Working with the DMSolver simulation/estimation software on the four EUROKIN test cases 12/05/07

Copyright (c) 2007 by F. D. Singleton, Digital Analytics, fdails@DAofTX.com, www.DAofTX.com

The EUROKIN authors (Berger, Hoorn, Verstraete, and Verwijs, www.EUROKIN.org/ paper_par_est_1.pdf) created or adapted the four EUROKIN test cases ".... to create a thorough inventory of modeling packages suitable for parameter estimation and capable of describing two or more dimensional reactor models."

It happens that the EUROKIN kinetics test cases fit mostly within the specifications of DMSolver's capabilities, but are rather challenging in their estimation aspects compared with previous simple estimation problems that were used to test DMSolver. Consequently in going from version 5.1 to 5.2 of DMSolver the estimation routines were revised and details of them improved to the point that results have been obtained for most parts of the four test cases.

The DMSolver least-squares estimation routines are called "add-ins" because the primary type of problem addressed by DMSolver is simulation by solution of simultaneous systems (such as used to model reactors), with parameter estimation optional. The add-in code is extra code linked to the simultaneous solver code. There are two estimation add-ins. The dynamic one is for when one is trying to fit data which are functions of time (e.g. batch reactor) with a mathematical model involving integration of differential equations. The static one is for fitting a set of observations with a static model involving a system of simultaneous algebraic equations (e.g. CSTR). In EUROKIN's Case study 3 for the first time a case was encountered in which there was no simultaneity, just a single equation. Minor changes were made so that DMSolver simultaneous solution code would be bypassed without incident.

Because of its capability for simulating larger complex systems, DMSolver input includes source code which is preprocessed, compiled, and linked so that large simulation problems are computed efficiently.

The DMSolver estimation routines compute numerical derivatives to linearize (in a local region) the nonlinear regression problem, and they output typical linear-style regression results including the variance/covariance matrix and a canonical variable analysis, both based on the linearization. To find a solution for the nonlinear regression problem, the estimation routines look for the minimum in the sum of squared deviations using a Gauss-Newton, or (as a fall-back) sometimes a gradient approach. It is also possible to request the typical regression results for the linearized problem even though the minimum has not been found. These results are not the correct answers but are sometimes useful for diagnostic purposes when it is difficult to find the minimum. Also, in the current version we have added a "grid search" mode which simply varies each parameter in equal steps over a specified range and computes the sum of squares. Sorting on the sum of squares may give a first trial set of values at which the search can start.

The test cases are presented at the website in www.EUROKIN.org/paper_par_est_1.pdf and its associated appendix. In this report we assume the reader refers to a copy of this paper, and we do not try to describe the test cases in detail.

In the first three test cases, the EUROKIN authors or their predecessors assumed a set of kinetic parameters, and generated artificial "experimental" data to which random values were added to simulate real experimental data. Thus the authors know the correct answers for the kinetic parameters and can judge estimation results accordingly. In the fourth test case, real experimental data were provided from earlier research.

In all the test cases, the authors specified very completely the mathematical formulation of the problems, so engineering judgement in formulating the problems is not supposed to be involved, but in the 4th test case we reduced the number of finite difference intervals to speed up the investigation, and we deviated from certain details in the 1st test case. In many cases we applied scaling factors to produce parameters with similar orders of magnitude.

Case study 1

For Case study 1 we did not follow exactly the directions since we did not use the proportional weighting method suggested. Also the batch reactor and CSTR were only run separately as DMSolver does not allow the combined approach.

1A The Batch Experments

The beginning of the listing of results for the 4 batch experiments reads:

```
LSQ Converged
Stopped after  7 Lsq iterations
Number of observations =  40
   Weighted sum of sq. err= 1.0790729E-02
   Overall R - squared    =  0.99878

Fitted Y variable          Mean        Rms error      R - squared   Weight
 ConcA                  2.792E-01   1.114354E-02      0.9989130   1.00E+00
 ConcB                  5.022E-01   1.264720E-02      0.9973654   1.00E+00
 ConcC                  2.065E-01   4.721992E-03      0.9997081   1.00E+00
 ConcD                  2.000E-03   1.370843E-04      0.9966547   1.00E+02
Parameter                  Value           Est Std error        T-ratio
 kkSc1                      1.0462890E-02   5.036493E-05   2.077416E+02
 kkSc2                      1.8625744E-03   1.481167E-05   1.257504E+02
 kkSc3                      1.5013959E-03   2.451480E-04   6.124446E+00
 Easc1                      6.3541945E-01   2.644627E-03   2.402681E+02
 Easc2                      4.0811112E-01   3.143255E-03   1.298371E+02
 Easc3                      9.2170265E-01   5.580905E-02   1.651529E+01
```

This was a very easy problem for the DMSolver least squares routine.

Applying the scaling factors to these results gives

$k_{ref1} = 0.0105$

$k_{ref2} = 0.00186$

$k_{ref3} = 0.0000150$

$E_{a1} = 63542$

$E_{a2} = 40811$

$E_{a3} = 92170$

All the parameters are very significantly different from zero.

Listing 1 in the appendix shows the user source code for the batch reactor problem. Procedure ADDSET provides the interface to the estimation routine. The data to be fit are in the database file EKFits under the data series directory heading EK1A3. Various named data series in this storage location are connected to variable names in the program by procedure calls in ADDSET. There are four time-integration runs named Batch1, ..., Batch4, and appropriate initial condition files were prepared with these names. Parameter scaling occurs in the IF _BEFORESOLVE statement.

Procedure PARSET has optional calls which set up a convenient GUI interface for changing various parameters manually if you want to run a reactor simulation independently of the estimation procedure.

Procedure SETUP is called before each reactor simulation run.

Procedures named so far are written in conventional (Borland-style) Pascal language, but defining a simultaneous system requires special notation involving what are called XPROC and MPROC procedures. There are 4 XPROC -type procedures to define the forms of the simultaneous equations and one MPROC-type procedure with calls to define instances of the equations in the simultaneous system. The whole simultaneous system is always defined by a procedure named SYSTEM which is the last item in the source code. There are 8 simultaneous variables. An introduction to DMSolver simultaneous systems programming methodology appears in the Digital Analytics website.

1B CSTR Experiments

The beginning of the results listing for the continuous reactor case, with 10 CSTR runs is:

```
LSQ Converged
Stopped after  8 Lsq iterations
Number of observations =  10; from    1 to   10 with    0 deleted
   Weighted sum of sq. err= 2.8367636E-04
   Overall R - squared    =  0.99988

Fitted Y variable          Mean         Rms error      R - squared   Weight
 ConcA                      5.985E-01   7.004891E-03      0.9998776   1.00E+00
 ConcB                      4.527E-01   2.326867E-03      0.9999145   1.00E+00
 ConcC                      5.899E-02   1.488570E-03      0.9997297   1.00E+00
 ConcD                      9.445E-04   1.913915E-05      0.9996810   1.00E+02
Parameter                  Value           Est Std error        T-ratio
 kkSc1                      9.5228801E-03   2.878041E-05   3.308806E+02
 kkSc2                      1.8692912E-03   2.274461E-05   8.218611E+01
 kkSc3                      1.4054618E-03   2.168866E-04   6.480169E+00
 Easc1                      6.0005919E-01   3.207517E-03   1.870790E+02
 Easc2                      3.8760602E-01   1.238545E-02   3.129527E+01
 Easc3                      8.9797081E-01   6.590503E-02   1.362522E+01
```

As can be seen results are similar to those of the batch case but with an even closer fit. With only 8 iterations it was also a very easy problem. The smallest T value is about 6.5, so all the parameters are determined to a relatively high degree of certainty.

The source code is listed in the Appendix. There are 6 simultaneous variables.

Case study 2

Case Study 2 involves 12 runs of a fixed-bed reactor under various conditions. The source code is listed in the Appendix, and there are 9 simultaneous variables.

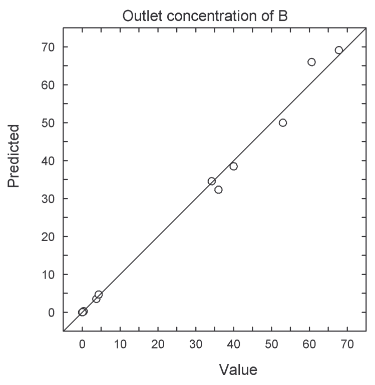
In Case Study 2 several of the parameters for which one is searching are highly interacting, and it is very difficult to determine values for some of them with any certainty. With this situation, the DMSolver least-squares algorithm had difficulty deciding that it reached a least squares point and several times continued to run until reaching an iteration limit (usually 50 iterations) or was stopped manually. Several runs were made. A representative output is

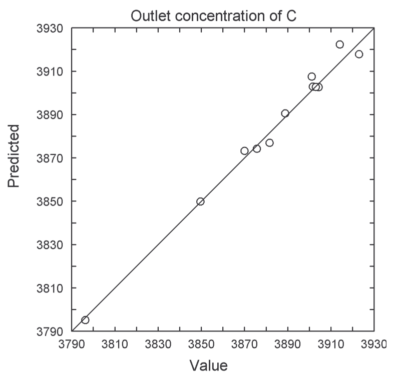
```
Stopped after 72 Lsq iterations
14784 Total dynamic integration runs
Number of observations =  12
   Weighted sum of sq. err= 1.6657428E+02
   Overall R - squared    =  0.99474

Fitted Y variable           Mean        Rms error      R - squared   Weight
CCB                          2.501E+01   2.642329E+00      0.9927164   1.00E+00
CCC                          3.884E+03   4.725586E+00      0.9862265   1.00E-03
CCD                          6.454E+02   3.717197E+00      0.9953872   1.00E+00
Parameter                   Value          Est Std error        T-ratio
kk1                          2.7716009E+02   4.001082E+01   6.927128E+00
kk2                          2.6596102E+02   2.751158E+02   9.667241E-01
K1mScaled                    4.1563004E+02   3.296342E+03   1.260883E-01
K2mScaled                    4.2900409E+02   1.668001E+03   2.571966E-01
K3mScaled                    1.0000000E+02   4.949798E+02   2.020285E-01
```

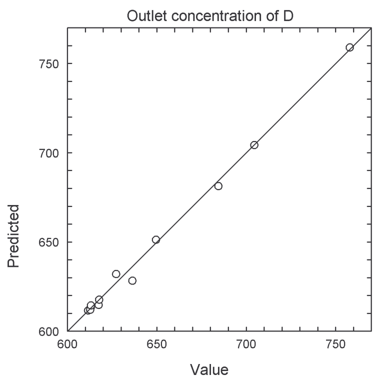
Removing the scaling factors gives

```
kk1 = 277       K1m = 4.2     K3m = 0.0010
kk2 = 266       K2m = 0.43
```

In the case of K3m the parameter was against the lower bound which was given in the problem statement. The statistical calculations done by DMSolver's least-squares routine show that kk1 is the only variable that a statistician would consider "significant" i.e. determined with relatively little uncertainty. kk2 is less-well determined and the other 3 K's are extremely poorly determined, and a statistician would remove them from the model, even though the mathematical model fits the results very well (overall R-squared = 0.99474). The data fits are to the three concentrations (of compounds B, C, and D) which are achieved at the reactor outlet. These can be displayed graphically as







Perfect fit occurs if a point falls exactly on the 45-degree line.

Note that with some poorly-determined parameters as indicated by the T-ratio values obtained in the linearized problem results, there were at least 10 times as many nonlinear search iterations than in Problem 1, where all the parameters were well-determined.

Case study 3

In this study there are 20 alternative mathematical models, each consisting of only one rate equation. Since there is only one equation there are no simultaneous variables and the entire DMSolver simultaneous solver mechanism is unnecessary and gets bypassed when the program runs, although DMSolver demands that the user go through the motions of setting up a simultaneous problem.

The results can be summarized as follows:

Summary of results listed by equation ("Significant"= <5% chance of being zero)

Eqn	Sum of Sq. of Err .	r^2	Signif. Param. / Total Param.	Comments
1	9.0699770E-13	0.99050	5 / 5	
2	9.0246291E-13	0.99055	5 / 5	
3	1.6383011E-12	0.98285	3 / 5	
4	5.1935121E-13	0.99456	5 / 5	
5	5.5694960E-13	0.99417	5 / 5	
6	7.6713921E-13	0.99197	5 / 5	
7	7.6713921E-13	0.99197	5 / 5	Same model as 6
8	6.6548898E-11	0.30329	0 / 5	
9	1.3464778E-12	0.98590	3 / 5	
10	1.4673276E-11	0.84638	2 / 5	
11	1.2348825E-12	0.98707	3 / 4	
12	1.0512817E-12	0.98899	3 / 4	
13	1.1557444E-11	0.87900	1 / 6	
14	5.7693922E-13	0.99396	3 / 6	
15	1.3685293E-13	0.99857	6 / 6	
16	4.9179105E-13	0.99485	3 / 8	
17	6.9249403E-13	0.99275	3 / 8	
18	6.9249403E-13	0.99275	4 / 8	
19	NA		? / 8	
20	3.5424752E-13	0.99629	2 /11	

There are many equations which show relatively good fits (high r-squared), but many of these have non-significant (poorly determined) parameters. It is informative to sort the results by these criteria:

```
These equations have all-significant parameters (< 5% chance of zero)
```

Eqn	Sum of Sq. of Err .	r^2	Signif. Param. / Total Param.	Comments
15	1.3685293E-13	0.99857	6 / 6	
4	5.1935121E-13	0.99456	5 / 5	
5	5.5694960E-13	0.99417	5 / 5	
6	7.6713921E-13	0.99197	5 / 5	
7	7.6713921E-13	0.99197	5 / 5	Same as 6
2	9.0246291E-13	0.99055	5 / 5	
1	9.0699770E-13	0.99050	5 / 5	

```
These equations have one or more non-significant parameters (> 5% chance of zero)
Eqn    Sum of Sq. of Err  .    r^2          Signif. Param. / Total Param.
20    3.5424752E-13   0.99629             2 /11
16    4.9179105E-13   0.99485             3 / 8
14    5.7693922E-13   0.99396             3 / 6
17    6.9249403E-13   0.99275             3 / 8
18    6.9249403E-13   0.99275             4 / 8
12    1.0512817E-12   0.98899             3 / 4
11    1.2348825E-12   0.98707             3 / 4
 9    1.3464778E-12   0.98590             3 / 5
 3    1.6383011E-12   0.98285             3 / 5
13    1.1557444E-11   0.87900             1 / 6
10    1.4673276E-11   0.84638             2 / 5
 8    6.6548898E-11   0.30329             0 / 5
19        NA                              ? / 8     Search difficult, no results
```

Equation 15 shows the very best behavior by these criteria, whereas equations in the second grouping should not be seriously considered as adequate models.

In many cases there were large numbers of nonlinear regression search iterations, but with no simultaneous problem computer cpu time was still small.

With the good behavior of equation 15, there were few search iterations, leading to the following output:

```
Stopped after  9 Lsq iterations
 3861 Total system solutions
Number of observations =  27; from    1 to   27 with    0 deleted
   Weighted sum of sq. err= 1.3685293E-13
   Overall R - squared    =  0.99857

Fitted Y variable          Mean        Rms error      R - squared   Weight
rMeOH                       3.149E-06   7.887070E-08      0.9985673   1.00E+00
Parameter                  Value          Est Std error        T-ratio
kkrefSC                     1.3219044E+00   1.701081E-01   7.770969E+00
EESC                        1.1546282E+00   3.911621E-02   2.951790E+01
K1                          2.0714506E-02   5.078206E-03   4.079099E+00
K3                          2.1946109E-01   3.108163E-02   7.060798E+00
DH3SC                      -4.1506515E-01   1.228759E-01  -3.377921E+00
K4                          2.1989204E-01   4.707813E-02   4.670789E+00
```

With scale factors removed and using the original notation:

$k_{ref} = 1.32*10^{-7}$ $\quad$ $K_1 = 0.0207$

$E_a = 1.15*10^{+5}$ $\quad$ $K_3 = 0.219$

$\Delta H_3 = - 0.415*10^{+5}$ $\quad$ $K_4 = 0.220$

Case study 4

Whereas Case study 3 has zero simultaneous variables, Case study 4 has the most. When the source code (see appendix) was set up and preprocessed, DMSolver found 3994 of them. Since DMSolver is designed with special modular features, the source code is only a little longer than the cases with 8 or 9 simultaneous variables, but of course the CPU time required is unavoidably much longer per iteration.

The first task specified by Case study 4 is to run a simulation with initial test parameters specified by

EUROKIN. This task was done successfully. Basically one conclusion was drawn from this: Case study 4 analyses done as directed would be very time consuming. Therefore, the discretization along the length of the reactor was changed from 500 to 100 intervals and the time discretization criteria were relaxed somewhat, and it was found that a solution similar to the original could be obtained much faster. In this case DMSolver finds 794 simultaneous variables. A typical well-behaved simulation takes about 20 seconds. (Admittedly if the equipment available to run the analyses ---Pentium III 733MHz--- were newer we might not have made such a major change.)

When one lets DMSolver run the least-squares linearization analysis using the initial test parameters specified by EUROKIN, it is obvious that these test parameters produce a very good fit and are perhaps considered by EUROKIN as THE answers to Case study 4. The initial part of the analysis output is:

```
Number of observations = 180
   Weighted sum of sq. err= 2.2600886E-01
   Overall R - squared    =  0.99086

Fitted Y variable          Mean        Rms error      R - squared   Weight
ThetaZ1                     1.064E+00   2.368565E-02      0.8839171   1.00E+00
ThetaZ2                     1.182E+00   1.252673E-02      0.9914098   1.00E+00
ThetaZ3                     1.192E+00   6.633642E-03      0.9977140   1.00E+00
ThetaZ4                     1.187E+00   1.024785E-02      0.9946783   1.00E+00
ThetaZ5                     1.177E+00   9.963080E-03      0.9949960   1.00E+00
ThetaZ6                     1.169E+00   8.745504E-03      0.9961202   1.00E+00
ThetaZ7                     1.163E+00   7.984577E-03      0.9967355   1.00E+00
ThetaEnd                    1.158E+00   1.332998E-02      0.9908066   1.00E+00
Parameter                  Value          Est Std error        T-ratio
gamTemp                     2.0200000E+01   1.313843E-01   1.537474E+02
DarSc                       4.1000000E+01   3.374676E-01   1.214931E+02
PeMrSc                      1.9600000E+01   3.103332E+00   6.315792E+00
PeHrSc                      4.2000000E+00   5.175807E-02   8.114676E+01
UstarSc                     1.6000000E+01   6.392248E+01   2.503032E-01
```

By the usual statistician's reasoning all the parameters except UstarSc are significantly different from zero, whereas the results show that the value of UstarSc is extremely poorly determined. (Sc in a parameter name means it is scaled compared with the actual parameter. Ustar = 10*UstarSc)

In subsequent nonlinear search trials starting from different starting points such as the one specified by EUROKIN, the search algorithm would not go near this point again. We decided to perform an exercise as if the only guidance from EUROKIN were the upper and lower bounds which they give for the parameters, namely (using the scaled values in the DMSolver program)

```
15 <= gamTemp<= 25
10 <= DarSc  <=100
 1 <= PeMrSc <= 25
 1 <= PeHrSc <= 25
10 <= UstarSc<= 20
```

These ranges were used in the "grid search" feature of the DMSLSQ estimation add-in. After a run during which a equi-spaced grid of 108 simulations was traversed, the best case encountered was

```
   gamTemp  =  15
   DarSc    = 100
   PeMrSc   =  25
   PeHrSc   =  13
   UstarSc  =  10

Wt'd Sum of squared errors = 2.9031
equiv r-squared            = 0.8826
```

The run took considerably longer than 108 times typical run time = 36 minutes because for odd unworkable combinations of parameters the algorithm takes longer, especially since it eventually gave up in a few of the 108 cases.

The best grid point case was used as a starting point for a nonlinear regression search. In the grid search the upper and lower bounds are used to form the grid, whereas in the nonlinear regression search the same input numbers are used to limit the space over which the search occurs, but hitting a bound causes the search algorithm to stall badly, so it is necessary to widen the bounds greatly, sometimes to surprising amounts. After several runs between which bounds were widened, the search algorithm arrived at the following linearized results:

```
Number of observations = 180
   Weighted sum of sq. err= 3.7222824E-01
   Overall R - squared    =  0.98495

Fitted Y variable          Mean        Rms error      R - squared   Weight
ThetaZ1                     1.064E+00   2.726881E-02      0.8461384   1.00E+00
ThetaZ2                     1.182E+00   2.522250E-02      0.9651742   1.00E+00
ThetaZ3                     1.192E+00   1.223017E-02      0.9922299   1.00E+00
ThetaZ4                     1.187E+00   9.508406E-03      0.9954186   1.00E+00
ThetaZ5                     1.177E+00   8.821074E-03      0.9960774   1.00E+00
ThetaZ6                     1.169E+00   8.912141E-03      0.9959709   1.00E+00
ThetaZ7                     1.163E+00   9.366873E-03      0.9955073   1.00E+00
ThetaEnd                    1.158E+00   1.581796E-02      0.9870545   1.00E+00
Parameter                  Value          Est Std error        T-ratio
gamTemp                     1.7346845E+01   1.571322E-01   1.103965E+02
DarSc                       4.9153015E+01   5.792363E-01   8.485831E+01
PeMrSc                      1.6372044E+02   2.135104E+02   7.668031E-01
PeHrSc                      3.9332324E+00   5.659902E-02   6.949295E+01
UstarSc                     4.9344530E+01   2.103243E+02   2.346117E-01
```

Comparing this vs the (probably "correct") EUROKIN test parameters we have

```
gamTemp                     17.4 vs  20.2
DarSc                       49.2 vs  41
PeMrSc                     163.7 vs  19.6
PeHrSc                       3.9 vs   4.2
UstarSc                     49.3 vs  16
```

Although the search algorithm was able to raise the equivalent r-squared from 0.8826 to 0.9850 it was not able to approach the value of 0.9909 obtained with the EUROKIN test parameters. Its final linearized results output identified a high level of correlation between the UstarSc and PeMrSc parameters as well as showing that these are poorly determined, and these parameters are the two which differ widely from the EUROKIN test parameters.

Also, even a fragment of linearization output using the grid search results parameters directly shows that UstarSc is not well-determined:

```
Parameter                  Value          Est Std error        T-ratio
gamTemp                     1.5000000E+01   6.077017E-01   2.468316E+01
DarSc                       1.0000000E+02   4.373078E+00   2.286719E+01
PeMrSc                      2.5000000E+01   3.676082E+00   6.800718E+00
PeHrSc                      1.3000000E+01   9.432151E-01   1.378265E+01
UstarSc                     1.0000000E+01   2.002732E+01   4.993180E-01
```

If the analysis is repeated starting from the grid search results but dropping UstarSc as a least-squares parameter the following results are obtained (after roughly an hour). Dropping a parameter simply involves putting comment marks around its call in ADDSET and its scaling statement (if any) and making sure its desired value (in this case, 100 unscaled) is manually entered into its menu.

```
Stopped after  8 Lsq iterations
  147 Total dynamic integration runs
Number of observations = 180
   Weighted sum of sq. err= 1.9508410E-01
   Overall R - squared    =  0.99211

Fitted Y variable          Mean        Rms error      R - squared   Weight
ThetaZ1                     1.064E+00   2.280395E-02      0.8917872   1.00E+00
ThetaZ2                     1.182E+00   1.434493E-02      0.9886712   1.00E+00
ThetaZ3                     1.192E+00   6.009614E-03      0.9981132   1.00E+00
ThetaZ4                     1.187E+00   6.225990E-03      0.9980246   1.00E+00
ThetaZ5                     1.177E+00   6.699178E-03      0.9977247   1.00E+00
ThetaZ6                     1.169E+00   6.552096E-03      0.9978099   1.00E+00
ThetaZ7                     1.163E+00   6.478120E-03      0.9978389   1.00E+00
ThetaEnd                    1.158E+00   1.310418E-02      0.9910649   1.00E+00
Parameter                  Value          Est Std error        T-ratio
gamTemp                     1.9785703E+01   1.022234E-01   1.935536E+02
DarSc                       4.2869673E+01   1.039489E-01   4.124111E+02
PeMrSc                      7.3851223E+01   3.062524E+01   2.411450E+00
PeHrSc                      4.6556722E+00   4.754986E-02   9.791138E+01
```

Again comparing vs the EUROKIN test parameters we have

```
gamTemp                     19.8 vs  20.2
DarSc                       42.9 vs  41
PeMrSc                      73.9 vs  19.6
PeHrSc                       4.7 vs   4.2
UstarSc                     10 assumed vs  16

Weighted sum of sq. err= 0.19508410 vs 0.22600886
Overall R - squared    = 0.99211    vs 0.99086
```

It appears that the values DMSolver selected are "better" than the EUROKIN test values, because the fit error is less and the r-squared higher, but the values of parameters Ustar especially, and PeMr both have considerable uncertainty.

How do we know that UstarSc = 10 ? This is just an assumption which came about from the grid search.

If we were working with conventional linear regression we might want to state a 95% confidence interval for PeMrSc as something like 74 +/- 62 (from the case presented on this page), or perhaps 19.6 +/- 6 from page 7. The page 7 results also would indicate UstarSc = 16 +/- 128. However, with nonlinear regression this would be an extreme oversimplification. The results still indicate sizeable uncertainty, i.e. the parameters are not significant.

In working with Case study 3 and its single equations there was not too much problem with running thousands of grid points and hundreds of least-squares iterations on problems with poorly-determined, insignificant parameters to see if the algorithm could find a minimum. With Case study 4 and its simulation, a strategy had to be developed to avoid excessive computation time. The DMSolver estimation routines are not as automatic as one might want, but a user is able to implement various strategies, as we have demonstrated.

Appendix ---- Program Listings---in Pascal with some special Solver keywords

Listing 1 ---- EUROKIN Problem 1(A) --- Isothermal Batch Reactor

```
VAR    { global variable defs }
   KA,KB,Keq,kk1,kk2,kk3,TK,CCAstart,kkref1,kkref2,kkref3
     ,Ea1,Ea2,Ea3,Easc1,Easc2,Easc3,kksc1,kksc2,kksc3,Tref,RcpTstar:DOUBLE;
CONST
   R:SINGLE=8.314; { J/(mol K) }

PROCEDURE ADDSET; {define least-squares problem and linkage to database }
begin
   _DLSQSETUP('EKFits','EK1A3'); {least-squares database name and DSDir}
   _DLSQRUN('Batch1',0,451,0.01,1,1,10);  {solver integration runs}
   _DLSQRUN('Batch2',0,451,0.01,2,11,20);
   _DLSQRUN('Batch3',0,451,0.01,3,21,30);
   _DLSQRUN('Batch4',0,451,0.01,4,31,40);
   _LSQBETA(kksc1,'kkSc1');  {parameters to be estimated -- scaled}
   _LSQBETA(kksc2,'kkSc2');
   _LSQBETA(kksc3,'kkSc3');
   _LSQBETA(Easc1,'Easc1');
   _LSQBETA(Easc2,'Easc2');
   _LSQBETA(Easc3,'Easc3');
   _LSQINDEP(TK,'TempK');            {independent variables}
   _LSQINDEP(CCAstart,'CCAstart');
   _DLSQDEP(_X.CCA.I,'ConcA');       {dependent variables to be fit}
   _DLSQDEP(_X.CCB.I,'ConcB');
   _DLSQDEP(_X.CCC.I,'ConcC');
   _DLSQDEP(_X.CCD.I,'ConcD');
   IF _BEFORESOLVE THEN{remove scaling factors to get actual parameters}
   begin
      kkref1:=kksc1;
      kkref2:=kksc2;
      kkref3:=0.01*kksc3;
      Ea1:=100000*Easc1;
      Ea2:=100000*Easc2;
      Ea3:=100000*Easc3;
   end;
end;

PROCEDURE SETUP; {set up values before each integration run}
begin
   KA:=EXP((35000/TK - 91)/R);
   KB:=EXP((20000/TK - 53)/R);
   Keq:=4.29E-04*(KA/KB)*EXP(30000/(R*TK));
   Tref:=330;
   RcpTstar:=1.0/TK - 1.0/Tref;
   kk1:=kkref1*EXP(-(Ea1/R)*RcpTstar);
   kk2:=kkref2*EXP(-(Ea2/R)*RcpTstar);
   kk3:=kkref3*EXP(-(Ea3/R)*RcpTstar);
   _X.CCA.I:=CCAstart;
end;

PROCEDURE TIMESET;
begin
   { NO ACTION ---Optional statements setting parameters as
   functions of time _T go here.}
end;

PROCEDURE REPORT;
begin
   { Optional reporting statements here---NONE }
end;
```

```
PROCEDURE PARSET;{to define convenient menus for manual setting of params}
begin
   _PARD(1,'Reactor temp','deg K','TK',TK,330);
   _PARD(3,'Starting A','mol/m3','CCAstart',CCAstart,1.10);
   _PARD(-1,'Rate parameter 1','mol/(m3 s)','kkref1',kkref1,0.05);
   _PARD(-1,'Rate parameter 2','mol/(m3 s)','kkref2',kkref2,0.05);
   _PARD(-1,'Rate parameter 3','mol/(m3 s)','kkref3',kkref3,0.0005);
   _PARD(1,'Activation Energy 1','J/mol','Ea1',Ea1,50000);
   _PARD(1,'Activation Energy 2','J/mol','Ea2',Ea2,50000);
   _PARD(1,'Activation Energy 3','J/mol','Ea3',Ea3,50000);
end;

PROCEDURE USERBPR;
begin
   _LSQGO;{required for all lsq: run least squares when U command given}
end;

(*-----------------------------------------------------------------------------*)
{   XPROC's and MPROC's to define simultaneous problem using special keywords  }
(*-----------------------------------------------------------------------------*)

XPROC DenomCalc(VAROT D:XVR; VARIN A,B:XVR);{simul var D = function of A, B}
VAR DV:DOUBLE;
begin
   DV:= 1.0 + KA*A + KB*B;
   IF DV<1.0E-06 THEN
   begin
      WRITELN('Denom ERR'); DV:=1.0E-06; ERRINT
   end;
   D:=DV;
end;

XPROC ProdCalc(VAROT Rate:XVR; VARIN CC,DD:XVR; kk,K:DOUBLE);
begin
   Rate:= kk*K*CC/DD;
end;

XPROC ARateCalc(VAROT Rate:XVR; VARIN AA,BB,DD,Drate:XVR);
begin
   Rate:= -kk1*KA*(AA-BB/Keq)/DD - Drate;
end;

XPROC MatlBal(VAROT BB:XVR; VARIN AA,CC,DD:XVR);
begin
   BB:=CCAstart - AA - CC - DD;
end;

MPROC SYSTEM; {simultaneous system}
VAR
   CCA,CCB,CCC,CCD,CCAdot,CCCdot,CCDdot,Denom:XVR;
begin
   DenomCalc(Denom, CCA, CCB);
   ARateCalc(CCAdot,CCA,CCB,Denom,CCDdot);
   ProdCalc(CCCdot, CCB, Denom, kk2, KB);
   ProdCalc(CCDdot, CCA, Denom, kk3, KA);
   MatlBal(CCB, CCA, CCC, CCD);
   PMSINTEG(CCA, CCAdot);          {integrator calls}
   PMSINTEG(CCC, CCCdot);
   PMSINTEG(CCD, CCDdot);
end;
```
{ end of source code }

Listing 2   EUROKIN Problem 1(B) --- CSTR Reactor

```
VAR
   KA,KB,Keq,kk1,kk2,kk3,TK,CCAstart,kkref1,kkref2,kkref3
  ,Ea1,Ea2,Ea3,Easc1,Easc2,Easc3,kksc1,kksc2,kksc3,Tref,RcpTstar,Tau:DOUBLE;
   StA,StB,StC,StD:SINGLE;

CONST
   R:SINGLE=8.314; { J/(mol K) }

PROCEDURE ADDSET;
begin
   _LSQSETUP('EKFits','EK1B1',1,10);
   _LSQBETA(kksc1,'kkSc1');
   _LSQBETA(kksc2,'kkSc2');
   _LSQBETA(kksc3,'kkSc3');
   _LSQBETA(Easc1,'Easc1');
   _LSQBETA(Easc2,'Easc2');
   _LSQBETA(Easc3,'Easc3');
   _LSQINDEP(TK,'TempK');
   _LSQINDEP(CCAstart,'CCAstart');
   _LSQINDEP(Tau,'TauSec');
   _LSQDEP(_X.CCA.I,'ConcA',StA);
   _LSQDEP(_X.CCB.I,'ConcB',StB);
   _LSQDEP(_X.CCC.I,'ConcC',StC);
   _LSQDEP(_X.CCD.I,'ConcD',StD);
   IF _BEFORESOLVE THEN
   begin
      kkref1:=kksc1;
      kkref2:=kksc2;
      kkref3:=0.01*kksc3;
      Ea1:=100000*Easc1;
      Ea2:=100000*Easc2;
      Ea3:=100000*Easc3;
   end;
end;

PROCEDURE SETUP;
begin
   KA:=EXP((35000/TK - 91)/R);
   KB:=EXP((20000/TK - 53)/R);
   Keq:=4.29E-04*(KA/KB)*EXP(30000/(R*TK));
   Tref:=330;
   RcpTstar:=1.0/TK - 1.0/Tref;
   kk1:=kkref1*EXP(-(Ea1/R)*RcpTstar);
   kk2:=kkref2*EXP(-(Ea2/R)*RcpTstar);
   kk3:=kkref3*EXP(-(Ea3/R)*RcpTstar);
end;

PROCEDURE TIMESET;
begin
   { Empty because static problem }
end;

PROCEDURE REPORT;
begin
   { Optional reporting statements here }
end;

PROCEDURE PARSET;
begin
   _PARD(1,'Reactor temp','deg K','TK',TK,330);
   _PARD(3,'Starting A','mol/m3','CCAstart',CCAstart,1.10);
   _PARD(-1,'Rate parameter 1','mol/(m3 s)','kkref1',kkref1,0.05);
   _PARD(-1,'Rate parameter 2','mol/(m3 s)','kkref2',kkref2,0.05);
   _PARD(-1,'Rate parameter 3','mol/(m3 s)','kkref3',kkref3,0.0005);
   _PARD(1,'Activation Energy 1','J/mol','Ea1',Ea1,50000);
   _PARD(1,'Activation Energy 2','J/mol','Ea2',Ea2,50000);
   _PARD(1,'Activation Energy 3','J/mol','Ea3',Ea3,50000);
   _PARD(1,'Residence time','sec','Tau',Tau,90);
end;

PROCEDURE USERBPR;
begin
   _LSQGO;
end;

XPROC DenomCalc(VAROT D:XVR; VARIN A,B:XVR);
VAR DV:DOUBLE;
begin
   DV:= (1.0 + KA*A + KB*B)/Tau;
   IF DV<1.0E-06 THEN
   begin
      WRITELN('Denom ERR'); DV:=1.0E-06; ERRINT
   end;
   D:=DV;
end;

XPROC ProdCalc(VAROT Rate:XVR; VARIN CC,DD:XVR; kk,K:DOUBLE);
begin
   Rate:= kk*K*CC/DD;
end;

XPROC AdifCalc(VAROT Rate:XVR; VARIN AA,BB,DD,Dconc:XVR);
begin
   Rate:= -kk1*KA*(AA-BB/Keq)/DD - Dconc;
end;

XPROC MatlBal(VAROT BB:XVR; VARIN AA,CC,DD:XVR);
begin
   BB:=CCAstart - AA - CC - DD;
end;

XPROC ACalc(VAROT AA:XVR; VARIN ADif:XVR);
begin
   AA:=CCAstart + ADif;
end;

MPROC SYSTEM;
VAR
   CCA,CCB,CCC,CCD,CCAdif,Denom:XVR;
begin
   DenomCalc(Denom, CCA, CCB);
   ADifCalc(CCAdif,CCA,CCB,Denom,CCD);
   ACalc(CCA,CCAdif);
   ProdCalc(CCC, CCB, Denom, kk2, KB);
   ProdCalc(CCD, CCA, Denom, kk3, KA);
   MatlBal(CCB, CCA, CCC, CCD);
end;
```

Listing 3   EUROKIN Case study 2 ---- Tubular reactor with mass transfer considerations

```
VAR
   CCA,CCB0,CCC0,CCD0,kk1,kk2,K1m,K2m,K3m,K1mSC,K2mSC,K3mSC:DOUBLE;

PROCEDURE SETUP;
begin
   _X.CCB.I:=CCB0;  {Set concentrations to initial vals at beginning}
   _X.CCC.I:=CCC0;
   _X.CCD.I:=CCD0;
end;

PROCEDURE TIMESET;
begin
   { Optional statements setting parameters as
   functions of time _T go here--none in this problem.}
end;

PROCEDURE REPORT;
begin
   { Optional reporting statements here }
end;

PROCEDURE PARSET;  { Interactive user setting of parameters. }
begin
  _PARD(4,'Concentration of A','','CCA',CCA,2.67);
  _PARD(4,'Inlet concentration of B','','CCB0',CCB0,25.39);
  _PARD(1,'Inlet concentration of C','','CCC0',CCC0,3918);
  _PARD(2,'Inlet concentration of D','','CCD0',CCD0,610.8);
  _PARD(2,'Kinetic constant kk1','','kk1',kk1,300);
  _PARD(2,'Kinetic constant kk2','','kk2',kk2,300);
  _PARD(4,'Kinetic constant K1m','','K1m',K1m,3);
  _PARD(5,'Kinetic constant K2m','','K2m',K2m,0.3);
  _PARD(6,'Kinetic constant K3m','','K3m',K3m,0.003);
end;

PROCEDURE ADDSET;  { Estimation of 5 parameters using 12 expt. runs. }
begin
   _DLSQSETUP('EKFITS','EK21');
   _DLSQRUN('Expt01',0,0.165,0.001,1,1,1);
   _DLSQRUN('Expt02',0,0.165,0.001,2,2,2);
   _DLSQRUN('Expt03',0,0.165,0.001,3,3,3);
   _DLSQRUN('Expt04',0,0.232,0.001,4,4,4);
   _DLSQRUN('Expt05',0,0.232,0.001,5,5,5);
   _DLSQRUN('Expt06',0,0.232,0.001,6,6,6);
   _DLSQRUN('Expt07',0,0.165,0.001,7,7,7);
   _DLSQRUN('Expt08',0,0.165,0.001,8,8,8);
   _DLSQRUN('Expt09',0,0.165,0.001,9,9,9);
   _DLSQRUN('Expt10',0,0.232,0.001,10,10,10);
   _DLSQRUN('Expt11',0,0.232,0.001,11,11,11);
   _DLSQRUN('Expt12',0,0.232,0.001,12,12,12);
   _LSQBETA(kk1,'kk1');
   _LSQBETA(kk2,'kk2');
   _LSQBETA(K1mSC,'K1mScaled');
   _LSQBETA(K2mSC,'K2mScaled');
   _LSQBETA(K3mSC,'K3mScaled');
   _LSQINDEP(CCA,'CCA');
   _LSQINDEP(CCB0,'CCB0');
   _LSQINDEP(CCC0,'CCC0');
   _LSQINDEP(CCD0,'CCD0');
   _DLSQDEP(_X.CCB,'CCB');
   _DLSQDEP(_X.CCC,'CCC');
   _DLSQDEP(_X.CCD,'CCD');
   IF _BEFORESOLVE THEN
   begin
      K1m:=K1mSC/100;  {scaled parameters to actual parameters}
      K2m:=K2mSC/1000;
      K3m:=K3mSC/100000;
   end;
end;

PROCEDURE USERBPR;
begin
   _LSQGO;
end;


XPROC DenomCalc(VAROT Den:XVR; VARIN AAS,BBS,C:XVR);
VAR DDD:DOUBLE;
begin
   DDD:=1.0 + Sqrt(K1M*AAS) + K2m*BBS + K3m*C;
   Den:=DDD*DDD*DDD;
end;

XPROC React1(VAROT RRR1:XVR; VARIN AAS,BBS,Den:XVR);
begin
   IF Den>1.0E-06 THEN RRR1:=kk1*K1m*K2m*AAS*BBS/Den
   ELSE
   begin
      WRITELN('Small denominator');
      ERRINTX(Den);
   end;
end;

XPROC React2(VAROT RRR2:XVR; VARIN AAS,CCC,Den:XVR);
begin
   IF Den>1.0E-06 THEN RRR2:=kk2*K1m*K3m*AAS*CCC/Den
   ELSE
   begin
      WRITELN('Small denominator');
      ERRINTX(Den);
   end;
end;

XPROC Eq1(VAROT Brate:XVR; VARIN BBB,BBS:XVR);
begin
   Brate:=-1442*(BBB-BBS);
end;

XPROC Eq2(VAROT Crate:XVR; VARIN RR1,RR2:XVR);
begin
   Crate:=28.8*(RR1-RR2);
end;

XPROC Eq3R(VAROT AAS:XVR; VARIN RR1,RR2:XVR);
begin
   AAS:=CCA-(RR1+RR2)/9.88;
end;

XPROC Eq4R(VAROT BBS:XVR; VARIN BBB,RR1:XVR);
begin
   BBS:=BBB-(28.8/1442)*RR1;
end;

XPROC MatlBal(VAROT DDD:XVR; VARIN BBB,CCC:XVR);
begin
   DDD:=CCB0+CCC0+CCD0-BBB-CCC;
end;
```

```
MPROC SYSTEM;
VAR
   CCB,CCC,CCD,CCAs,CCBs,CCBdot,CCCdot,R1,R2,Denom:XVR;
begin
   DenomCalc(Denom,CCAs,CCBs,CCC);
   React1(R1,CCAs,CCBs,Denom);
   React2(R2,CCAs,CCC,Denom);
   Eq1(CCBdot,CCB,CCBs);
   Eq2(CCCdot,R1,R2);
   Eq3R(CCAs,R1,R2);
   Eq4R(CCBs,CCB,R1);
   MatlBal(CCD,CCB,CCC);
   PMSINTEG(CCB,CCBdot);
   PMSINTEG(CCC,CCCdot);
end;
```

Listing 4   Typical Case study 3 Program Code --- Equation 15 (the best one)

```
VAR
   kkref,kkrefSC,EE,EESC,K1,K3,DH3,DH3SC,K4,Keq:DOUBLE;
   DumR:SINGLE;

PROCEDURE SETUP;
begin
end;

PROCEDURE TIMESET;
begin
end;

PROCEDURE REPORT;
begin
end;

PROCEDURE PARSET;
begin
end;

PROCEDURE ADDSET;
begin
   _LSQSETUP('EK3DB','EK315A',1,27);
   _LSQBETA(kkrefSC,'kkrefSC');
   _LSQBETA(EESC,'EESC');
   _LSQBETA(K1,'K1');
   _LSQBETA(K3,'K3');
   _LSQBETA(DH3SC,'DH3SC');
   _LSQBETA(K4,'K4');
   _LSQINDEP(_X.pCO.I,'pCO');
   _LSQINDEP(_X.pH2.I,'pH2');
   _LSQINDEP(_X.pMeOH.I,'pMeOH');
   _LSQINDEP(_X.RRTstar.I,'RRTstar');
   _LSQINDEP(Keq,'Keq');
   _LSQDEP(_X.Rate.I,'rMeOH',DumR);
   IF _BEFORESOLVE THEN
   begin
      kkref:=kkrefSC*1.0E-07;
      EE:=EESC*1.0E+05;
      DH3:=DH3SC*1.0E+05;
   end;
end;

PROCEDURE USERBPR;
begin
   _LSQGO;
end;

XPROC EK315eqn(VAROT RR:XVR; VARIN PC,PH,PM,RRT:XVR);
VAR DENOM:DOUBLE;
begin
   DENOM:=(1.0+K1*PC+K3*EXP(-DH3*RRT)*PM+K4*PH/PC)*PC*PH;
   RR:=kkref*EXP(-EE*RRT)*(PC*SQR(PH)-PM/Keq)/DENOM;
end;

MPROC SYSTEM;
VAR
   pCO,pH2,pMeOH,RRTstar,Rate:XVR;
begin
   EK315eqn(Rate,pCO,pH2,pMeOH,RRTstar);
end;
```

Listing 5 EUROKIN Case study 4 ---- Tubular reactor with reaction, dispersion, and wall heat transfer

```
VAR
   PhiV,gamTemp,Dar,PeMr,PeHr,Ustar,DelTadr,omegaH:DOUBLE;
   PhiVA,PsiBA,TimeA,Theta0A:ARRAY[0..180]OF SINGLE;
   HH,HHSQ:SINGLE;
   CValM,CVal0,CValP,TValM,TVal0,TValP:DOUBLE;
   DarSc,PeMrSc,PeHrSc,UstarSc:DOUBLE;

PROCEDURE SETUP;
VAR TF:TextFile; I:INTEGER;
CONST
    NeedTheTimData:BOOLEAN=TRUE;
begin
   IF NeedTheTimData THEN
   begin
      WRITELN('Reading data file');
      ASSIGNfile(TF,'TimVars.txt');
      RESET(TF);
      FOR I:=0 TO 180 DO READLN(TF,TimeA[I],PhiVA[I],PsiBA[I],Theta0A[I]);
      CloseFile(TF);
      HH:=1/100;
      HHSQ:=HH*HH;
      NeedTheTimData:=FALSE;
   end;
   CValP:=1/(PeMr*HHSQ);
   TValP:=1/(PeHr*HHSQ);
end;

procedure GetTimVars(TT:DOUBLE; VAR VV,BB,T0:DOUBLE);
CONST ITM:INTEGER=0; IT:INTEGER=1;
VAR
   FAC,UMF:DOUBLE;
begin
   IF TT<TimeA[ITM] THEN
   begin
      ITM:=0; IT:=1
   end;
   WHILE (TT>=TimeA[IT])AND(IT<180) DO
   begin
      ITM:=IT;
      INC(IT);
   end;
   IF TT=TimeA[ITM] THEN
   begin
      VV:=PhiVA[ITM]; BB:=PsiBA[ITM]; T0:=Theta0A[ITM]
   end
   ELSE IF TT>TimeA[IT] THEN
   begin
      VV:=PhiVA[IT]; BB:=PsiBA[IT]; T0:=Theta0A[IT]
   end
   ELSE
   begin
      FAC:=(TT-TimeA[ITM])/(TimeA[IT]-TimeA[ITM]);
      UMF:=1.0-FAC;
      VV:=FAC*PhiVA[IT]+UMF*PhiVA[ITM];
      BB:=FAC*PsiBA[IT]+UMF*PsiBA[ITM];
      T0:=FAC*Theta0A[IT]+UMF*Theta0A[ITM];
   end;
end;
```

```
PROCEDURE TIMESET; { Time-varying specifications}
begin
   GetTimVars(_T,PhiV,_X.PsiB.I,_X.Theta0.I);
   CValM:=(1/(PeMr*HH)+PhiV)/HH;
   CVal0:=(2/(PeMr*HH)+PhiV)/HH;
   TValM:=(1/(PeHr*HH)+PhiV)/HH;
   TVal0:=(2/(PeHr*HH)+PhiV)/HH;
end;

PROCEDURE REPORT;
begin
   { Optional reporting statements here--none }
end;

PROCEDURE PARSET;{manual changing of parameters}
begin
   _PARD(3,'Activation Temp','dimensionless','gamTemp',gamTemp,20.2);
   _PARD(3,'Damkoehler No.','dimensionless','Dar',Dar,0.41);
   _PARD(3,'Peclet No., Mass','dimensionless','PeMr',PeMr,196);
   _PARD(3,'Peclet No., Heat','dimensionless','PeHr',PeHr,42);
   _PARD(3,'Heat transf. coeff.','dimensionless','Ustar',Ustar,160);
   _PARD(3,'Adiabatic Temp Rise','dimensionless','DelTadr',DelTadr,0.34);
   _PARD(3,'Heat Cap. Ratio','dimensionless','omegaH',omegaH,11.67);
end;

PROCEDURE ADDSET; {define least-squares problem}
begin
   _DLSQSETUP('EK4DBA','EK4A1');
   _DLSQRUN('RunA',0,6,0.001,1,1,180);
   _DLSQDEP(_X.Theta.Z[17].I,'ThetaZ1');
   _DLSQDEP(_X.Theta.Z[39].I,'ThetaZ2');
   _DLSQDEP(_X.Theta.Z[50].I,'ThetaZ3');
   _DLSQDEP(_X.Theta.Z[60].I,'ThetaZ4');
   _DLSQDEP(_X.Theta.Z[70].I,'ThetaZ5');
   _DLSQDEP(_X.Theta.Z[80].I,'ThetaZ6');
   _DLSQDEP(_X.Theta.Z[90].I,'ThetaZ7');
   _DLSQDEP(_X.Theta.Z[100].I,'ThetaEnd');
   _LSQBETA(gamTemp,'gamTemp');
   _LSQBETA(DarSc,'DarSc');
   _LSQBETA(PeMrSc,'PeMrSc');
   _LSQBETA(PeHrSc,'PeHrSc');
   _LSQBETA(UstarSc,'UstarSc');
   IF _BEFORESOLVE THEN
   begin
      Dar:=0.01*DarSc;
      PeMr:=10*PeMrSc;
      PeHr:=10*PeHrSc;
      Ustar:=10*UstarSc;
   end;
end;

PROCEDURE USERBPR;
begin
   _LSQGO; { run LSQ analysis when U key is pressed }
end;


{ Special DMSolver simultaneous system code starts below }
```

```
XVTYPE
   AxialArrayTy=RECORD  Z:ARRAY[1..100]OF XVR   end;

XPROC IDENT1(VAROT B:XVR; VARIN A:XVR);
begin
   B:=A;
end;

XPROC CBDcalc(VAROT CBZdot:XVR; VARIN CBZM,CBZ,CBZP,RR:XVR);
begin
   CBZdot:=CValM*CBZM - CVal0*CBZ + CValP*CBZP - RR;
end;

XPROC THDcalc(VAROT THZdot:XVR; VARIN THZM,THZ,THZP,RR,QQ:XVR);
begin
   THZdot:=TValM*THZM - TVal0*THZ + TValP*THZP - QQ + DelTadr*RR;
end;

XPROC THWDcalc(VAROT THWdot:XVR; VARIN QQ:XVR);
begin
   THWdot:=omegaH*QQ;
end;

XPROC RateCalc(VAROT RR:XVR; VARIN THZ,CBZ:XVR);
begin
   RR:=Dar*CBZ*exp(gamTemp*(1.0-1.0/THZ));
end;

XPROC Qcalc(VAROT QQ:XVR; VARIN THZ,THW:XVR);
begin
   QQ:=Ustar*Dar*(THZ-THW);
end;

MPROC ZCell(ZID:MODID; VAR CBZM,CBZ,CBZP,THZM,THZ,THZP:XVR);
VAR
   CbDot,ThetaDot,ThetaW,ThetaWdot,Rate,Q:XVR;
begin
   CBDcalc(CbDot,CBZM,CBZ,CBZP,Rate);
   THDcalc(ThetaDot,THZM,THZ,THZP,Rate,Q);
   THWDcalc(ThetaWdot,Q);
   RateCalc(Rate,THZ,CBZ);
   Qcalc(Q,THZ,ThetaW);
   PMSINTEG(CBZ,CbDot);
   PMSINTEG(THZ,ThetaDot);
   PMSINTEG(ThetaW,ThetaWdot);
end;

MPROC SYSTEM;
VAR
   PsiB,Theta0:XVR;
   ConcB,Theta:AxialArrayTY;
   _J:SMALLINT;
begin
   ZCell('Z1',PsiB,ConcB.Z[1],ConcB.Z[2],Theta0,Theta.Z[1],Theta.Z[2]);
   MFOR _J:=2 TO 99 DO
      ZCell('ZZ',ConcB.Z[_J-1],ConcB.Z[_J]
                ,ConcB.Z[_J+1],Theta.Z[_J-1],Theta.Z[_J],Theta.Z[_J+1]);
   IDENT1(ConcB.Z[100],ConcB.Z[99]);
   IDENT1(Theta.Z[100],Theta.Z[99]);
end;
```