GDX: Advanced GAMS + Databases + Applications

Erwin Kalvelagen

Agenda

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

Ordering

- The way GAMS and GDX order data is somewhat unique
- Often automatically what you want, but sometimes a surprise
- We need some background info to understand the behavior

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
```

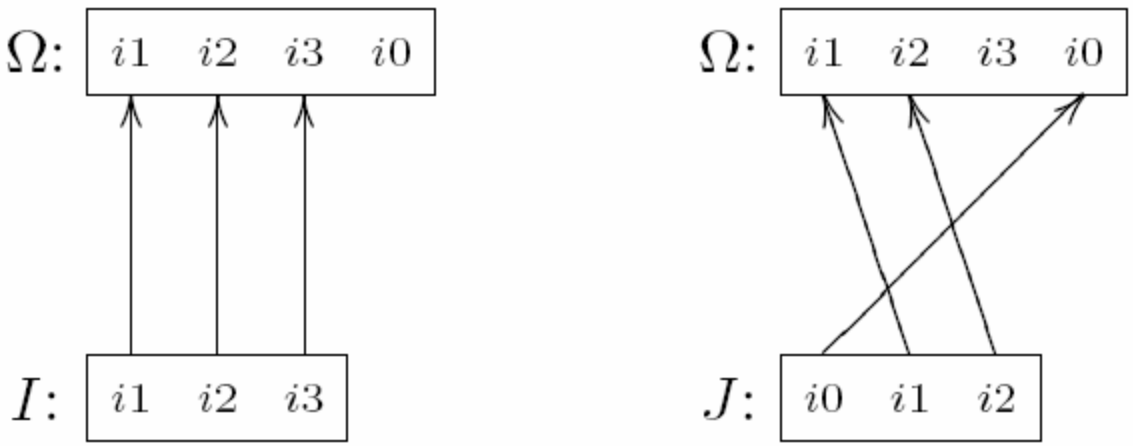
Unordered sets

- 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)



Pool of uels

Ordered if arrows don't cross

Figure 1.5: Ordered and unordered set

Explanatory text set elements

- Limited value, not visible in display

```
set
   ja 'modelcrops, for use with A,AB' /
         Corn                    01
         Sorghum                 02
         Oats                    03
         Barley                  04
         Wheat                   05
         Soybeans                06
         Cotton                  07
         FeedGrains              08
         FoodGrains              09
         OilCrops                10
    /;
```

01, 02 is explanatory text for set elements. I use it mainly for self-documentation. They are preserved in the GDX file.

Corn	"01"
Sorghur	"02"
Oats	"03"
Barley	"04"
Wheat	"05"
Soybea	"06"
Cotton	"07"
FeedGr	"08"
FoodGr	"09"
OilCrop	"10"

Solving Linear Equations

- Solve Ax=b for x
- Often not a good idea to calculate A^{-1}
- In GAMS we can solve by specifying Ax=b as equations

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

Inverse

- 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}

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

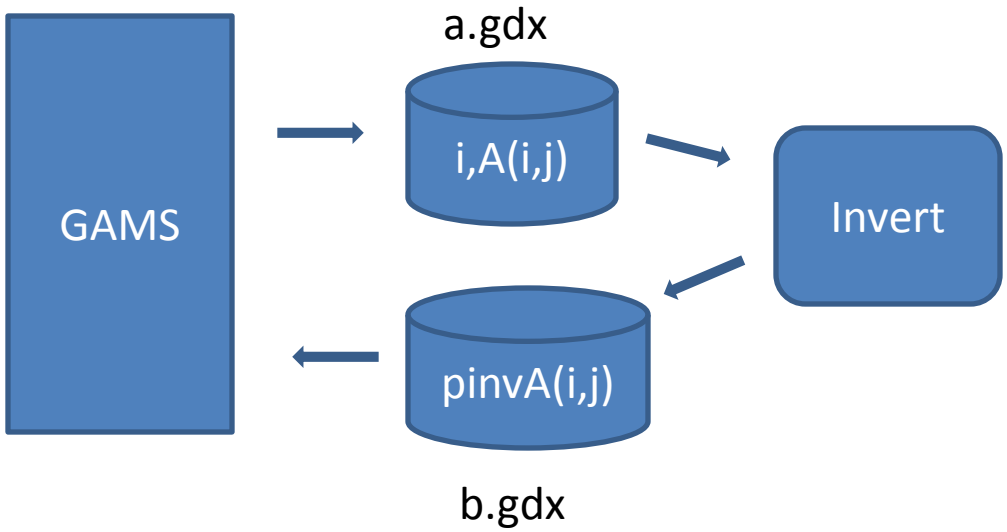
Speed Up: Advanced Basis

- We can provide advanced basis so the calculation takes 0 Simplex iterations
  - Inv.m(i,j) = 0; (var: basic)
  - Inverse.m(i,j) = 1; (equ: non-basic)

	method=1	method=2
n=50	0.637	0.433
n=100	8.267	4.036
n=200	313.236	53.395

External Solver using GDX

```
execute_unload 'a.gdx',i,a;
execute '=invert.exe a.gdx i a b.gdx pinva';
parameter pinva(i,j);
execute_load 'b.gdx',pinva;
```



Test Matrix:Pei Matrix

$$\begin{pmatrix} 1+\alpha & 1 & 1 & 1 \\ 1 & 1+\alpha & 1 & 1 \\ 1 & 1 & 1+\alpha & 1 \\ 1 & 1 & 1 & 1+\alpha \end{pmatrix}$$

	method=1	method=2	method=3
n=50	0.637	0.433	0.027
n=100	8.267	4.036	0.055
n=200	313.236	53.395	0.118

Other tools

- Cholesky
- Eigenvalue
- Eigenvector

Max Likelihood Estimation

- NLP solver can find optimal values: estimates
- But to get covariances we need:
  - Hessian
  - Invert this Hessian
- We can do this now in GAMS
  - Klunky, but at least we can now do this
  - GDX used several times

MLE Estimation Example

```
*  Data:
*  Number of days until the appearance of a carcinoma in 19
*  rats painted with carcinogen DMBA.


set i /i1*i19/;
table data(i,*)
             days   censored
    i1        143
    i2        164
    i3        188
    i4        188
    i5        190
    i6        192
    i7        206
    i8        209
    i9        213
    i10       216
    i11       220
    i12       227
    i13       230
    i14       234
    i15       246
    i16       265
    i17       304
    i18       216     1
    i19       244     1
;
```

```
set k(i) 'not censored';
k(i)$(data(i,'censored')=0) = yes;

parameter x(i);
x(i) = data(i,'days');

scalars
  p 'number of observations'
  m 'number of uncensored observations'
;

p = card(i);
m = card(k);

display p,m;
```

MLE Estimation

```
*-------------------------------------------------------------------------------
* estimation
*-------------------------------------------------------------------------------

scalar theta 'location parameter' /0/;

variables
     sigma 'scale parameter'
     c     'shape parameter'
     loglik 'log likelihood'
;

equation eloglike;

c.lo = 0.001;
sigma.lo = 0.001;


eloglike.. loglik =e= m*log(c) - m*c*log(sigma)
                 + (c-1)*sum(k,log(x(k)-theta))
                 - sum(i,((x(i)-theta)/sigma)**c);

model mle /eloglike/;
solve mle maximizing loglik using nlp;
```

Get Hessian

```
*-------------------------------------------------------------------------------
* get hessian
*-------------------------------------------------------------------------------
option nlp=convert;
$onecho > convert.opt
hessian
$offecho
mle.optfile=1;
solve mle minimizing loglik using nlp;


*
* gams cannot add elements at runtime so we declare the necessary elements here
*
set dummy /e1,x1,x2/;

parameter h(*,*,*) '-hessian';
execute_load "hessian.gdx",h;
display h;

set j /sigma,c/;
parameter h0(j,j);
h0('sigma','sigma') = h('e1','x1','x1');
h0('c','c') = h('e1','x2','x2');
h0('sigma','c') = h('e1','x1','x2');
h0('c','sigma') = h('e1','x1','x2');
display h0;
```

H: individual row Hessians

		x1	x2
e1	x1	0.0114575560266001	-0.0257527570747253
	x2		0.934221386133885

Invert Hessian

```
*-------------------------------------------------------------------------------
* invert hessian
*-------------------------------------------------------------------------------

execute_unload "h.gdx",j,h0;
execute "=invert.exe h.gdx j h0 invh.gdx invh";
parameter invh(j,j);
execute_load "invh.gdx",invh;
display invh;
```

Normal Quantiles

```
*-------------------------------------------------------------------------------
* quantile of normal distribution
*-------------------------------------------------------------------------------

* find
*   p = 0.05
*   q = probit(1-p/2)

scalar prob /.05/;

* we don't have the inverse error function so we calculate it
* using a small cns model
equations e;
variables probit;
e.. Errorf(probit) =e= 1-prob/2;
model inverterrorf /e/;
solve inverterrorf using cns;

display probit.l;

* verification:
*> qnorm(0.975);
*[1] 1.959964
*>
```

Or just use 1.96

Finally: confidence intervals

```
*-------------------------------------------------------------------------------
* calculate standard errors and confidence intervals
*-------------------------------------------------------------------------------

parameter result(j,*);
result('c','estimate') = c.l;
result('sigma','estimate') = sigma.l;

result(j,'stderr') = sqrt(abs(invh(j,j)));

result(j,'conf lo') = result(j,'estimate') - probit.l*result(j,'stderr');
result(j,'conf up') = result(j,'estimate') + probit.l*result(j,'stderr');

display result;
```

```
----    168 PARAMETER result

         estimate      stderr     conf lo     conf up

sigma     234.319       9.646     215.413     253.224
c           6.083       1.068       3.989       8.177
```

LS: linear regression

- Solver for linear regression
- Returns number of statistics in GDX file
  - Variance-covariance matrix
  - Confidence intervals
  - Etc.
- GAMS distribution has forgotten to include the documentation

Coefficients become variables

```
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;
```

```
----     81 VARIABLE constant.L            =        7.383  estimate constant term coefficient
            VARIABLE income.L              =        0.232  estimate income coefficient
            VARIABLE sse.L                 =     1780.413  sum of squared errors
```

Log

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

        Parameter      Estimate    Std. Error       t value      Pr(>|t|)
         constant   7.38322E+00   4.00836E+00   1.84196E+00   7.32959E-02 .
           income   2.32253E-01   5.52934E-02   4.20038E+00   1.55136E-04 ***
     ---
     Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

Estimation statistics:
Cases: 40  Parameters: 2  Residual sum of squares:   1.78041E+03
Residual standard error:   6.84492E+00   on  38 degrees of freedom
Multiple R-squared:   3.17077E-01  Adjusted R-squared:   2.99105E-01
F statistic:   1.76432E+01 on 1 and 38 DF, p-value:   1.55136E-04

GDX library version: GDX Library      BETA  4Nov08 22.9.1 WIN 7279.7544 VIS x86
/MS Windows
GDX file: ls.gdx
```

Ls.gdx

Entry	Symbol	Type	Dim	Nr Elem
1	r2	Par	0	1
2	rss	Par	0	1
3	df	Par	0	1
4	sigma	Par	0	1
5	resvar	Par	0	1
6	estimate	Par	1	2
7	se	Par	1	2
8	tval	Par	1	2
9	pval	Par	1	2
10	resid	Par	1	40
11	fitted	Par	1	40
12	covar	Par	2	4
13	confint	Par	3	16

se: Standard errors

Plane Index (empty)

constant	4.00835633479187
income	0.0552934293700572

Ls.gdx

confint: confidence intervals

		LO	UP
90%	constant	0.625311768027148	14.1411233181389
	income	0.139031132885651	0.325475527770996
95%	constant	-0.731274947250641	15.4977100334167
	income	0.120317643998824	0.344189016657823
97.5	constant	-1.97116743676858	16.7376025229346
	income	0.103213898206427	0.36129276245022
99%	constant	-3.48567081944515	18.2521059056112
	income	0.0823220216300179	0.382184639026629

covar: variance-covariance matrix

	constant	income
constant	16.0669205066661	-0.213403960538805
income	-0.213403960538805	0.0030573633315015

Read ls.gdx

```
sets
   v /constant,income/
   bnd /lo,up/
   perc /'90%'/
;
parameter covar(v,v),confint(perc,v,bnd);
execute_load 'ls.gdx',covar,confint;
display covar,confint;
```

Ordering Problem

```
----     91 PARAMETER covar

              income    constant

income         0.003      -0.213
constant      -0.213      16.067


----     91 PARAMETER confint

                       lo          up

90%.income          0.139       0.325
90%.constant        0.625      14.141
```

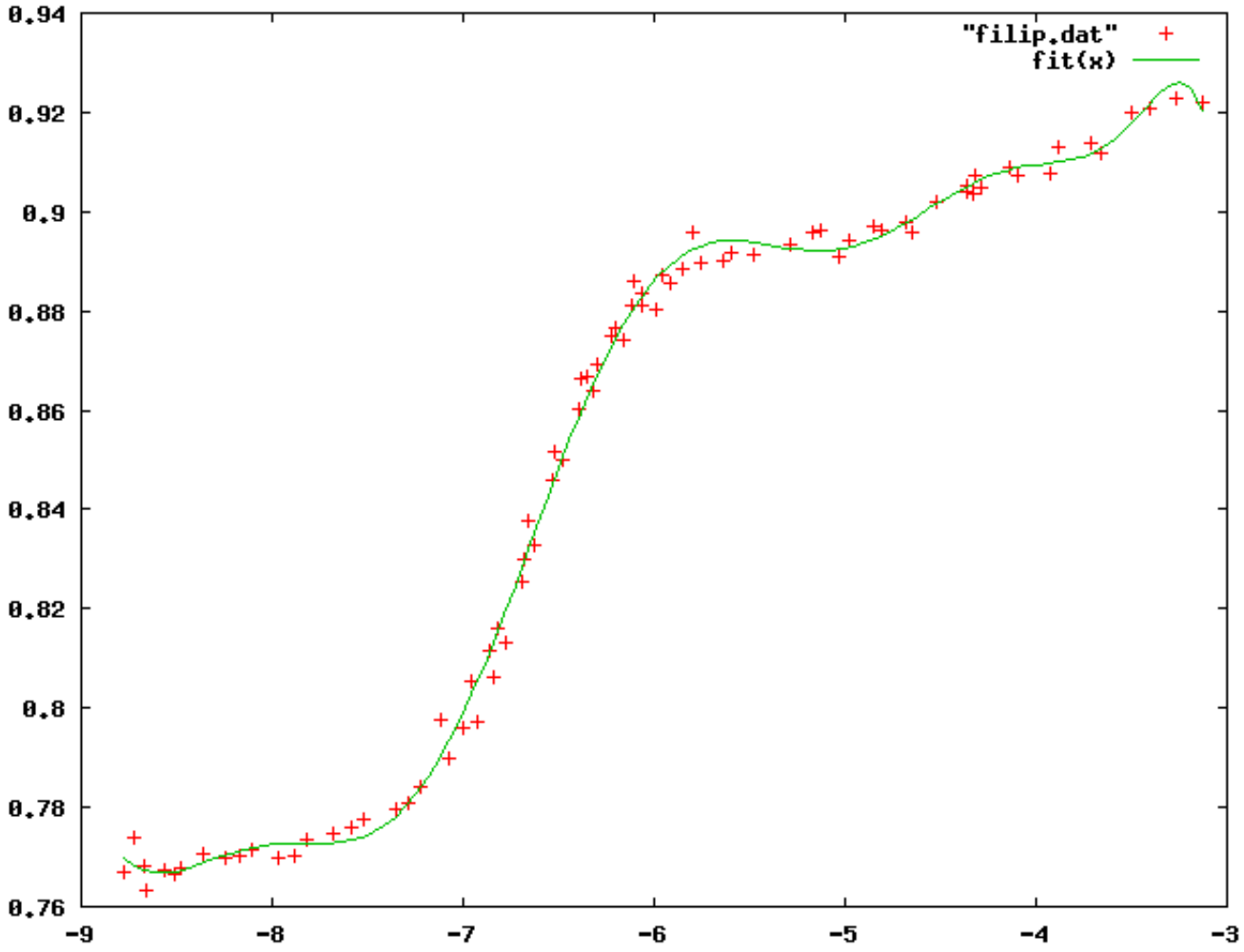
Multiple Regression

- Regression is still linear

$$y = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4 + b_5 x^5 + b_6 x^6 + b_7 x^7 + b_8 x^8 + b_9 x^9 + b_{10} x^{10} + \varepsilon$$

Parameter	Estimate	Std. Error	t value	Pr(>|t|)	
b(j0)	-1.46749E+03	2.98085E+02	-4.92307E+00	5.34684E-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	8.75366E-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	***

Plots: GNUPLOT



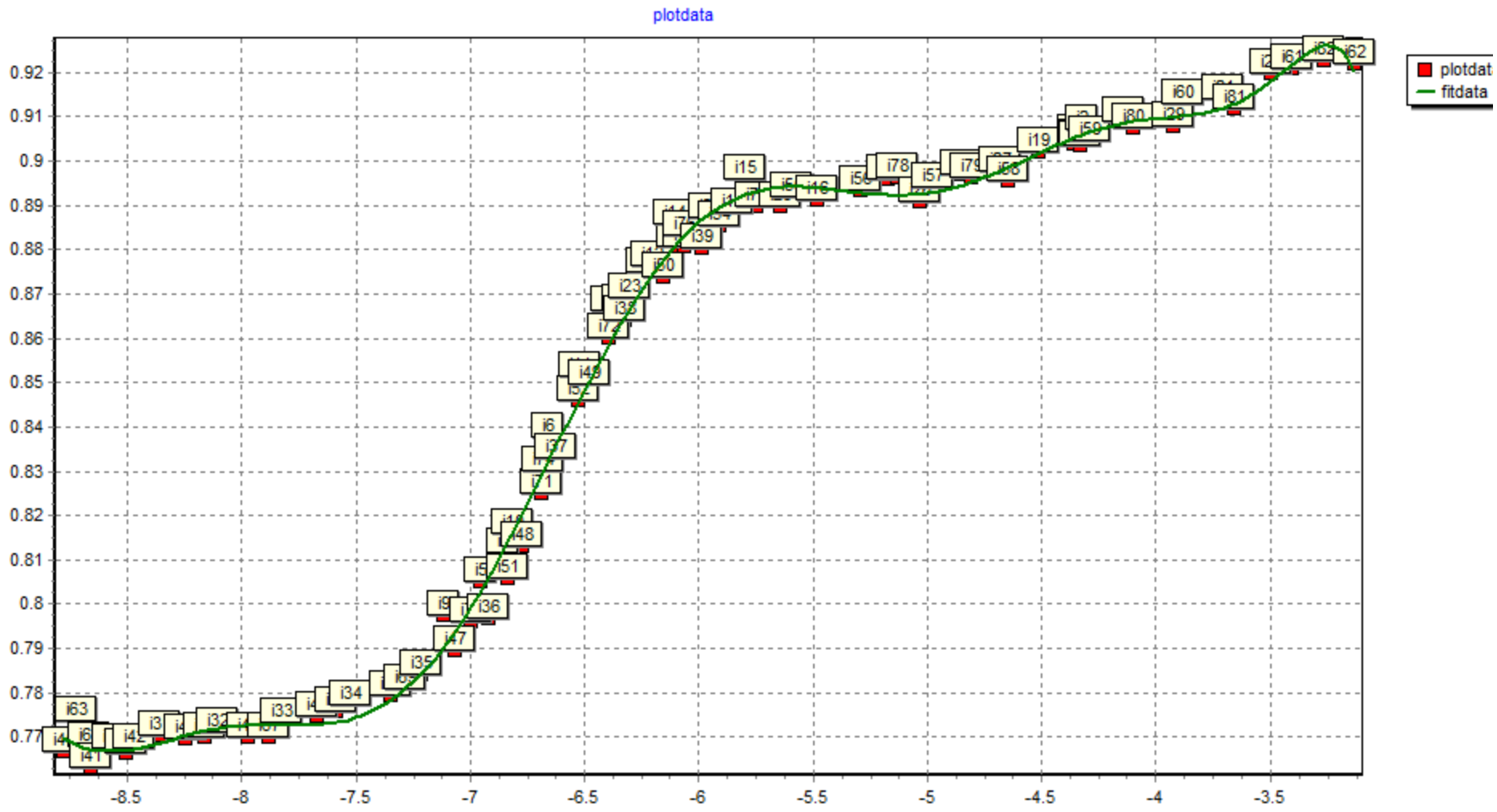
Gnuplot Input

```
b0=-1.46748962028755E+3
b1=-2.77217960148754E+3
b2=-2.31637108806308E+3
b3=-1.12797394340319E+3
b4=-354.478234242291000
b5=-75.124201807597610
b6=-10.875318038818510
b7=-1.062214985565215
b8=-0.067019115397592
b9=-0.002467810778920
b10=-0.000040296252419
fit(x)=b0+b1*x**1+b2*x**2+b3*x**3+b4*x**4+b5*x**5+b6*x**6+b7*x**7+b8*x**8+b9*x**9+b10*x**10
set term png
set output "filip.png"
plot "filip.dat",fit(x)
```

Generated by PUT

```
file pltdat /filip.dat/;
loop(i,
    put pltdat data(i,'x'):17:5,data(i,'y'):17:5/;
);
putclose;


file plt /filip.plt/;
put plt;
loop(j,
   put "b",v(j):0:0,"=",b.l(j):0:15/;
);
put "fit(x)=b0";
loop(j1,
   put "+b",v(j1):0:0,"*x**",v(j1):0:0
);
put /;
putclose 'set term png'/
         'set output "filip.png"'/
         'plot "filip.dat",fit(x)'/;

execute 'type filip.plt';

execute '=wgnuplot.exe filip.plt';
execute '=shellexecute filip.png';
```

Plot by IDE


plotdata
plotdat
fitdata
0.92
0.91
0.9
0.89
0.88
0.87
0.86
0.85
0.84
0.83
0.82
0.81
0.8
0.79
0.78
0.77
-8.5
-8
-7.5
-7
-6.5
-6
-5.5
-5
-4.5
-4
-3.5

IDE via GDX file

```
*
* plot results
*
* first we need to make sure x comes before y.
* we had before declared 'y' before 'x' so we introduce
* 'x0' and 'y0' where we make sure 'x0' comes first.
* if you load plotdata in the gdxviewer you will see that
* indeed 'x0','y0' are ordered correctly. If this step
* was not performed, we would have seen an inverted
* graph.
*

parameter plotdata(i,*);
plotdata(i,'x0') = EPS+data(i,'x');
plotdata(i,'y0') = EPS+data(i,'y');

set k/point1*point200/;
scalar minx, maxx, stepx;
minx = smin(i,data(i,'x'));
maxx = smax(i,data(i,'x'));
stepx = (maxx-minx)/(card(k)-1);
parameter xfit(k), yfit(k);
xfit(k) = minx+stepx*(ord(k)-1);
yfit(k) = sum(j, b.l(j)*power(xfit(k),v(j)));
parameter fitdata(k,*);
fitdata(k,'x0') =  EPS+xfit(k);
fitdata(k,'y0') =  EPS+yfit(k);
```

```
execute_unload 'chartdata.gdx',plotdata,fitdata;

$onecho > filip_gch.gch
[CHART]
VERID=GAMSIDE Chart(s) V1
GDXFILE=chartdata.gdx
TITLE=plotdata

[SERIES1]
SYMBOL=plotdata
TYPE=scatter2d

[SERIES2]
SYMBOL=fitdata
TYPE=function
$offecho


execute '=idecmds FileOpen filip_gch.gch'
```

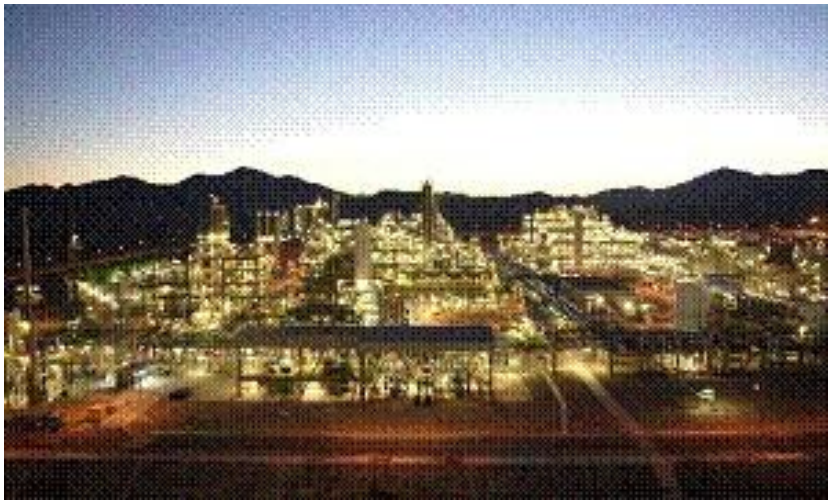
Gnuplot vs IDE vs Excel

- Whatever you are used to (taste)
- Function plotting better in GNUPLOT
- Gantt charts easiest in IDE
- Interfaces:
  - Gnuplot: text files, PUT statement
  - Gnuplot:Uwe Schneider tools
  - IDE: gdx files
  - Excel: gdxxrw

Plastic Pellet Production

GE Plastics, now SABIC

Carthagena (Spain) Plant

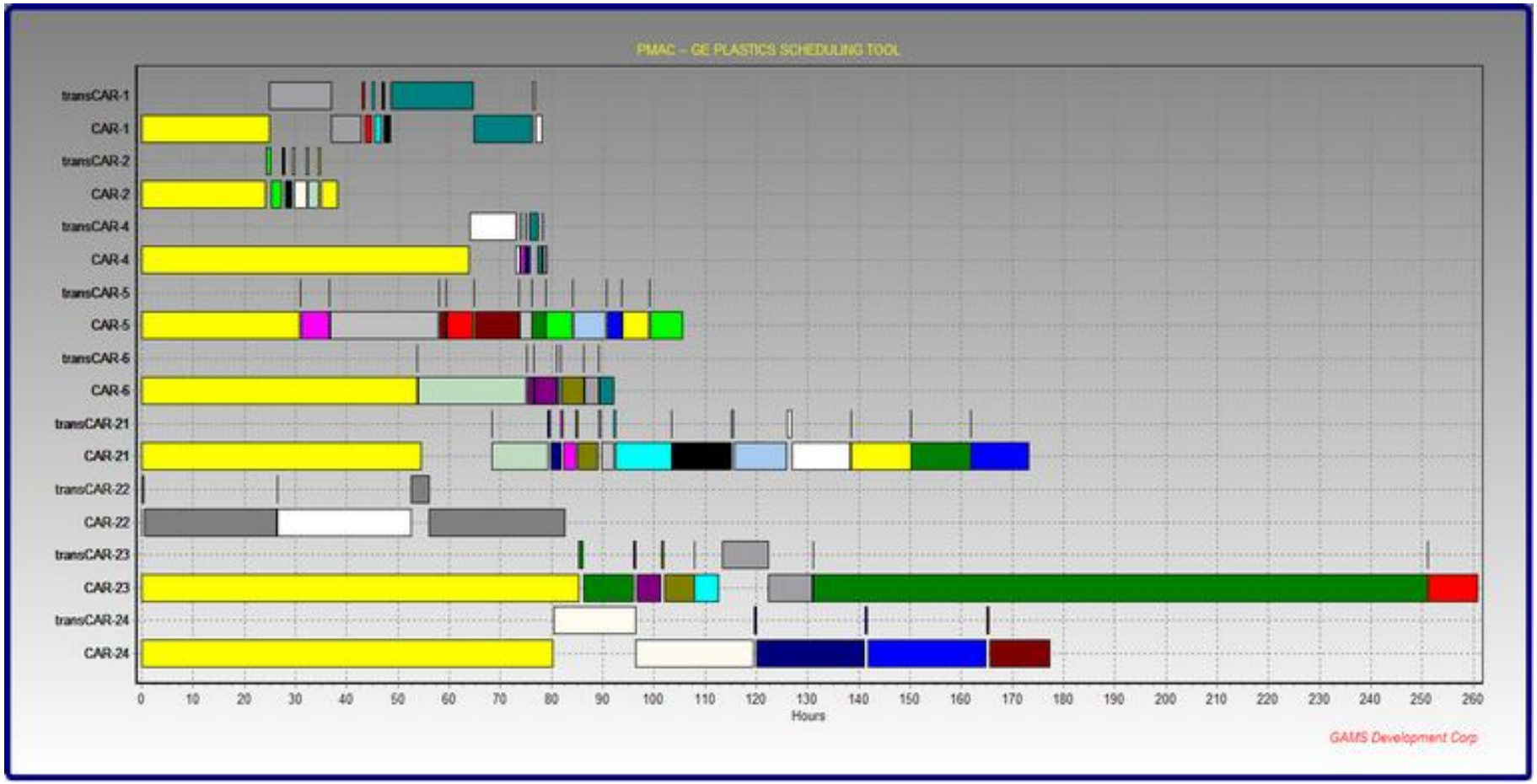


Similar Application

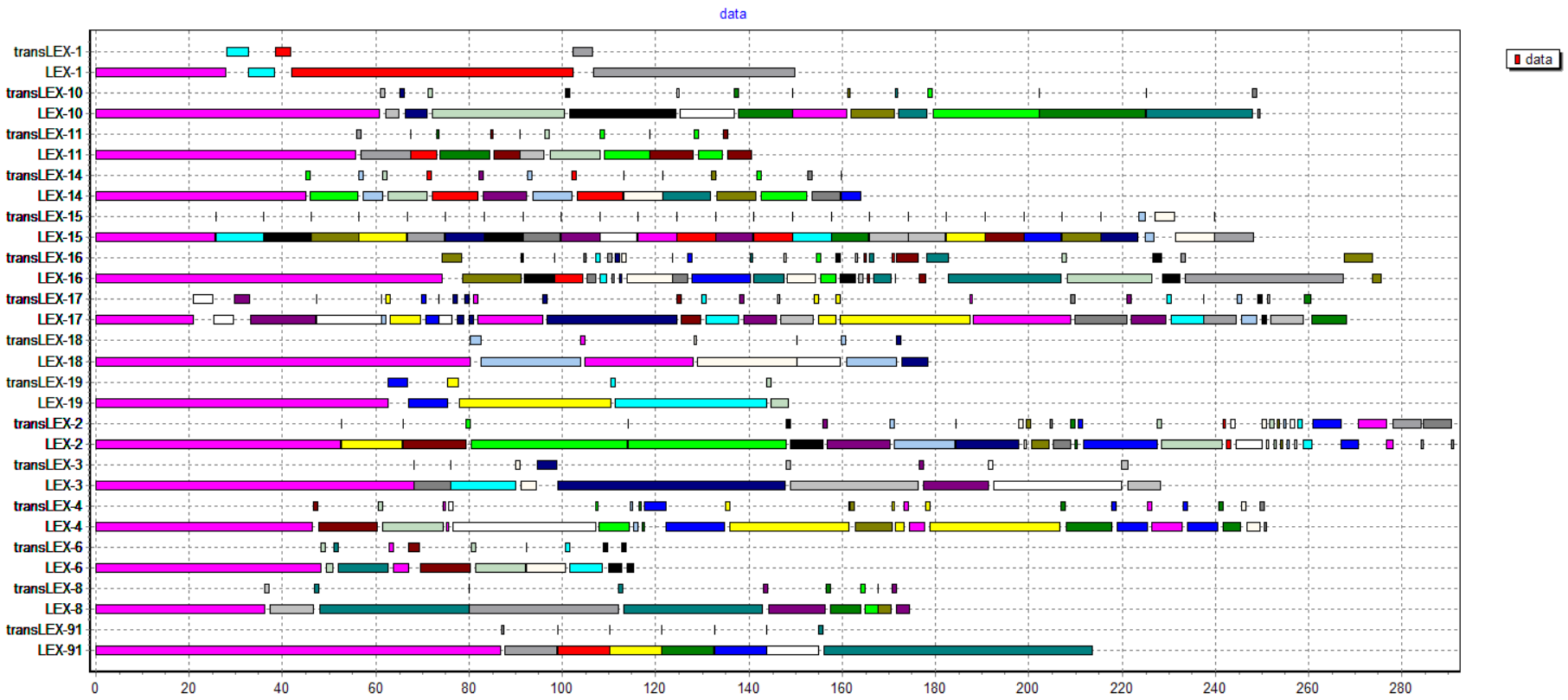
INTERNATIONAL PAPER

- Printing of colored paper
- Color difference determines setup time
  - Close colors are cheap
  - White → anything is cheap
  - Black → anything (except black) is expensive
  - Longer cleaning

Example


PMAC – GE PLASTICS SCHEDULING TOOL
transCAR-1
CAR-1
transCAR-2
CAR-2
transCAR-4
CAR-4
transCAR-5
CAR-5
transCAR-6
CAR-6
transCAR-21
CAR-21
transCAR-22
CAR-22
transCAR-23
CAR-23
transCAR-24
CAR-24
0 10 20 30 40 50 60 70 80 90 100 110 120 130 140 150 160 170 180 190 200 210 220 230 240 250 260
Hours
GAMS Development Corp

A bigger instance



Nonlinear Regression

- Engine: NL2SOL
- Uses GAMS provided gradients
- Not in GAMS distribution
- Can be downloaded

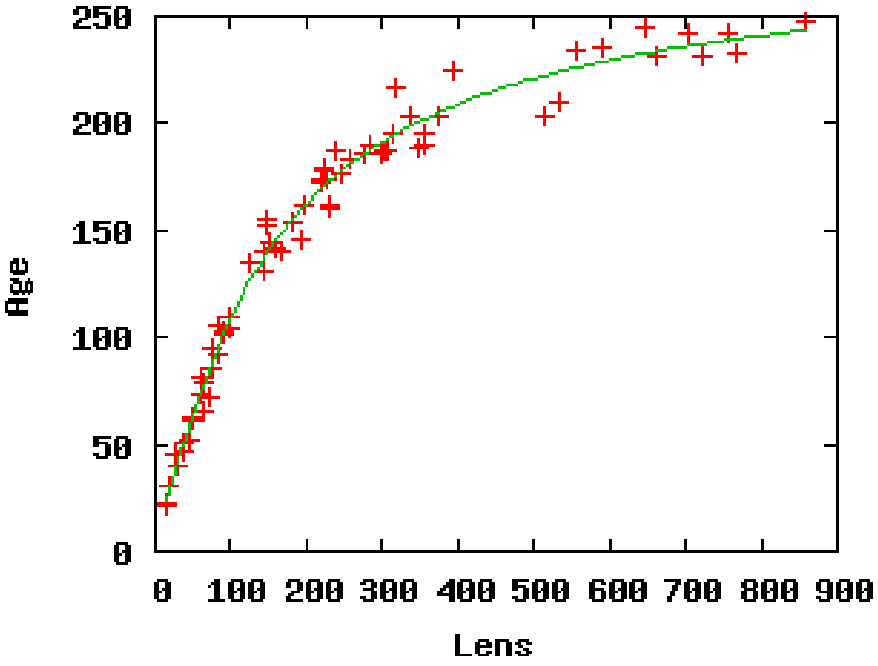
Formulation

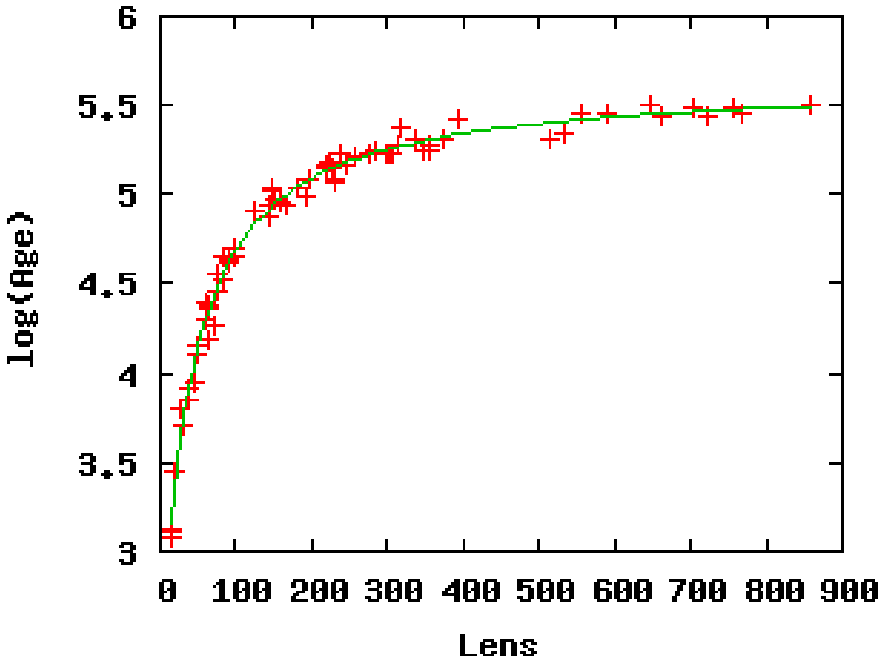
```
parameters
   Age(i)          'Age of rabbit in days '
   Lens(i)         'Dry weight of eye lens in milligrams'
;

Age(i) = data(i,'Age');
Lens(i) = data(i,'Lens');


variables
   a1  'parameter to be estimated'
   a2  'parameter to be estimated'
   a3  'parameter to be estimated'
   sse 'sum of squared errors'
;

equation
   fit(i)      'equation to fit'
   sumsq       'dummy objective'
;

sumsq..    sse =n= 0;
fit(i)..   log(Lens(i)) =e= a1 - a2/(Age(i)+a3);

option nlp=nls;

models m /sumsq,fit/;
solve m minimizing sse using nlp;
```

Example

```
   it    nf      f        reldf      preldf     reldx

    0     1  0.842E+03
    1     2  0.178E+01  0.998E+00  0.998E+00  0.100E+01
    2     3  0.432E+00  0.757E+00  0.836E+00  0.121E+00
    3     4  0.210E+00  0.514E+00  0.406E+00  0.276E-01
    4     5  0.145E+00  0.309E+00  0.323E+00  0.166E-01
    5     6  0.135E+00  0.727E-01  0.729E-01  0.862E-02
    6     7  0.135E+00  0.692E-03  0.691E-03  0.514E-03
    7     8  0.135E+00  0.153E-07  0.153E-07  0.264E-05
    8     9  0.135E+00  0.462E-13  0.466E-13  0.678E-08

X- and relative function convergence.

 function     0.134624E+00   reldx        0.678399E-08
 func. evals       9         grad. evals       9
 preldf       0.465683E-13   npreldf      0.465683E-13
 ==================================================================

        Parameter      Estimate    Std. Error       t value      Pr(>|t|)
               a1   5.63991E+00   1.99670E-02   2.82461E+02   4.09093-106 ***
               a2   1.30584E+02   5.72503E+00   2.28093E+01   1.41196E-33 ***
               a3   3.76029E+01   2.32299E+00   1.61873E+01   5.09560E-25 ***
     ---
     Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

 Residual standard error:   6.29247E-02   on  68 degrees of freedom
```

nls plots


250
200
150
100
50
0
Age
0 100 200 300 400 500 600 700 800 900
Lens


6
5.5
5
4.5
4
3.5
3
log(Age)
0 100 200 300 400 500 600 700 800 900
Lens

nls.gdx

Entry	Symbol	Type	Dim	Nr Elem
1	rss	Par	0	1
2	df	Par	0	1
3	sigma	Par	0	1
4	resvar	Par	0	1
5	estimate	Par	1	3
6	se	Par	1	3
7	tval	Par	1	3
8	pval	Par	1	3
9	resid	Par	1	71
10	covar	Par	2	9
11	confint	Par	3	24
12	grad	Par	1	3
13	jac	Par	2	213

confint: confidence intervals

		LO	UP
90%	a1	5.60661495047289	5.67320794499147
	a2	121.036813881872	140.130602337118
	a3	33.7291008706744	41.4766094822408
95%	a1	5.60006782257375	5.67975507289061
	a2	119.159597276743	142.007818942247
	a3	32.9674001820916	42.2383101708236
97.5%	a1	5.59414461671684	5.68567827874752
	a2	117.461273895637	143.706142323353
	a3	32.2782873299624	42.9274230229527
99%	a1	5.58699714466823	5.69282575079613
	a2	115.411924406663	145.755491812327
	a3	31.4467418851624	43.7589684677527

Bootstrap Statistics

```
*
* get statistics
*
parameter bbar(e) "Averaged estimates";
bbar(e) = sum(s, sb(s,e)) / card(s);

parameter sehat(e) "Standard errors of bootstrap algorithm";
sehat(e) = sqrt(sum(s, sqr(sb(s,e)-bbar(e)))/(card(s)-1));

parameter tbootstrap(e) "t statistic for bootstrap";
tbootstrap(e) = bhat(e)/sehat(e);

scalar df 'degrees of freedom';
df = card(i) - (card(e) - 1) - 1;
parameter pbootstrap(e) "p-values for bootstrap";

*
*  pvalue = 2 * pt( abs(tvalue), df)
*         = 2 * 0.5 * pbeta( df / (df + sqr(abs(tvalue))), df/2, 0.5)
*         = betareg( df / (df+sqr(tvalue)), df/2, 0.5)
*
pbootstrap(e) = betareg( df / (df+sqr(tbootstrap(e))), df/2, 0.5);

parameter bootstrap(e,*);
bootstrap(e,'estimates') = bhat(e);
bootstrap(e,'std.error') = sehat(e);
bootstrap(e,'t value') = tbootstrap(e);
bootstrap(e,'p value') = pbootstrap(e);


display
  "------------------------------------ BOOTSTRAP MODEL -------------------------",
  bootstrap;
```

Database Connectivity

- Mdb2gms
  - Import data from Access data bases
- Sql2gms
  - Import data from other databases
- Gdx2access
  - Export data to Access

Example: USDA Farm database

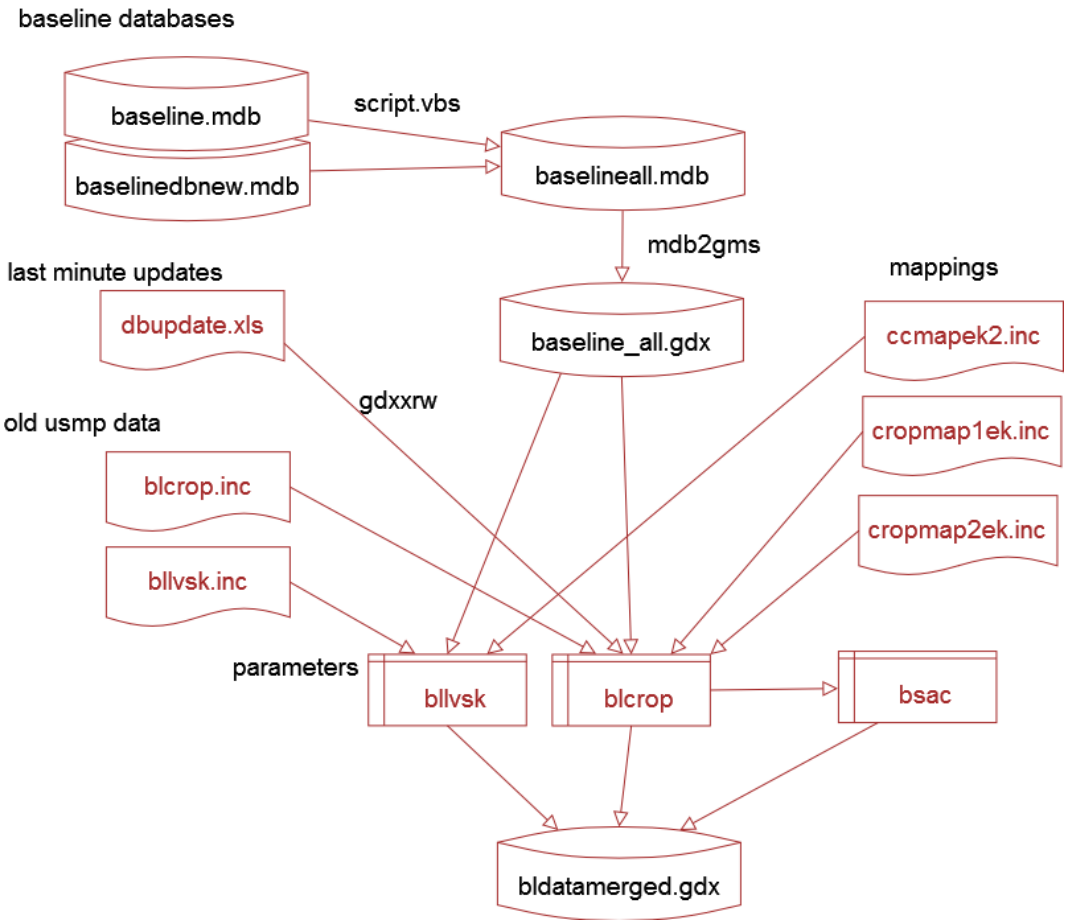
- Imports 3 MDB + 1 XLS file $\rightarrow$ 1 GDX file
  - $\pm$ 5 million records (raw data)

File	Size (bytes)
FARMLandWaterResourcesDraft.mdb	80,650,240
FARMResourcesProductionDraft.mdb	429,346,816
GTAP54-NonLandIntensive.mdb	303,616,000
FIPS2&FAOCountries.xls	29,184
farm.gdx	119,811,434
farm.gdx (compressed)	52,924,664

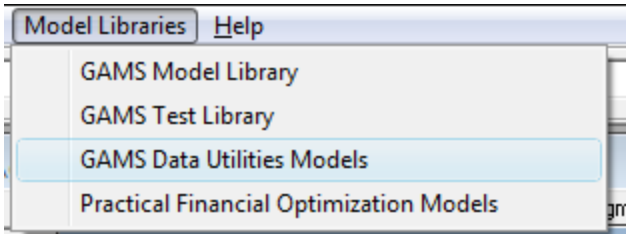
- Good, compact way to distribute and reuse large data sets (input or output)
- Aggregation easier in GAMS than in Access!

Example: USDA Reap Model

- Merge data from different sources



Example



033	SalesProfitDB1	gms, mdb	mdb2gms	Single-valued tables

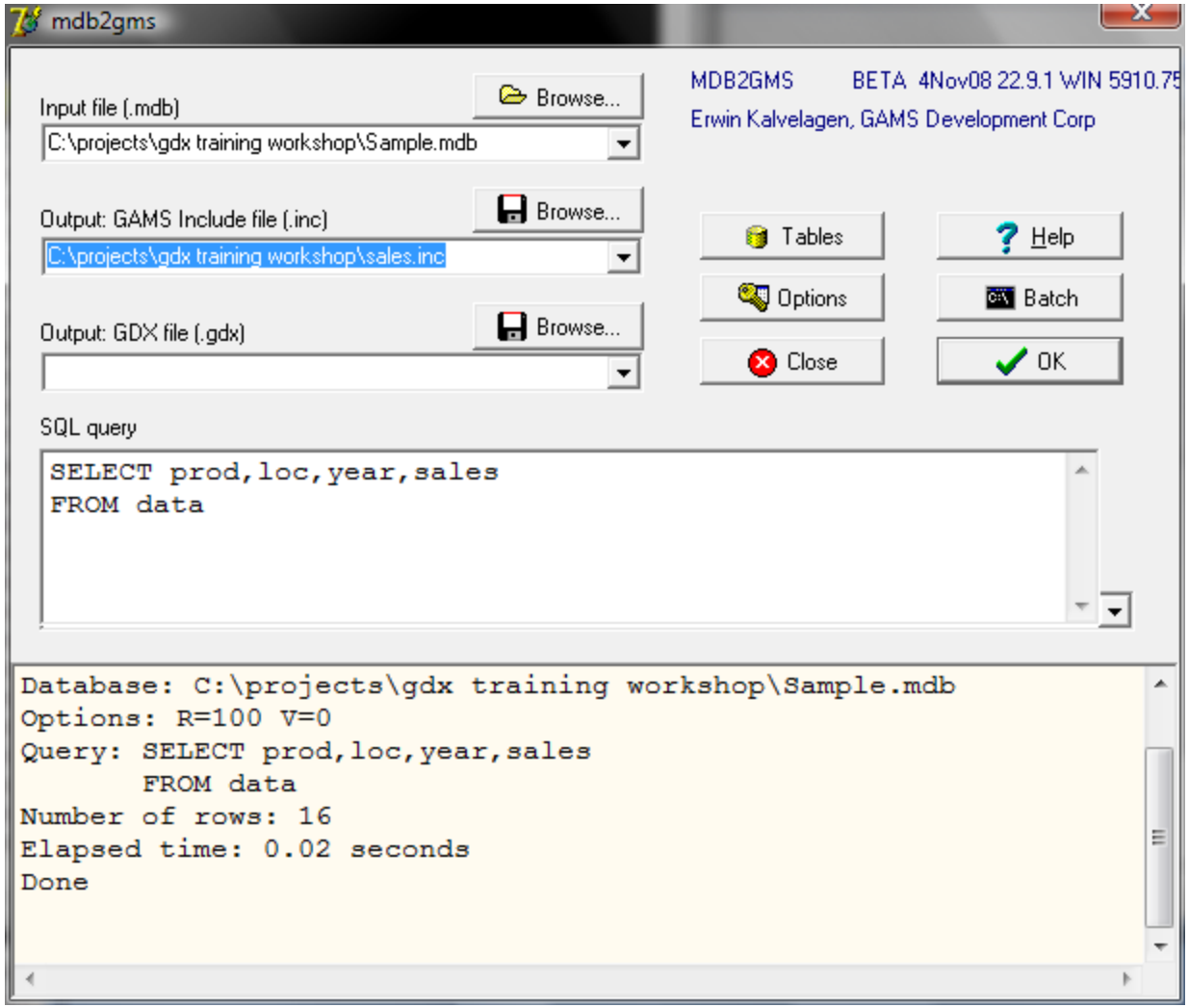
Sample.mdb
Table: data

data

year	loc	prod	sales	profit
1997	la	hardware	80	8
1997	la	software	60	16
1997	nyc	hardware	110	5
1997	nyc	software	100	10
1997	sfo	hardware	80	9
1997	sfo	software	50	10
1997	was	hardware	120	7
1997	was	software	70	20
1998	la	hardware	70	6
1998	la	software	70	10
1998	nyc	hardware	120	7

Record: 1 of 16 No Filter Search

Mdb2gms interactively



Mdb2gms batch

```
set y years /1997*1998/;
set loc locations /nyc,was,la,sfo/;
set prd products /hardware, software/;

parameter sales(prd,loc,y) /
$call =mdb2gms I=Sample.mdb Q="select prod,loc,year,sales from data" O=sales.inc
$include sales.inc
/;
display sales;

parameter profit(prd,loc,y) /
$call =mdb2gms I=Sample.mdb Q="select prod,loc,year,profit from data" O=profit.inc
$include profit.inc
/;
display profit;
```

Response file

```
set y   'years'      /1997*1998/;
set loc 'locations'  /nyc,was,la,sfo/;
set prd 'products'   /hardware, software/;
set q   'quantities' /sales, profit/;

$onecho > cmd.txt
I=Sample.mdb
Q=select prod,loc,year,'sales',sales from data \
  union \
  select prod,loc,year,'profit',profit from data
O=salesprofit.inc
$offecho

$call =mdb2gms @cmd.txt

parameter data(prd,loc,y,q) /
$include salesprofit.inc
/;
display data;
```

Multiple queries

```
$onecho > cmd.txt
I=Sample.mdb

Q1=select distinct(year) from data
O1=year.inc

Q2=select distinct(loc) from data
O2=loc.inc

Q3=select distinct(prod) from data
O3=prod.inc

Q4=select prod,loc,year,sales from data
O4=sales.inc

Q5=select prod,loc,year,profit from data
O5=profit.inc
$offecho
```

```
$call =mdb2gms @cmd.txt

set y years /
$include year.inc
/;
set loc locations /
$include loc.inc
/;
set prd products /
$include prod.inc
/;

parameter sales(prd,loc,y) /
$include sales.inc
/;
display sales;

parameter profit(prd,loc,y) /
$include profit.inc
/;
display profit;
```

Via GDX File

```
$onecho > cmd.txt
I=Sample.mdb
X=sample.gdx

Q1=select distinct(year) from data
s1=year

Q2=select distinct(loc) from data
s2=loc

Q3=select distinct(prod) from data
s3=prd

Q4=select prod,loc,year,sales from data
p4=sales

Q5=select prod,loc,year,profit from data
p5=profit
$offecho
```

```
$call =mdb2gms @cmd.txt

$call =shellExecute gdxviewer sample.gdx

set y 'years';
set loc 'locations';
set prd 'products';
parameter sales(prd,loc,y);
parameter profit(prd,loc,y);

$gdxin 'sample.gdx'
$load y=year loc prd sales profit


display sales;
display profit;
```

ERS Problem

- Convert feedgrains database to GAMS
- MDB2GMS

tblFG_UpDate

FG_Key	PUB	TABL	YEAR	PERIOD	COMMODITY	ATTRIBUTE	GEOCODE	UNIT	VALUE	
JANOATAMFPCUSRP	FED	13		JAN	OAT	AMFP	CUS	RP	0.055	FED13OA
AUGBARAPFMCUSPB	FED	13	1908	AUG	BAR	APFM	CUS	PB	0.57	FED13BAI
APROATAPFMCUSPB	FED	13	1908	APR	OAT	APFM	CUS	PB	0.5	FED13OA
DECCORAPFMCUSPB	FED	13	1908	DEC	COR	APFM	CUS	PB	0.61	FED13CO
DECBARAPFMCUSPB	FED	13	1908	DEC	BAR	APFM	CUS	PB	0.56	FED13BAI
AUGOATAPFMCUSPB	FED	13	1908	AUG	OAT	APFM	CUS	PB	0.48	FED13OA
NOVBARAPFMCUSPB	FED	13	1908	NOV	BAR	APFM	CUS	PB	0.55	FED13BAI
NOVCORAPFMCUSPB	FED	13	1908	NOV	COR	APFM	CUS	PB	0.62	FED13CO
MAYOATAPFMCUSPB	FED	13	1908	MAY	OAT	APFM	CUS	PB	0.51	FED13OA
OCTBARAPFMCUSPB	FED	13	1908	OCT	BAR	APFM	CUS	PB	0.55	FED13BAI
NOVOATAPFMCUSPB	FED	13	1908	NOV	OAT	APFM	CUS	PB	0.47	FED13OA
OCTOATAPFMCUSPB	FED	13	1908	OCT	OAT	APFM	CUS	PB	0.47	FED13OA
OCTCORAPFMCUSPB	FED	13	1908	OCT	COR	APFM	CUS	PB	0.68	FED13CO
JUNOATAPFMCUSPB	FED	13	1908	JUN	OAT	APFM	CUS	PB	0.51	FED13OA
MAROATAPFMCUSPB	FED	13	1908	MAR	OAT	APFM	CUS	PB	0.49	FED13OA
DECOATAPFMCUSPB	FED	13	1908	DEC	OAT	APFM	CUS	PB	0.48	FED13OA
JANOATAPFMCUSPB	FED	13	1908	JAN	OAT	APFM	CUS	PB	0.47	FED13OA
JULBARAPFMCUSPB	FED	13	1908	JUL	BAR	APFM	CUS	PB	0.58	FED13BAI
JULOATAPFMCUSPB	FED	13	1908	JUL	OAT	APFM	CUS	PB	0.5	FED13OA

Record: 1 of 378347 Unfiltered Search

Conversion mdb -> gdx

```
$onecho > cmd.txt
I=FeedGrainsData.mdb
X=FeedGrainsData.gdx


q1=select commodity from tblCommodities
s1=commodity

q2=select attribute from tblAttributes
s2=attribute

q3=select period from tblTimePeriods
s3=period

q4=select unit from tblUnits
s4=unit

q5=select distinct(iif(isnull(isource),'blank',isource)) \
   from tblFG_update where \
   not isnull(year)
s5=isource

q6=select geo from tblgeography
s6=geocode

q7=select commodity,attribute,unit,iif(isnull(isource),'blank',isource),geocode,year,period,value \
   from tblFG_update where \
   not isnull(year)
p7=feedgrains
$offecho

$call mdb2gms @cmd.txt
```

Usage

```
set
   year /1908*2009/
   period
   geocode
   commodity
   attribute
   unit
   isource
;

parameter feedgrains(commodity,attribute,unit,isource,geocode,year,period);

$gdxin  FeedGrainsData.gdx
$load commodity,attribute,unit,isource,geocode,period
$load feedgrains


parameter barley(attribute,year);

barley(attribute,year) = feedgrains('BAR',attribute,'AAM','38001a','CUS',year,'ANN');

display barley;
```

Typical Problems

- NULL's
- Duplicate records
- Multivalued tables
- More difficult processing:
  - Get latest available number
    - Difficult in SQL and in GAMS

comm	year	forecasty	value
corn	2010	2006	12
corn	2010	2007	12.5
barley	2010	2005	9
barley	2010	2006	9.1

comm	year	value
corn	2010	12.5
barley	2010	9.1

Sql2gms

- Works with any database
  - Odbc
  - ADO
- Such as:
  - Oracle
  - SQL Server
  - Interbase
  - Mysql
  - IBM DB2
  - Etc.

gdx2access

- Dumps gdx file to an MDB file

```
set
  i /i1*i10/
  j /j1*j100/
;

parameter a(i,j);
a(i,j) = uniform(0,1);

execute_unload 'a.gdx',a;
execute 'gdx2access a.gdx';

execute 'shellexecute a.mdb';
```

IO matrix

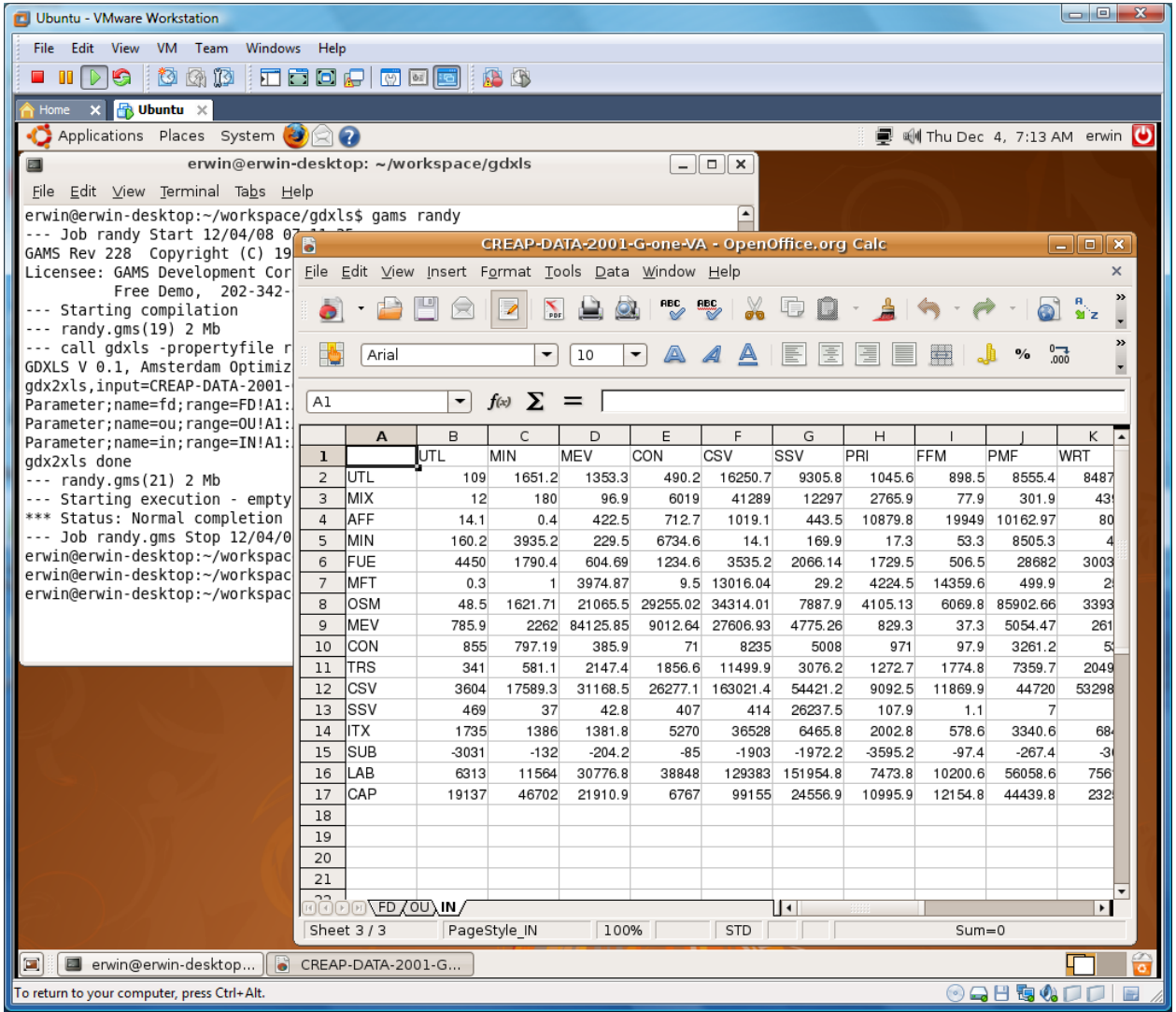
```
* column headers in spreadsheet
set col /1*430/;

parameters makea(ic,col),makeb(ic,col);

$CALL  'GDXXRW make1.xls se=0 index=Index!A1'
$gdxin make1.gdx
$LOADdc makea

$CALL  'GDXXRW make2.xls se=0 index=Index!A1'
$gdxin make2.gdx
$LOADdc makeb

display makea,makeb;
                *
                * combine makea and makeb
                *
                * check one of them is always zero
                abort$sum((ic,col),makea(ic,col)*makeb(ic,col)) "oops, wrong data";

                parameter makec(ic,col);
                makec(ic,col) = makea(ic,col)+makeb(ic,col);

                display makec;
```

IO Matrix 2

```
*
* map col -> ic
*
set mapper(ic,col);
mapper(ic,col)$(ord(ic)=ord(col)) = yes;
display mapper;


alias(ic,ic2)
parameter make(ic,ic2);

* alternative 1
* make(ic,ic2) = sum(mapper(ic2,col), makec(ic,col));

* alternative 2
loop(mapper(ic2,col),
  make(ic,ic2) = makec(ic,col);
);
```

1111A0	1	Y
1111B0	2	Y
111200	3	Y
1113A0	4	Y
111335	5	Y
111400	6	Y
111910	7	Y
111920	8	Y
1119A0	9	Y
1119B0	10	Y

Linux



	A	B	C	D	E	F	G	H	I	J	K
1		UTL	MIN	MEV	CON	CSV	SSV	PRI	FFM	PMF	WRT
2	UTL	109	1651.2	1353.3	490.2	16250.7	9305.8	1045.6	898.5	8555.4	8487
3	MIX	12	180	96.9	6019	41289	12297	2765.9	77.9	301.9	43
4	AFF	14.1	0.4	422.5	712.7	1019.1	443.5	10879.8	19949	10162.97	80
5	MIN	160.2	3935.2	229.5	6734.6	14.1	169.9	17.3	53.3	8505.3	4
6	FUE	4450	1790.4	604.69	1234.6	3535.2	2066.14	1729.5	506.5	28682	3003
7	MFT	0.3	1	3974.87	9.5	13016.04	29.2	4224.5	14359.6	499.9	2
8	OSM	48.5	1621.71	21065.5	29255.02	34314.01	7887.9	4105.13	6069.8	85902.66	3393
9	MEV	785.9	2262	84125.85	9012.64	27606.93	4775.26	829.3	37.3	5054.47	261
10	CON	855	797.19	385.9	71	8235	5008	971	97.9	3261.2	5
11	TRS	341	581.1	2147.4	1856.6	11499.9	3076.2	1272.7	1774.8	7359.7	2049
12	CSV	3604	17589.3	31168.5	26277.1	163021.4	54421.2	9092.5	11869.9	44720	53298
13	SSV	469	37	42.8	407	414	26237.5	107.9	1.1	7	
14	ITX	1735	1386	1381.8	5270	36528	6465.8	2002.8	578.6	3340.6	68
15	SUB	-3031	-132	-204.2	-85	-1903	-1972.2	-3595.2	-97.4	-267.4	-3
16	LAB	6313	11564	30776.8	38848	129383	151954.8	7473.8	10200.6	56058.6	756
17	CAP	19137	46702	21910.9	6767	99155	24556.9	10995.9	12154.8	44439.8	232
18											
19											
20											
21											

Applications

- Excel, Access are good tools to build applications around
- They have enough GUI widgets to allow to build front-ends

Portfolio

Microsoft Excel - Portfolio2.xls

File Edit View Insert Format Tools Data Window Help DPlot

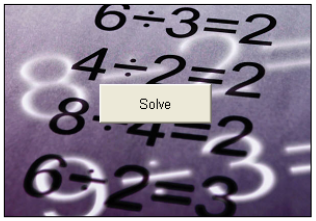
W29

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	0.873
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

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

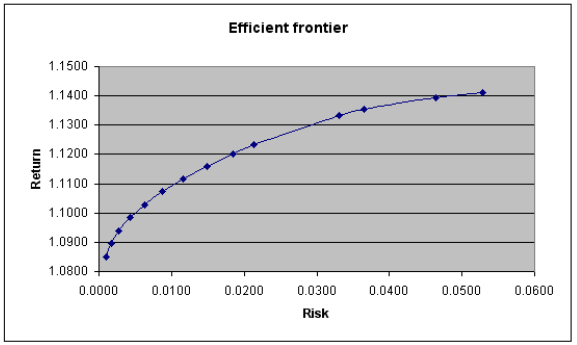


FRONTIER(λ)

$$\begin{array}{ll} \underset{x}{\text{minimize}} & x^T Q x - \lambda r^T x \\ \text{subject to} & \sum_i x_i = 1 \\ & x \geq 0 \end{array}$$

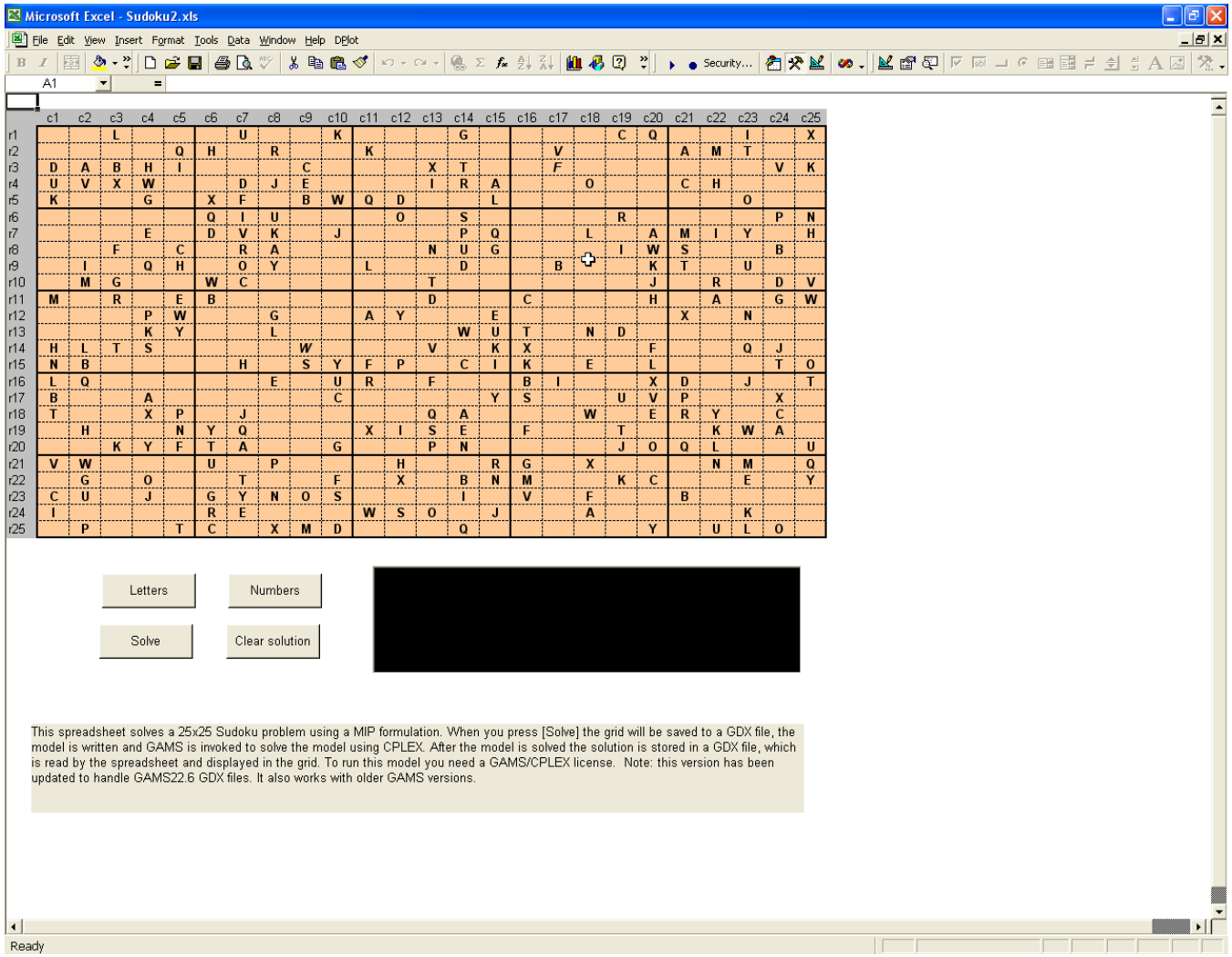
Optimal portfolios (percentages)

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

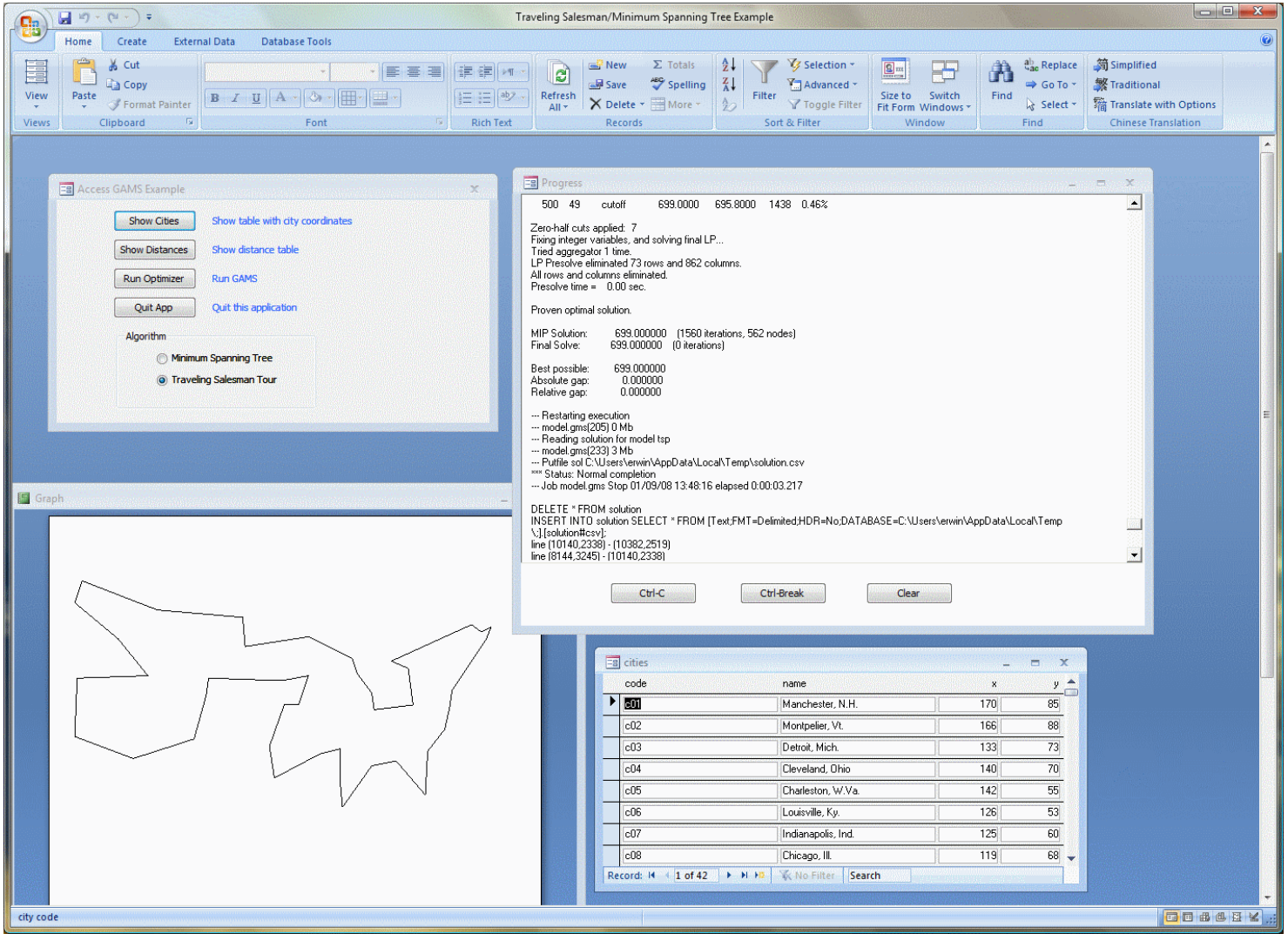


Ready

Sudoku


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

Access


Traveling Salesman/Minimum Spanning Tree Example
Access GAMS Example
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
Progress
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)
Ctrl-C
Ctrl-Break
Clear
Graph
cities
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
Record: 1 of 42
No Filter
Search
city code

Scenario Analysis

CleanAir8.xls [Compatibility Mode]

Systematic Sensitivity Analysis

item	description	default	values				
eta	*Demand elasticity of textile exports*	*10*	7.00	8.00	9.00	10.00	11.00
epsilon	*Supply elasticity of cotton imports*	*10*	7.00	8.00	9.00	10.00	11.00
etaL	*Elasticity of transformation in land*	*10*	7.00	8.00	9.00	10.00	11.00
tm0	*Base year tariff on grain*	*0.54*	0.54				

Run

Help

Clean Up

SSA | PSA | Configuration | ScenarioSetti

Gams Form

Scenario 125 of 125

```
Maximum of X. . . . . . . . . .  4.1373e+001 var: (E.tex)
Maximum of F. . . . . . . . . .  5.5788e-011 eqn: (M.grn)
Maximum of Grad F . . . . . . .  5.2284e+002 eqn: (PC)
                                             var: (PC)

 ** EXIT - solution found.

Major Iterations. . . . 6
Minor Iterations. . . . 6
Restarts. . . . . . . . 0
Crash Iterations. . . . 0
Gradient Steps. . . . . 0
Function Evaluations. . 7
Gradient Evaluations. . 7
Total Time. . . . . . . 0.007000
Residual. . . . . . . . 5.654577e-011


--- Executing MCP subsolver PATH

--- Restarting execution
--- model.gms(982) 0 Mb
--- Reading solution for model UK1841
--- model.gms(1021) 3 Mb
--- Putfile MPS D:\office\qtool\225a\gamscge.scr
--- GDX File D:\office\qtool\Output\EK_125.gdx
*** Status: Normal completion
--- Job model.gms Stop 01/28/08 12:30:55 elapsed 0:00:00.252
```

Pivot Tables

Close

Pivot Tables