Advanced GAMS

Erwin Kalvelagen
Amsterdam Optimization

Names

- Choosing meaningful names is very important for large models
- Choose shorter names for much used symbols
- Don’t overdo: very long names add clutter

Set Implementation

- GAMS stores a single large pool of set elements. This pool can be displayed as follows:
  - Alias (*,pool);
  - Display pool;

```
----    21 SET pool  Aliased with *
seattle  ,   san-diego,   new-york ,   chicago  ,   topeka
```

Set Ordering

```
set
i /a,c/
j /b,c/
;
display j;
```

```
---- 5 SET j
c, b
```

The ordering looks strange

```
alias (pool,*);
display pool;
```

```
---- 8 SET pool Aliased with *
a, c, b
```

Reason: the ordering in the pool (aka universe)

Note: set j is called unordered

Set Ordering 2

```
set
t1 /2001*2005/
t2 /2000*2006/;
display t2;
```

```
---- 5 SET t2
2001, 2002, 2003, 2004, 2005, 2000, 2006
```

This is probably unwanted. Also parameters based on t2 will be displayed reordered.

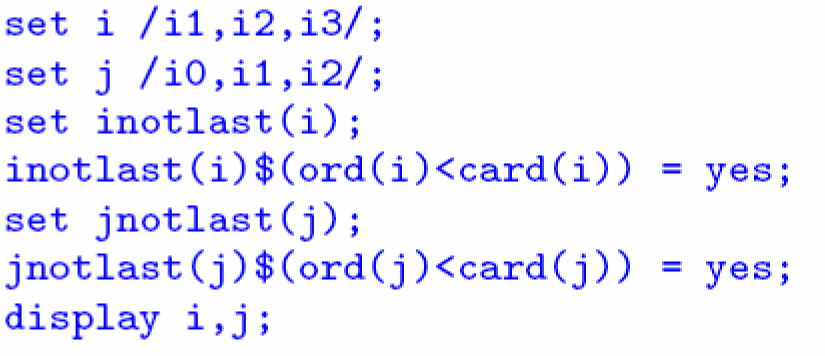
Quick solution: use a dummy set

```
set
dummy /2000*2010/
t1 /2001*2005/
t2 /2000*2006/;
display t2;
```

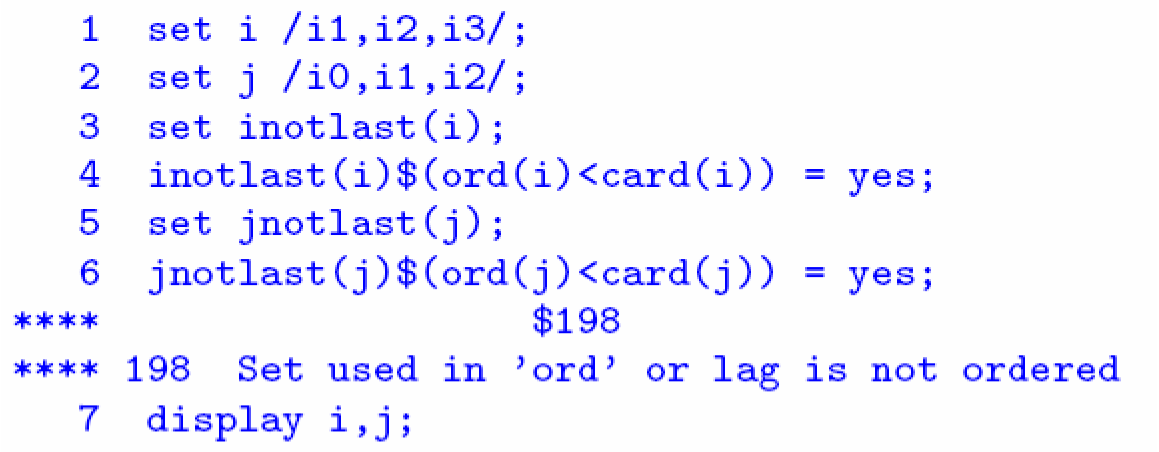
```
----      6 SET t2
2000, 2001, 2002, 2003, 2004, 2005, 2006
```

Unordered sets

- Cannot use ord() on unordered sets



```
set i /i1,i2,i3/;
set j /i0,i1,i2/;
set inotlast(i);
inotlast(i)$(ord(i)<card(i)) = yes;
set jnotlast(j);
jnotlast(j)$(ord(j)<card(j)) = yes;
display i,j;
```



```
   1  set i /i1,i2,i3/;
   2  set j /i0,i1,i2/;
   3  set inotlast(i);
   4  inotlast(i)$(ord(i)<card(i)) = yes;
   5  set jnotlast(j);
   6  jnotlast(j)$(ord(j)<card(j)) = yes;
****                   $198
**** 198  Set used in ’ord’ or lag is not ordered
   7  display i,j;
```

Unordered Sets (2)

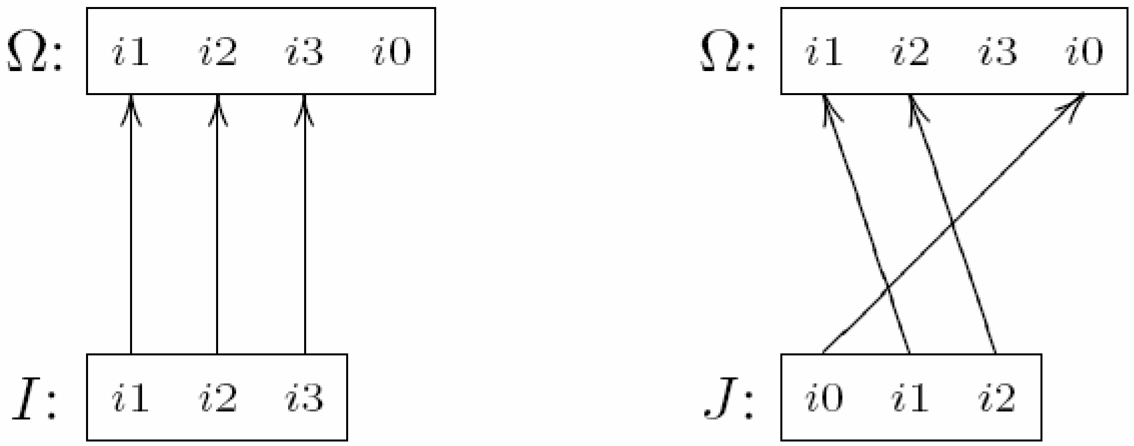


Figure 1.5: Ordered and unordered set

Pool of uels

Ordered if arrows don't cross

Explanatory text set elements

- Limited value, not visible in display

```
ccc(jb)  /
      "FOR Reserve"              015
      "CCC Inventory"             016
  /
```

015, 016 is explanatory text for set elements. I use it mainly for self-documentation.

Find the source of the error

```
  19
  20
  21    Sets
  22         i   canning, plants    / seattle, san-diego /
  23         j   markets           / new-york, chicago, topeka / ;
  24
  25    Parameters
  26
  27         a(i)  capacity of plant i in cases
  28           /     seattle     350
****                       $361
**** 361  Values for domain 1 are unknown - no checking possible
  29                 san-diego   600   /
  30
  31         b(j)  demand at market j in cases
  32           /     new-york    325
  33                 chicago     300
  34                 topeka      275   / ;
```

2-pass execution

- 1: Compilation
  - Parser
  - Handle declarations
  - Do all $commands
- 2: Execution
  - Execute compiled code

```
if(1,
$set name "hello"
else
$set name "world"
);

display "%name%";
```

Example

- Put is execution time, $include run-time
- $onecho compile time, solve execution time

```
   1  file f /x.inc/;
   2  putclose f "Display 'hello';"/;
   3  $include x.inc;
****                      $282
**** 282  Unable to open include file
   4
```

```
Model m/all/;
m.optfile=1;
Solve m minimizing z using lp;

$onecho > cplex.opt
lpmethod 4
$offecho
```

Special Values

- Unexpected behavior

```
scalar a /EPS/;
scalar b;
b$a=1/a;
display a,b;
```

Division by zero because EPS is not the same as zero.

```
scalar s /NA/;
s$(s>1) = 3.14;
display s;
```

Returns 3.14. Because any operation on NA returns NA, $(s>1) is $(NA) which is evaluated as true.

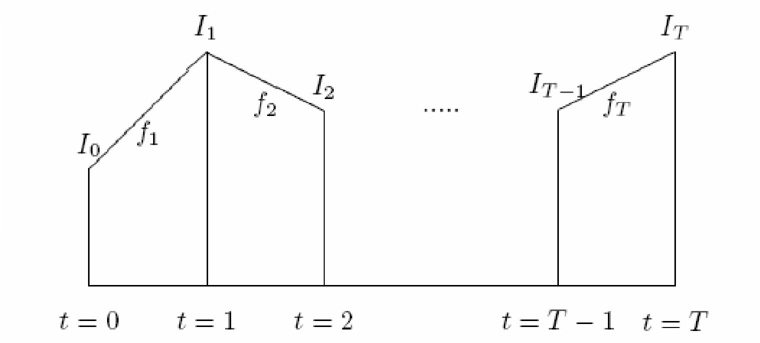
Leads and Lags

- Inventory balance

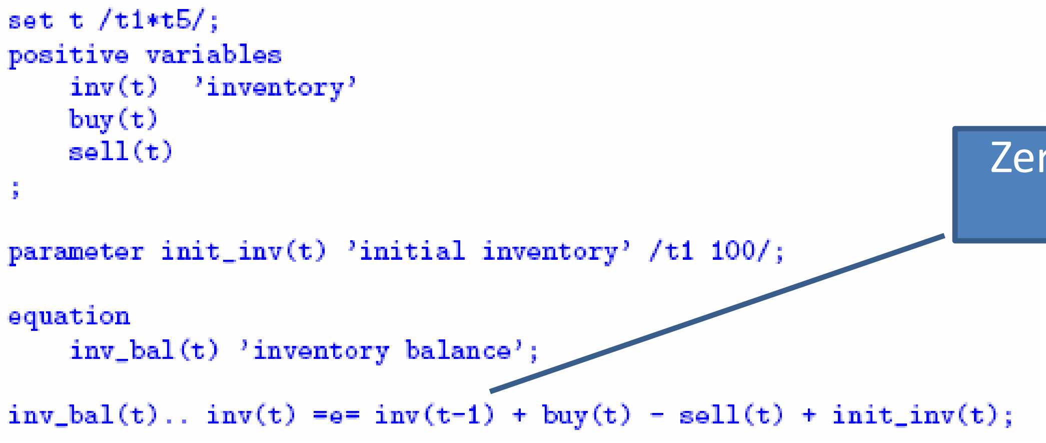
Always check equation listing in listing file

$$inventory_t = inventory_{t-1} - outflow_t + inflow_t$$

$$inventory_{t+1} = inventory_t - outflow_t + inflow_t.$$



Stock and flow variables

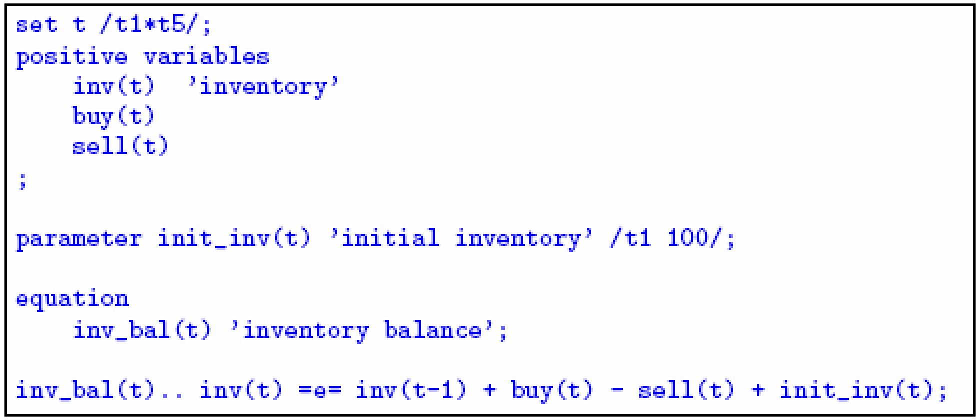


```
set t /t1*t5/;
positive variables
    inv(t)  'inventory'
    buy(t)
    sell(t)
;

parameter init_inv(t) 'initial inventory' /t1 100/;

equation
    inv_bal(t) 'inventory balance';

inv_bal(t).. inv(t) =e= inv(t-1) + buy(t) - sell(t) + init_inv(t);
```

Zero if t-1 outside domain

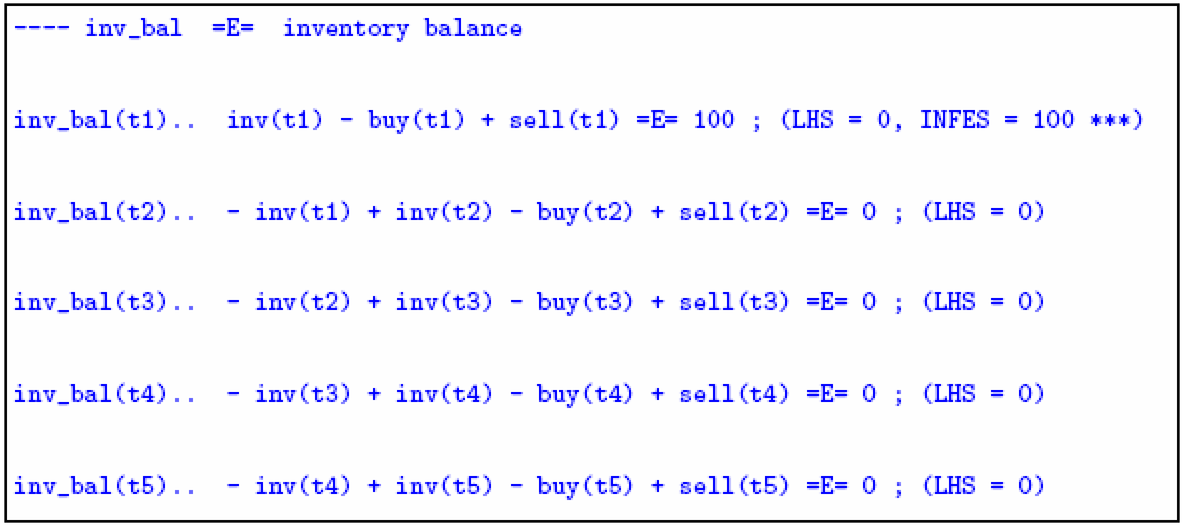
Trick: initial inventory is parameter indexed by t but only contains a value in first period.

Leads and lags (2)



```
set t /t1*t5/;
positive variables
    inv(t)  'inventory'
    buy(t)
    sell(t)
;

parameter init_inv(t) 'initial inventory' /t1 100/;

equation
    inv_bal(t) 'inventory balance';

inv_bal(t).. inv(t) =e= inv(t-1) + buy(t) - sell(t) + init_inv(t);
```

Equation listing shows expanded result.

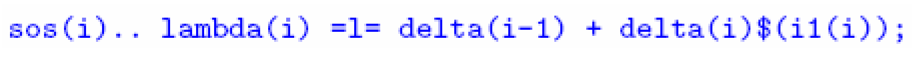


```
---- inv_bal  =E=  inventory balance


inv_bal(t1)..  inv(t1) - buy(t1) + sell(t1) =E= 100 ; (LHS = 0, INFES = 100 ***)


inv_bal(t2)..  - inv(t1) + inv(t2) - buy(t2) + sell(t2) =E= 0 ; (LHS = 0)


inv_bal(t3)..  - inv(t2) + inv(t3) - buy(t3) + sell(t3) =E= 0 ; (LHS = 0)


inv_bal(t4)..  - inv(t3) + inv(t4) - buy(t4) + sell(t4) =E= 0 ; (LHS = 0)


inv_bal(t5)..  - inv(t4) + inv(t5) - buy(t5) + sell(t5) =E= 0 ; (LHS = 0)
```

Debugging constraints

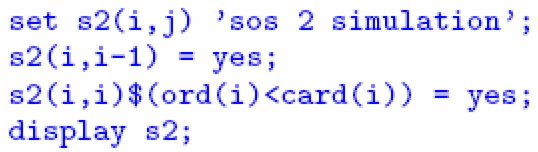
- Equations are difficult to debug
- Only once you have a complete running model you can use the equation and column listing
- Try to move as much complexity out of the equations to sets
- Then you can use DISPLAY to debug in an early development stage

Debugging constraints (2)

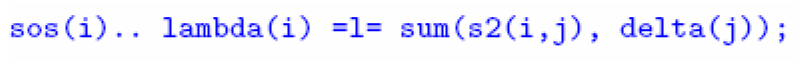


```
sos(i).. lambda(i) =l= delta(i-1) + delta(i)$(i1(i));
```

To understand this we could have used:

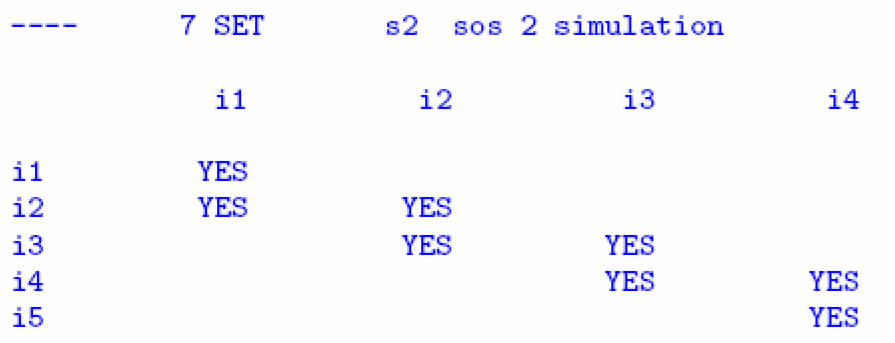


```
set s2(i,j) 'sos 2 simulation';
s2(i,i-1) = yes;
s2(i,i)$(ord(i)<card(i)) = yes;
display s2;
```



```
sos(i).. lambda(i) =l= sum(s2(i,j), delta(j));
```

Now we can look at s2 before developing the rest of the model:



```
----      7 SET       s2  sos 2 simulation

              i1          i2          i3          i4

i1           YES
i2           YES         YES
i3                       YES         YES
i4                                   YES         YES
i5                                               YES
```

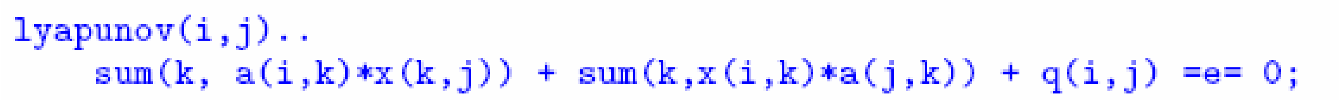
Stop GAMS in the middle

- $stop
  - Not inside loop
- Abort$1 “Stopped”;
  - Works inside loop

Matrix expression

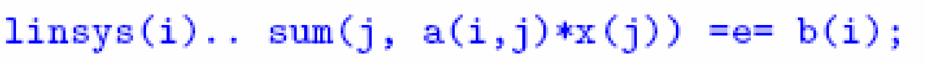
- Gams does not have matrix expressions
- Example (Lyapunov Equation):

$$AX + XA^T + Q = 0$$



```
lyapunov(i,j)..
    sum(k, a(i,k)*x(k,j)) + sum(k,x(i,k)*a(j,k)) + q(i,j) =e= 0;
```

- There is no inv(A). Often not needed: write Ax=b as equation:

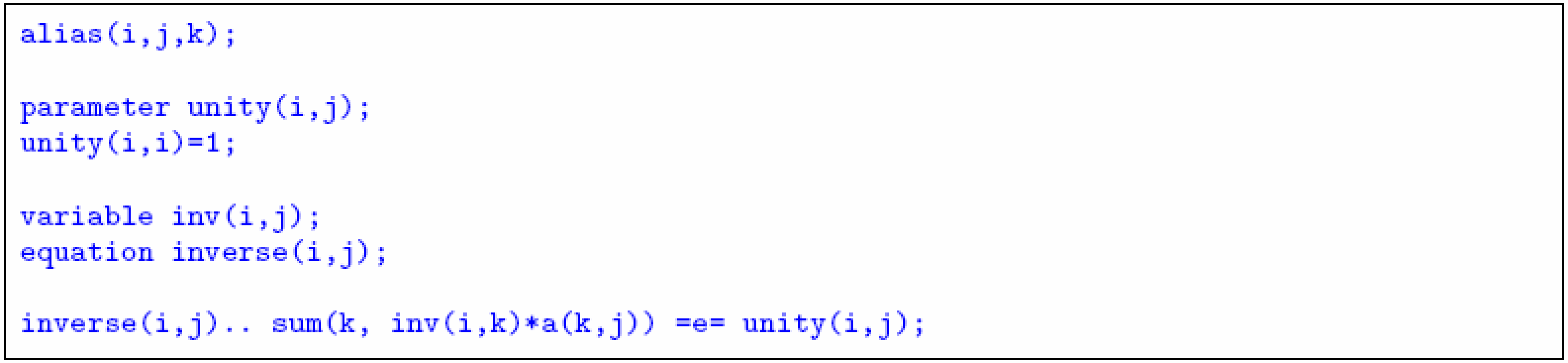


```
linsys(i).. sum(j, a(i,j)*x(j)) =e= b(i);
```

- This will automatically be solved for *x*

Matrix Expressions

- If you really want the inverse of a matrix:



```
alias(i,j,k);

parameter unity(i,j);
unity(i,i)=1;

variable inv(i,j);
equation inverse(i,j);

inverse(i,j).. sum(k, inv(i,k)*a(k,j)) =e= unity(i,j);
```

i.e. solve for A^{-1}

$$A^{-1}A = I$$

Tracing a loop

This does not work:

```
set t /2000*2005/;

loop(t,
  display t;
);
```

```
----      4 SET t

2000,    2001,    2002,    2003,    2004,    2005


----      4 SET t

2000,    2001,    2002,    2003,    2004,    2005


----      4 SET t

2000,    2001,    2002,    2003,    2004,    2005


----      4 SET t

2000,    2001,    2002,    2003,    2004,    2005


----      4 SET t

2000,    2001,    2002,    2003,    2004,    2005


----      4 SET t

2000,    2001,    2002,    2003,    2004,    2005
```

Tracing a loop (2)

Alternative 1

```
set
  t /2000*2005/
  tnow(t) /2000/
;


loop(t,
   tnow(tnow) = no;
   tnow(t) = yes;
   display tnow;
);
```

Alternative 2

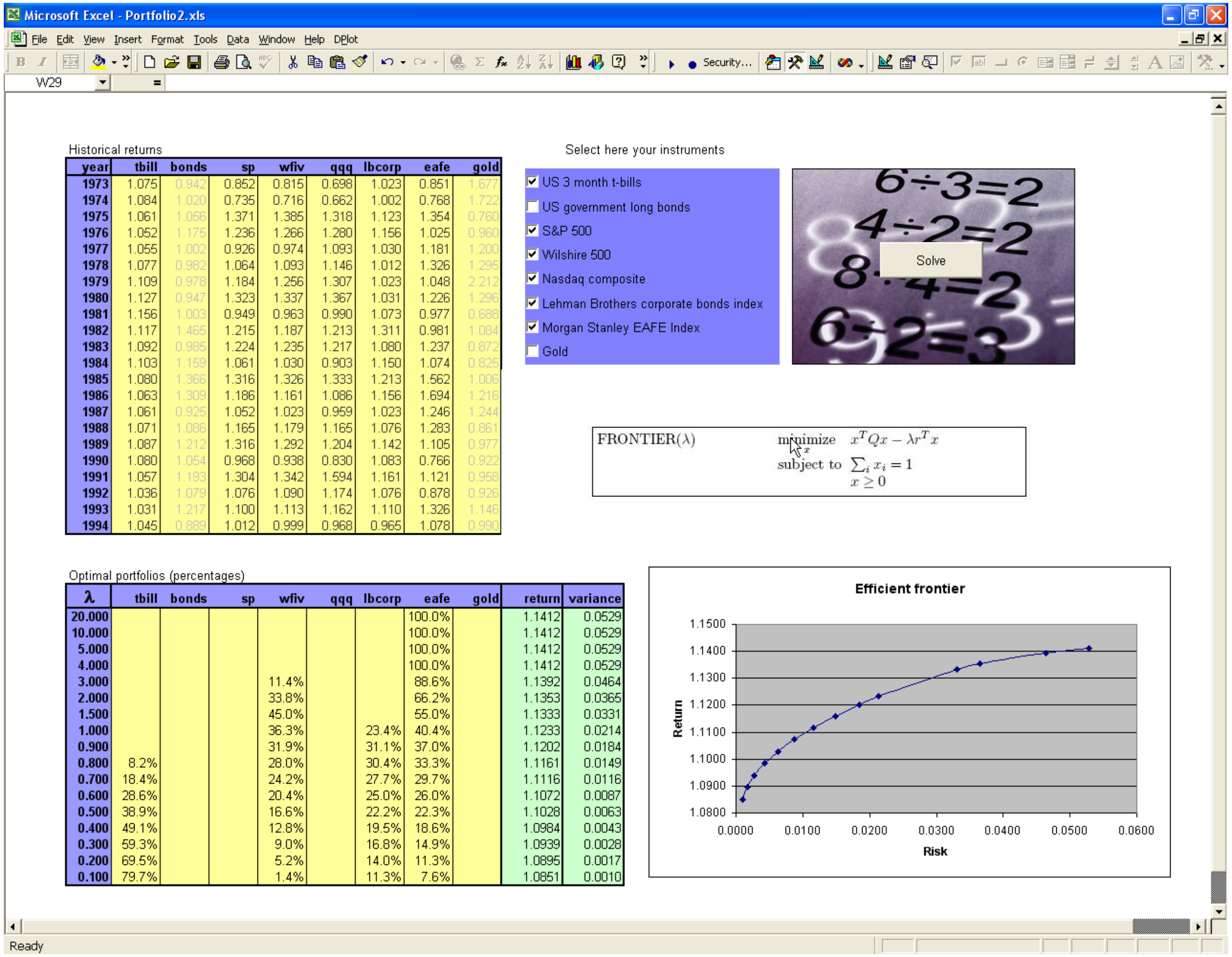
```
set
  t /2000*2005/
  tnow(t) /2000/
;

alias(t,tt);

loop(t,
   display tnow;
   tnow(tt)=tnow(tt-1);
);
```

Alternative 3

```
set
  t /2000*2005/
  tnow(t)
;


loop(t,
   tnow(t) = yes;
   display tnow;
   tnow(t) = no;
);
```

```
----          9 SET tnow

2000

----          9 SET tnow

2001

----          9 SET tnow

2002

----          9 SET tnow

2003

----          9 SET tnow

2004

----          9 SET tnow

2005
```

Packaging in Excel



Historical returns

year	tbill	bonds	sp	wfiv	qqq	lbcorp	eafe	gold
1973	1.075	0.942	0.852	0.815	0.698	1.023	0.851	1.677
1974	1.084	1.020	0.735	0.716	0.662	1.002	0.768	1.722
1975	1.061	1.056	1.371	1.385	1.318	1.123	1.354	0.760
1976	1.052	1.175	1.236	1.266	1.280	1.156	1.025	0.960
1977	1.055	1.002	0.926	0.974	1.093	1.030	1.181	1.200
1978	1.077	0.982	1.064	1.093	1.146	1.012	1.326	1.295
1979	1.109	0.978	1.184	1.256	1.307	1.023	1.048	2.212
1980	1.127	0.947	1.323	1.337	1.367	1.031	1.226	1.296
1981	1.156	1.003	0.949	0.963	0.990	1.073	0.977	0.688
1982	1.117	1.465	1.215	1.187	1.213	1.311	0.981	1.084
1983	1.092	0.985	1.224	1.235	1.217	1.080	1.237	0.872
1984	1.103	1.159	1.061	1.030	0.903	1.150	1.074	0.825
1985	1.080	1.366	1.316	1.326	1.333	1.213	1.562	1.006
1986	1.063	1.309	1.186	1.161	1.086	1.156	1.694	1.216
1987	1.061	0.925	1.052	1.023	0.959	1.023	1.246	1.244
1988	1.071	1.086	1.165	1.179	1.165	1.076	1.283	0.861
1989	1.087	1.212	1.316	1.292	1.204	1.142	1.105	0.977
1990	1.080	1.054	0.968	0.938	0.830	1.083	0.766	0.922
1991	1.057	1.193	1.304	1.342	1.594	1.161	1.121	0.958
1992	1.036	1.079	1.076	1.090	1.174	1.076	0.878	0.926
1993	1.031	1.217	1.100	1.113	1.162	1.110	1.326	1.146
1994	1.045	0.889	1.012	0.999	0.968	0.965	1.078	0.990

Select here your instruments

- ☑ US 3 month t-bills
- ☐ US government long bonds
- ☑ S&P 500
- ☑ Wilshire 500
- ☑ Nasdaq composite
- ☑ Lehman Brothers corporate bonds index
- ☑ Morgan Stanley EAFE Index
- ☐ Gold

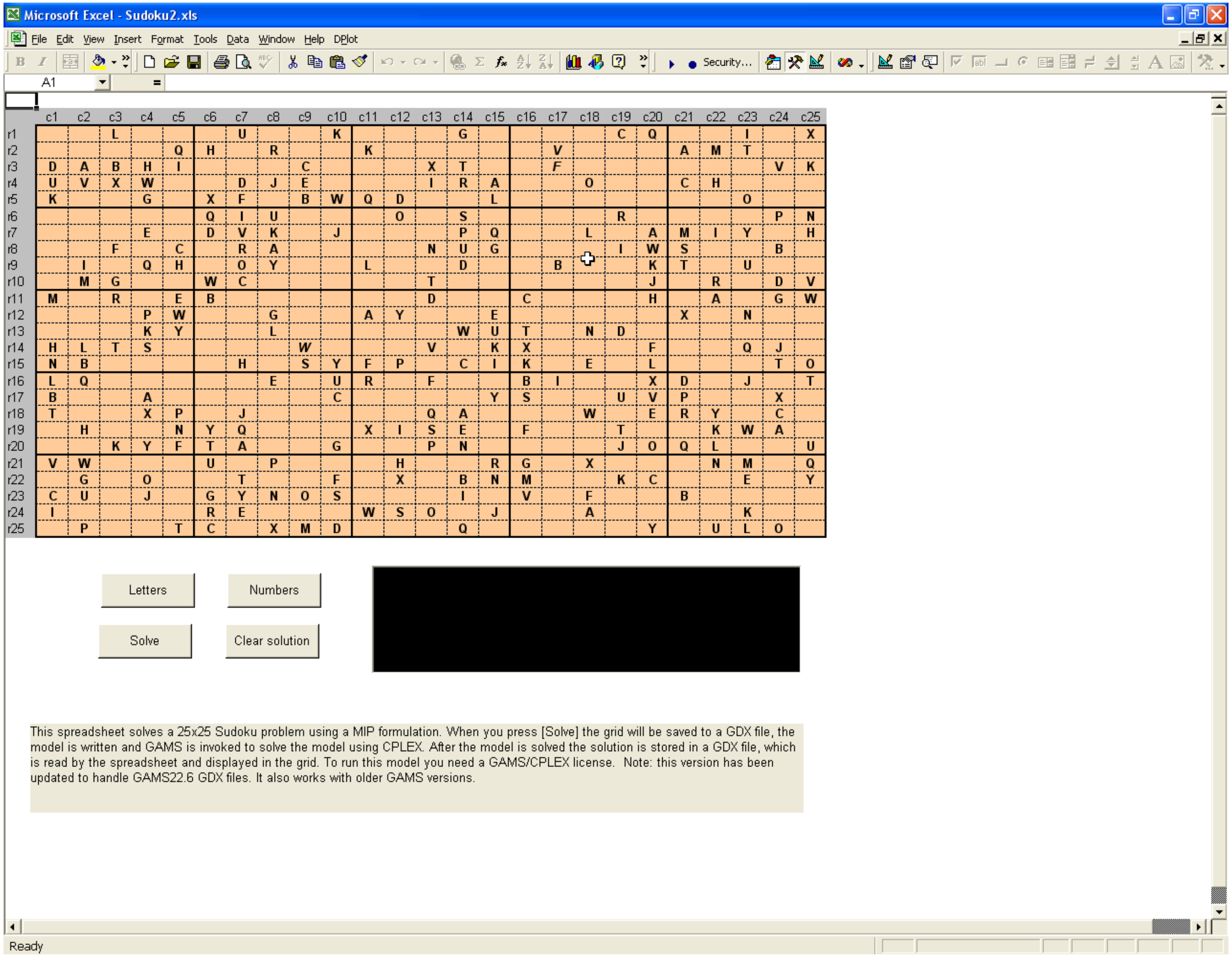
FRONTIER(λ)

$$\begin{aligned} \text{minimize}_x \quad & x^T Q x - \lambda r^T x \\ \text{subject to} \quad & \textstyle\sum_i x_i = 1 \\ & x \geq 0 \end{aligned}$$

Optimal portfolios (percentages)

λ	tbill	bonds	sp	wfiv	qqq	lbcorp	eafe	gold	return	variance
20.000							100.0%		1.1412	0.0529
10.000							100.0%		1.1412	0.0529
5.000							100.0%		1.1412	0.0529
4.000							100.0%		1.1412	0.0529
3.000				11.4%			88.6%		1.1392	0.0464
2.000				33.8%			66.2%		1.1353	0.0365
1.500				45.0%			55.0%		1.1333	0.0331
1.000				36.3%		23.4%	40.4%		1.1233	0.0214
0.900				31.9%		31.1%	37.0%		1.1202	0.0184
0.800	8.2%			28.0%		30.4%	33.3%		1.1161	0.0149
0.700	18.4%			24.2%		27.7%	29.7%		1.1116	0.0116
0.600	28.6%			20.4%		25.0%	26.0%		1.1072	0.0087
0.500	38.9%			16.6%		22.2%	22.3%		1.1028	0.0063
0.400	49.1%			12.8%		19.5%	18.6%		1.0984	0.0043
0.300	59.3%			9.0%		16.8%	14.9%		1.0939	0.0028
0.200	69.5%			5.2%		14.0%	11.3%		1.0895	0.0017
0.100	79.7%			1.4%		11.3%	7.6%		1.0851	0.0010

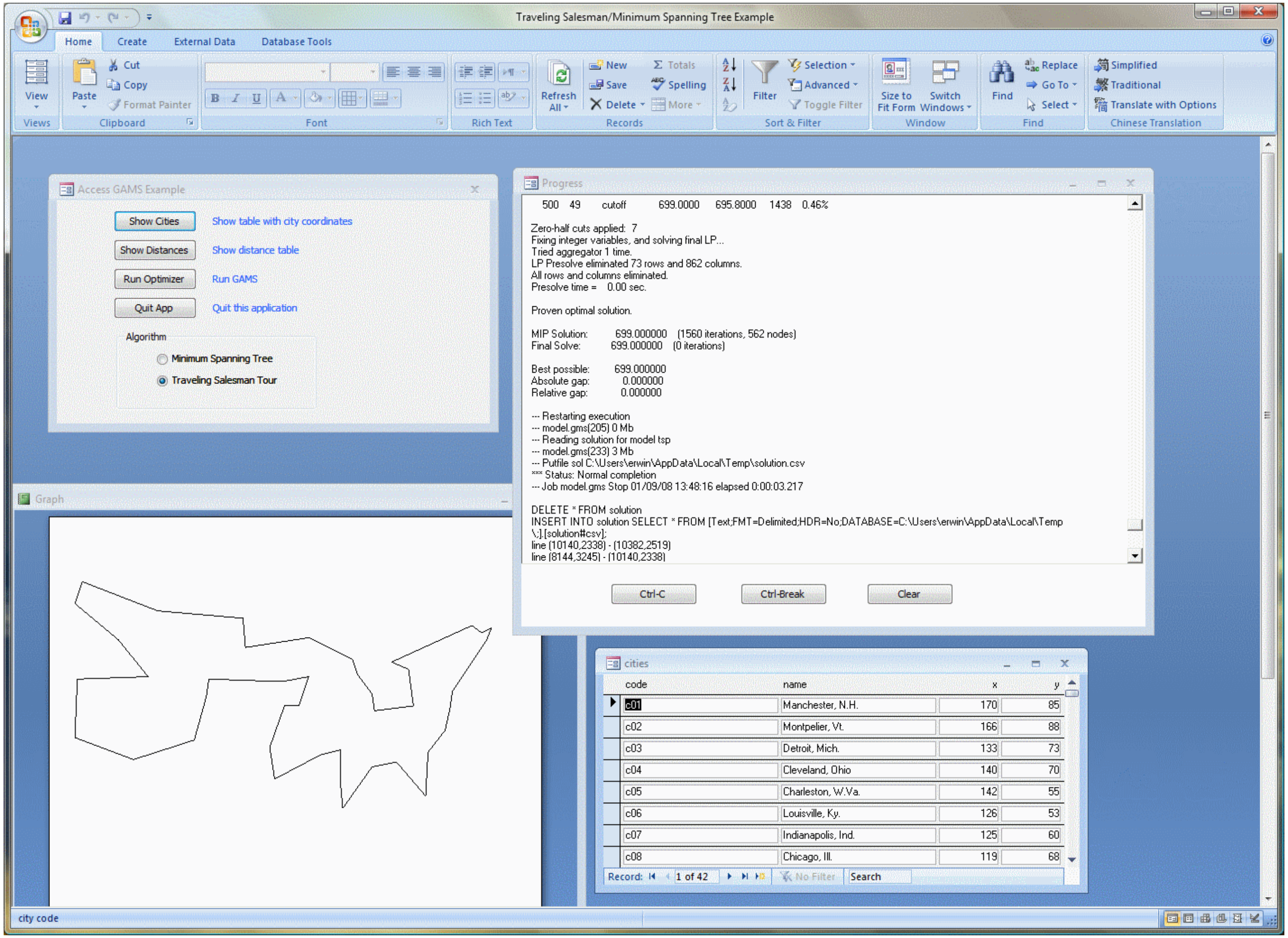
Packaging in Excel (2)


Microsoft Excel - Sudoku2.xls
File Edit View Insert Format Tools Data Window Help DPlot
Security...
A1
=
Letters
Numbers
Solve
Clear solution
Ready

	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10	c11	c12	c13	c14	c15	c16	c17	c18	c19	c20	c21	c22	c23	c24	c25
r1			L				U			K				G					C	Q			I		X
r2					Q	H		R			K						V				A	M	T		
r3	D	A	B	H	I				C				X	T			*F*							V	K
r4	U	V	X	W			D	J	E				I	R	A			O			C	H			
r5	K			G		X	F		B	W	Q	D			L								O		
r6						Q	I	U				O		S					R					P	N
r7				E		D	V	K		J				P	Q			L		A	M	I	Y		H
r8			F		C		R	A					N	U	G				I	W	S			B	
r9		I		Q	H		O	Y			L			D			B			K	T		U		
r10		M	G			W	C						T							J		R		D	V
r11	M		R		E	B							D			C				H		A		G	W
r12				P	W			G			A	Y			E						X		N		
r13				K	Y			L						W	U	T		N	D						
r14	H	L	T	S					*W*				V		K	X				F			Q	J	
r15	N	B					H		S	Y	F	P		C	I	K		E		L				T	O
r16	L	Q						E		U	R		F			B	I			X	D		J		T
r17	B			A						C					Y	S			U	V	P			X	
r18	T			X	P		J						Q	A				W		E	R	Y		C	
r19		H			N	Y	Q				X	I	S	E		F			T			K	W	A	
r20			K	Y	F	T	A			G			P	N					J	O	Q	L			U
r21	V	W				U		P				H			R	G		X				N	M		Q
r22		G		O			T			F		X		B	N	M			K	C			E		Y
r23	C	U		J		G	Y	N	O	S				I		V		F			B				
r24	I					R	E				W	S	O		J			A					K		
r25		P			T	C		X	M	D				Q						Y		U	L	O	

This spreadsheet solves a 25x25 Sudoku problem using a MIP formulation. When you press [Solve] the grid will be saved to a GDX file, the model is written and GAMS is invoked to solve the model using CPLEX. After the model is solved the solution is stored in a GDX file, which is read by the spreadsheet and displayed in the grid. To run this model you need a GAMS/CPLEX license. Note: this version has been updated to handle GAMS22.6 GDX files. It also works with older GAMS versions.

Packaging in Access



Show Cities — Show table with city coordinates

Show Distances — Show distance table

Run Optimizer — Run GAMS

Quit App — Quit this application

Algorithm

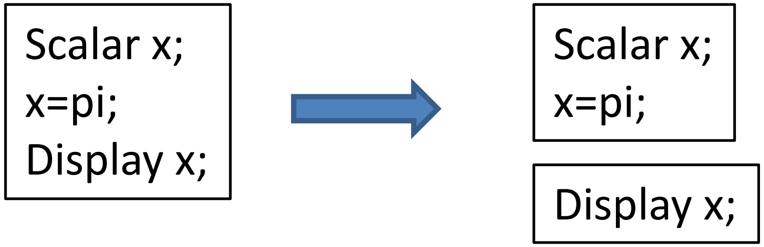
○ Minimum Spanning Tree

● Traveling Salesman Tour

```
   500   49      cutoff        699.0000    695.8000    1438   0.46%

Zero-half cuts applied:  7
Fixing integer variables, and solving final LP...
Tried aggregator 1 time.
LP Presolve eliminated 73 rows and 862 columns.
All rows and columns eliminated.
Presolve time =    0.00 sec.

Proven optimal solution.

MIP Solution:        699.000000    (1560 iterations, 562 nodes)
Final Solve:         699.000000    (0 iterations)

Best possible:       699.000000
Absolute gap:          0.000000
Relative gap:          0.000000

--- Restarting execution
--- model.gms(205) 0 Mb
--- Reading solution for model tsp
--- model.gms(233) 3 Mb
--- Putfile sol C:\Users\erwin\AppData\Local\Temp\solution.csv
*** Status: Normal completion
--- Job model.gms Stop 01/09/08 13:48:16 elapsed 0:00:03.217

DELETE * FROM solution
INSERT INTO solution SELECT * FROM [Text;FMT=Delimited;HDR=No;DATABASE=C:\Users\erwin\AppData\Local\Temp
\;].[solution#csv];
line (10140,2338) - (10382,2519)
line (8144,3245) - (10140,2338)
```

code	name	x	y
c01	Manchester, N.H.	170	85
c02	Montpelier, Vt.	166	88
c03	Detroit, Mich.	133	73
c04	Cleveland, Ohio	140	70
c05	Charleston, W.Va.	142	55
c06	Louisville, Ky.	126	53
c07	Indianapolis, Ind.	125	60
c08	Chicago, Ill.	119	68

Save/restart



```
Scalar x;
x=pi;
Display x;
```

```
Scalar x;
x=pi;
```

```
Gams m1.gms s=savem1
```

```
Display x;
```

```
Gams m2.gms r=savem1
```

Hide code: G00 file

- Compile only, save to restart file
  - Main.gms a=c s=xxx
- Read restart, execute only
  - Dummy.gms r=xxx

Hide Code Complete Example

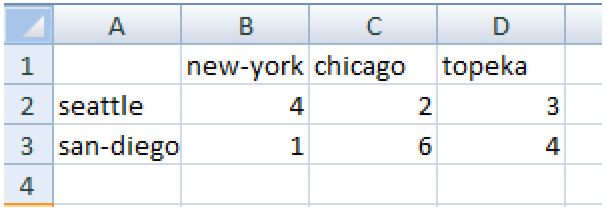
In trnsport.gms change how c(i,j) is calculated:

```
Parameter c(i,j)  transport cost in thousands of dollars per case ;
 execute_load  "trnsport.gdx", c;
```

Compile as: > gams trnsport a=c s=tmodel

Give tmodel.g00 to user

User creates spreadsheet:



	A	B	C	D
1		new-york	chicago	topeka
2	seattle	4	2	3
3	san-diego	1	6	4
4				

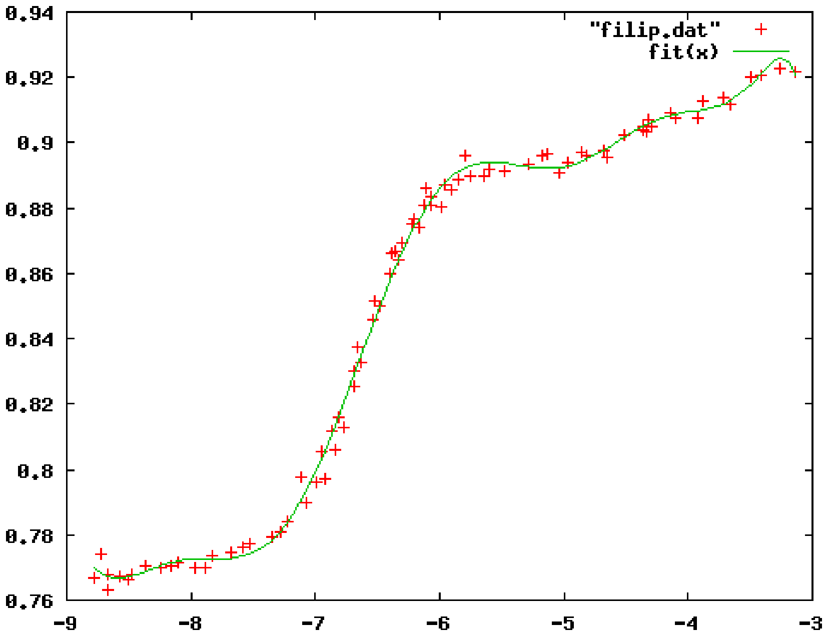
User model:

```
$log *****************************
$log *   POLYSYS                  *
$log *****************************
$call 'gdxxrw.exe i=trnsport.xlsx o=trnsport.gdx par=c rdim=1 cdim=1'
```

Run as: > gams user.gms r=tmodel

```
--- Job user.gms Start 07/14/08 23:18:55
GAMS Rev 227  Copyright (C) 1987-2008 GAMS Development. All rights reserved
Licensee: Erwin Kalvelagen                          G070509/0001CE-WIN
        GAMS Development Corporation                         DC4572
--- Starting continued compilation
****************************
*  POLYSYS               *
****************************
--- user.gms(4) 2 Mb
--- call gdxxrw.exe i=trnsport.xlsx o=trnsport.gdx par=c rdim=1 cdim=1

GDXXRW          May  1, 2008 22.7.1 WIN 4378.4700 VIS x86/MS Windows
Input file : C:\projects\Polysys\VersionJuly\model based on Energy Reserve\gams\restart\trnsport.xlsx
Output file: C:\projects\Polysys\VersionJuly\model based on Energy Reserve\gams\restart\trnsport.gdx
Total time = 359 Ms
--- user.gms(4) 2 Mb
--- Starting execution: elapsed 0:00:00.374
--- user.gms(51) 3 Mb
--- GDXin=C:\projects\Polysys\VersionJuly\model based on Energy Reserve\gams\restart\trnsport.gdx
--- Generating LP model transport
--- user.gms(72) 3 Mb
---   6 rows  7 columns  19 non-zeroes
--- Executing CPLEX: elapsed 0:00:00.380
…..
```

LS Solver



```
variables
   constant    'estimate constant term coefficient'
   income      'estimate income coefficient'
   sse         'sum of squared errors'
;

equations
   fit(i)     'the linear model'
   obj        'objective'

;

obj..     sse =n= 0;
fit(i)..  data(i,'expenditure') =e= constant + income*data(i,'income');

option lp=ls;
model ols1 /obj,fit/;
solve ols1 minimizing sse using lp;

display constant.l, income.l, sse.l;
```

Regression Output

```
=======================================================================
  Least Square Solver V2
  Erwin Kalvelagen, Amsterdam Optimization Modeling Group
  www.amsterdamoptimization.com
=======================================================================

        Parameter      Estimate     Std. Error        t value       Pr(>|t|)
          b('j0')  -1.46749E+03   2.98085E+02  -4.92307E+00   5.34685E-06 ***
          b('j1')  -2.77218E+03   5.59780E+02  -4.95227E+00   4.78349E-06 ***
          b('j2')  -2.31637E+03   4.66478E+02  -4.96566E+00   4.54488E-06 ***
          b('j3')  -1.12797E+03   2.27204E+02  -4.96458E+00   4.56374E-06 ***
          b('j4')  -3.54478E+02   7.16479E+01  -4.94751E+00   4.87122E-06 ***
          b('j5')  -7.51242E+01   1.52897E+01  -4.91338E+00   5.54762E-06 ***
          b('j6')  -1.08753E+01   2.23691E+00  -4.86176E+00   6.74865E-06 ***
          b('j7')  -1.06221E+00   2.21624E-01  -4.79286E+00   8.75365E-06 ***
          b('j8')  -6.70191E-02   1.42364E-02  -4.70760E+00   1.20510E-05 ***
          b('j9')  -2.46781E-03   5.35617E-04  -4.60741E+00   1.74863E-05 ***
         b('j10')  -4.02963E-05   8.96633E-06  -4.49418E+00   2.65146E-05 ***
     ---
     Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

Estimation statistics:
Cases: 82  Parameters: 11  Residual sum of squares:   7.95851E-04
Residual standard error:   3.34801E-03   on  71 degrees of freedom
Multiple R-squared:   9.96727E-01  Adjusted R-squared:   9.96266E-01
F statistic:   2.16244E+03 on 10 and 71 DF, p-value:   0.00000E+00

DLL version: _GAMS_GDX_237_2007-01-09
GDX file: ls.gdx
```