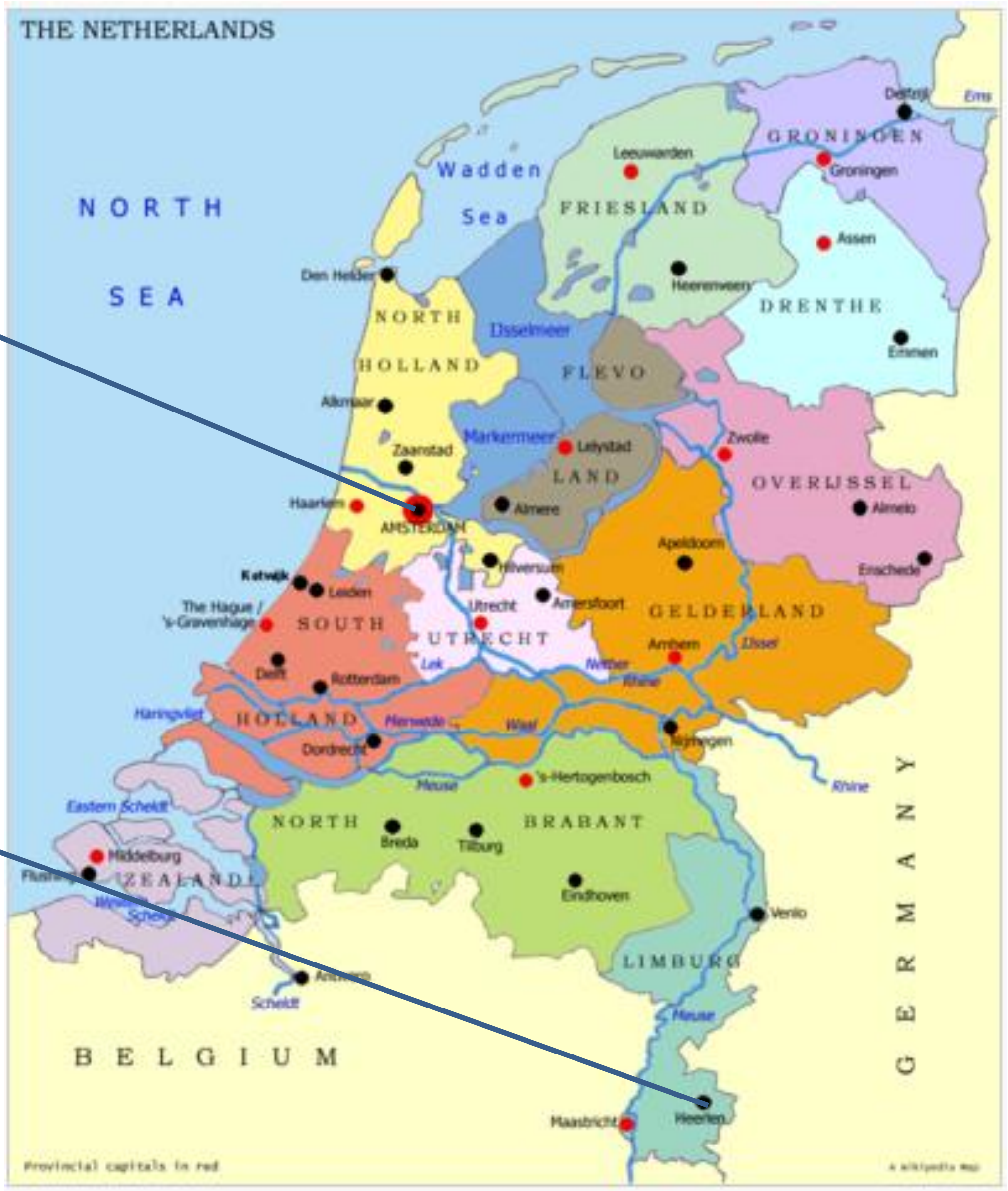
GAMS/GDX

Paul van der Eijk (paul@gams.com)
Erwin Kalvelagen (erwin@gams.com)

Amsterdam

Heerlen



Agenda

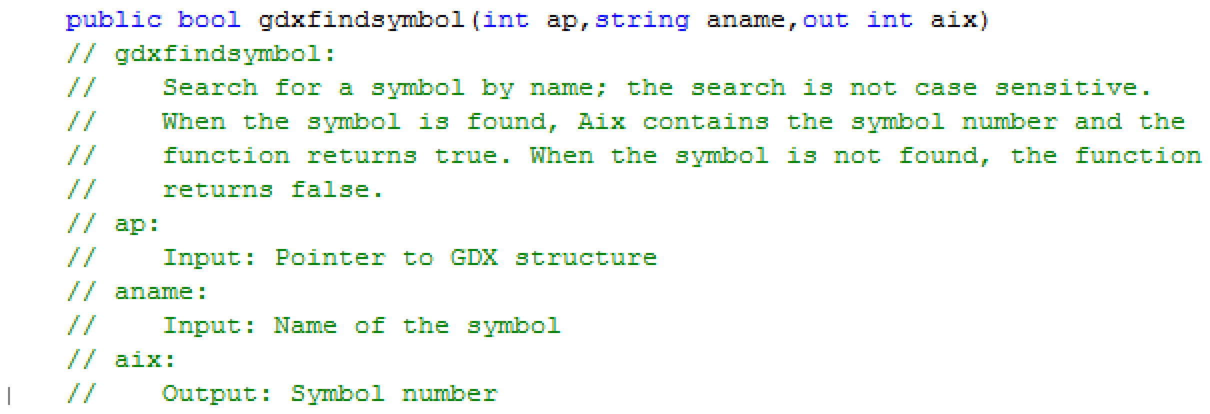
- Wednesday 9:00-12:00,
  - Erwin: Intro, GAMS/GDX
- Wednesday 13:30-16:00
  - Paul: Gdxviewer, Excel
- Thursday 9:00-12:00
  - Erwin: Advanced GAMS/GDX, Databases, Applications
- Thursday 13:30-16:00
  - Paul: Tools, Charting, Other subjects (let us know)

GDX

- Gams Data eXchange
  - API: Application Programming Interface
    - I.e. a set of functions a programmer can use
  - File format/type: gdx file (*.gdx)
    - Binary file
- Only for GAMS **Data**
  - Parameters, Sets, Variables, Equation values
  - No Symbolic Equations
  - Limited Meta Information (domains recently added)

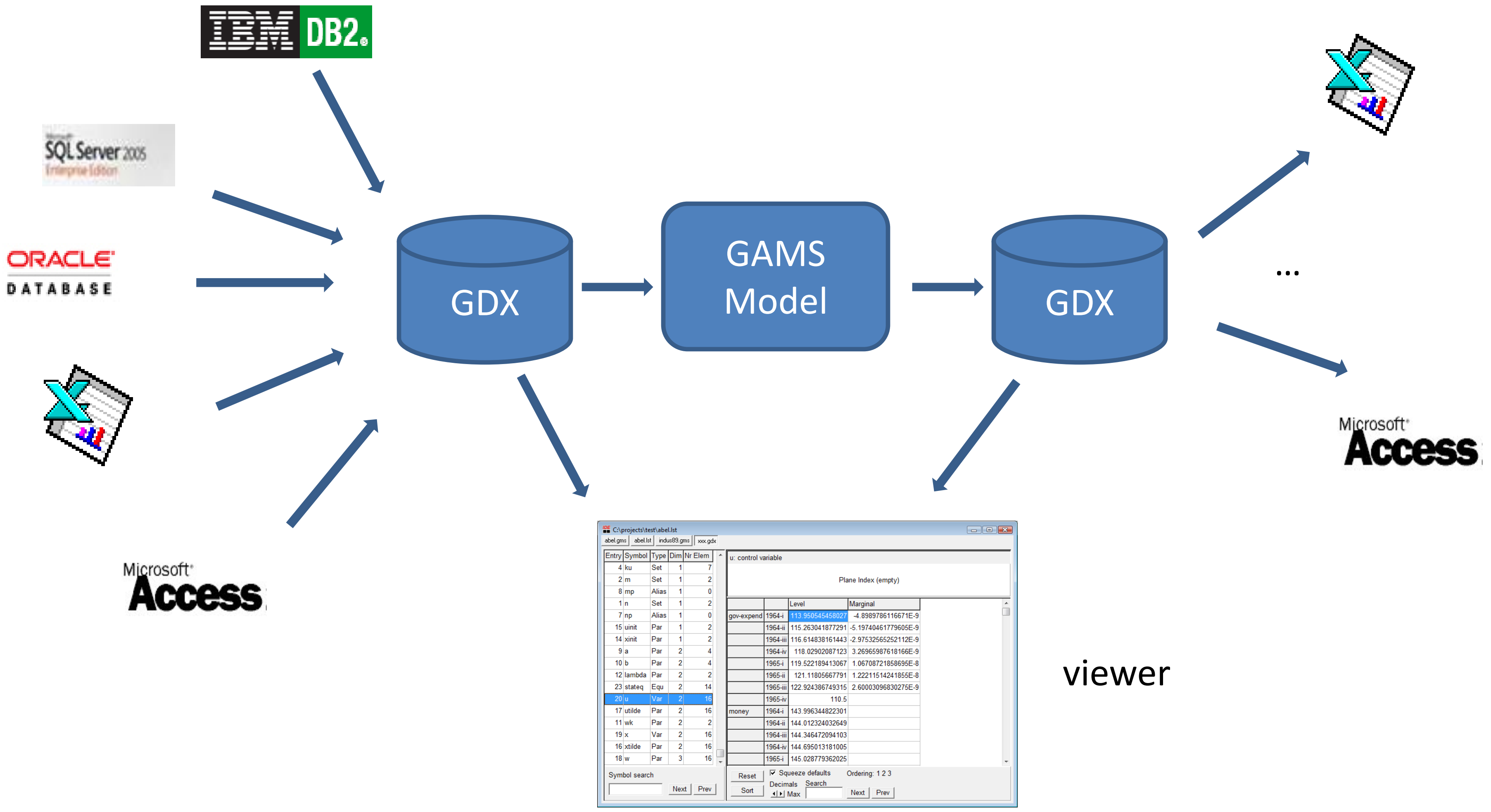
GDX API

- This is for programmers
- API available for several programming languages (VBA, C, VB.NET, C#, Fortran, Delphi)



```
public bool gdxfindsymbol(int ap,string aname,out int aix)
// gdxfindsymbol:
//    Search for a symbol by name; the search is not case sensitive.
//    When the symbol is found, Aix contains the symbol number and the
//    function returns true. When the symbol is not found, the function
//    returns false.
// ap:
//    Input: Pointer to GDX structure
// aname:
//    Input: Name of the symbol
// aix:
//    Output: Symbol number
```

GDX: Data Hub



C:\projects\test\abel.lst

abel.gms | abel.lst | indus89.gms | xxx.gdx

Entry	Symbol	Type	Dim	Nr Elem
4	ku	Set	1	7
2	m	Set	1	2
8	mp	Alias	1	0
1	n	Set	1	2
7	np	Alias	1	0
15	uinit	Par	1	2
14	xinit	Par	1	2
9	a	Par	2	4
10	b	Par	2	4
12	lambda	Par	2	2
23	stateq	Equ	2	14
20	u	Var	2	16
17	utilde	Par	2	16
11	wk	Par	2	2
19	x	Var	2	16
16	xtilde	Par	2	16
18	w	Par	3	16

Symbol search

u: control variable

Plane Index (empty)

		Level	Marginal
gov-expend	1964-i	113.950545458027	-4.8989786116671E-9
	1964-ii	115.263041877291	-5.19740461779605E-9
	1964-iii	116.614838161443	-2.97532565252112E-9
	1964-iv	118.02902087123	3.26965987618166E-9
	1965-i	119.522189413067	1.06708721858695E-8
	1965-ii	121.11805667791	1.22211514241855E-8
	1965-iii	122.924386749315	2.60003096830275E-9
	1965-iv	110.5	
money	1964-i	143.996344822301	
	1964-ii	144.012324032649	
	1964-iii	144.346472094103	
	1964-iv	144.695013181005	
	1965-i	145.028779362025	

Reset | Sort | ✓ Squeeze defaults | Decimals Max | Search | Ordering: 1 2 3 | Next | Prev

GDX File is Not:

- A database
- A data container

Because GDX file is ‘**immutable**:’

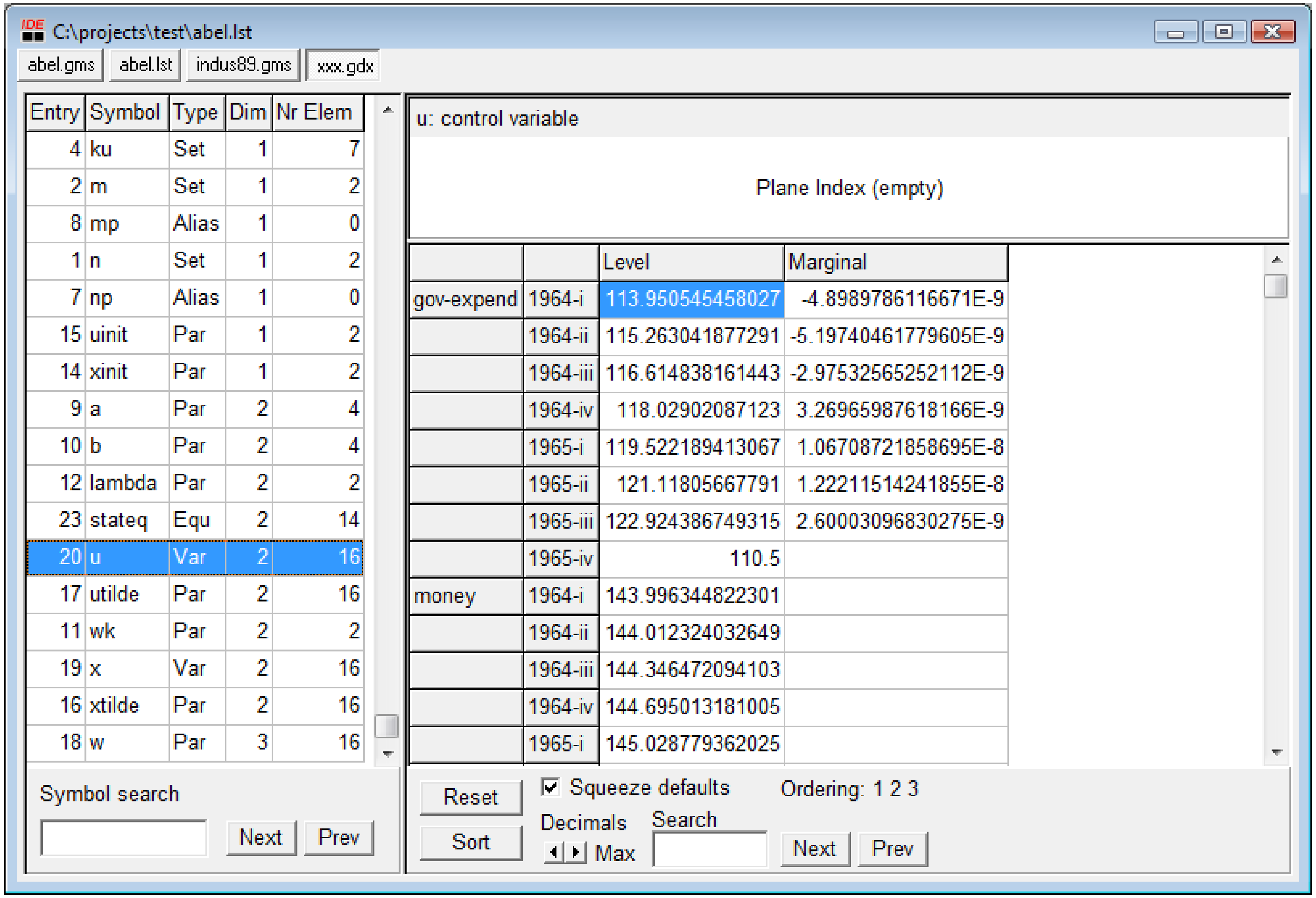
cannot add records
cannot delete records

cannot change records

(this looks worse than it is)

So I have a Gdx File, What Now?

- File|Open



C:\projects\test\abel.lst

Entry	Symbol	Type	Dim	Nr Elem
4	ku	Set	1	7
2	m	Set	1	2
8	mp	Alias	1	0
1	n	Set	1	2
7	np	Alias	1	0
15	uinit	Par	1	2
14	xinit	Par	1	2
9	a	Par	2	4
10	b	Par	2	4
12	lambda	Par	2	2
23	stateq	Equ	2	14
20	u	Var	2	16
17	utilde	Par	2	16
11	wk	Par	2	2
19	x	Var	2	16
16	xtilde	Par	2	16
18	w	Par	3	16

u: control variable

Plane Index (empty)

		Level	Marginal
gov-expend	1964-i	113.950545458027	-4.8989786116671E-9
	1964-ii	115.263041877291	-5.19740461779605E-9
	1964-iii	116.614838161443	-2.97532565252112E-9
	1964-iv	118.02902087123	3.26965987618166E-9
	1965-i	119.522189413067	1.06708721858695E-8
	1965-ii	121.11805667791	1.22211514241855E-8
	1965-iii	122.924386749315	2.60003096830275E-9
	1965-iv	110.5	
money	1964-i	143.996344822301	
	1964-ii	144.012324032649	
	1964-iii	144.346472094103	
	1964-iv	144.695013181005	
	1965-i	145.028779362025	

Symbol search

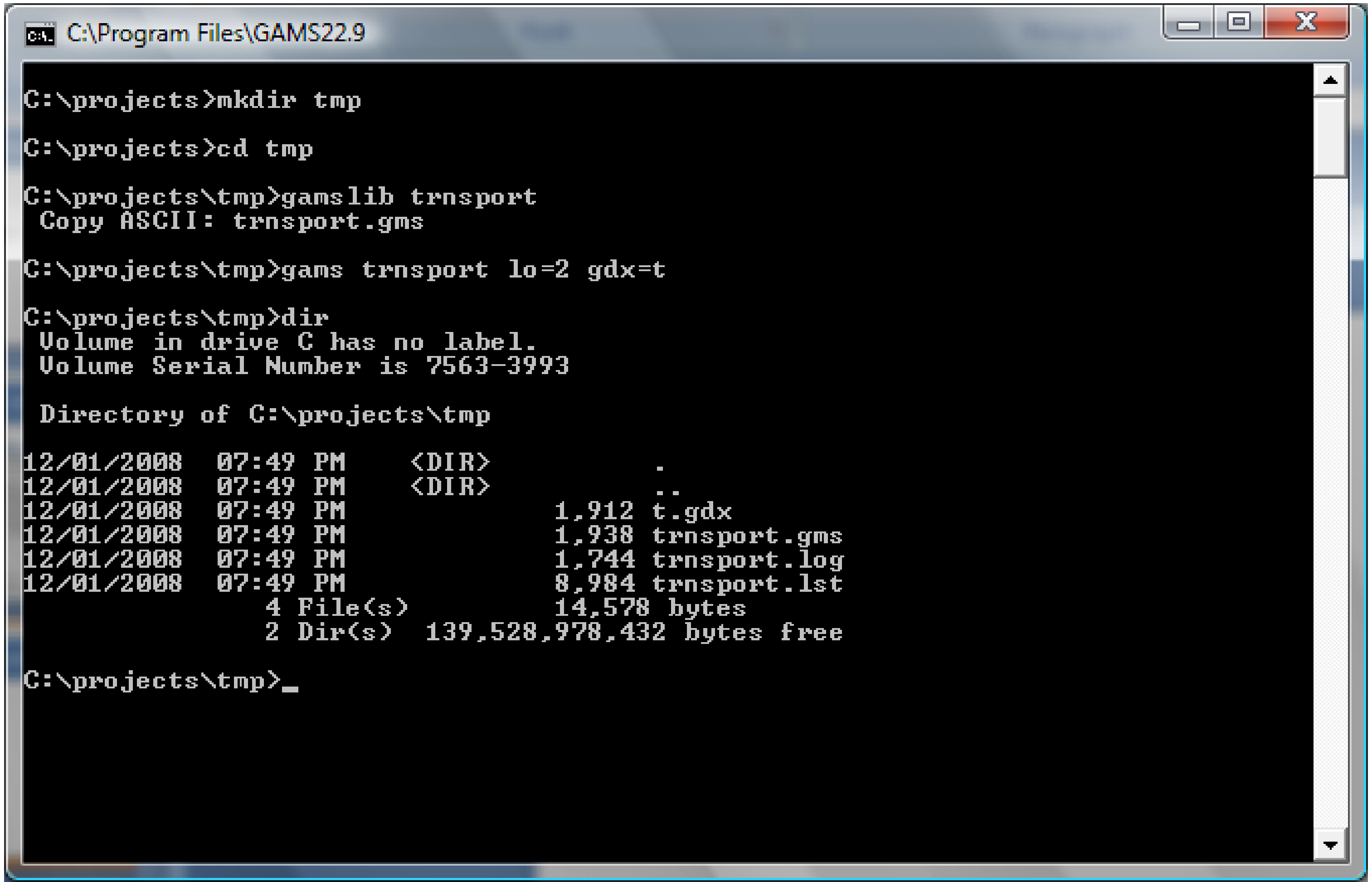
[✓] Squeeze defaults

Ordering: 1 2 3

Decimals Max

How to create a GDX file

- Method 1: Command line parameter, GDX=xxx



```
C:\projects>mkdir tmp

C:\projects>cd tmp

C:\projects\tmp>gamslib trnsport
 Copy ASCII: trnsport.gms

C:\projects\tmp>gams trnsport lo=2 gdx=t

C:\projects\tmp>dir
 Volume in drive C has no label.
 Volume Serial Number is 7563-3993

 Directory of C:\projects\tmp

12/01/2008  07:49 PM    <DIR>          .
12/01/2008  07:49 PM    <DIR>          ..
12/01/2008  07:49 PM             1,912 t.gdx
12/01/2008  07:49 PM             1,938 trnsport.gms
12/01/2008  07:49 PM             1,744 trnsport.log
12/01/2008  07:49 PM             8,984 trnsport.lst
               4 File(s)         14,578 bytes
               2 Dir(s)  139,528,978,432 bytes free

C:\projects\tmp>_
```

Trnsport

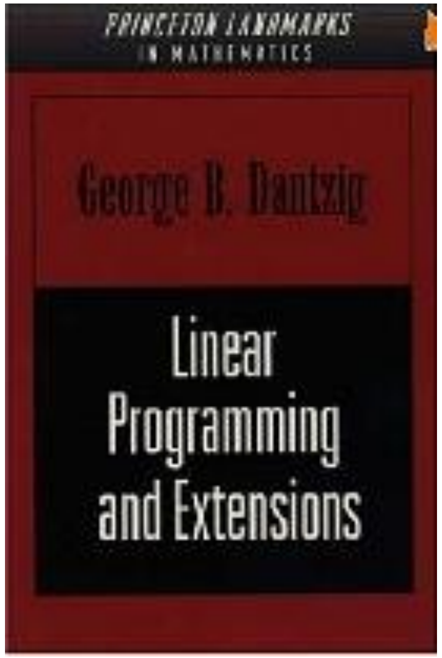
```
Sets
       i   canning plants   / seattle, san-diego /
       j   markets          / new-york, chicago, topeka / ;

  Parameters

       a(i)  capacity of plant i in cases
         /    seattle     350
              san-diego   600  /

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

  Table d(i,j)  distance in thousands of miles
                    new-york       chicago      topeka
      seattle          2.5            1.7          1.8
      san-diego        2.5            1.8          1.4   ;

  Scalar f  freight in dollars per case per thousand miles
/90/ ;

  Parameter c(i,j)  transport cost in thousands of dollars
per case ;

            c(i,j) = f * d(i,j) / 1000 ;

  Variables
       x(i,j)  shipment quantities in cases
       z       total transportation costs in thousands of
dollars ;

  Positive Variable x ;
```

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

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

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

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

 Model transport /all/ ;

 Solve transport using lp minimizing z ;

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

Trnsport.1

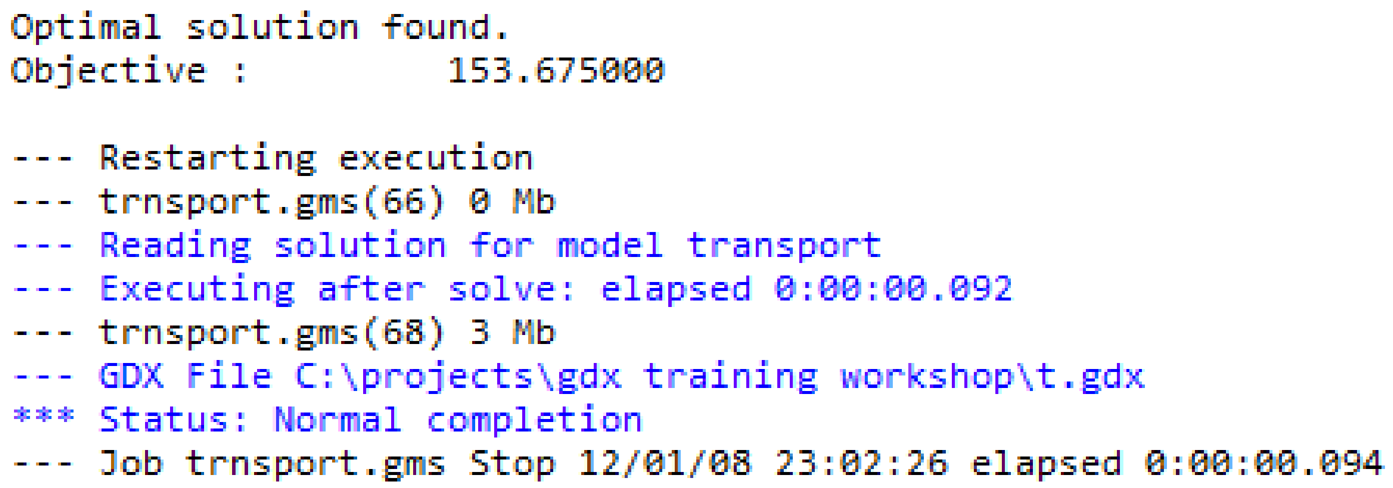
- Model is first model in famous Dantzig 1963 book
- Slightly changed to introduce degeneracy

$$\begin{array}{ll}
\min & \sum_{i,j} c_{i,j} x_{i,j} \\
\hline
\text{s.t.} & \sum_{j} x_{i,j} \le a_i \quad \forall i \\
& \sum_{i} x_{i,j} \ge b_j \quad \forall j \\
& x_{i,j} \ge 0
\end{array}$$


PRINCETON LANDMARKS IN MATHEMATICS
George B. Dantzig
Linear Programming and Extensions

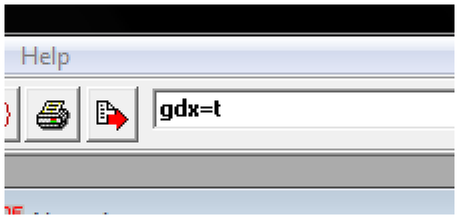
IDE Trick

- Instead of File|Open just click on blue line



```
Optimal solution found.
Objective :          153.675000

--- Restarting execution
--- trnsport.gms(66) 0 Mb
--- Reading solution for model transport
--- Executing after solve: elapsed 0:00:00.092
--- trnsport.gms(68) 3 Mb
--- GDX File C:\projects\gdx training workshop\t.gdx
*** Status: Normal completion
--- Job trnsport.gms Stop 12/01/08 23:02:26 elapsed 0:00:00.094
```

IDE Command Line Parameters

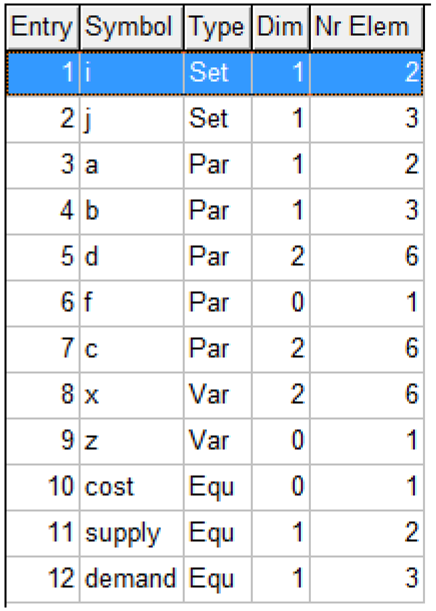


```
gdx=t
```

- Project file determines location
- Also allowed **gdx=t.gdx**

t.gdx

- >Gams trnsport gdx=t
  - Runs the model
  - Saves all data in the model to the gdx file
  - In Goobledegook: saves the whole *symbol table*

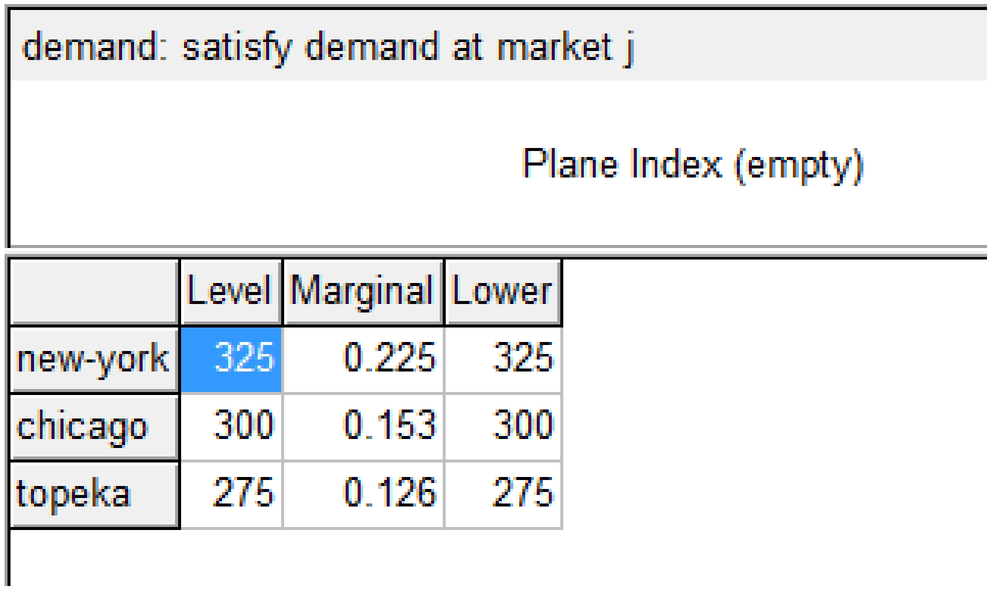


Entry	Symbol	Type	Dim	Nr Elem
1	i	Set	1	2
2	j	Set	1	3
3	a	Par	1	2
4	b	Par	1	3
5	d	Par	2	6
6	f	Par	0	1
7	c	Par	2	6
8	x	Var	2	6
9	z	Var	0	1
10	cost	Equ	0	1
11	supply	Equ	1	2
12	demand	Equ	1	3

•Set
•Par (Parameter/scalar/table)
•Var (Variable)
•Equ (Equation)

Equations

- These are just the values, not the symbolic formulas
- Often not so interesting (look at vars instead)



demand: satisfy demand at market j

Plane Index (empty)

	Level	Marginal	Lower
new-york	325	0.225	325
chicago	300	0.153	300
topeka	275	0.126	275

GAMS/GDX Set element names

- If contain blanks then need to be quoted

```
Set jx 'for use with X/XB variable'  /
      Imports
      "Food,Seed & Industial"
      Production
      ‘Paid Diversion’
/;
```

Explanatory text: these quotes are not needed if we had no / in the text

Double quotes

Single quotes. This can be important if the string already contains a single or double quote.

A valid set element can not contain both ‘ and “

UELS, Symbol limits

- Symbol names (parameter, variable name)
  - Identifier (starts with a letter, up to 63 chars, no blanks etc.)
- UEL (set element) names
  - Up to 63 chars
  - May need quoting
  - UEL use different storage than symbols
    - Set i /i/; is allowed

Special Values

- GAMS has special values
  - EPS, NA, UNDEF, INF, -INF
- They can be stored in a GDX file
- But note that they cannot always be handled by other programs. E.g. numeric field in a database.

Other ways to read/write GDX files

- From GAMS:
  - $load
  - $loaddc
  - $unload
  - Execute_load
  - Execute_unload
- No command line to read a gdx file
- First we need to understand difference between compile time and execution time

2 pass system

- Pass 1: Compile time
  - Parser
  - Handle all declarations
    - Set, parameter, table statement
  - Handle all $ control options
    - $include, $set, $onecho etc.
- Pass 2: Execution time
  - Execute compiled statements

Often no problem but…

- Sometimes surprises….

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

display "%name%";
```

```
$set name "hello"
$set name "world"

if(1,
else
);

display "%name%";
```

Examples

- Put is execution time, $include run-time

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

- $onecho compile time, solve execution time

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

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

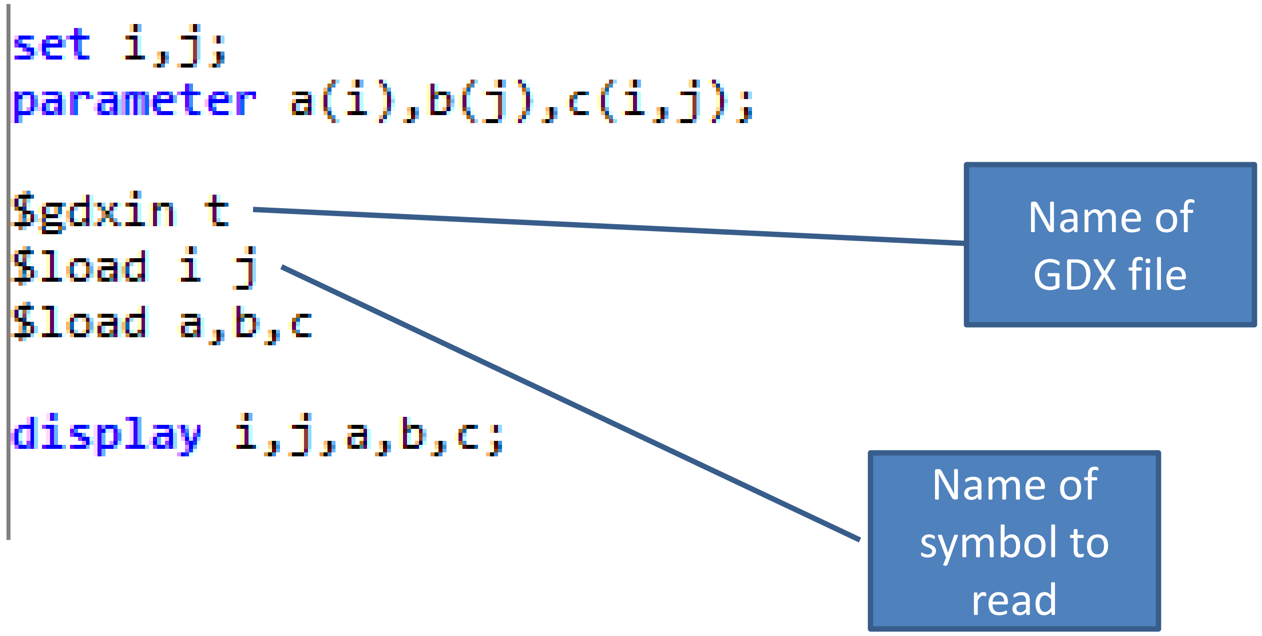
Last one

```
* exotic but I have seen this happening

scalar s;
$onmulti


* .... lots of stuff here

s = 3;
scalar s /2/;
display s;
```

```
----      11 PARAMETER s                  =        3.000
```

GDX $load

- Read symbol from gdx file at compile time



```
set i,j;
parameter a(i),b(j),c(i,j);

$gdxin t
$load i j
$load a,b,c

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

Oops

```
set i,j;
*parameter a(i),b(j),c(i,j);
parameter a(i),b(i),c(i,j);

$gdxin t
$load i j
$load a,b,c

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

GDX file has b(j) not b(i)

```
----    11 PARAMETER b  demand at market j in cases

                ( ALL       0.000 )
```

$loaddc

- $load Domain Checked

```
set i,j;
*parameter a(i),b(j),c(i,j);
parameter a(i),b(i),c(i,j);

$gdxin t
$load i j
$loaddc a,b,c

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

No need here

Now triggers a syntax error

```
**** 3 Domain errors for symbol b
     new-york
     chicago
     topeka
--- LOAD  c = 7:c
   9  $loaddc a,b,c
****            $649
**** 649  Domain violation when loading from GDX file
```

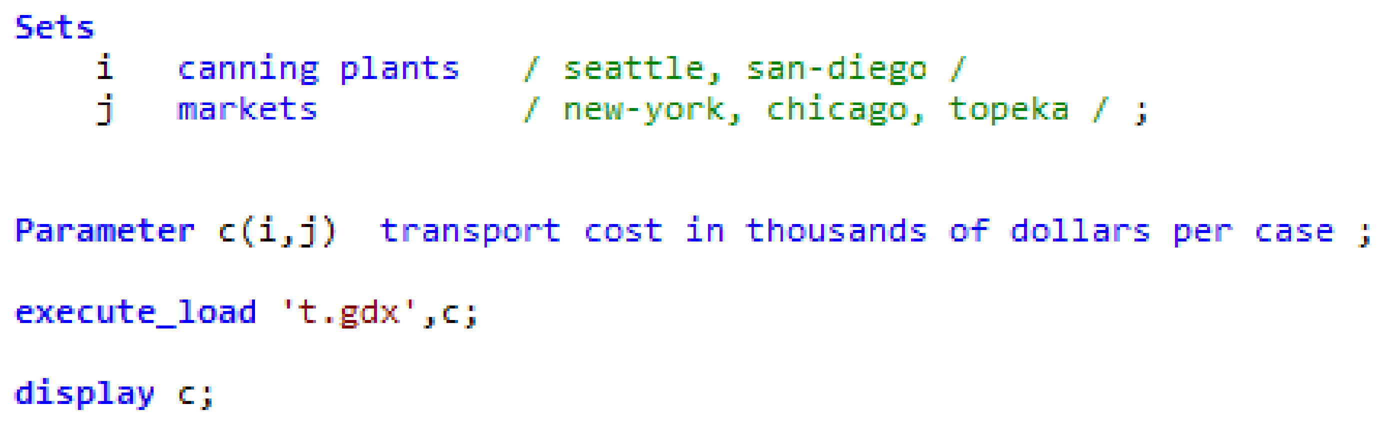
Trick: table of contents

```
$gdxin t
$load
```

```
Number Type       Dim       Count  Name

     1 Set          1           2  i           canning plants
     2 Set          1           3  j           markets
     3 Parameter    1           2  a           capacity of plant i in cases
     4 Parameter    1           3  b           demand at market j in cases
     5 Parameter    2           6  d           distance in thousands of miles
     6 Parameter    0           1  f           freight in dollars per case per thousand miles
     7 Parameter    2           6  c           transport cost in thousands of dollars per case
     8 Variable     2           6  x           shipment quantities in cases
     9 Variable     0           1  z           total transportation costs in thousands of dollars
    10 Equation     0           1  cost        define objective function
    11 Equation     1           2  supply      observe supply limit at plant i
    12 Equation     1           3  demand      satisfy demand at market j
```

Execute_load

- At execution time
  - Needed when you must read after some execution
  - E.g. after a solve where solver writes gdx file



```
Sets
    i   canning plants   / seattle, san-diego /
    j   markets          / new-york, chicago, topeka / ;

Parameter c(i,j)  transport cost in thousands of dollars per case ;

execute_load 't.gdx',c;

display c;
```

No New Elements

- Execute_load will never add new elements (UELs)

Needed (*) to say 1 dimensional

No domain checking

```
Sets
    i(*)    canning plants
    j(*)    markets
    dummy /new-york/
;


Parameter c(*,*)  transport cost in thousands of dollars per case ;

execute_load 't.gdx',i,j,c;

display i,j,c;
```

Result:	i=empty
	j=‘new york’
	c=0

Write GDX file

- Compile time
  - $gdxout/$unload
  - Seldomly used
- Execution time
  - Execute_unload
  - Simple and often used

$gdxout/$unload vs Execute_unload

- Example
  - Add to bottom of trnsport model:

```
$gdxout gdxout.gdx
$unload i c d x
```

```
Execute_unload ‘execute_unload.gdx’,
    i, c, d, x;
```

  - Then result is:

Set i is ok
Parameter d is ok (table)
Parameter c is empty
Variable x is empty

All are ok

Display vs GDX

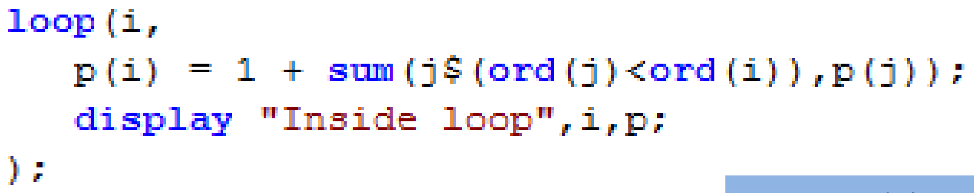
- Display Advantages
  - We can Display a string
    - Display “Updated with results”,p,a;
  - Sometimes easier to follow an algorithm
    - Display “Current value”,iter,x;
  - Same thing can be displayed several times
- GDX Advantages
  - Large data easier to inspect
  - All data available with simple gdx=xxx
  - Layout easier to control than option symb:a:b:c;

Example debugging loop

```
set i /i1*i5/;
alias(i,j);

parameter p(i);
scalar iteration;

p(i) = 0;

loop(i,
   p(i) = 1 + sum(j$(ord(j)<ord(i)),p(j));
   iteration = ord(i);
   display "Inside loop",iteration,p;
);

display "final",p;
```

```
----     12 Inside loop
            PARAMETER iteration            =        1.000

----     12 PARAMETER p

i1 1.000

----     12 Inside loop
            PARAMETER iteration            =        2.000

----     12 PARAMETER p

i1 1.000,    i2 2.000

----     12 Inside loop
            PARAMETER iteration            =        3.000

----     12 PARAMETER p

i1 1.000,    i2 2.000,    i3 4.000

. . . . .

----     15 final

----     15 PARAMETER p

i1  1.000,    i2  2.000,    i3  4.000,    i4  8.000,    i5 16.000
```

Aside: debugging loops



```
loop(i,
   p(i) = 1 + sum(j$(ord(j)<ord(i)),p(j));
   display "Inside loop",i,p;
);
```

```
----    16 Inside loop

----    16 SET i

i1,   i2,   i3,   i4,   i5


----    16 PARAMETER p

i1 1.000


----    16 Inside loop

----    16 SET i

i1,   i2,   i3,   i4,   i5


----    16 PARAMETER p

i1 1.000,    i2 2.000
```

Display of set inside loop displays whole set instead of current element.

In previous example solved by displaying a scalar.

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
```

Display Option

```
set i /i1*i5/;
alias (i,i1,i2,i3);

parameter p(i1,i2,i3);
p(i1,i2,i3) = uniform(0,1);

display p;

option p:4:1:2; display p;
option p:4:2:1; display p;
option p:4:0:1; display p;
```

```
----      8 PARAMETER p

               i1          i2          i3          i4          i5

i1.i1       0.172       0.843       0.550       0.301       0.292
i1.i2       0.224       0.350       0.856       0.067       0.500
i1.i3       0.998       0.579       0.991       0.762       0.131
```

```
----      10 PARAMETER p

           i1.i1       i1.i2       i1.i3       i1.i4

i1        0.1717      0.8433      0.5504      0.3011
i2        0.8309      0.2308      0.6657      0.7759
i3        0.6611      0.7558      0.6274      0.2839
```

```
----      11 PARAMETER p

                i1          i2          i3          i4          i5

i1.i1       0.1717      0.8433      0.5504      0.3011      0.2922
i1.i2       0.2241      0.3498      0.8563      0.0671      0.5002
i1.i3       0.9981      0.5787      0.9911      0.7623      0.1307
i1.i4       0.6397      0.1595      0.2501      0.6689      0.4354
```

```
----      12 PARAMETER p

i1.i1.i1 0.1717
i1.i1.i2 0.8433
i1.i1.i3 0.5504
i1.i1.i4 0.3011
i1.i1.i5 0.2922
i1.i2.i1 0.2241
i1.i2.i2 0.3498
```

Debugging:Stop GAMS in the middle

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

This works very well with GDX=xxx command line parameter

Big Models: modules

- Big models are often split into modules:
  - Data import
  - Data preparation
  - Calibration
  - Solving
  - Reporting

Save/Restart

```
scalar s;
s = 12;
```

```
s = s + 1;
display s;
```

```
scalar s;
s = 12;
```

> Gams file1 save=f1

```
s = s + 1;
display s;
```

> Gams file2 restart=f1

Restart file is f1.g00

```
--- Job file2.gms Start 12/02/08 10:57:45 WEX-WEI 22.9.1 x86_64/MS Windows
GAMS Rev 229  Copyright (C) 1987-2008 GAMS Development. All rights reserved
Licensee: Erwin Kalvelagen                                 G080731/0001CJ-WIN
          GAMS Development Corporation                                 DC4572
--- Starting continued compilation
--- file2.gms(2) 2 Mb
--- Starting execution: elapsed 0:00:00.004
--- file2.gms(4) 3 Mb
*** Status: Normal completion
--- Job file2.gms Stop 12/02/08 10:57:45 elapsed 0:00:00.004
```

GDX Files

```
scalar s;
s = 12;
```

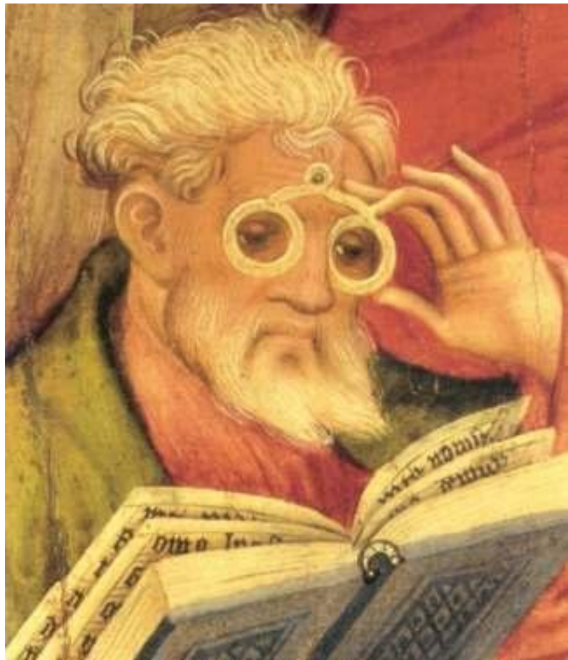
```
> Gams file1 gdx=f1
```

```
$gdxin f1.gdx
scalar s;
$load s

s = s + 1;
display s;
```

```
> Gams file2
```

GDX vs Restart

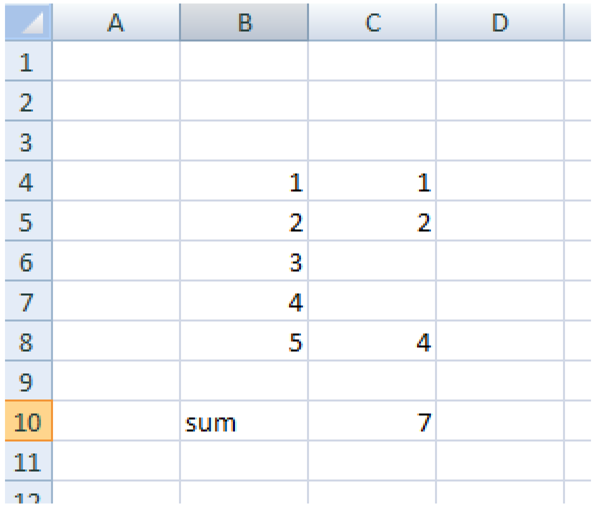
- GDX File only has data
  - Equations are only values
- Restart File includes symbolic equations
- But Restart Files are black boxes
- Big advantage:
  One can **look** at a GDX file



GAMS+GDX: sparse storage

- Both GAMS and GDX use sparse storage
  - Advantage: can store very large sparse data structures + performance
  - These have the same meaning:
    - Zero
    - Does not exist
  - Can occasional give surprises.

Excel Sparse Sum

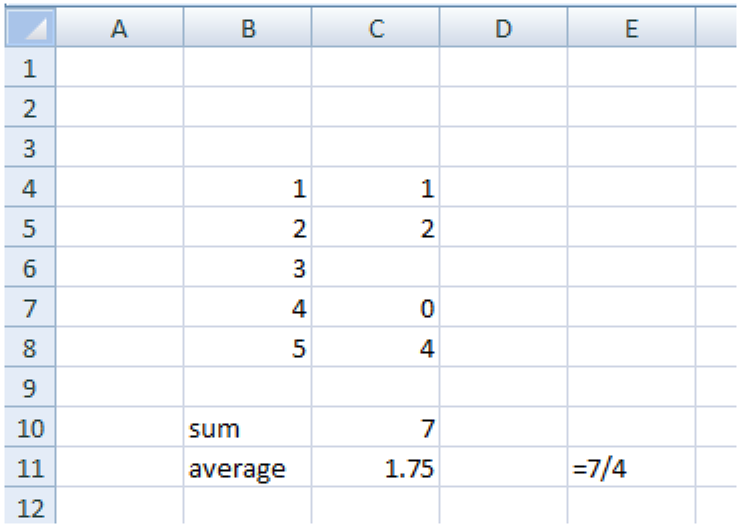


	A	B	C	D
1				
2				
3				
4		1	1	
5		2	2	
6		3		
7		4		
8		5	4	
9				
10		sum	7	
11				

But what about average?

=SUM(C4:C8)

Excel Average



	A	B	C	D	E
1					
2					
3					
4		1	1		
5		2	2		
6		3			
7		4	0		
8		5	4		
9					
10		sum	7		
11		average	1.75		=7/4
12					

Excel makes difference between 0 and blank

GAMS Average

```
set i /i1*i5/;

table p(i,*)
    values
 i1  1
 i2  2
 i3
 i4  0
 i5  4

;


scalars psum, paverage;

psum = sum(i, p(i,'values'));
paverage = psum/card(p);

display psum, paverage;
```

Here average is 7/3

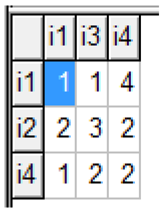
```
----      19 PARAMETER psum                    =        7.000
             PARAMETER paverage                =        2.333
```

Export matrix

```
set i /i1*i4/;

table p(i,i)

        i1   i2   i3   i4
   i1    1         1    4
   i2    2         3    2
   i3
   i4    1         2    2

;
```

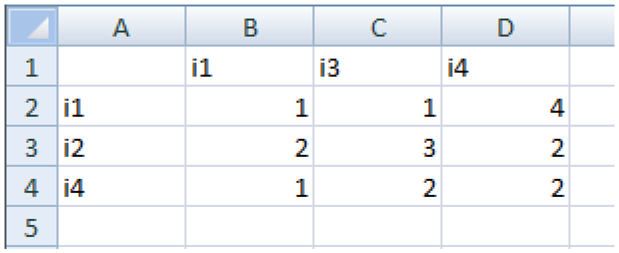
4x4 matrix arrives as 3x3 matrix

Gdx:



	i1	i3	i4
i1	1	1	4
i2	2	3	2
i4	1	2	2

Exported to excel:



	A	B	C	D
1		i1	i3	i4
2	i1	1	1	4
3	i2	2	3	2
4	i4	1	2	2
5				