york chicago topeka

1060

1061 s2 32.500

1062 s3 65.000 30.000 27.500

1086 lines 1031 / 17 RO UTF-8

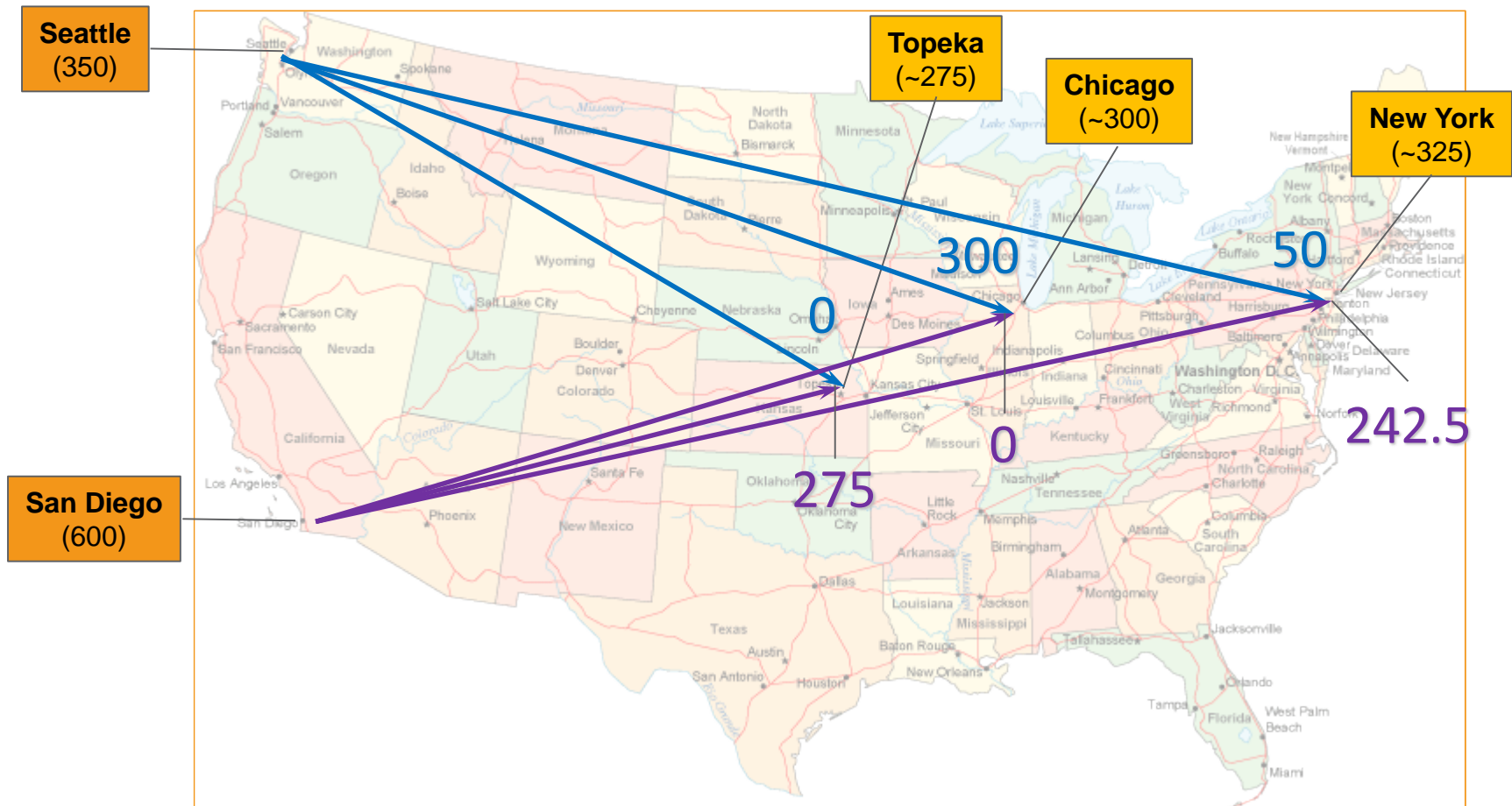
Stochastic Program: **Solution**

Canning Plants (supply)

shipments

(Number of cases)

Markets (demand)



Freight: \$90 case / thousand miles

Total cost: \$158,588

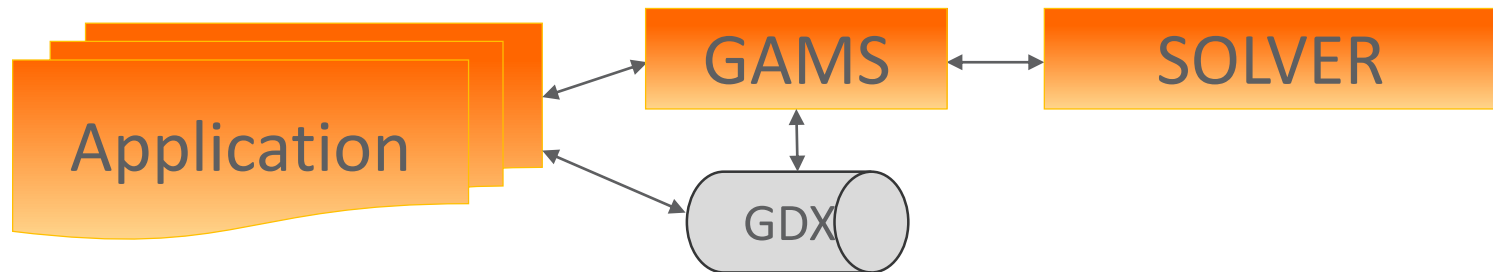
Stochastic Programming in GAMS

- 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:
https://www.gams.com/latest/docs/UG_EMP_SP.html

Outlook: Deployment of GAMS models

- APIs – Application Programming Interfaces to GAMS
- Using R/Shiny to deploy and visualize GAMS models in a Web Interface
- Using GAMS Jupyter Notebooks to tell “optimization stories”

Calling GAMS from your Application



Creating Input for GAMS Model

→ Data handling using **GDX** API

Callout to GAMS

→ GAMS option settings using **Option** API

→ Starting GAMS using **GAMS** API

Reading Solution from GAMS Model

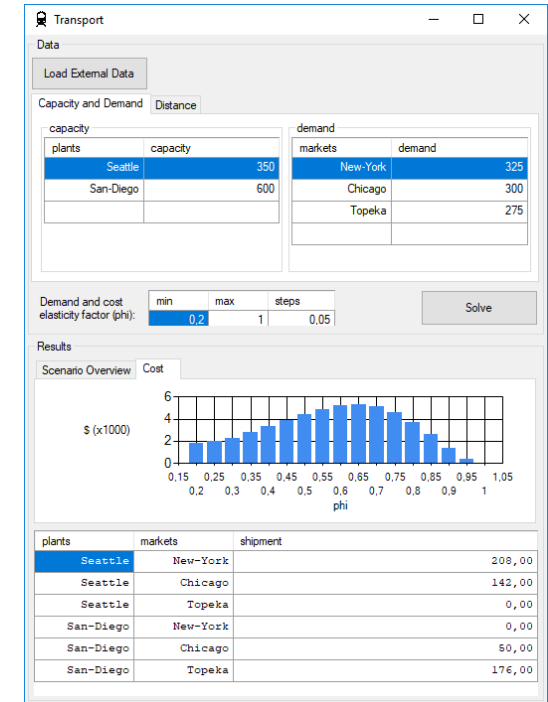
→ Data handling using **GDX** API

Low level APIs → Object Oriented API

- Low level APIs
 - GDX, OPT, GAMSX, GMO, ...
 - High performance and flexibility
 - Automatically generated imperative APIs for several languages (C, C++, C#, Delphi, Java, Python, VBA, ...)
- Object Oriented GAMS API
 - Additional layer on top of the low level APIs
 - Object Oriented
 - Written by hand to meet the specific requirements of different Object Oriented languages

Transport Application GUI Example

- Scenario solves of the transportation problem
- Features:
 - Preparation of input data
 - Loading data from Access file
 - Solving multiple scenarios of a model
 - Displaying results
- Four implementation steps:
 1. Graphical User Interface
 2. Preparation of GAMS model
 3. Implementation of scenario solving using GAMSJob
 4. GAMSModelInstance for performance improvements

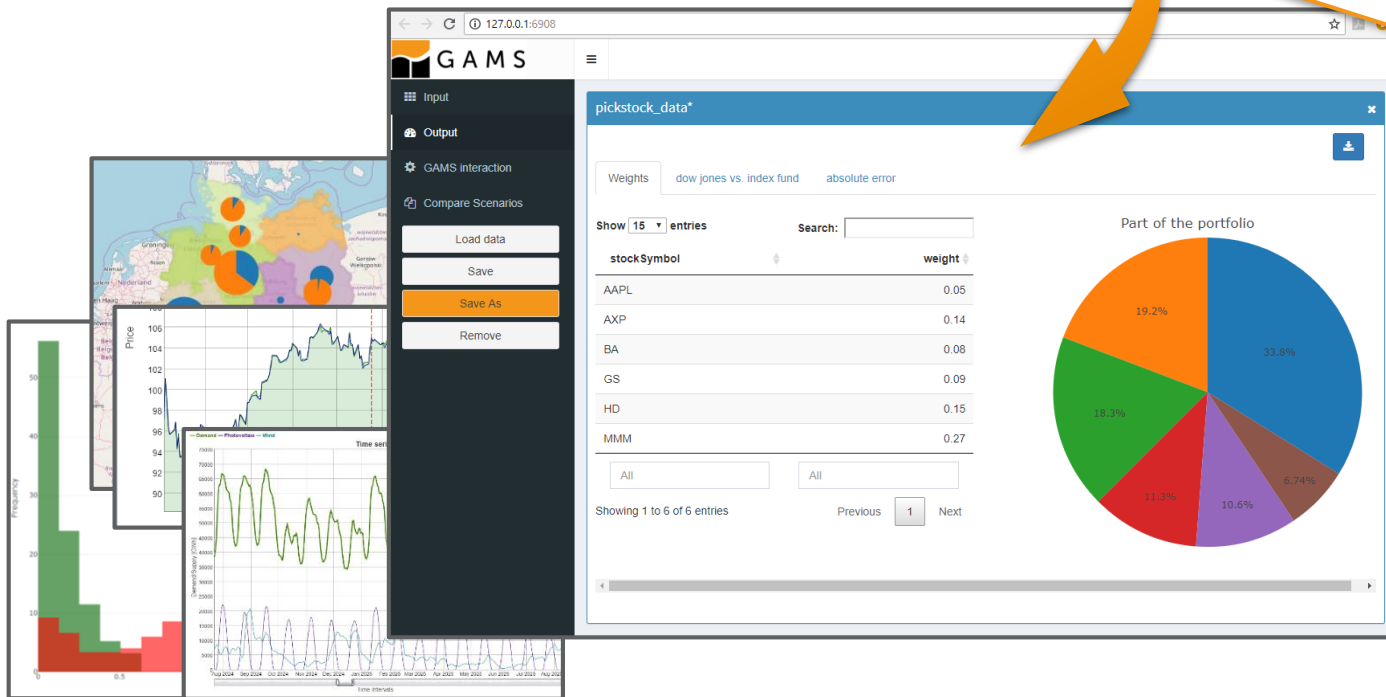


Hands-On

From GAMS Model to Visual Web User Interface

```
54 Equation
55   deffit(date)      'fit to Dow Jones index'
56   defpick(symbol)   'can only use stock if picked'
57   defnumstock       'few stocks allowed'
58   defobj            'absolute violation (L1 norm) from index';
59
60 deffit(ds)..  sum(s, price(ds,s)*w(s)) =e= index(ds) + slpos(ds) - slneg(ds);
61
62 defpick(s)..  w(s) =l= p(s);
63
64 defnumstock.. sum(s, p(s)) =l= maxstock;
65
66 defobj..      obj =e= sum(ds, slpos(ds) + slneg(ds));
```

Currently under
Development



Setting the Model Input Data

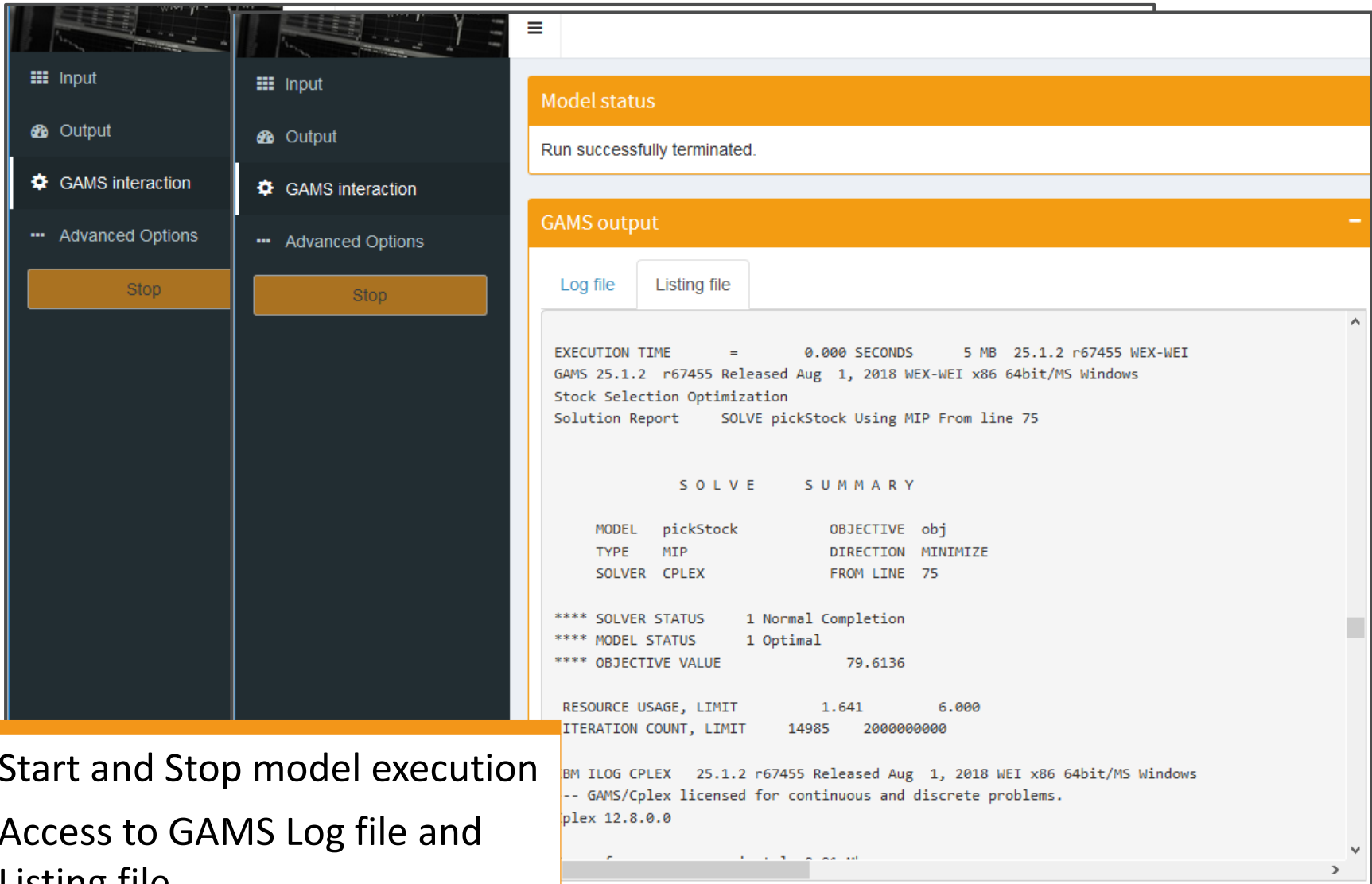
- ✓ Data exchange via local files or database connection
- ✓ Visualization and modification of input data with intuitive controls
- ✓ From a GAMS model to the first interface within minutes
- ✓ Comprehensive configurability

The screenshot displays a software interface with a dark sidebar on the left containing menu items: Input, Output, GAMS interaction, and Advanced Options. The main area features a 'Load data' button and a 'Solve model' button. A modal window titled 'maximum number of stocks to select' is open, showing a slider control ranging from 1 to 30, with the current value set to 2. Below the modal, a table displays data for three stocks: CAT, CSCO, and CVX, with their respective values for 2016-01-04.

Stock	Date	Value
CAT	2016-01-04	67.99
CSCO	2016-01-04	26.41
CVX	2016-01-04	88.85

An 'Import' button is visible at the bottom of the interface.

Communication with GAMS



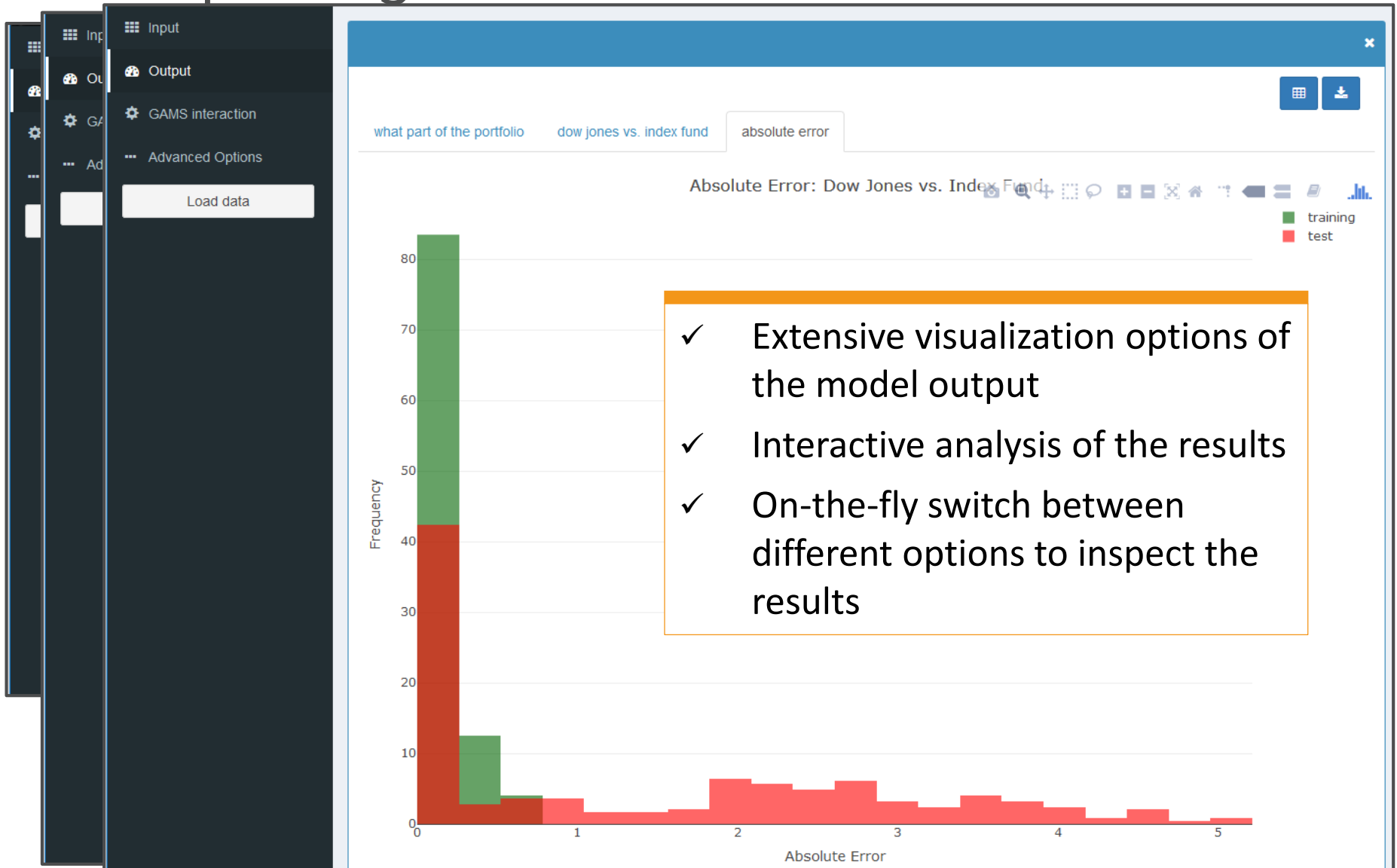
The screenshot displays the GAMS software interface. On the left, a dark sidebar contains a menu with 'Input', 'Output', 'GAMS interaction' (selected), and 'Advanced Options', each with a corresponding icon. Below the menu are two 'Stop' buttons. The main area on the right is divided into sections. The 'Model status' section shows 'Run successfully terminated.' Below it, the 'GAMS output' section has tabs for 'Log file' and 'Listing file'. The 'Log file' tab is active, displaying execution details: 'EXECUTION TIME = 0.000 SECONDS 5 MB 25.1.2 r67455 WEX-WEI', 'GAMS 25.1.2 r67455 Released Aug 1, 2018 WEX-WEI x86 64bit/MS Windows', 'Stock Selection Optimization', and 'Solution Report SOLVE pickStock Using MIP From line 75'. A 'SOLVE SUMMARY' table follows, showing model details and solver status. At the bottom, resource usage and iteration limits are listed.

S O L V E S U M M A R Y			
MODEL	pickStock	OBJECTIVE	obj
TYPE	MIP	DIRECTION	MINIMIZE
SOLVER	CPLEX	FROM LINE	75
**** SOLVER STATUS 1 Normal Completion			
**** MODEL STATUS 1 Optimal			
**** OBJECTIVE VALUE 79.6136			
RESOURCE USAGE, LIMIT		1.641	6.000
ITERATION COUNT, LIMIT		14985	2000000000

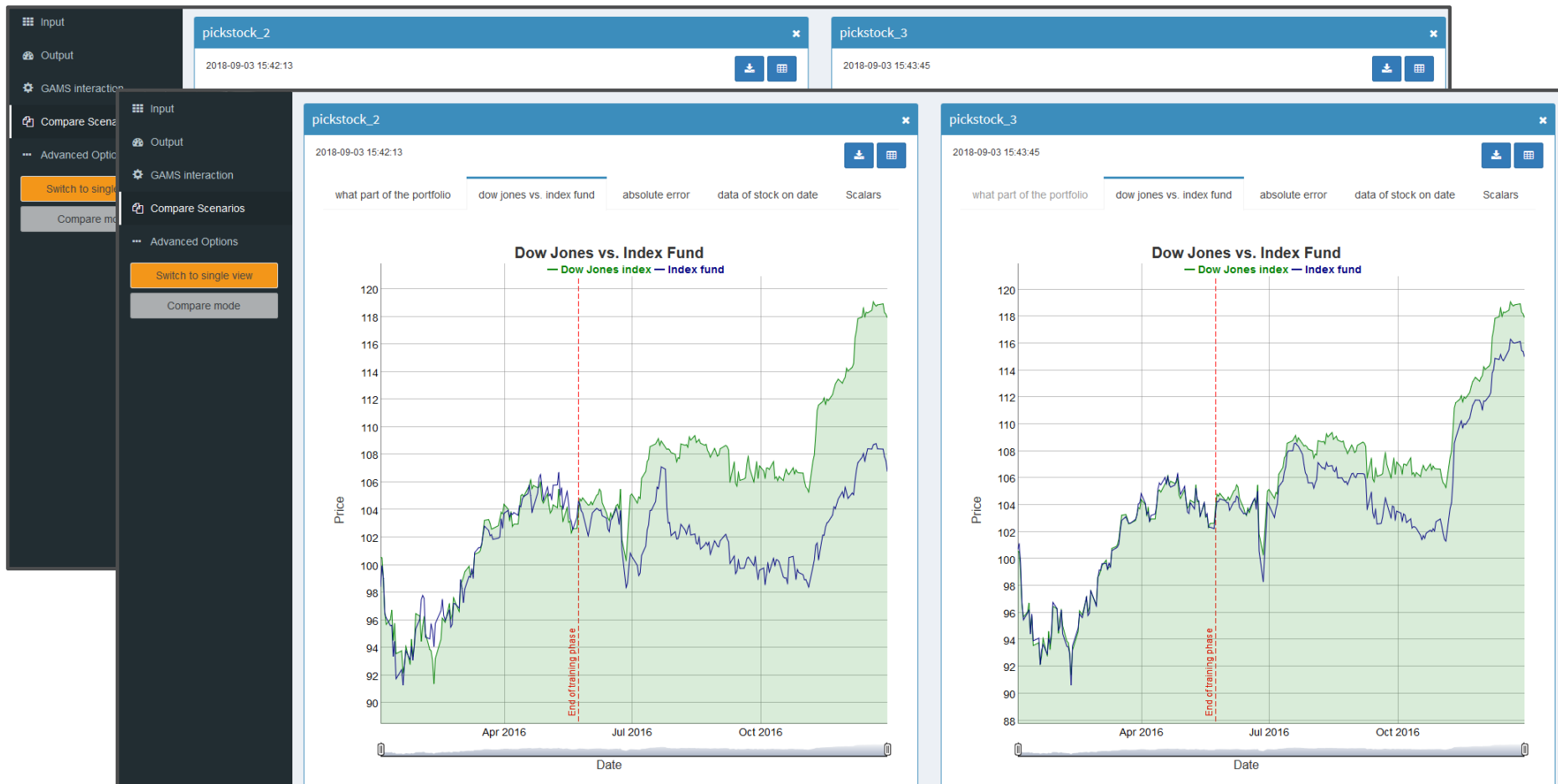
BM ILOG CPLEX 25.1.2 r67455 Released Aug 1, 2018 WEI x86 64bit/MS Windows
-- GAMS/Cplex licensed for continuous and discrete problems.
plex 12.8.0.0

- ✓ Start and Stop model execution
- ✓ Access to GAMS Log file and Listing file

Inspecting the Results



Scenario Management



- ✓ Solve multiple scenarios or load saved data for comparison

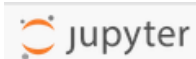
Hands-On

Enhanced Model Deployment in GAMS using R/Shiny

- Application connects with a GAMS model
- User Interface allows
 - ✓ Data exchange via local files or data
 - ✓ Modification of the input data
 - ✓ Extensive visualization options
 - ✓ Comparison of different scenarios
 - ✓ Multi-user support based on Docker technology
 - ✓ User authentication
- Tool with intuitive interface for planners (configuration vs. programming)
- This “product” is currently **under development**. If you are interested in getting involved, please contact support@gams.com

Model Deployment in GAMS
Wednesday, September 12, 4:10 pm - 5:50 pm
(Session WD-16)

Welcome to Jupyter @ GAMS!



Jupyter @ GAMS

Currently under
Development

Sign in

Username:

Password:

Sign In



G A M S

Welcome to Jupyter @ GAMS!

Enter your credentials in order to sign in or contact [GAMS Support](#) for further information.

Getting Started

- [Introduction](#)
- [Milco Example](#)
- [PickStock Example](#)
- [A GAMS Tutorial by Richard E. Rosenthal](#)

Further Help

- [Jupyter Notebook Users Manual \(from Bryn Mawr College\)](#)
- [GAMS World Forum](#)
- [Contact GAMS](#)

GAMS Jupyter Example

[JUPYTER](#)[FAQ](#)

```
In [17]: %%gams
Parameter fund(date) 'Index fund report parameter'; fund(d) = sum(s, price(d, s)*w.l(s));
Parameter error(date) 'Absolute error'; error(d) = abs(index(d)-fund(d));
```

Plotting of the results

```
In [18]: %%gams pull -d fund error
fig, ax = plt.subplots()
index.plot(y="value", ax=ax, xticks=[0, trainingDays, len(date)], yticks=[], label="Dow Jones")
fund.plot(y="value", ax=ax, xticks=[0, trainingDays, len(date)], yticks=[], label="Index Fund")
```



Hands-On

Using GAMS Jupyter Notebooks to tell “optimization stories”

- Runs in a browser/on a server
→ No local installation needed
- Allows to use notebook technology in combination with GAMS
- Notebooks allow to combine GAMS and Python
 - GAMS works great with well structured data and optimization models
 - Python is very rich in features to retrieve, manipulate, and visualize data that comes in all sort of ways
 - ➔ Combining GAMS and Python in a notebook it is relatively easy to tell an optimization story with text, data, graphs, math, and models
- This “product” is currently **under development**.
Give it a try at <https://jupyterhub.gams.com/hub/login>

Outlook: Solving Multiple Models Efficiently

- Solvelink & the GAMS Grid Facility
- GUSS – The Gather-Update-Solve-Scatter

Motivation

- Solving challenging real-world problems often involves the solution of many optimization problems
 - Decomposition Methods
 - Scenario Analysis
 - Heuristics
 - ...
- Such approaches are often chosen, if solving the problem at hand does not work with a single monolithic model, e.g.
 - Due to its size and the required resources (e.g. memory)
 - Due to time restrictions (Problem should be solved in minutes but it takes days)
 - ...

→GAMS Grid Facility

→Gather-Update-Solve-Scatter

Solverlink Option

Controls GAMS function when linking to solve.

```
Model transport /all/ ;
```

```
Option solverlink = {0    %Solverlink.ChainScript%,  
                      1    %Solverlink.CallScript%,  
                      2    %Solverlink.CallModule%,  
                      3    %Solverlink.AsyncGrid%,  
                      4    %Solverlink.AsyncSimulate%,  
                      5    %Solverlink.LoadLibrary%,  
                      6    %solveLink.aSyncThreads%,  
                      7    %solveLink.threadsSimulate%};
```

```
solve transport using lp minimizing z;
```

Solverlink Option – Sequential Solves

- ChainScript [0]: Solver process, GAMS vacates memory
 - + Maximum memory available to solver
 - + protection against solver failure (*hostile* link)
 - swap to disk
- 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

Solverlink Option – Sequential Solves

```
6_transport_solverlink_seq.gms 6_transport_solverlink_seq.lst
62 Model transport /all/ ;
63
64 set s scenarios / s1*s100 /
65     sl solverlink / ChainScript, CallModule, LoadLibrary /;
66 parameter dd(s,i,j) distance by scenario
67             ff(s) freight cost by scenario
68             time(*) time for 100 scenarios
69             sl_val(sl) solverlink value / ChainScript 0, CallModule 2, LoadLibrary 5 /;
70 scalar tmp;
71
72 dd(s,i,j) = uniform(0.9,1.1)*d(i,j);
73 ff(s) = uniform(0.9,1.1)*f;
74
75 option limrow=0, limcol=0, solprint=off;
76
77 * SERIAL SOLVE
78 loop(sl,
79     tmp = jnow;
80     transport.solverlink=sl_val(sl);
81     loop(s,
82         d(i,j) = dd(s,i,j);
83         f = ff(s);
84         Solve transport using lp minimizing z ;
85     );
86     time(sl) = (jnow-tmp)*24*60*60;
87 );
88 display time;
```

```
----- 88 PARAMETER time time for 100 scenarios
ChainScript 17.042, CallModule 2.150, LoadLibrary 0.631
```

Solverlink Option – Asynchronous Solves

- aSyncGrid [3]: GAMS starts the solution and continues in a Grid computing environment
- aSyncThreads [6]: The problem is passed to the solver in core without use of temporary files, GAMS does not wait for the solver to come back

Solverlink Option – Asynchronous Solves

```
7_transport_solverlink_async.gms x 7_transport_solverlink_async.lst x
62 Model transport /all/ ;
63
64 option lp=cplexd;
65 set s scenarios / s1*s100 /
66     sl solverlink / aSyncGrid, aSyncThreads /;
67 parameter dd(s,i,j) distance by scenario
68             ff(s) freight cost by scenario
69             time(*) time for 100 scenarios
70             sl_val(sl) solverlink value / aSyncGrid 3, aSyncThreads 6 /
71             h(s) scenario handle;
72 scalar tmp;
73
74 dd(s,i,j) = uniform(0.9,1.1)*d(i,j);
75 ff(s) = uniform(0.9,1.1)*f;
76 option limrow=0, limcol=0, solprint=silent;
77 * Async SOLVE
78 loop(sl,
79     tmp = jnow;
80     transport.solverlink=sl_val(sl);
81
82     loop(s,
83         d(i,j) = dd(s,i,j);
84         f = ff(s);
85         Solve transport using lp minimizing z ;
86         h(s) = transport.handle; // save instance handle
87     );
88     repeat
89         display$readycollect(h) 'Waiting for next instance to complete';
90         loop(s$handlecollect(h(s)),
91             display$handledelete(h(s)) 'trouble deleting handles';
92             h(s) = 0; // indicate that we have loaded the solution
93         );
94     until card(h) = 0 or timeelapsed > 180; // wait until all models are loaded
95     time(sl) = (jnow-tmp)*24*60*60;
96 );
97 display time;
```

```
----- 97 PARAMETER time time for 100 scenarios
aSyncGrid 2.991, aSyncThreads 0.144
```

Solverlink Option – Asynchronous Solves

- Helpful, if large ratio of solver time / GAMS time

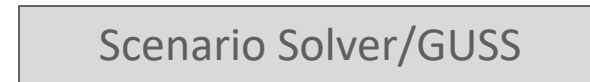
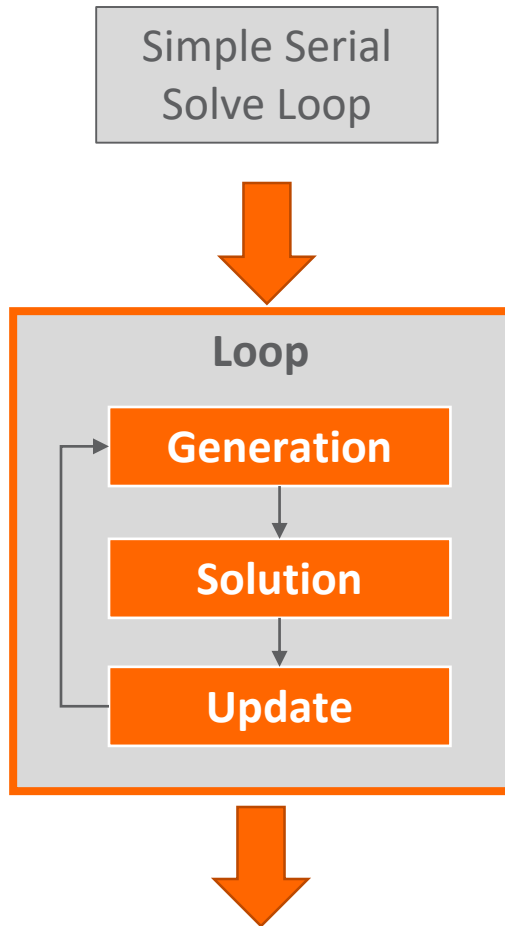
```
7_dicex_solverlink.gms x 7_dicex_solverlink.lst x
95 option mip=cplexd;
96 * SEQUENTIAL SOLVE
97 loop(seq(sl),
98     tmp = jnow;
99     dice2.solverlink=sl_val(sl);
100     loop(s,
101         solve dice2 using mip maximizing wnx;
102     );
103     time(sl) = (jnow-tmp)*24*60*60;
104 );
105
106 * Async SOLVE
107 loop(async(sl),
108     tmp = jnow;
109     dice2.solverlink=sl_val(sl);
110     loop(s,
111         solve dice2 using mip maximizing wnx;
112         h(s) = dice2.handle;    // save instance handle
113     );
114     repeat
115         display$readycollect(h) 'Waiting for next instance to complete';
116         loop(s$hndlc collect(h(s)),
117             display$handledelete(h(s)) 'trouble deleting handles';
118             h(s) = 0;    // indicate that we have loaded the solution
119         );
120     until card(h) = 0 or timeelapsed > 180; // wait until all models are loaded
121     time(sl) = (jnow-tmp)*24*60*60;
122 );
123 option time:3:0:1;
124 display time;
```

----- 124 PARAMETER time
time for 4 scenarios

ChainScript	21.523
CallModule	21.215
LoadLibrary	20.814
aSyncGrid	6.521
aSyncThreads	5.316

GUSS – Gather-Update-Solve-Scatter

aka Scenario Solver



Generates model once and updates the algebraic model **keeping the model “hot”** inside the solver.

GUSS – Gather-Update-Solve-Scatter

aka Scenario Solver

```

8_transport_GUSS_solvelink.gms* x 8_transport_GUSS_solvelink.lst x
69 parameter dd(s,i,j) distance by scenario
70          ff(s) freight cost by scenario
71          time(*) time for 100 scenarios;
72 scalar tmp;
73
74 dd(s,i,j) = uniform(0.9,1.1)*d(i,j);
75 ff(s) = uniform(0.9,1.1)*f;
76 option limrow=0, limcol=0, solprint=off;
77
78 * GUSS
79 transport.solvelink = 0;
80 tmp = jnow;
81 Set mattrib / system.GUSSModelAttributes /;
82 Parameter
83     xxGUSS(s,i,j) collector for level of x
84     srep(s, mattrib) model attributes like modelstat etc
85     o(*) GUSS options / SkipBaseCase 1 /
86
87 Set dict / s . scenario.'
88          o . opt .srep
89          d . param .dd
90          f . param .ff
91          x . level .xxGUSS /
92
93 Solve transport using lp minimizing z scenario dict;
94 time('GUSS') = (jnow-tmp)*24*60*60;
95
96 display time;
97

```

```

----      137 PARAMETER time
time for 100 scenarios

```

ChainScript	4.65
CallModule	1.80
LoadLibrary	0.44
aSyncGrid	3.22
aSyncThreads	3.18
GUSS	0.36

Grid & GUSS – Examples from the model library

- trnsgrid: https://www.gams.com/latest/gamslib_ml/libhtml/gamslib_trnsgrid.html
 - Simple asynchronous solves in a loop, separate collection loop
- tgridmix: https://www.gams.com/latest/gamslib_ml/libhtml/gamslib_tgridmix.html
 - Asynchronous solves in combined submission & collection loop. Keep number of submitted models \leq #threads
- gussgrid: https://www.gams.com/latest/gamslib_ml/libhtml/gamslib_gussgrid.html
 - Asynchronous GUSS-solves in combined submission & collection loop. Keep number of submitted models \leq #threads

GAMS and High Performance Computing

*Thursday, September 13, 9:00 am - 10:15 am
(Session TA-5)*

Thank You

Meet us at the GAMS booth!