

ASSESSMENT

MTDATA HANDBOOK ASSESSMENT MODULE



Documentation for the NPL Databank for Materials Thermochemistry

April 2000

NATIONAL PHYSICAL LABORATORY

MTDATA HANDBOOK: ASSESSMENT MODULE

R H Davies, A T Dinsdale, J A Gisby, S M Martin

Centre for Materials Measurement and Technology
National Physical Laboratory, Teddington, Middlesex, TW11 0LW

Abstract

The ASSESSMENT module provides tools to help in the task of deriving values for thermodynamic model parameters, which, when used as a basis for phase equilibrium calculations, give the best possible representation of a collection of experimental data. Any property or combination of properties that can be calculated using MTDATA can feature in an experimental data file. The values of parameters can be stepped to observe their effect or optimised automatically using a choice of algorithms. Calculated results and target data can be compared graphically or by means of tabular output.

© Crown copyright 2000
Reproduced by permission of the Controller of HMSO

ISSN 1361-4061

National Physical Laboratory
Teddington, Middlesex TW11 0LW, UK

Extracts from this handbook may be reproduced provided that the source
is acknowledged and the extract is not taken out of context

Approved on behalf of Managing Director, NPL, by Dr C Lea,
Head of Centre for Materials Measurement and Technology.

IMPORTANT NOTICE

Limitations of MTDATA and responsibilities of users.

Although MTDATA can be used as an aid for teaching thermodynamics, phase equilibria and their interdependency, it is primarily a tool for use by informed research workers and process designers.

The user has a great many options at his/her disposal and may perform calculations on data from a variety of sources. Not all combinations of options nor all combinations of data with these options will have been or can be tested. For this reason it is not possible to be certain that MTDATA will perform any particular calculation correctly and therefore no guarantee is given or implied to that effect.

The user can minimise the risk of error by checking the results of calculations against existing knowledge, for example by making calculations on compositions and for conditions similar to those for which well established experimental data are available or for which experience of plant operation has already been gained.

CONTENTS

1 OVERVIEW

1.1	Purpose	1-1
1.2	Options available and their functions	1-2
1.3	Organisation of this manual	1-4

2 WORKED EXAMPLES

2.1	Preliminary considerations	2-1
2.2	The List command	2-3
2.3	Experimental files	2-6
2.4	Modifying and stepping parameters	2-12
2.5	Optimisation.....	2-14
2.6	Plotting and saving results	2-16

3 EXERCISES

3.1	A simple eutectic.....	3-1
3.2	Miscibility gaps	3-2
3.3	A system with a congruently melting compound	3-2
3.4	Tables and figures.....	3-5

4 OPTION DEFINITIONS

4.1	Define.....	4-1
4.2	List.....	4-4
4.3	Classify	4-13
4.4	Modify_parameter	4-15
4.5	Range	4-18
4.6	Set.....	4-19
4.7	Step_parameter	4-26
4.8	Show.....	4-27
4.9	Plot	4-29
4.10	Save	4-30
4.11	Compute	4-33
4.12	Optimise	4-35
4.13	Laser.....	4-37
4.14	Restore	4-38
4.15	Undo_optimisation	4-38
4.16	Return	4-39
4.17	Swap.....	4-39
4.18	End_of_experiments	4-39
4.19	<"Macro name">	4-39

5 QUICK REFERENCE SECTION

6 SUMMARY OF COMMANDS

1 OVERVIEW

1.1 PURPOSE

The ASSESSMENT module provides tools to help in the task of deriving values for thermodynamic model parameters, which, when used as a basis for phase equilibrium calculations, give the best possible representation of a collection of experimental measurements obtained for a particular chemical system. This overall task of gathering data and generating values for parameters is referred to as *data assessment* or sometimes *critical assessment* since it involves comparing data of many different types, from many different sources, obtained using many different experimental techniques. The specific task of generating model parameters appropriate to a set of experimental data is called *optimisation*.

Experimental data for a particular system are supplied to the ASSESSMENT module via an experimental data file. Any property or combination of properties that can be calculated using MTDATA can feature in this file, which takes the form of a macro giving instructions for the calculation of each property and associating each calculation with an experimental value and uncertainty (using the LIST command). Properties typically considered in experimental files include activities, heats of mixing, equilibrium phase compositions and phase boundary temperatures and compositions. Best results are obtained where a good mix of thermodynamic measurements and phase equilibrium data are available.

Individual thermodynamic parameters from the data input (.mpi) file are referred to by number (or index) during the process of setting up an optimisation run. The SHOW command is used to reveal the index appropriate to each parameter. Once a parameter's index is known the parameter can be flagged for possible change during an optimisation run using the MODIFY command. Sets of parameters can also be linked together such that any change made to one will be mirrored in its partner(s). This is useful in ensuring that data remain consistent across temperature range boundaries.

A optimisation run based upon a chosen algorithm can be initiated using the OPTIMISE command. During such a run the values of parameters flagged for optimisation are changed automatically to minimise differences between calculated properties defined in the experimental data file and their experimental values. The actual function minimised is the weighted sum of squared errors where an error is defined as the difference between calculated and experimental values of a property divided by the stated experimental uncertainty.

$$\text{Weighted sum of squares of errors} = \sum_{i=1}^n W_i ((C_i - E_i)/U_i)^2$$

where n is the number of properties considered in the optimisation run, C_i is the calculated value of property i , E_i is the experimental value of property i , U_i is the uncertainty associated with E_i and W_i is the weight assigned to property i .

Optimisation runs can be monitored graphically or by means of text output. In the former case plots illustrate changes in the optimised parameters and corresponding changes in the weighted sum of squared errors as they happen. Typically an

assessment will involve many optimisation runs with different numbers of parameters being changed and more experimental data being added at each stage.

As a prelude to an optimisation run it is often useful to obtain a crude impression of the effects of varying individual thermodynamic parameters. This helps in selecting suitable parameters for change. To facilitate this, one or two parameters can be flagged for stepping using the MODIFY command. On issue of the STEP_PARAMETER command the values of these parameters are varied automatically based upon a specified SCALE (magnitude of changes) and number of steps, a comparison between calculated properties and experimental data being made at each stage.

Properties calculated using parameter values resulting from an optimisation run can be compared with experimental data graphically using the PLOT command or in the form of a results table using the LIST command. In order to make more complex calculations using the new parameters it is possible to SWAP to other MTDATA modules while still retaining the optimised data in memory. If the new parameters are found to be unsatisfactory, perhaps due to a lack of experimental data to constrain their values, it is possible to UNDO the effects of an optimisation run and recover the parameter values appropriate at its start.

Optimised values of parameters can be saved in an in-memory store or saved in a data loading file for the purposes of updating an MTDATA database. Alternatively a set up file (MTDATA macro) can be written, allowing the exact state of the existing optimisation to be recreated at a later date.

1.2 OPTIONS AVAILABLE AND THEIR FUNCTIONS

DEFINE

is used to retrieve thermodynamic data for a system to be studied, either from a list of databases or from a previously created data input (.mpi) file. Additionally it is used to provide the name of the file (macro) containing calculation instructions and experimental data for any planned optimisation runs.

LIST

provides the basic tools for setting up an optimisation run in that it allows target experimental values and uncertainties, labels, weights and suitable abscissa settings (for later plotting) to be associated with properties to be calculated. It can also be used, after optimisation, to examine thermodynamic parameter values and differences between calculated properties and experimental data. Alternatively it displays the elements, components, substances, unaries and phases in the currently defined system, their status (classifications) and any initial or equilibrium settings (including ranges) established previously.

CLASSIFY

is used to remove particular components, phases or substances from consideration in calculations or to reintroduce those previously removed, to establish reference states for the thermodynamic properties of components, to flag "PARA" components, the relative amounts of which are to be fixed in all phases in which they appear, and to indicate potential miscibility gaps in phases. It is also allows computational parameters appropriate to open system calculations to be specified and the amounts of components to be set free to change.

MODIFY_PARAMETER

is used to assign new absolute or relative values to thermodynamic parameters, to flag those parameters to be changed during stepping or optimisation, to link parameters together so they are changed in an identical manner and to provide scales for changes to parameters.

RANGE

is used to specify temperature, pressure or composition ranges for consideration in phase boundary searches along with temperature, pressure or composition tolerances indicating the accuracy to which individual boundaries are to be found. Note that currently only temperature can be ranged.

SET

is most commonly used to fix the system temperature, pressure or volume and initial amounts or masses of components, individually or in total, prior to a calculation. It can also be used to fix equilibrium partial pressures, molalities and amounts or masses of individual species, to associate group names with system compositions and to assign values to named user variables.

STEP_PARAMETER

is used to initiate calculations in which the values of one or two thermodynamic parameters are stepped automatically, the level of agreement between calculated properties and experimental data from the current experimental data file being indicated at each stage in terms of the weighted sum of squared errors divided by the sum of weights. This is sometimes a useful prelude to a full optimisation run. The required number of steps must be specified.

SHOW

lists the names, current values and indices of thermodynamic parameters from the current data input (.mpi) file to the screen. Parameters can be selected for listing by type (unary or interaction) or by phase.

PLOT

is used to obtain diagrams comparing calculated properties with experiment. Two types of diagram can be plotted, bar charts of errors (differences between calculated and experimental results divided by experimental uncertainties) or lines joining calculated values associated with the same abscissa variable overlaid with symbols indicating experimental data. Weights are not shown in these diagrams.

SAVE

is used to save the values of fixed parameters, parameters flagged for change during stepping or optimisation and parameters linked to others flagged for change, either to an in-memory store or to a data loading file ready for creating or updating an MTDATA database. Alternatively a set up macro can be written allowing the exact state of the current optimisation to be recreated at a later date.

COMPUTE

is used to initiate phase equilibrium calculations based upon the current settings and classifications. The required type of calculation (stage or combination of stages) and method of initiation (initial point option) can be chosen.

OPTIMISE

is used to initiate an optimisation run in which thermodynamic parameters flagged as optimised or linked are adjusted automatically to minimise differences between calculated properties and experimental data from an experimental data file. An appropriate optimisation algorithm, tolerance, maximum number of comparisons to be made between calculation and experiment and method of monitoring progress (text or graphical output) can be selected.

LASER

is used on some MTDATA platforms to request hard copies of graphs plotted but not previously sent to a printer. It is not used in MTDATA for Windows 9x and NT.

RESTORE

is used to retrieve the values of fixed parameters, parameters flagged for change during stepping or optimisation and parameters linked to others flagged for change, from in-memory store. Existing values are overwritten.

UNDO_OPTIMISATION

negates the effects of the last optimisation run performed. New values for thermodynamic parameters generated during the last optimisation run are overwritten by those current at the start of that run.

RETURN

is used to leave the ASSESSMENT module ready to begin work in another module or to leave MTDATA's character interface altogether. The current optimisation set up is lost unless previously saved.

SWAP

is used to leave the ASSESSMENT module and enter another module, retaining the current optimisation set up. This allows test calculations to be performed in modules other than ASSESSMENT using the current set of optimised parameters.

1.3 ORGANISATION OF THIS MANUAL

- | | |
|-----------|--|
| Section 1 | has attempted to provide an overall view of the purpose of the ASSESSMENT module, its structure and its functions, without the main features being obscured by too much detail. |
| Section 2 | is an introduction to using the options available in the ASSESSMENT module. It contains annotated examples, each directed towards a particular outcome, showing typical combinations of commands used during data assessment work. |
| Section 3 | contains some problems based upon the use of the ASSESSMENT module which you might like to solve. Solutions are not included in this binding but can be provided. |
| Section 4 | gives a full definitive explanation of all of the ASSESSMENT module keywords. |
| Section 5 | provides a quick reference guide to the ASSESSMENT module. |

Section 6 is a simple summary of the commands available within the ASSESSMENT module.

2 WORKED EXAMPLES

The intention of this section is to provide a guided introduction to the various commands and combinations of commands used to set up and execute stepping and optimisation runs within the ASSESSMENT module of MTDATA. This is done through illustrative examples in which typical commands are described and corresponding output is shown in separate tables. Tables of output appropriate to each section of text are given the same number and title as that section. The text should always be read in conjunction with the sample output provided. Actual MTDATA commands issued are shown in italics.

The emphasis in these examples is on commands and techniques specific to the ASSESSMENT module of MTDATA. Some knowledge is assumed of how MTDATA data input (.mpi) files and databases are created, how data are loaded into MTDATA databases and how phase equilibrium calculations are set up and executed using commands such as SET, CLASSIFY and COMPUTE. Please refer to the ACCESS, UTILITY and MULTIPHASE handbooks for guidance on these tasks.

2.1 PRELIMINARY CONSIDERATIONS

The ASSESSMENT module of MTDATA is designed to help in refining the values of thermodynamic parameters from existing data input (.mpi) files such that differences between calculated properties and target experimental data are minimised. This means that every ASSESSMENT session must begin with the loading of a data input (.mpi) file containing preliminary values for all thermodynamic parameters to be considered during that session. New parameters, not present in the loaded .mpi file, cannot be introduced interactively during an ASSESSMENT session.

Typically an initial data input (.mpi) file is created by typing preliminary values for those parameters to be refined into a MTDATA data loading file, populating a temporary database, then performing a database search (see the UTILITY and ACCESS module handbooks). The database search might cover more than one database in order to incorporate well established data items, such as standard unary data for the elements from the UNARY database. Database loading files, used to update values stored in the temporary database, can be created automatically during an ASSESSMENT session.

An experimental file is also required in any ASSESSMENT session involving parameter stepping or optimisation. In practise this means most ASSESSMENT sessions. The experimental file contains instructions for the calculation of individual properties, written as MTDATA commands, and target experimental values. Its name should be given as part of the same command in which the data input (.mpi) file is loaded. For example:

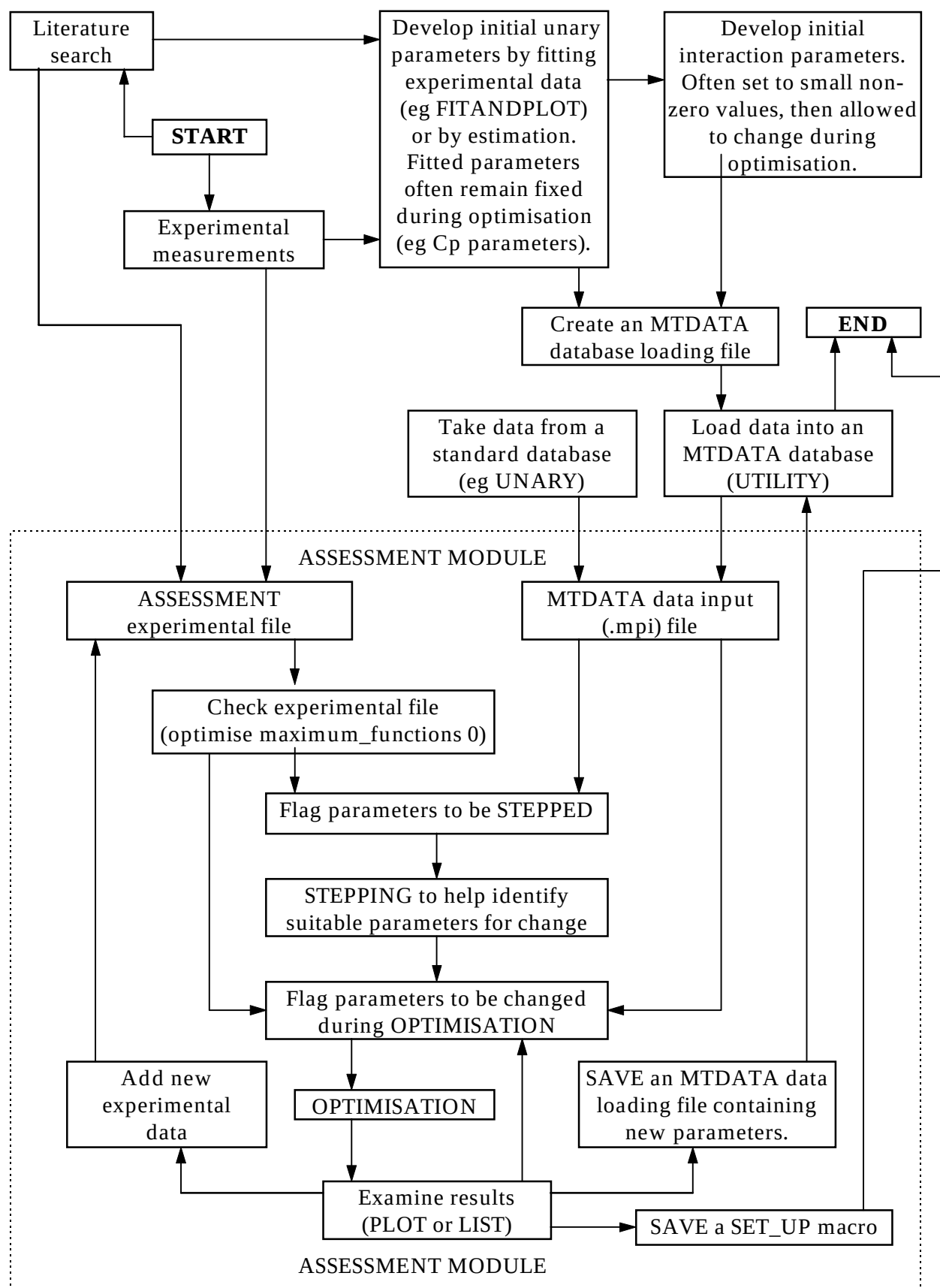
define data 'mydata' experiment 'mydata.amt' !

The structure of experimental files is discussed in Section 2.3.

Prior to the creation of an experimental file it is necessary to collect suitable experimental data for inclusion. This is typically done by means of a literature

search. The commissioning of new measurements, or failing that, estimation, may be necessary where existing data are scarce.

Figure 2.1 DATA ASSESSMENT FLOW CHART



Any property that can be calculated using MULTIPHASE can be included in an experimental data file. However, a combination of phase diagram information (such as phase boundary temperatures, compositions of phases in equilibrium) and thermodynamic measurements (such as activities, enthalpies of mixing) generally leads to the best results.

The flow chart in Figure 2.1 gives an overview of the steps that might be taken in developing suitable thermodynamic parameters for a new system. The preliminary considerations outlined above are shown outside the dotted ASSESSMENT module box as they must be addressed before entering the ASSESSMENT module. The sections that follow deal with tasks performed within the ASSESSMENT module, and so are shown inside the dotted ASSESSMENT module box.

2.2 THE LIST COMMAND

LIST is the ASSESSMENT module command used to associate target experimental data and uncertainties with calculated properties or combinations of properties. A phase equilibrium calculation or a phase boundary search must have been completed prior to using LIST in this way.

```
define system = "KCl,CaCl2" source = demo_1 !
```

Data for the KCl-CaCl₂ system are retrieved from the DEMO_1 database. These data will be used to illustrate the commands used during a typical ASSESSMENT session.

```
set temperature = 900 n(1) = 0.99 n(2) = 0.01 ! compute !
```

Using the same commands as would be entered in MULTIPHASE, an overall system temperature and composition are established, the composition expressed in terms of individual amounts of KCl and CaCl₂, then a calculation is initiated. Unlike in MULTIPHASE no results are automatically printed to the screen or to the results (.mpr) file.

```
list property = x(CaCl2<LIQUID>) !  
list property = gibbs_energy !
```

The calculated mole fraction of component CaCl₂ in the LIQUID phase is accessed and printed to the screen, then the calculated system Gibbs energy is accessed and printed to the screen. Many other calculated properties can be accessed using the LIST PROPERTY command, such as system enthalpies, partial Gibbs energies, partial pressures of gaseous substances, mass fractions of components in phases and component activities.

```
set temperature = 1050 n(1) = 0.8 n(2) = 0.2 !  
classify reference(1) = liquid reference(2) = liquid ! compute !  
list property = h !  
list property = gibbs_energy(KCl) !  
list prop = activity(KCl) !
```

It is sensible to establish reference phases for each component before initiating calculations in which thermodynamic properties such as activities, partial Gibbs energies and enthalpies are to be accessed. The liquid is chosen as the reference

phase for both components in the example above (this could also have been done using the single command *classify reference liquid !*) then the calculated system enthalpy, partial Gibbs_energy with respect to KCl and activity of KCl are accessed and printed to the screen.

```
set user_variable(hold) = h !
set temperature = 900 ! compute !
list property = difference(h:user_variable(hold)) !
```

The calculated value of the system enthalpy is stored in a user variable named “hold”, then a new calculation is performed with a different system temperature. The difference between the new calculated enthalpy and the old value from “hold” is accessed and printed to the screen.

Any property accessible using the LIST PROPERTY command can have its value assigned to a named user variable for future reference. SUM, PRODUCT, RATIO, LOG10 and LN operators are available in addition to DIFFERENCE. Arguments can be properties, user variables or numbers.

```
list property = ln(activity(KCl)) !
```

The natural logarithm of the calculated activity of KCl, reference phase previously established using the CLASSIFY command, is accessed and printed to the screen.

```
range temperature = 1100 500 0.1 ! compute !
list property = temperature(1) !
list property = temperature(2) !
```

If a temperature RANGE is specified, rather than a single temperature SETting, the COMPUTE command initiates a search for phase boundaries within that temperature RANGE, rather than a single phase equilibrium calculation. In the example above the temperatures of boundaries between 1100 K and 500 K are determined to an accuracy of ± 0.1 K. The temperatures of each of the first two boundaries in turn are then accessed and printed to the screen.

```
list property = temp(1) value = 924 uncert = 1 weight = 2 label = 'tliq' !
```

This command accesses the temperature appropriate to the first phase boundary, as above, and prints it to the screen, then associates a target experimental VALUE, an experimental UNCERTAINTY, a LABEL and a WEIGHTing factor (for use during optimisation runs) with the result obtained. The difference between the calculated property and its target value is divided by the specified uncertainty, squared, then multiplied by the specified weight in order to establish a contribution to the “sum of squares of errors”, minimised during optimisation runs.

```
list result = experiment(tliq) !
```

All data associated with the property labelled “tliq”, whether calculated or entered explicitly as part of a LIST command, are tabulated and printed to the screen. If no label or number is provided in brackets then all previously LISTed properties are included in the output.

Note that LISTed properties can also be associated with particular ABSCISSA variables for reference during the plotting of results.

list prop = temp(2) val = 872 unc = 1 lab = 'tsol' abscissa x(CaCl2) !

The temperature appropriate to the second phase boundary is accessed and printed to the screen. It is associated with a particular abscissa variable, $x(\text{CaCl}_2)$, as well as a target value, an experimental uncertainty and a label. This means that, in output requested using PLOT VALUE, its value will be plotted using the mole fraction of component CaCl_2 as ABSCISSA, along with the values of other properties associated with the same ABSCISSA.

Table 2.2 Sample output for THE LIST COMMAND

```
ASSESSMENT OPTION ? define system="KCl,CaCl2" source=demo_1 !
SEARCHING FOR SYSTEM KCl,CaCl2
SEARCHING DATABASE(S) :
demo_1          - Test database - use only with macro 'testmt.mac'
*****
GOOD DATAFILE CREATED
*****
Date and time of run 7-APR-2000 14:50:11
* DATAFILE = C:\MTDATA\Work\def.mpi - CREATED 14:50:11 7-APR-2000
* SYSTEM = KCl,CaCl2,
* NUMBER OF PHASES = 4
* NUMBER OF SPECIES = 5
*
ASSESSMENT OPTION ? set temperature=900 n(1)=0.99 n(2)=0.01 ! compute !
ASSESSMENT OPTION ? list property x(CaCl2<LIQUID>) !
The mole fraction of CaCl2<LIQUID> is .2258012730732503
ASSESSMENT OPTION ? list property=gibbs_energy !
The Gibbs energy is : -536164.039895464
ASSESSMENT OPTION ? set temperature=1050 n(1)=0.8 n(2)=0.2 !
ASSESSMENT OPTION ? classify reference(1)=liquid reference(2)=liquid ! compute !
ASSESSMENT OPTION ? list property=h !
The system enthalpy is : -6831.52000104793
ASSESSMENT OPTION ? list property=gibbs_energy(KCl) !
The partial Gibbs energy of KCl is : -3122.58871327958
ASSESSMENT OPTION ? list property=activity(KCl) !
The activity of KCl is : .6993001720395908
ASSESSMENT OPTION ? set user_variable(hold)=h !
ASSESSMENT OPTION ? set temperature=900 ! compute !
ASSESSMENT OPTION ? list property difference(h:user_variable(hold)) !
H-USER_VARIABLE(hold) = -2.673027E+03
ASSESSMENT OPTION ? list property=ln(activity(KCl)) !
ln(ACTIVITY(KCl)) = -4.828780E-01
ASSESSMENT OPTION ? range temperature=1100 500 0.1 ! compute !
ASSESSMENT OPTION ? list property temperature(1) !
The phase boundary temperature is : 923.6987559042488
ASSESSMENT OPTION ? list property temperature(2) !
The phase boundary temperature is : 871.8755138955712
ASSESSMENT OPTION ? list property=temp(1) value=924 uncert=1 weight=2 label='tliq' !
The phase boundary temperature is : 923.6987559042488

No.    Label    Calculated    Experimental    Uncertainty    Error    Weight
      value (C)    value (E)    (U)    (C-E)/U
10 tliq    9.23699E+02    9.24000E+02    1.0000E+00    -3.01244E-01    2.000E+00
```

ASSESSMENT OPTION ? **list result=experiment(tliq) !**

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
10	tliq	9.23699E+02	9.24000E+02	1.0000E+00	-3.01244E-01	2.000E+00

Mean weighted square of errors of selected experiments
(omitting any undefined values) = 9.07480D-02

ASSESSMENT OPTION ? **list prop=temp(2) val=872 unc=1 lab='tsol' abscissa x(CaCl2) !**

The phase boundary temperature is : 871.8755138955712

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
11	tsol	8.71876E+02	8.72000E+02	1.0000E+00	-1.24486E-01	1.000E+00

2.3 EXPERIMENTAL FILES

Experimental files contain instructions for the calculation of individual properties, written as MTDATA commands, each property being associated with a target experimental value, an uncertainty, a label, a weight (optional) and an abscissa variable for use in the plotting of calculated results (optional). Differences between calculated values and their targets, divided by the uncertainty in the target value, then squared, contribute to the sum of squares of errors which is minimised during optimisation runs.

A typical experimental file consists of a series of command blocks. Each block begins with commands like SET, CLASSIFY and RANGE, used to set up a phase equilibrium or phase boundary calculation. This calculation is executed by means of a COMPUTE command. LIST commands, as introduced in Section 2.2, are then used to associate target values, experimental uncertainties, labels, weights and abscissa variables with selected calculated properties.

An example of an experimental file for the KCl-CaCl₂ system, named "kcacl.amt", is shown in Table 2.3.1. Properties considered include enthalpies of mixing and KCl activities, with the liquid phase as reference, phase boundary compositions and the congruent melting temperature of a compound (KCaCl₃). Note that the latter has been entered in terms of the difference in Gibbs energy between the compound and a liquid of the same composition, set to zero at the congruent melting temperature. An alternative would have been to establish a temperature RANGE and carry out a phase boundary temperature calculation using commands such as those shown in Table 2.3.2. This approach is less favoured because the calculations are likely to be more time consuming. In addition, difficulties could be encountered if the values of individual parameters were to alter during a stepping or optimisation run such that the corresponding boundary temperature moved outside the specified temperature RANGE.

Note that the experimental file "kcacl.amt" is completed by the command *end_of_experiments*. This is preceded by *classify normal phase(*) !* to restore phases classified as absent earlier on.

Table 2.3.1 An experimental file for the KCl-CaCl₂ system - "kcacl.amt"

```

; KCl activities at 1073 K
set temp 1073 ! classify reference(1) liquid reference(2) liquid !
set n(1) 0.9 n(2) 0.1 ! classify absent phase(*) normal phase(liquid) ! compute !
list prop act(KCl) value 0.872 uncert 0.005 weight 1 label "act1" !
set n(1) 0.8 n(2) 0.2 ! classify absent phase(*) normal phase(liquid) ! compute !
list prop act(KCl) value 0.703 uncert 0.005 weight 1 label "act2" !
set n(1) 0.7 n(2) 0.3 ! classify absent phase(*) normal phase(liquid) ! compute !
list prop act(KCl) value 0.522 uncert 0.005 weight 1 label "act3" !
set n(1) 0.6 n(2) 0.4 ! classify absent phase(*) normal phase(liquid) ! compute !
list prop act(KCl) value 0.356 uncert 0.005 weight 1 label "act4" !
set n(1) 0.5 n(2) 0.5 ! classify absent phase(*) normal phase(liquid) ! compute !
list prop act(KCl) value 0.222 uncert 0.005 weight 1 label "act5" !
set n(1) 0.4 n(2) 0.6 ! classify absent phase(*) normal phase(liquid) ! compute !
list prop act(KCl) value 0.127 uncert 0.005 weight 1 label "act6" !
;
; Liquid in equilibrium with KCl<HALITE> and KCaCl3
set n(1) 0.99 n(2) 0.01 ! set t 872.0 !
classify absent phase(*) normal phase(liquid,halite) ! compute !
lis pro x(CaCl2<LIQUID>) val 0.254 un 0.005 we 5 lab "elo1" !
set n(1) 0.51 n(2) 0.49 ! set t 872.0 !
classify absent phase(*) normal phase(liquid,CaCl3K) ! compute !
lis pro x(CaCl2<LIQUID>) val 0.254 un 0.005 we 5 lab "elo2" !
;
; Liquid in equilibrium with KCaCl3 and CaCl2
set n(1) 0.01 n(2) 0.99 ! set t 900.5 !
classify absent phase(*) normal phase(liquid,CaCl2) ! compute !
lis pro x(CaCl2<LIQUID>) val 0.734 un 0.005 we 5 lab "ehi1" !
set n(1) 0.49 n(2) 0.51 ! set t 900.5 !
classify absent phase(*) normal phase(liquid,CaCl3K) ! compute !
lis pro x(CaCl2<LIQUID>) val 0.734 un 0.005 we 5 lab "ehi2" !
;
; Liquid in equilibrium with KCaCl3
set n(1) 0.51 n(2) 0.49 !
classify absent phase(*) normal phase(liquid,CaCl3K) !
set t 893 ! comp ! lis pro x(CaCl2<LIQUID>) val 0.272 un 0.005 we 1 lab "com1" !
set t 916 ! comp ! lis pro x(CaCl2<LIQUID>) val 0.294 un 0.005 we 1 lab "com2" !
set t 931 ! comp ! lis pro x(CaCl2<LIQUID>) val 0.310 un 0.005 we 1 lab "com3" !
set t 967 ! comp ! lis pro x(CaCl2<LIQUID>) val 0.356 un 0.005 we 1 lab "com4" !
set n(1) 0.49 n(2) 0.51 !
set t 935 ! comp ! lis pro x(CaCl2<LIQUID>) val 0.694 un 0.005 we 1 lab "com5" !
set t 971 ! comp ! lis pro x(CaCl2<LIQUID>) val 0.642 un 0.005 we 1 lab "com6" !
;
; Congruent melting point of KCaCl3 (G(KCaCl3)=G(Liquid) at melting point)
set n(1) 0.5 n(2) 0.5 t 1013.1 !
classify absent phase(*) normal phase(CaCl3K) ! compute !
set user_variable(gcomp) g !
classify absent phase(*) normal phase(liquid) ! compute !
list prop difference(g:user_v(gcomp)) value 0.0 uncert 1 weight 5 label "melt" !
;
; Enthalpy of mixing
classify absent phase(*) normal phase(liquid) !
set temp 1000 ! classify ref(1) liquid ref(2) liquid !
set n(1) 0.35 n(2) 0.65 ! compute !
list prop h val -7980 uncert 5 weight 1 label "hmx1" abs x(KCl) !
set n(1) 0.25 n(2) 0.75 ! compute !
list prop h val -6300 uncert 5 weight 1 label "hmx2" abs x(KCl) !
set n(1) 0.15 n(2) 0.85 ! compute !
list prop h val -4100 uncert 5 weight 1 label "hmx3" abs x(KCl) !
;
classify normal phase(*) !
end_of_experiments

```

Table 2.3.2 An alternative way of entering the melting point of KCaCl_3

```
; Congruent melting point of KCaCl3 (phase boundary calculation)
set n(1) 0.5 n(2) 0.5 ! range t 1100 500 0.1 !
classify absent phase(*) normal phase(liquid,CaCl3K) ! compute !
list prop temperature(1) value 1013.1 uncert 0.1 weight 5 lab "melt" !
;
```

It can be difficult to pick out the numeric data supplied in an experimental file from the surrounding MTDATA commands. The experimental file "kcacl.amt" illustrates this well. In order to simplify such files, MTDATA commands to set up and perform particular types of calculation can be entered in separate macros. This means that an experimental file need only contain a list of macros to be called along with any numbers or character data required as input.

Table 2.3.3 A simplified KCl-CaCl_2 experimental file - "kcaclsim.amt"

```
; KCl activities at 1073 K
; macro T/C x(CaCl2) activity uncertainty weight label
"actkcl.mac" 800 0.1 0.872 0.005 1 "act1"
"actkcl.mac" 800 0.2 0.703 0.005 1 "act2"
"actkcl.mac" 800 0.3 0.522 0.005 1 "act3"
"actkcl.mac" 800 0.4 0.356 0.005 1 "act4"
"actkcl.mac" 800 0.5 0.222 0.005 1 "act5"
"actkcl.mac" 800 0.6 0.127 0.005 1 "act6"
;
; Liquid in equilibrium with KCl<HALITE> and KCaCl3
; macro phase T/K x(CaCl2) value uncertainty weight label
"xcac12.mac" p(halite) 872 0.01 0.254 0.005 5 "elo1"
"xcac12.mac" p(CaCl3K) 872 0.49 0.254 0.005 5 "elo2"
;
; Liquid in equilibrium with KCaCl3 and CaCl2
; macro phase T/K x(CaCl2) value uncertainty weight label
"xcac12.mac" p(CaCl2) 900.5 0.99 0.734 0.005 5 "ehi1"
"xcac12.mac" p(CaCl3K) 900.5 0.51 0.734 0.005 5 "ehi2"
;
; Liquid in equilibrium with KCaCl3
; macro phase T/K x(CaCl2) value uncertainty weight label
"xcac12.mac" p(CaCl3K) 893 0.49 0.272 0.005 1 "com1"
"xcac12.mac" p(CaCl3K) 916 0.49 0.294 0.005 1 "com2"
"xcac12.mac" p(CaCl3K) 931 0.49 0.310 0.005 1 "com3"
"xcac12.mac" p(CaCl3K) 967 0.49 0.356 0.005 1 "com4"
"xcac12.mac" p(CaCl3K) 935 0.51 0.694 0.005 1 "com5"
"xcac12.mac" p(CaCl3K) 971 0.51 0.642 0.005 1 "com6"
;
; Congruent melting point of KCaCl3
; macro phase T/K x(CaCl2) value uncert weight label
"meltp1.mac" p(CaCl3K) 1013.1 0.5 0.0 1 5 "melt"
;
; Enthalpy of mixing
; macro T/K x(CaCl2) value uncert weight label abscissa
"enthmx.mac" 1000 0.65 -7980 5 1 "hmx1" x(CaCl2)
"enthmx.mac" 1000 0.75 -6300 5 1 "hmx2" x(CaCl2)
"enthmx.mac" 1000 0.85 -4100 5 1 "hmx3" x(CaCl2)
;
classify normal phase(*) !
end_of_experiments
```

An example of how the experimental file "kcacl.amt" might be simplified using macros called "actkcl.mac", "xcac12.mac", "meltp1.mac" and "enthmx.mac" is given

in Table 2.3.3. The simplified experimental file is called “kcaclsim.amt”. The contents of the macros “actkcl.mac”, “xcac2.mac”, “meltpt.mac” and “enthmx.mac” are shown in Tables 2.3.4, 2.3.5, 2.3.6 and 2.3.7 respectively. Note how, in the macro “actkcl.mac”, the SUM command is used to convert temperatures in Celsius, as entered in the experimental file, to temperatures in kelvin.

The experimental files “kcacl.amt” and “kcaclsim.amt” would lead to identical results if used in optimisation runs.

Table 2.3.4 The macro “actkcl.mac”

```
; Macro to calculate KCl activities with liquid reference
;
classify absent phase(*) normal phase(liquid) !
classify reference liquid !
set user_variable(tccl) *****
!
set u(tkcl) sum(user_variable(tccl):273.15) !
set temperature u(tkcl) !
set n 1 n(1) undef n(2) *****
! compute !
list property activity(KCl) value *****
uncertainty *****
weight *****
label *****
!
```

Table 2.3.5 The macro “xcac2.mac”

```
; Macro to calculate x(CaCl2<LIQUID>) in equilibrium with a specified phase
;
classify absent phase(*) normal p(liquid) normal *****
!
set t *****
!
set n 1 n(1) undef n(2) *****
! compute !
list property x(CaCl2<LIQUID>) value *****
uncertainty *****
weight *****
label *****
!
```

Table 2.3.6 The macro “meltpt.mac”

```
; Macro to calculate the congruent melting point of a specified compound
;
classify absent phase(*) normal *****
!
set n 1 n(1) undef t *****
n(2) *****
!
compute ! set user_variable(gcomp) g !
classify absent phase(*) normal phase(liquid) ! compute !
list property difference(g:user_variable(gcomp)) value *****
uncertainty *****
weight *****
label *****
!
```

Table 2.3.7 The macro "enthmx.mac"

```
; Macro to calculate enthalpy of mixing in liquid phase
;
classify absent phase(*) normal phase(liquid) !
classify reference(1) liquid reference(2) liquid !
set t *****
!
set n 1 n(1) undef n(2) *****
! compute !
list property h value *****
uncertainty *****
weight *****
label *****
abscissa *****
!
```

As already mentioned in Section 2.1, the experimental file to be used during an ASSESSMENT session should be specified as part of the same DEFINE command used to load in thermodynamic data.

define system = "KCl,CaCl2" sou = demo_1 out = 'kcacl' experiment = "kcacl.amt" !

Experimental files should be always be tested for consistency before attempting optimisation runs based upon them. This can be done by entering the command,

optimise maximum_functions = 0 !

Stepping or optimisation with *maximum_functions* > 0 should only be attempted if no errors are reported in the output generated by the above command. If errors are reported, the DEFINEd experimental file should be corrected, then tested again.

The sample output shown in Table 2.3 features typical entries from the results table generated using *optimise maximum_functions = 0 !* The full table is not reproduced.

Table 2.3 Sample output for EXPERIMENTAL FILES

```
ASSESSMENT OPTION ? define sys="KCl,CaCl2" sou=demo_1 experiment="kcacl.amt" !
SEARCHING FOR SYSTEM KCl,CaCl2
SEARCHING DATABASE(S) :
demo_1          - Test database - use only with macro 'testmt.mac'
*****
GOOD DATAFILE CREATED
*****
Date and time of run 7-APR-2000 15:41:03
* DATAFILE = C:\MTDATA\Work\def.mpi - CREATED 15:41:03 7-APR-2000
* SYSTEM = KCl,CaCl2,
* NUMBER OF PHASES = 4
* NUMBER OF SPECIES = 5
*
ASSESSMENT OPTION ? optimise maximum_functions=0 !
The activity of KCl is : .872066284292496

  No.    Label    Calculated    Experimental    Uncertainty    Error    Weight
          value (C)    value (E)    (U)    (C-E)/U
    1 act1    8.72067E-01    8.72000E-01    5.0000E-03    1.33257E-02    1.000E+00
```

The activity of KCl is : .7032910972923411

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
2	act2	7.03291E-01	7.03000E-01	5.0000E-03	5.82195E-02	1.000E+00

The activity of KCl is : .5220842656482606

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
3	act3	5.22084E-01	5.22000E-01	5.0000E-03	1.68531E-02	1.000E+00

The activity of KCl is : .3558643895908951

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
4	act4	3.55864E-01	3.56000E-01	5.0000E-03	-2.71221E-02	1.000E+00

The activity of KCl is : .2223956559107486

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
5	act5	2.22396E-01	2.22000E-01	5.0000E-03	7.91312E-02	1.000E+00

The activity of KCl is : .127062033008948

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
6	act6	1.27062E-01	1.27000E-01	5.0000E-03	1.24066E-02	1.000E+00

The mole fraction of CaCl₂<LIQUID> is .253912782657581

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
7	elo1	2.53913E-01	2.54000E-01	5.0000E-03	-1.74435E-02	5.000E+00

The mole fraction of CaCl₂<LIQUID> is .2539809054967812

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
8	elo2	2.53981E-01	2.54000E-01	5.0000E-03	-3.81890E-03	5.000E+00

The mole fraction of CaCl₂<LIQUID> is .7343676630240455

... etc

The system enthalpy is : -7979.84687253414

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
18	hmx1	-7.97985E+03	-7.98000E+03	5.0000E+00	3.06255E-02	1.000E+00

The system enthalpy is : -6297.42187688802

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
19	hmx2	-6.29742E+03	-6.30000E+03	5.0000E+00	5.15625E-01	1.000E+00

The system enthalpy is : -4101.70687937515

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
20	hmx3	-4.10171E+03	-4.10000E+03	5.0000E+00	-3.41376E-01	1.000E+00

2.4 MODIFYING AND STEPPING PARAMETERS

It is useful, as a prelude to a full optimisation run, to be able to monitor changes in the sum of squares of errors, calculated using the DEFINED experimental file, as the values of one or two thermodynamic parameters are stepped automatically between predefined limits. This is possible using the STEP_PARAMETER command. Firstly, however, the parameters to be stepped have to be selected.

show unary_data = halite !

The names, current values and upper temperature limits of all parameters contributing to data for unaries in the halite phase are listed to the screen. A number, or INDEX, is associated with each parameter.

modify index = 65 type = stepped scale = 50000 !

The parameter corresponding to INDEX 65 is flagged for stepping. It will be varied from its current value minus the specified SCALE to its current value plus the specified SCALE. Note that parameter 65 is valid only up to 700 K. All of its counterparts within the same unary data set, valid over different temperature ranges, must be varied in an identical way. This is achieved by LINKing parameters together.

modify index = 71 link = 65 index = 77 link = 65 !

modify index = 83 link = 65 index = 89 link = 65 !

Parameters 71, 77, 83 and 89 are LINKed to parameter 65. This ensures that any changes made to parameter 65 will automatically be repeated for parameters 71, 77, 83 and 89.

list parameter = value !

The values of all FIXED, STEPPED and OPTIMISED parameters, and parameters LINKed to them, are displayed on the screen. In the case of LINKed parameters, the INDEX of the parameter to which they are LINKed is shown in the column headed Type.

list parameter = name(65) !

Lists the name and temperature limits of the parameter specified by INDEX.

step_parameter number_of_steps = 10 !

Initiates a stepping run during which parameters flagged as STEPPED are varied automatically from their current value minus their SCALE to their current value plus their SCALE using the specified NUMBER_OF_STEPS. The sum of squares of errors appropriate to properties LISTed in the experimental file is reported at each stage. The sensitivity of the sum of squares of errors to changes in the values of particular parameters can be explored in this way and the most suitable parameters for optimisation identified.

Although useful, the *step_parameter* command is no substitute for a sound theoretical understanding of how individual parameters influence calculated thermodynamic properties and phase equilibria.

In the current example, see Table 2.4, marked changes are apparent in the calculated sum of squares of errors as parameter 65 is STEPPED. The minimum sum of squares of errors is associated with the data input (.mpi) value of parameter 65, or a step addition of 0.

Table 2.4 Sample output for MODIFYING AND STEPPING PARAMETERS

ASSESSMENT OPTION ? **show unary_data=halite !**

Index	T(high) /K	Current Value	Parameter Name
65	700.00	-4.52490E+05	CLK<HALITE> *xi
66	700.00	2.63150E+02	CLK<HALITE> *xi*T
67	700.00	-5.12948E+01	CLK<HALITE> *xi*T*ln(T)
68	700.00	-1.40523E-03	CLK<HALITE> *xi*T**2
69	700.00	-1.73100E-06	CLK<HALITE> *xi*T**3
70	700.00	7.67320E+04	CLK<HALITE> *xi/T
71	800.00	-4.87176E+05	CLK<HALITE> *xi
72	800.00	7.62308E+02	CLK<HALITE> *xi*T
73	800.00	-1.27777E+02	CLK<HALITE> *xi*T*ln(T)
74	800.00	7.29682E-02	CLK<HALITE> *xi*T**2
75	800.00	-1.51909E-05	CLK<HALITE> *xi*T**3
76	800.00	3.00201E+06	CLK<HALITE> *xi/T
77	1045.00	-7.29641E+05	CLK<HALITE> *xi
78	1045.00	3.63572E+03	CLK<HALITE> *xi*T
79	1045.00	-5.53395E+02	CLK<HALITE> *xi*T*ln(T)
80	1045.00	4.04611E-01	CLK<HALITE> *xi*T**2
81	1045.00	-6.35871E-05	CLK<HALITE> *xi*T**3
82	1045.00	2.88679E+07	CLK<HALITE> *xi/T
83	1100.00	-9.29276E+06	CLK<HALITE> *xi
84	1100.00	8.37326E+04	CLK<HALITE> *xi*T
85	1100.00	-1.19459E+04	CLK<HALITE> *xi*T*ln(T)
86	1100.00	7.09895E+00	CLK<HALITE> *xi*T**2
87	1100.00	-7.95735E-04	CLK<HALITE> *xi*T**3
88	1100.00	1.22924E+09	CLK<HALITE> *xi/T
89	2000.00	-4.69544E+05	CLK<HALITE> *xi
90	2000.00	4.29820E+02	CLK<HALITE> *xi*T
91	2000.00	-7.33994E+01	CLK<HALITE> *xi*T*ln(T)

ASSESSMENT OPTION ? **modify index=65 type=stepped scale=50000 !**

ASSESSMENT OPTION ? **modify index=71 link=65 index=77 link=65 !**

ASSESSMENT OPTION ? **modify index=83 link=65 index=89 link=65 !**

ASSESSMENT OPTION ? **list parameter=value !**

Index	Type	Scale	Data File Parameter	Current Parameter	Stored Parameter
65	Step	5.000E+04	-4.52490E+05	-4.52490E+05	-4.52490E+05
71	65	5.000E+04	-4.87176E+05	-4.87176E+05	-4.87176E+05
77	65	5.000E+04	-7.29641E+05	-7.29641E+05	-7.29641E+05
83	65	5.000E+04	-9.29276E+06	-9.29276E+06	-9.29276E+06
89	65	5.000E+04	-4.69544E+05	-4.69544E+05	-4.69544E+05

ASSESSMENT OPTION ? **list parameter=name(65) !**

Index	T(low) /K	T(high) /K	Parameter Name
65	298.15	700.00	CLK<HALITE> *xi

ASSESSMENT OPTION ? **step_parameter number_of_steps=10 !**

Table of residual sum squares errors/sum of weights			
Step addn.	Total addn.	Total parameter	Sum of squares of errors
-5.000000E+04	-5.000000E+04	-5.024899E+05	2.663836E+03
-4.000000E+04	-4.000000E+04	-4.924899E+05	2.393326E+03
-3.000000E+04	-3.000000E+04	-4.824899E+05	1.823360E+03
-2.000000E+04	-2.000000E+04	-4.724899E+05	1.071325E+03
-1.000000E+04	-1.000000E+04	-4.624899E+05	3.766122E+02
.000000E+00	.000000E+00	-4.524899E+05	7.521895E-01
1.000000E+04	1.000000E+04	-4.424899E+05	2.984322E+02
2.000000E+04	2.000000E+04	-4.324899E+05	2.984322E+02
3.000000E+04	3.000000E+04	-4.224899E+05	2.984322E+02
4.000000E+04	4.000000E+04	-4.124899E+05	2.984322E+02
5.000000E+04	5.000000E+04	-4.024899E+05	2.984322E+02

2.5 OPTIMISATION

In an optimisation run the values of selected parameters are varied automatically so as to obtain the best possible agreement between calculated properties and target values from an experimental file. This is done by minimising the sum of squares of errors. To illustrate the commands used to set up and execute an optimisation run, a parameter from the current data input (.mpi) file will be set to zero, then optimised.

modify index = 65 type = unchanged !

Parameter 65, previously STEPPED, is now said to be UNCHANGED. This means that it is removed from the list of FIXED, STEPPED, OPTIMISED and LINKed parameters and its data input (.mpi) file value is restored. All parameters LINKed to parameter 65 are treated similarly.

show interaction_data = liquid !

modify index = 52 type = fixed value = 0.0 !

The names, indices, current values and upper temperature limits of all parameters contributing to data for interactions in the liquid phase are listed to the screen. The current value of parameter 52 is FIXED as zero. This is the temperature independent regular solution parameter of the Redlich-Kister model.

list parameter = offset !

The attributes of all FIXED, STEPPED, OPTIMISED and LINKed parameters, just number 52 in this case, are listed to the screen. Note that the value of the parameter is expressed as an OFFSET from that in the data input (.mpi) file. Its actual value would have been shown if LIST PARAMETER VALUE had been entered. Since no SCALE was specified in the MODIFY statement above, a zero SCALE is shown. This triggers the automatic calculation of suitable SCALES for optimisation runs.

modify index = 52 type = optimised !

Parameter 52 is marked for automatic change during subsequent optimisation runs.

optimise algorithm = upolyt maximum_functions = 100 !

An optimisation run, in which the UPOLYT algorithm is used to minimise the sum of squares of errors, is initiated. The process is terminated if convergence has not been achieved after 50 steps, each step resulting in the production of a lower value for the sum of squares of errors.

Table 2.5 Sample output for OPTIMISATION

ASSESSMENT OPTION ? **modify index=65 type=unchanged !**

ASSESSMENT OPTION ? **show interaction_data=liquid !**

Index	T(high) /K	Current Value	Parameter Name
52	3000.00	-3.74554E+04	CaCl2, CLK<LIQUID> *xi*xj
53	3000.00	9.84500E+00	CaCl2, CLK<LIQUID> *xi*xj*T
54	3000.00	8.19000E+03	CaCl2, CLK<LIQUID> *xi*xj*(xi-xj)
55	3000.00	-5.74000E+00	CaCl2, CLK<LIQUID> *xi*xj*(xi-xj)*T
56	3000.00	-9.25000E+02	CaCl2, CLK<LIQUID> *xi*xj*(xi-xj)**2
57	3000.00	1.92900E+00	CaCl2, CLK<LIQUID> *xi*xj*(xi-xj)**2*T

ASSESSMENT OPTION ? **modify index=52 type=fixed value=0.0 !**

ASSESSMENT OPTION ? **list parameter=offset !**

Index	Type	Scale	Data File Parameter	Current Offset	Stored Offset
52	Fix	.000E+00	-3.74500E+04	3.74500E+04	.00000E+00

ASSESSMENT OPTION ? **modify index=52 type=optimised !**

ASSESSMENT OPTION ? **optimise algorithm=upolyt maximum_functions=100 !**

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
1	act1	8.72057E-01	8.72000E-01	5.0000E-03	1.14482E-02	1.000E+00
2	act2	7.03261E-01	7.03000E-01	5.0000E-03	5.21632E-02	1.000E+00
3	act3	5.22034E-01	5.22000E-01	5.0000E-03	6.73774E-03	1.000E+00
4	act4	3.55803E-01	3.56000E-01	5.0000E-03	-3.93792E-02	1.000E+00
5	act5	2.22336E-01	2.22000E-01	5.0000E-03	6.71630E-02	1.000E+00
6	act6	1.27013E-01	1.27000E-01	5.0000E-03	2.56072E-03	1.000E+00
7	elo1	2.53890E-01	2.54000E-01	5.0000E-03	-2.20953E-02	5.000E+00
8	elo2	2.54097E-01	2.54000E-01	5.0000E-03	1.93347E-02	5.000E+00
9	ehi1	7.34397E-01	7.34000E-01	5.0000E-03	7.94281E-02	5.000E+00
10	ehi2	7.34198E-01	7.34000E-01	5.0000E-03	3.96131E-02	5.000E+00
11	com1	2.72174E-01	2.72000E-01	5.0000E-03	3.47930E-02	1.000E+00
12	com2	2.94195E-01	2.94000E-01	5.0000E-03	3.90628E-02	1.000E+00
13	com3	3.10178E-01	3.10000E-01	5.0000E-03	3.56399E-02	1.000E+00
14	com4	3.56661E-01	3.56000E-01	5.0000E-03	1.32239E-01	1.000E+00
15	com5	6.94035E-01	6.94000E-01	5.0000E-03	7.05489E-03	1.000E+00
16	com6	6.41532E-01	6.42000E-01	5.0000E-03	-9.35231E-02	1.000E+00
17	melt	3.30445E-02	.00000E+00	1.0000E+00	3.30445E-02	5.000E+00
18	hmx1	-7.98203E+03	-7.98000E+03	5.0000E+00	-4.06391E-01	1.000E+00
19	hmx2	-6.29922E+03	-6.30000E+03	5.0000E+00	1.55444E-01	1.000E+00
20	hmx3	-4.10293E+03	-4.10000E+03	5.0000E+00	-5.86297E-01	1.000E+00

Index	Type	Start Value	Final Value	Difference
52	Opt.	.00000E+00	-3.74596E+04	-3.74596E+04

Mean weighted square of errors = 1.55369E-02

Graphs are produced during the optimisation run to show how individual parameters are changing and the effect this is having on the mean square of errors (Figure 2.2). Changes in parameters are expressed in units of the parameter's SCALE. This allows all parameters to be represented on a single graph. When the

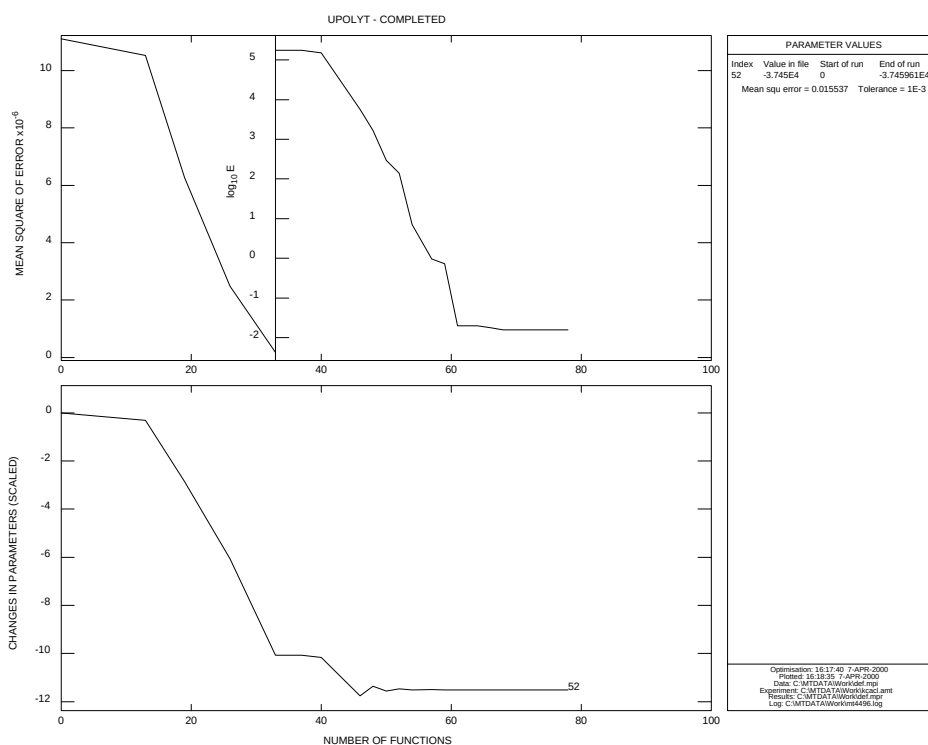
mean square of errors is reduced below about a tenth of its original value a log scale is used to show further reductions.

A table comparing calculated and target values for all properties considered during a optimisation run is output to the screen at the end of that run. The starting and final values of OPTIMISED parameters are also shown. Entry of the command,

list result = experiment !

will generate an equivalent table at other times.

Figure 2.2 GRAPHS PRODUCED DURING AN OPTIMISATION RUN



2.6 PLOTTING AND SAVING RESULTS

list parameter = value !

The current (optimised) value of parameter 52, that is -37459.6, is very similar to the original data input file (.mpi) file value of -37450.0. Note that the latter value is retained, by default, in the in-memory store.

save parameter = 52 ! list parameter = value !

This command saves the current (optimised) value of parameter 52 to the in-memory store. The stored value LISTed out for parameter 52 is now the same as the current (optimised) value, not the same as the data input (.mpi) file value.

undo

list parameter = value !

The UNDO command negates the effects of the last optimisation run. This means that all parameters are returned to the values they had at the start of that run. The current value of parameter 52, for example, becomes 0.0.

restore parameter = all ! list parameter = value !

The values of all previously SAVED parameters are retrieved from the in-memory store. Existing values are overwritten. The current LISTed value for parameter 52 is, once again, -37459.6, the same as the stored value.

Table 2.6 Sample output for PLOTTING AND SAVING RESULTS

ASSESSMENT OPTION ? *list parameter=value !*

Index	Type	Scale	Data File Parameter	Current Parameter	Stored Parameter
52	Opt.	3.255E+03	-3.74500E+04	-3.74596E+04	-3.74500E+04

ASSESSMENT OPTION ? *save parameter=52 ! list parameter=value !*

Index	Type	Scale	Data File Parameter	Current Parameter	Stored Parameter
52	Opt.	3.255E+03	-3.74500E+04	-3.74596E+04	-3.74596E+04

ASSESSMENT OPTION ? *undo*

ASSESSMENT OPTION ? *list parameter=value !*

Index	Type	Scale	Data File Parameter	Current Parameter	Stored Parameter
52	Opt.	3.255E+03	-3.74500E+04	.00000E+00	-3.74596E+04

ASSESSMENT OPTION ? *restore parameter=all ! list parameter=value !*

Index	Type	Scale	Data File Parameter	Current Parameter	Stored Parameter
52	Opt.	3.255E+03	-3.74500E+04	-3.74596E+04	-3.74596E+04

ASSESSMENT OPTION ? *plot errors=experiment !*

ASSESSMENT OPTION ? *plot values=experiment(hmx) !*

ASSESSMENT OPTION ? *swap binary*

BINARY OPTION ? *step t=500 1100 10 ! set n(1)=0.1 n(2)=0.9 t=500 ! compute !*

BINARY OPTION ? *replot experiment 'kcacsim.amt' format assessment(5,3) label 'com' !*

BINARY OPTION ? *swap assessment*

ASSESSMENT OPTION ? *save set_up="kcac1_setup.mac" !*

ASSESSMENT OPTION ? *save data_loading_file="kcac1_new.loa" !*

ASSESSMENT OPTION ? *return utility*

UTILITY OPTION ? *create_database database='kcac1' !*

UTILITY OPTION ? *load_data file='kcac1_new.loa' !*

CaCl₂, Cl₂<LIQUID>

UTILITY OPTION ? *return assessment*

ASSESSMENT OPTION ? *define system='KCl,CaCl2' source=kcac1,demo_1 output 'kcac1' !*

SEARCHING FOR SYSTEM KCl,CaCl₂

SEARCHING DATABASE(S) :

```

kcac1
demo_1          - Test database - use only with macro 'testmt.mac'

*****
GOOD DATAFILE CREATED
*****

Date and time of run 7-APR-2000 17:34:32
* DATAFILE = C:\MTdata\Work\kcac1.mpi - CREATED 17:34:32 7-APR-2000
* SYSTEM = KCl,CaCl2,
* NUMBER OF PHASES = 4
* NUMBER OF SPECIES = 5
*

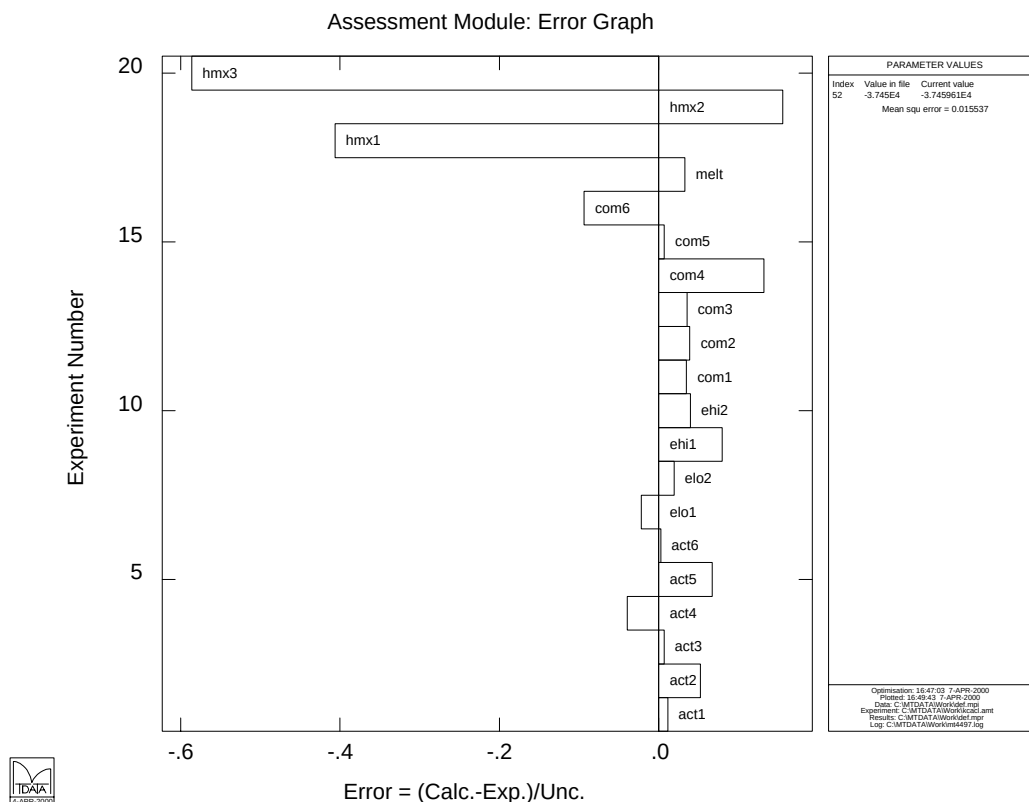
ASSESSMENT OPTION ? 'kcac1_setup.mac'
Date and time of run 7-APR-2000 17:40:30
* DATAFILE = C:\MTDATA\Work\def.mpi - CREATED 15:41:03 7-APR-2000
* SYSTEM = KCl,CaCl2,
* NUMBER OF PHASES = 4
* NUMBER OF SPECIES = 5
*

```

plot errors = experiment !

Generates a bar chart in which each bar represents the difference between a calculated PROPERTY and its target VALUE, expressed in units of the associated experimental UNCERTAINTY. This type of chart provides a useful graphical overview of the results of an optimisation run, enabling properties or groups of properties that are difficult to reproduce to be easily identified.

Figure 2.3 A GRAPH PRODUCED USING *plot errors experiment !*



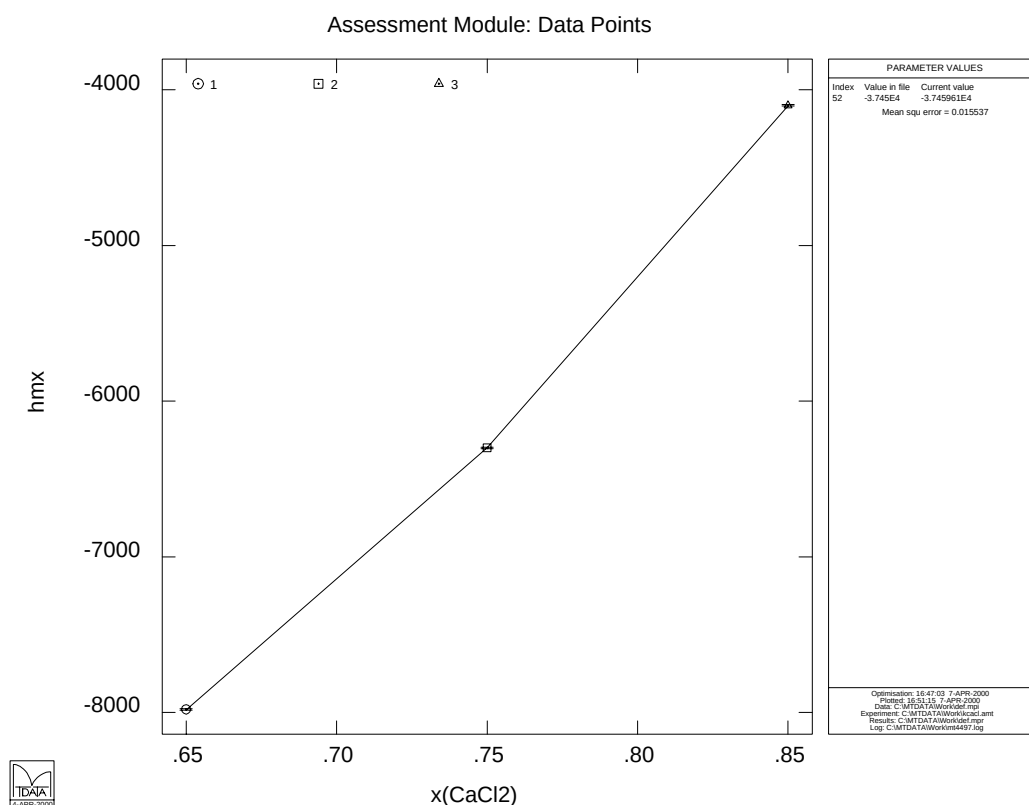
In the current example (Figure 2.3) all target values are reproduced to within their associated uncertainty. The longest bars are those with labels beginning “hmx”,

corresponding to calculated enthalpies of mixing. These properties have had smaller uncertainties, relatively speaking, assigned to them, than others in the experimental file “kcacl.amt”.

plot values = experiment(hmx) !

Generates a graph which takes the form of a line joining the calculated values of properties associated with the same ABSCISSA variable, plotted against that variable. Properties are chosen according to their labels. In the example above, all properties having labels starting with the letters “hmx” are selected (Figure 2.4). Symbols representing experimental data are superimposed.

Figure 2.4 A GRAPH PRODUCED USING *plot values experiment* !



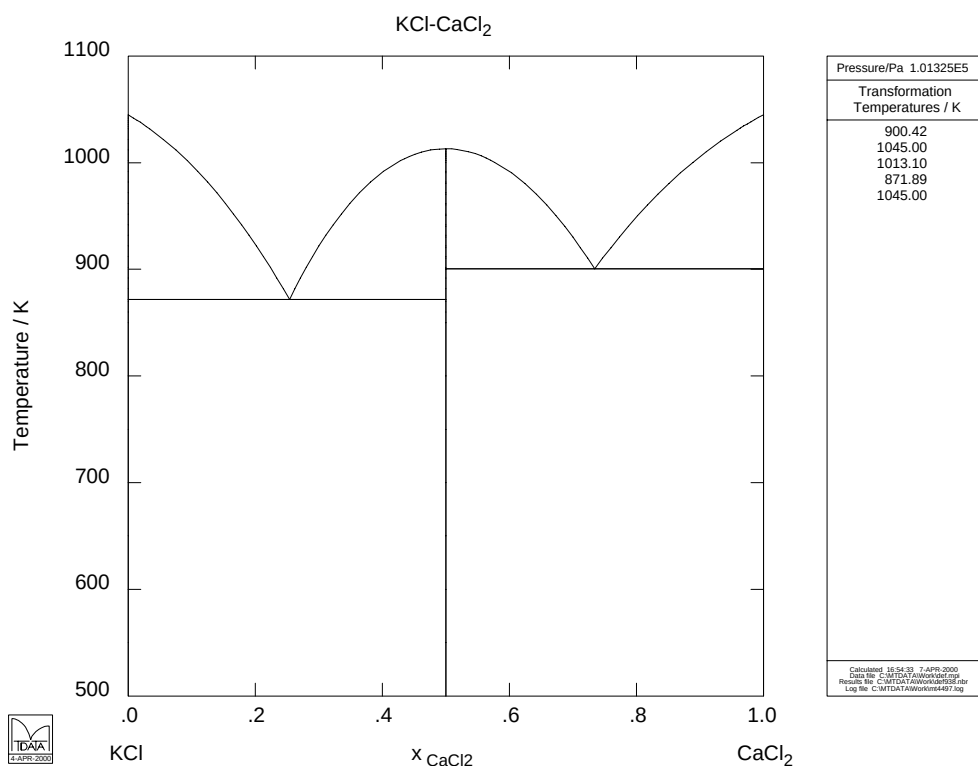
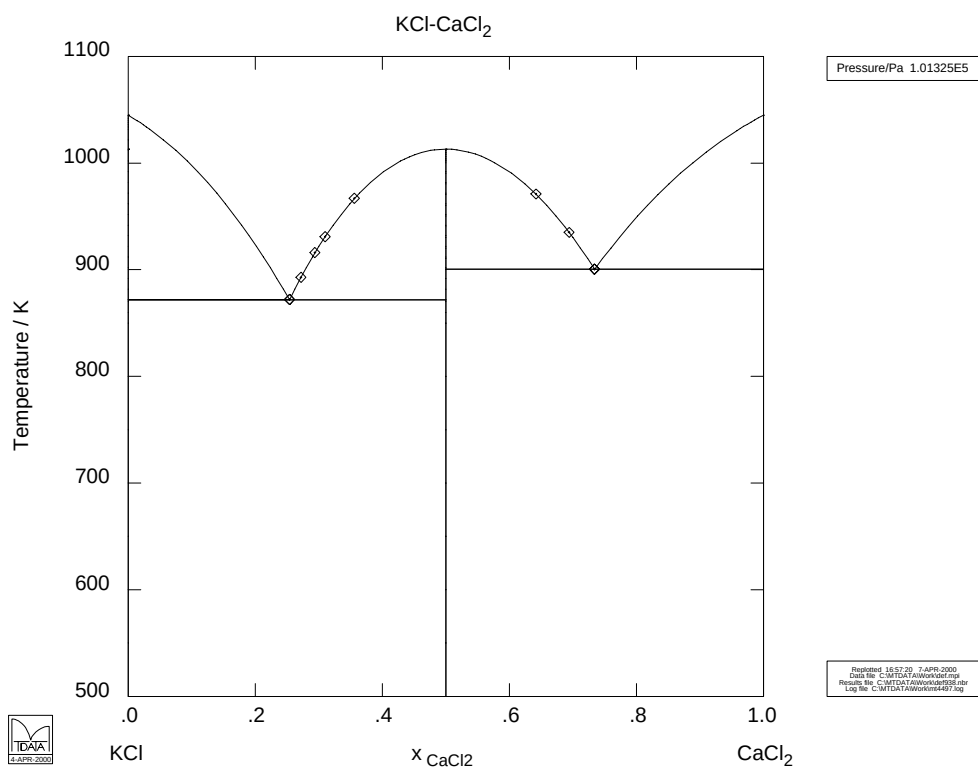
swap binary

step t = 500 1100 10 ! set n(1) = 0.1 n(2) = 0.9 t = 500 ! compute !

SWAP is used to leave the ASSESSMENT module and enter another MTDATA module, BINARY in this case, retaining the current thermodynamic parameters, and any classifications and settings, in memory. This allows parameters to be tested in other modules without the need for a new data input (.mpi) file to be created. Note that the DEFINE command is not used in the new module. The above commands calculate a KCl-CaCl₂ phase diagram in the BINARY module (Figure 2.5).

replot experiment 'kcaclsim.amt' format assessment(5,3,4) label 'com' !

REPLOTS the calculated BINARY diagram with symbols added to represent data points taken from the table in the experimental file “kcaclsim.amt” (Figure 2.6).

Figure 2.5 CALCULATED KCl-CaCl₂ BINARY DIAGRAM**Figure 2.6 REPLOTTED DIAGRAM WITH EXPERIMENTAL DATA ADDED**

Abscissa and ordinate values for each data point are taken from the fifth and third columns respectively of rows containing the characters “com” (see Table 2.3.3). The symbol plotted is that corresponding to the fourth letter of the alphabet (D). For a full list of symbols available and their corresponding letters see the UTILITY module handbook. Error bars representing specified UNCERTAINTY values are also shown on the replotted diagram.

swap assessment

Swaps back to the ASSESSMENT module, retaining the current thermodynamic parameters in memory. Any settings and classifications made in the module being swapped from, BINARY in this case, are also retained.

save set_up = 'kcac1_setup.mac' !

Writes a macro which can be used to re-establish the exact state of the current ASSESSMENT module session at a later date. The contents of the file “kcac1_setup.mac” produced by the above command are shown in Table 2.6.1.

Table 2.6.1 The ASSESSMENT module set up file “kcac1_setup.mac”

```
; MTDATA ASSESSMENT MODULE SET UP FILE CREATED 17:23:15 07-APR-2000
; LOG FILE NAME IS C:\MTDATA\Work\mt4370.log
DEFINE DATA 'C:\MTDATA\Work\def.mpi' EXPERIMENT 'C:\MTDATA\Work\kcac1.amt' !
MODIFY INDEX 52 TYPE OPT !
MODIFY INDEX 52 VALUE -3.745960D+04 SCALE 3.255078D+03 !
SET USER_VAR(penalty_function) 1E12 !
```

save data_loading_file = 'kcac1_new.loa' !

Writes an MTDATA data loading file suitable for updating all unary or interaction data sets which include OPTIMISED, STEPPED, FIXED or LINKed parameters. The contents of the file “kcac1_new.loa” produced by the above command is shown in Table 2.6.2.

Table 2.6.2 The MTDATA data loading file “kcac1_new.loa”

```
; MTDATA ASSESSMENT MODULE LOAD FILE CREATED 17:26:06 7-APR-2000
; LOG FILE NAME IS C:\MTDATA\Work\mt4370.log

CaCl2, ClK<LIQUID> E/J
***** GEX(REDLICH-KISTER) DATASET FORMAT *****
  298.150      3
              2
-3.74596047835D+04 9.84500000000D+00
              2
  8.19000000000D+03 -5.74000000000D+00
              2
-9.25000000000D+02 1.92900000000D+00
  3000.000
    0      0
```

Typically the contents of a file such as “kcac1_new.loa” would be loaded into a new database created within the UTILITY module. An updated data input (.mpi) file

would then be produced by means of a database search encompassing the new database.

return utility

create_database database = 'kcac1' !

load_data file = 'kcac1_new.loa' !

In the UTILITY module, a database named “kcac1” is created and data are loaded into this database from the file “kcac1_new.loa”.

return assessment

define system = 'KCl,CaCl2' source = kacac1,demo_1 output 'kcac1' !

Back in the ASSESSMENT module an updated data input file, given the name “kcac1.mpi”, is created. Note that the new database “kcac1” is placed first on the database search list so that the data it contains take preference.

'kcac1_setup.mac

The set up macro SAVED earlier can be used to re-establish the ASSESSMENT session as it was before RETURNing to the UTILITY module.

Note that data are retrieved from the original data input file (def.mpi) and not from the new data input file (kcac1.mpi). The new file may have a different structure to the original file and so parameters may be numbered differently.

Care should be taken to ensure that the contents of a data input (.mpi) file accessed in a set up macro remain unchanged in between saving the macro and using it to re-establish an ASSESSMENT session, otherwise parameter numbers in the set up macro may become invalid.

3 EXERCISES

Some exercises involving the use of the ASSESSMENT module are suggested below. The target data provided for use during these exercises do not necessarily represent genuine experimental measurements and so should not be used elsewhere.

3.1 A SIMPLE EUTECTIC

One of the simplest types of systems to model is one featuring a eutectic between two crystalline components which exhibit no mutual solubility. Be-Si is an example of such a system.

Retrieve data for the Be-Si system from the UNARY database and plot a binary diagram. Interactions in the liquid, BCC_A2, FCC_A1 and HCP_A3 phases should be treated as ideal, at least temporarily, by classifying all phases NORMAL. A comparison between the diagram obtained in this way and that accepted in the scientific literature reveals the following problems:

- The solubility of Si in pure Be, HCP_A3 or BCC_A2 at high temperatures, should be negligible but is calculated to be about 12 mole % at its maximum.
- The eutectic temperature should be 1363 K but is calculated to be 1275 K.
- The composition of the liquid at the eutectic should contain 36 mole % Si but is calculated to contain 31.2 mole % Si.

The ASSESSMENT module can be used to generate interaction parameters which correct these problems.

Create a database loading file containing initial data for interactions between Be and Si in the BCC_A2, FCC_A1, HCP_A3 and liquid phases. The easiest way to do this is to edit the "misbin.dbl" file created during the UNARY database search above. Make sure all parameters are given non zero values and add extra composition dependent terms for the liquid in case they are found to be necessary at a later stage. Use this database loading file to populate a newly created database, then search this database along with UNARY to generate an initial data input (.mpi) file for the Be-Si system.

Next create an experimental file specifying that the solubility of Si in pure Be (HCP_A3) is, at most, about 0.1 mole% and that the eutectic between Si and pure Be (HCP_A3) occurs at 1363 K with a Si content of 36 mole % in the liquid phase. The file which you produce might not be identical to the example given in Table 3.2, but might work equally well. There is not a unique way of setting up a suitable experimental file.

Now set up an optimisation run in the ASSESSMENT module. It is often best to optimise parameters incrementally rather than all at once. Adopt this approach by optimising the single HCP_A3 interaction parameter first. Then swap to the BINARY module and plot a new binary diagram. The solubility of Si in the HCP_A3 phase should have been dramatically reduced, probably allowing the BCC_A2 phase to take over from HCP_A3 in equilibrium with the liquid. The solubility of Si in the BCC_A2 and the FCC_A1 phases must now be reduced as well. This can be done by making suitable entries in the experimental file and optimising again or, more

elegantly, by linking interaction parameters in the BCC_A2 and FCC_A2 phases to that in HCP_A3 phase.

Last of all, try optimising interaction parameters appropriate to the liquid phase. It may be that more than one parameter is necessary, although care should be taken not to use more parameters than are justified by the available experimental data. The binary diagram shown in Figure 3.1 was produced using two temperature independent parameters, A and B, to model the excess Gibbs energy of the Be-Si liquid phase as follows:

$$G^{\text{ex}} = A x_{\text{Be}} x_{\text{Si}} + B x_{\text{Be}} x_{\text{Si}} (x_{\text{Be}} - x_{\text{Si}})$$

Experimental files are generally much larger than the simple one used in this example, ideally containing both phase boundary information and thermodynamic measurements such as activities or enthalpies of mixing.

3.2 MISCIBILITY GAPS

Miscibility gaps, where two or more examples of the same phase exist in equilibrium, occur in many systems. It is important, therefore, to be able to enter information about the extent of such gaps in experimental files. The ability to flag the absence of a miscibility gap is also useful, since spurious gaps can be generated during an optimisation run if, for example, interaction parameters become too large and positive.

The CaO-SiO₂ system is one which exhibits a miscibility gap in the liquid phase. Liquids containing between 99 and about 90 mole % SiO₂ coexist with liquids containing between 72 mole % and about 90 mole % SiO₂ at temperatures between 1962 K and 2173 K. If you were creating an experimental file for this system, what commands would you use;

- to specify that liquids containing 70 mole % SiO₂ and 99 mole % SiO₂ exist in equilibrium at 1962 K ?
- to specify that a single liquid phase containing 90 mole % SiO₂ exists at 2200 K ?

Examples of the types of commands that might be used are provided for reference in Table 3.3. Other solutions are, of course, possible.

To access the different liquid compositions at the two ends of a miscibility gap, the qualifiers “,1” and “,2” can be appended to the phase name “LIQUID” in LIST PROPERTY commands. For example:

```
list property x(SiO2<LIQUID,1>) !
list property x(SiO2<LIQUID,2>) !
```

3.3 A SYSTEM WITH A CONGRUENTLY MELTING COMPOUND

The phase diagram for the In-Sb system features a congruently melting compound with the stoichiometry InSb and eutectics between this compound and both pure components. The aim of this exercise is to derive thermodynamic parameters to model the In-Sb system.

Create a data input (.mpi) file for the In-Sb system by searching the UNARY database and list the phases. Note the names of the solution phases classified as absent due to a lack of interaction data. Enter initial In-Sb interaction data for these phases into an MTDATA database loading file, using “misbin.dbl” as a template. Also enter preliminary data for the InSb compound, taken from the SGTE substance database. Only data appropriate below the melting point of this compound (about 800 K) should be used since the remaining parameters apply to liquid InSb, modelled as a separate solution phase in the current exercise. In reality, checks would be made to ensure that there were no problems with the extrapolation of these data to high temperatures. The heat capacity, for example, might have to be modified using FITANDPLOT.

Next use the loading file just created to populate a new database. Search this database along with UNARY to generate an initial data input (.mpi) file for the In-Sb system with no missing data. Then enter the information presented in Table 3.1 into an experimental file. As a final task before optimisation can begin, test this file for consistency using the command *optimise maximum_functions 0* !

Now use the ASSESSMENT module to generate parameters representing In-Sb interactions in the liquid phase consistent with the above data. Could a better fit be obtained if the first parameter in the expression for the Gibbs energy of InSb was allowed to change as well?

It is not uncommon for the first two parameters in the expression for the Gibbs energy of a unary to be optimised, the constant and T terms, provided that there is information about the unary's enthalpy and entropy in the experimental file to guide any changes. The remaining parameters, which describe the variation of Cp with temperature, are usually left unchanged during optimisation as suitable values will have been determined prior to entering the ASSESSMENT module, for example by using FITANDPLOT.

Results calculated using the optimised parameters generated in this exercise can be compared with those obtained using data for the In-Sb system from the NPL MTSOL solution database.

A typical error plot obtained by modelling the excess Gibbs energy of the In-Sb liquid phase using four coefficients, A, B, C and D as follows:

$$G^{XS} = (A + B T) x_{In} x_{Sb} + (C + D T) x_{In} x_{Sb} (x_{In} - x_{Sb})$$

can be seen in Section 4.9. It is interesting to note that the data from the NPL MTSOL solution database include coefficients of $T \ln(T)$ as well. A graphical comparison between calculated Sb activities at 900 K and values from Table 3.1 is also made in Section 4.9.

Binary diagrams calculated using the optimised parameters are shown in Figures 3.2 and 3.3, the latter being a replotted diagram with data points from Table 3.1 superimposed.

A sample experimental file for the In-Sb system, containing the data recommended for use in the current exercise, is shown in Table 3.4. Macros have been used at the

top of the file to avoid having to repeat blocks of MTDATA commands. At the bottom of the file, for comparison, commands have been entered in full.

Table 3.1 Data to be used in creating an experimental file for the In-Sb system		
PHASE BOUNDARY DATA (TA6 = TETRAGONAL_A6, RA7 = RHOMBOHEDRAL_A7)		
Equilibrium	Temperature /K	x(Sb) in LIQUID
TA6 + LIQUID + InSb	427	0.005
InSb + LIQUID + RA7	767	0.688
LIQUID + InSb	463	0.010
LIQUID + InSb	515	0.026
LIQUID + InSb	598	0.079
LIQUID + InSb	643	0.123
LIQUID + InSb	704	0.202
LIQUID + InSb	744	0.272
LIQUID + InSb	780	0.363
LIQUID + InSb	789	0.606
LIQUID + InSb	774	0.667
LIQUID + InSb	800	0.500
LIQUID + RA7	790	0.724
LIQUID + RA7	841	0.823
LIQUID + RA7	867	0.890
Sb ACTIVITIES IN THE LIQUID PHASE (LIQUID REFERENCE)		
Temperature /K	x(Sb)	a(Sb) in LIQUID
800	0.1	0.014
800	0.2	0.042
800	0.3	0.095
900	0.1	0.017
900	0.2	0.049
900	0.3	0.108
900	0.4	0.202
900	0.6	0.492
900	0.8	0.794
ENTHALPIES (TA6 = TETRAGONAL_A6, RA7 = RHOMBOHEDRAL_A7)		
Reaction	Temperature /K	ΔH / kJ mol ⁻¹ of InSb
In(TA6) + Sb(RA7) = InSb	298.15	-32.47
InSb = LIQUID	800	49.46

3.4 TABLES AND FIGURES

Section 3.4 should not be examined until the exercises above have been attempted.

Table 3.2 EXAMPLE EXPERIMENTAL FILE FOR THE Be-Si SYSTEM

```

classify absent phase(*) miscibility(LIQUID) 1 miscibility(HCP_A3) 1 !
set n(1) 0.9 n(2) 0.1 temperature 1363 !
compute !
list property x(Si<HCP_A3>) value 0.001 uncertainty 0.0001 label 'HCP' !
list property x(Si<LIQUID>) value 0.36 uncertainty 0.01 label 'Be_LIQ' !
;
classify absent phase(*) miscibility(LIQUID) 1 no p(DIAMOND) !
set n(1) 0.1 n(2) 0.9 t 1363 ! compute !
list property x(Si<LIQUID>) value 0.36 uncertainty 0.01 label 'Si_LIQ' !
;
classify normal phase(*)
end_of_experiments

```

Figure 3.1 CALCULATED BINARY DIAGRAM FOR THE Be-Si SYSTEM

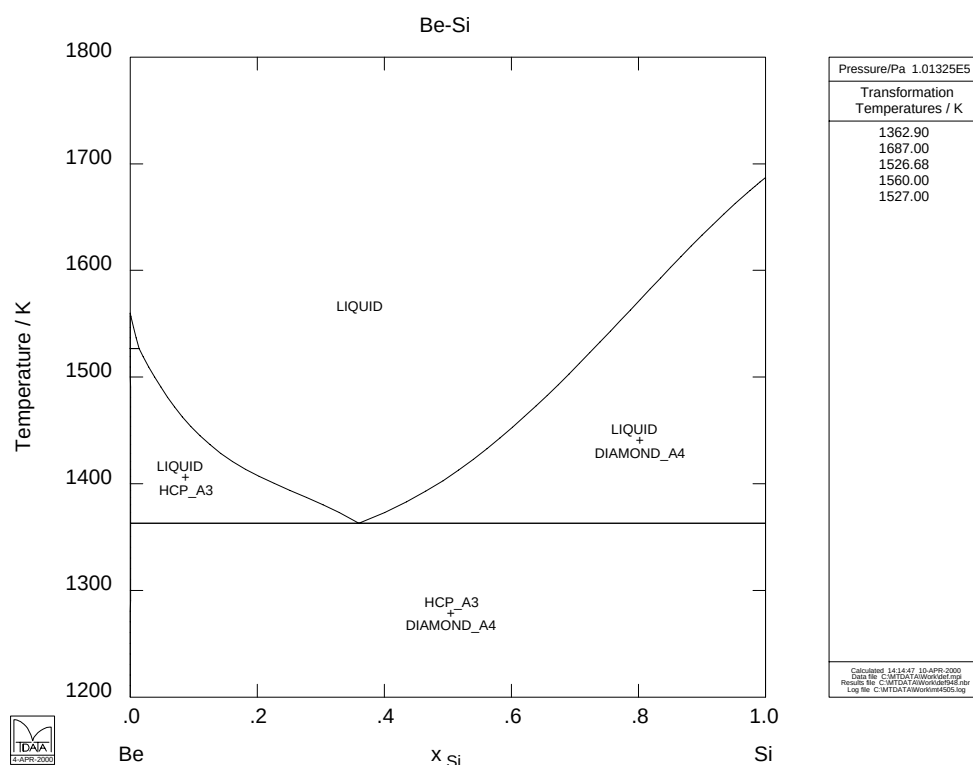


Table 3.3 DEALING WITH MISCIBILITY GAPS IN EXPERIMENTAL FILES

```

; ENTERING INFORMATION ABOUT A MISCIBILITY GAP
classify absent phase(*) miscibility(liquid) 1 ! classify reference liquid !
set n(1) 0.3 n(2) 0.7 t 1962 ! compute !
; STORE THE PARTIAL GIBBS ENERGY OF SiO2 AT ONE END OF THE GAP
set user_variable(G1) gibbs_energy(SiO2) !
set n(1) 0.01 n(2) 0.99 t 1962 ! compute !
; STORE THE PARTIAL GIBBS ENERGY OF SiO2 AT THE OTHER END OF THE GAP
set user_variable(G2) gibbs_energy(SiO2) !

```

```

; THE PARTIAL GIBBS ENERGY OF SiO2 MUST BE THE SAME AT EACH END OF THE GAP
list property difference(u_var(G1):u_var(G2)) value 0.0 uncert 1 label 'GAP1' !
;
classify absent phase(*) miscibility(liquid) 1 ! classify reference liquid !
set n(1) 0.3 n(2) 0.7 t 1962 ! compute !
; STORE THE PARTIAL GIBBS ENERGY OF CaO AT ONE END OF THE GAP
set user_variable(G1) gibbs_energy(CaO) !
set n(1) 0.01 n(2) 0.99 t 1962 ! compute !
; STORE THE PARTIAL GIBBS ENERGY OF CaO AT THE OTHER END OF THE GAP
set user_variable(G2) gibbs_energy(CaO) !
; THE PARTIAL GIBBS ENERGY OF CaO MUST BE THE SAME AT EACH END OF THE GAP
list property difference(u_var(G1):u_var(G2)) value 0.0 uncert 1 label 'GAP2' !
;
; SPECIFYING THAT NO MISCIBILITY GAP EXISTS ERRONEOUSLY AT HIGHER TEMPERATURES
classify absent phase(*) miscibility(liquid) 1 !
set n(1) 0.1 n(2) 0.9 t 2200 ! compute !
; STORE THE GIBBS ENERGY FROM A CALCULATION WITH A MISCIBILITY GAP CLASSIFIED
set user_variable(G1) gibbs_energy !
classify absent phase(*) normal phase(liquid) !
set n(1) 0.1 n(2) 0.9 t 2200 ! compute !
; STORE THE GIBBS ENERGY FROM A CALCULATION WITH NO MISCIBILITY GAP CLASSIFIED
set user_variable(G2) gibbs_energy !
; IF A SPURIOUS MISCIBILITY GAP IS FOUND THEN G1 AND G2 WILL BE DIFFERENT
list property difference(u_var(G1):u_var(G2)) value 0.0 uncert 1 label 'NOGAP' !

```

Table 3.4 EXAMPLE EXPERIMENTAL FILE FOR THE In-Sb SYSTEM

```

; COMPOSITIONS OF LIQUIDS IN EQUILIBRIUM WITH InSb
; macro x(Sb) T/K value uncert weight label
'insbpb1.mac' 0.49 427 0.005 0.001 2 'PB:eutectic'
'insbpb1.mac' 0.51 767 0.688 0.01 2 'PB:eutectic'
'insbpb1.mac' 0.49 463 0.010 0.01 1 'PB:liquidus'
'insbpb1.mac' 0.49 515 0.026 0.01 1 'PB:liquidus'
'insbpb1.mac' 0.49 598 0.079 0.01 1 'PB:liquidus'
'insbpb1.mac' 0.49 643 0.123 0.01 1 'PB:liquidus'
'insbpb1.mac' 0.49 704 0.202 0.01 1 'PB:liquidus'
'insbpb1.mac' 0.49 744 0.272 0.01 1 'PB:liquidus'
'insbpb1.mac' 0.49 780 0.363 0.01 1 'PB:liquidus'
'insbpb1.mac' 0.51 789 0.606 0.01 1 'PB:liquidus'
'insbpb1.mac' 0.51 774 0.667 0.01 1 'PB:liquidus'
;
; COMPOSITIONS OF LIQUIDS IN EQUILIBRIUM WITH Sb(RHOMBOHEDRAL_A7)
; macro x(Sb) T/K value uncert weight label
'insbpb2.mac' 0.99 767 0.688 0.01 2 'PB:eutectic'
'insbpb2.mac' 0.99 790 0.724 0.01 1 'PB:liquidus'
'insbpb2.mac' 0.99 841 0.823 0.01 1 'PB:liquidus'
'insbpb2.mac' 0.99 867 0.890 0.01 1 'PB:liquidus'
;
; Sb ACTIVITIES IN THE LIQUID PHASE
; macro x(Sb) T/K value uncert weight label
'insbac.mac' 0.1 800 0.014 0.02 1 'Activity:800'
'insbac.mac' 0.2 800 0.042 0.02 1 'Activity:800'
'insbac.mac' 0.3 800 0.095 0.02 1 'Activity:800'
'insbac.mac' 0.1 900 0.017 0.02 1 'Activity:900'
'insbac.mac' 0.2 900 0.049 0.02 1 'Activity:900'
'insbac.mac' 0.3 900 0.108 0.02 1 'Activity:900'
'insbac.mac' 0.4 900 0.202 0.02 1 'Activity:900'
'insbac.mac' 0.6 900 0.492 0.02 1 'Activity:900'
'insbac.mac' 0.8 900 0.794 0.02 1 'Activity:900'
;
; EUTECTIC TEMPERATURES
classify absent phase(*) normal phase(liquid,tetragonal_a6,InSb) !
set n(1) 0.51 n(2) 0.49 ! range t 1500 300 0.1 ! compute !
list property t(2) value 427 uncertainty 1 weight 2 label 'Teutectic' !

```

```

classify absent phase(*) normal phase(liquid,rhombohedral_a7,InSb) !
set n(1) 0.01 n(2) 0.99 ! range t 1500 300 0.1 ! compute !
list property t(2) value 767 uncertainty 1 weight 2 label 'Teutectic' !
;
; ENTHALPIES
classify absent phase(*) normal phase(InSb) reference default !
set t 298.15 n(1) 1 n(2) 1 ! compute !
list property h value -32470 uncertainty 2500 weight 1 label 'DeltaHf' !
;
set t 800 ! compute !
set user_variable(G1) g ! set user_variable(H1) h !
classify absent phase(*) normal phase(liquid) ! compute !
list property diff(h:u_v(H1)) value 49460 unc 1000 weight 1 label 'Hmelt' !
;
; MELTING TEMPERATURE
list property diff(g:U_v(G1)) value 0.0 unc 5 weight 1 label 'Tmelt' !
;
classify normal phase(*) !
end_of_experiments
;

```

insbpb1.mac

```

; MOLE FRACTION OF Sb IN LIQUID IN EQUILIBRIUM WITH InSb
classify absent p(*) normal phase(liquid,InSb) !
set n(1) undef n 1 n(2) *****
t *****
!
compute !
list property x(Sb<LIQUID>) value *****
uncertainty *****
weight *****
label *****
abscissa T !

```

insbpb2.mac

```

; MOLE FRACTION OF Sb IN LIQUID IN EQUILIBRIUM WITH Sb(RHOMBOHEDRAL_A7)
classify absent p(*) normal phase(liquid,RHOMBOHEDRAL_A7) !
set n(1) undef n 1 n(2) *****
t *****
! compute !

list property x(Sb<LIQUID>) value *****
uncertainty *****
weight *****
label *****
abscissa T !

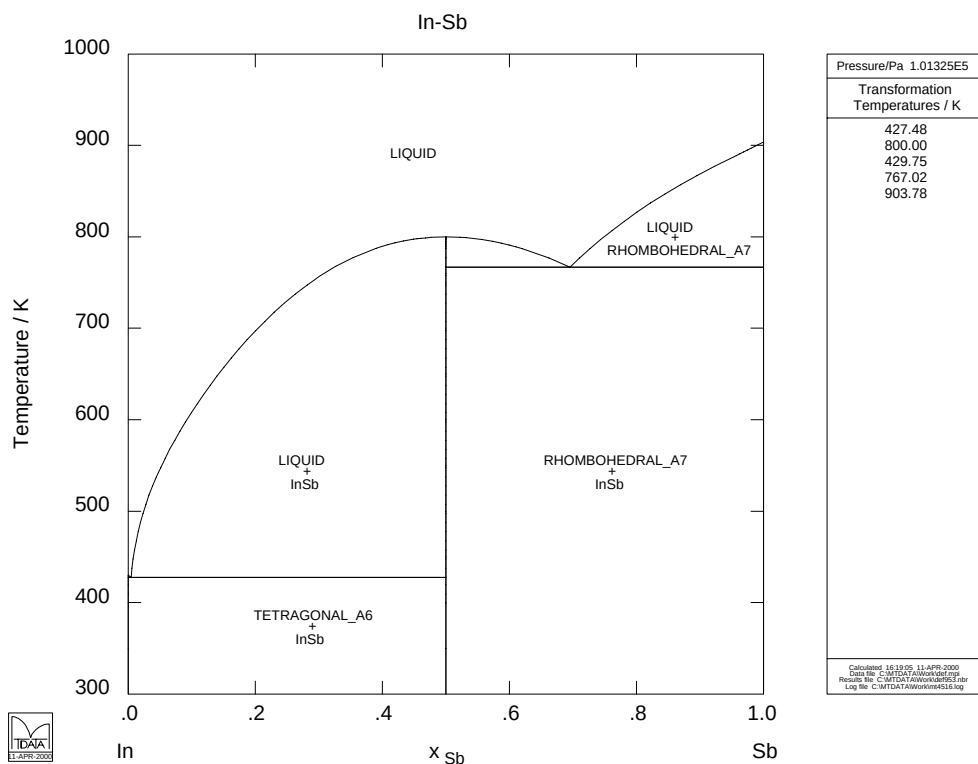
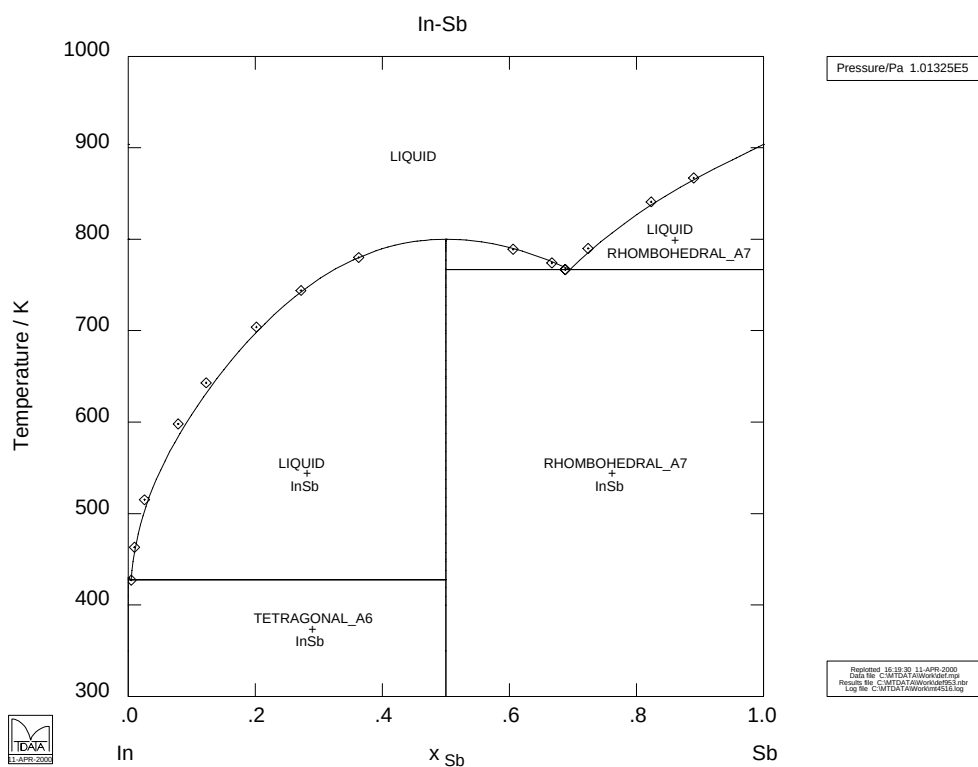
```

insbac.mac

```

; ACTIVITY OF Sb IN LIQUID PHASE (LIQUID REFERENCE)
classify absent phase(*) normal phase(liquid) !
classify reference liquid !
set n(1) undef n 1 n(2) *****
t *****
! compute !
list property activity(Sb) value *****
uncertainty *****
weight *****
label *****
abscissa x(Sb) !

```

Figure 3.2 CALCULATED BINARY DIAGRAM FOR THE In-Sb SYSTEM**Figure 3.3 REPLOTTED DIAGRAM WITH DATA FROM TABLE 3.1 ADDED**

4 OPTION DEFINITIONS

4.1 DEFINE

Function: The DEFINE command is used to obtain thermodynamic data for a system to be studied, either from listed databases, in which case the system must be specified in terms of its components, or from a previously created data input (.mpi) file. Data found during a database search are written automatically to a named or default data output file using the .mpi format. The DEFINE command is also used to specify the file (macro) which contains calculation instructions and experimental data for planned optimisation runs.

Parameters: DATA_INPUT_FILE EXPERIMENTAL_FILE
 OUTPUT DATA/RESULTS SOURCE
 SYSTEM

Defaults: DATA_INPUT_FILE = "def.mpi"
 OUTPUT DATA/RESULTS = "def" (for example "def.mpr")
 SOURCE = first database from the database menu

No default system or experimental_file.

Examples: *define data = "proj403" !*
define system "C, H, O" source = sgte mydata !
define data default !
def !
define system "Al2O3, CaO, SiO2, O/-2" !
define system "Al, H, Cl" output = "mydata" !
define system "Al, Si" source = "mt-aldata"
define data = "cumg" experiment = "cumg.amt" !

Parameter values:

DATA_INPUT_FILE takes the values <"file name"> / DEFAULT

If the DEFINE statement contains a reference to a DATA_INPUT_FILE the program reads thermodynamic data for the system to be studied from the specified data input (.mpi) file. This file could have been created during a previous MTDATA session from within a different MTDATA module such as ACCESS or MULTIPHASE. If DATA = DEFAULT is entered (or DATA or no parameter at all) thermodynamic data are read from the last data input file generated with the default name (def.mpi).

OUTPUT DATA/RESULTS takes the value <"file root">

The OUTPUT parameter is used to specify a root name for any files created during the current ASSESSMENT session, such as .mpi files resulting from database searches and .mpr files to which diagnostic information and some calculated results are written. The OUTPUT parameter should form part of a DEFINE statement containing either

a SYSTEM or a DATA_INPUT_FILE specification. Use on its own will cause thermodynamic data to be read from the default data input (.mpi) file. If no OUTPUT file root is DEFINED, results (.mpr) files are created with the same root name as the data input (.mpi) file current at the time of their creation and data input (.mpi) files are created (on entry of DEFINE SYSTEM command) with the default name "def.mpi".

SYSTEM takes the value <"component list">

The SYSTEM parameter is used to DEFINE components of interest prior to searching one or more databases. Components may be elements, compounds or charged species. The best choice will depend on the models used to represent phases within the system under study. Please refer to the handbook for the ACCESS module for more details.

Use of the DEFINE SYSTEM command causes the program to search the current list of databases (specified using the SOURCE parameter) for substances formed by taking linear combinations of the DEFINED components and for interactions between any such species associated with the same phase. Thermodynamic data for these substances and interactions are automatically written to a data input (.mpi) file, with a name based upon the DEFINED OUTPUT_DATA/RESULTS file root, then read into memory.

SOURCE takes the values <database list> / NONE_BUT <database list> / ALL / <"database path">

The SOURCE parameter is used to provide a list of database(s) for consideration in database searches. The databases should be specified in the order in which they are to be searched. The effects of multiple entries of the SOURCE command are cumulative except when NONE_BUT is specified. Use of the SOURCE parameter other than as part of a DEFINE statement containing either a SYSTEM or a DATA_INPUT_FILE specification will cause data to be read in from the default data input (.mpi) file. If no database SOURCE is provided then the last database list entered previously is used. If no list has been entered previously then the first database from the database menu is searched.

<database list>

a list containing the names of one or more databases available from the database menu.

<"database path">

the path to a database that does not appear on the database menu.

ALL

causes the complete list of databases available from the database menu to be searched in the menu order.

NONE_BUT <database list>

provides a mechanism for clearing a database search list already established and specifying a new one.

EXPERIMENT takes the value <"filename">

The EXPERIMENT parameter is used to specify the name of the file containing calculation instructions and experimental data for planned optimisation runs. During such runs thermodynamic parameters from the DEFINED data input (.mpi) file, flagged using the MODIFY INDEX and MODIFY TYPE commands, are altered automatically to reproduce experimental data as closely as possible. The file specified, an MTDATA macro, may call other macros. The contents of the DEFINED file can be checked for errors by entering the command OPTIMISE MAXIMUM_FUNCTIONS 0 !

General comments:

When the DEFINE SYSTEM command is executed, databases on the database list established using the SOURCE parameter are searched for all species formed by taking linear combinations of the specified components and for interactions between any such species associated with the same phase. Thermodynamic data for the species and interactions found are automatically written to a data input (.mpi) file, then read into memory.

If the number of species is larger than can be supported by the ASSESSMENT module it is possible to use the ACCESS module to perform a preliminary screening of species and produce a data input (.mpi) file which ASSESSMENT is able to deal with. The maximum number of species in the ASSESSMENT module is the same as that in MULTIPHASE and can be inspected for a particular MTDATA installation using the LIMITS command of the UTILITY module. A typical value is 500.

If data are found for the same species or interaction from different databases, those from the database which appears earliest on the database search list will be saved in the data input (.mpi) file. The ACCESS module should be used to select data items on a case by case basis rather than on a first found basis. Data saved in a .mpi file using ACCESS can be read into the ASSESSMENT module using the command DEFINE DATA_INPUT_FILE.

If no OUTPUT_DATA/RESULTS file name is specified, thermodynamic data are saved to a data input (.mpi) file with the default name "def.mpi". These data can be read in subsequently using the commands:

```

define data = "def" !
define data = default !
define data !
or
define !

```

Data input files created with the default name are overwritten if subsequent database searches are carried out with no OUTPUT_DATA/RESULTS file name specified.

4.2 LIST

Function: The LIST command is used to display the elements, components, substances, unaries and phases in the system under study, their classifications (see CLASSIFY) and any initial or equilibrium settings (or RANGED values) established by the user. It also provides the basic tools for setting up an optimisation run in that it allows target experimental values and uncertainties to be associated with calculated properties or combinations of properties. Appropriate labels, weights for use during optimisation runs and abscissa settings for use during the plotting of results can be assigned to individual calculations or “experiments”. In addition, the LIST command allows the differences between calculated and experimental results or the identities and values of FIXED, STEPPED, LINKed and OPTIMISED parameters (see MODIFY) to be listed to the screen.

Parameters:	ABSCISSA	LABEL_OF_EXPERIMENT
	PARAMETER	PROPERTY
	RESULT	SYSTEM
	VALUE	UNCERTAINTY
	WEIGHT	

Defaults: LABEL_OF_EXPERIMENT = current PROPERTY name
 PARAMETER = all fixed, stepped, optimised parameters
 RESULT = all experiments
 WEIGHT = 1.0

Examples: *list system = elements substances !*
list system = components settings phases !
list property h !
list property = gibbs_energy(Cr) !
list property = pressure(O2<g>) !
list property = diff(h:user_variable(h1)) !
list prop h val -123000 unc 5000 weight 1 lab 'jones1' abs x(SiO2) !
list result = experiment !
list result experiment(2,5,7,9,12) !
list parameter name(12) !
list parameter value !

Parameter values:

ABSCISSA takes the values TEMPERATURE / USER_VARIABLE(*) / X(*) / W(*)

The ABSCISSA parameter is used to associate appropriate abscissa variables with individual calculations (or experiments) established using the PROPERTY parameter. Calculated values of properties associated with the same abscissa variable, selected via their LABEL, can be plotted against that variable, with symbols representing experimental data overlaid, using the PLOT VALUE command.

TEMPERATURE

selects the current TEMPERATURE as the appropriate ABSCISSA for the currently defined PROPERTY (see LIST PROPERTY) in graphs generated using the PLOT VALUE command.

USER_VARIABLE(*)

selects the current value of the USER_VARIABLE named in brackets as the appropriate ABSCISSA for the currently defined PROPERTY (see LIST PROPERTY) in graphs generated using the PLOT VALUE command.

X(*), W(*)

selects the current mole or mass fraction, respectively, of the component named in brackets as the appropriate ABSCISSA for the currently defined PROPERTY (see LIST PROPERTY) in graphs generated using the PLOT VALUE command.

LABEL_OF_EXPERIMENT takes the value <"main label [: additional label]">
where [] indicates an optional entry

The LABEL_OF_EXPERIMENT parameter is used to associate labels, entered in quotes, with individual calculations (or experiments) established using the PROPERTY parameter (see LIST PROPERTY). If no label is given, the appropriate PROPERTY name (Gibbs energy or temperature, for example) is adopted. Labels are used to identify individual calculations in tables generated using LIST RESULT and in graphs generated using PLOT ERROR and PLOT VALUE. In the case of PLOT VALUE, labels are also used to select results for plotting. Each label may consist of a main part, typically a PROPERTY name, and an additional part, typically an author's name or reference, separated from the main part by a colon. The main part is used to provide ordinate annotation in PLOT VALUE graphs.

PARAMETER takes the values NAME[(*)] / OFFSET[(*)] / VALUE[(*)]
where [] indicates an optional entry

LIST PARAMETER is used to inspect the status of parameters previously flagged as FIXED, OPTIMISED, STEPPED or LINKED via the MODIFY command. Options allow parameter NAMES, absolute VALUES and OFFSETs from the values in the current data input (.mpi) file to be listed to the screen. The VALUE and OFFSET lists also display parameter values and offsets from the in memory store (see the SAVE and RESTORE commands). The status of an individual parameter can be inspected by entering its number or INDEX (see MODIFY INDEX) in brackets after the choice of NAME, OFFSET or VALUE. If no parameter INDEX is given, all FIXED, OPTIMISED, STEPPED and LINKED parameters are featured.

NAME(*)

displays the NAMES of all FIXED, OPTIMISED, STEPPED and LINKED parameters. Alternatively, displays the NAME of one such

parameter if a valid INDEX is supplied in brackets. Typical output is shown below.

Index	T(low) /K	T(high) /K	Parameter Name
79	298.15	591.15	Cl2Zn<Cl2Zn> *xi
83	591.15	1200.00	Cl2Zn<Cl2Zn> *xi
53	298.15	3000.00	ClK,Cl2Zn<LIQUID> *xi*xj
55	298.15	3000.00	ClK,Cl2Zn<LIQUID> *xi*xj*(xi-xj)

VALUE(*)

displays the current VALUES of all FIXED, OPTIMISED, STEPPED or LINKED parameters. Alternatively, displays the current value of one such parameter if a valid INDEX is supplied in brackets. Typical output is shown below.

Index	Type	Scale	Data File Parameter	Current Parameter	Stored Parameter
79	Opt.	.000E+00	-4.35419E+05	-4.25419E+05	-4.37145E+05
83	79	.000E+00	-4.56887E+05	-4.46887E+05	-4.58613E+05
53	Fix	.000E+00	-8.27400E+04	-1.45267E+05	-1.45267E+05
55	Opt.	5.000E+03	-4.00800E+04	-3.85740E+04	.00000E+00

OFFSET(*)

displays the OFFSETs, or differences between current and data input (.mpi) file values, for all FIXED, OPTIMISED, STEPPED and LINKED parameters. Alternatively, displays the OFFSET appropriate to one such parameter if a valid parameter INDEX is supplied in brackets. Typical output is shown below.

Index	Type	Scale	Data File Parameter	Current Offset	Stored Offset
79	Opt.	.000E+00	-4.35419E+05	1.00000E+04	-1.72600E+03
83	79	.000E+00	-4.56887E+05	1.00000E+04	-1.72600E+03
53	Fix	.000E+00	-8.27400E+04	-6.25270E+04	-6.25270E+04
55	Opt.	5.000E+03	-4.00800E+04	1.50600E+03	4.00800E+04

PROPERTY takes the values H / GIBBS_ENERGY[(*)] / TEMPERATURE[(*)]
 / PRESSURE[(*)] / VOLUME / X(*) / W(*) / ACTIVITY(*)
 / USER_VARIABLE(*) / SUM(*:*) / DIFFERENCE(*:*)
 / PRODUCT(*:*) / RATIO(*:*) / LOG10(*) / LN(*)
 where [] indicates an optional entry

LIST PROPERTY allows the values of certain calculated properties such as activities, enthalpies and mole fractions of components in phases to be listed to the screen following a successful phase equilibrium calculation. Operations such as SUM, DIFFERENCE, PRODUCT, RATIO, LOG10 and LN can also be performed, with calculated properties, constants or user variables as arguments.

By making use of PROPERTY, LABEL, ABSCISSA, VALUE, UNCERTAINTY and WEIGHT parameters within a single LIST command it is possible to associate a label, an appropriate abscissa variable, a target experimental value, an experimental uncertainty and a weight, used during optimisation, with each listed property. For example:

```
list prop h val -123000 unc 5000 wei 2 lab 'H:Jones97' abs x(Mo) !
```

Such commands, along with those used to set up phase equilibrium calculations, form the basis of a typical EXPERIMENTAL_FILE (see DEFINE).

H

lists the total enthalpy of the system under consideration.

GIBBS_ENERGY(*)

lists the total Gibbs energy of the system under consideration or the partial Gibbs energy of a component named in brackets. The CLASSIFY command should be used to establish appropriate reference states.

TEMPERATURE(*)

lists the temperature of the system under consideration or, in the case of a phase boundary calculation, the temperature of the nth phase boundary within a specified temperature RANGE, the required boundary number n being supplied in brackets.

PRESSURE(*)

lists the total pressure of the system under consideration or the partial pressure of a gaseous species named in brackets.

VOLUME

lists the total volume of the system under consideration.

X(*)

lists the mole fraction of a component, named in brackets, either in a particular phase (if a phase name in angular brackets is appended to the component name) or in the system as a whole. For example:

```
list property x(Si) !  
list prop x(CaO<LIQUID_OXIDE>) !
```

W(*)

lists the mass fraction of a component, named in brackets, either in a particular phase (if a phase name in angular brackets is appended to the component name) or in the system as a whole. For example:

```
list prop w(CaCl2) !  
list property x(Fe<BCC_A2>) !
```

ACTIVITY(*)

lists the activity of a component named in brackets. The CLASSIFY command should be used to establish appropriate reference states.

USER_VARIABLE(*)

lists the value assigned to the user variable named in brackets.

SUM(*:*)

lists the result of adding one calculated property, user variable or constant to another. The property names, user variable names or numbers should be entered in brackets, separated by a colon, following the SUM keyword. For example:

```
list property sum(gibbs_energy:user_v(g12)) !  
list property sum(x(Cr<LIQUID>):x(Ni<LIQUID>)) !
```

DIFFERENCE(*:*)

lists the result of subtracting one calculated property, user variable or constant from another. The property names, user variable names or numbers should be entered in brackets, separated by a colon, following the DIFFERENCE keyword. That specified second is subtracted from that specified first. For example:

```
list property difference(h:user_v(h1)) !  
list property diff(temperature:273.15) !
```

PRODUCT(*:*)

lists the result of multiplying one calculated property, user variable or constant by another. The property names, user variable names or numbers should be entered in brackets, separated by a colon, following the PRODUCT keyword. For example:

```
list property product(5:gibbs_energy) !
```

RATIO(*:*)

lists the result of dividing one calculated property, user variable or constant by another. The property names, user variable names or numbers should be entered in brackets, separated by a colon, following the RATIO keyword. That specified first is divided by that specified second. For example:

```
list property ratio(gibbs_energy:2) !  
list prop rat(x(FeO<LIQUID>):x(Fe2O3<LIQUID>)) !
```

LOG10(*)

lists the base ten logarithm of a calculated property, user variable or constant. The property name, user variable name or number should be given in brackets following the LOG10 keyword.

LN(*)

lists the natural logarithm of a calculated property, user variable or constant. The property name, user variable name or number should

be given in brackets following the LN keyword.

RESULT takes the value EXPERIMENT[(*)] where [] indicates an optional entry

The RESULT parameter is used to generate a table comparing the calculated values of all or selected properties accessed using LIST PROPERTY with target experimental data entered via LIST VALUE. Differences between calculation and experiment (errors) are expressed as multiples of experimental uncertainties entered using LIST UNCERTAINTY. This allow results corresponding to different types of property (such as Gibbs energies and mole fractions) to be compared on a common basis.

EXPERIMENT(*)

lists the differences between calculated properties accessed using LIST PROPERTY and experimental values entered via LIST VALUE. Particular calculations or groups of calculations may be selected by entering the appropriate calculation numbers or LABELs in brackets after the EXPERIMENT keyword. For example:

list result experiment(1,5-9,10,12) !

list result exp(enthalpy) !

list res exp(98RED/HOT,57OLD/HAT) !

No.	Label	Calculated value (C)	Experimental value (E)	Uncertainty (U)	Error (C-E)/U	Weight
1	PB:eutectic	3.04904E-03	5.00000E-03	1.0000E-03	-1.95096E+00	5.000E+00
2	PB:eutectic	7.13620E-01	6.88000E-01	1.0000E-02	2.56196E+00	5.000E+00
3	PB:liquidus	8.42863E-03	1.00000E-02	1.0000E-02	-1.57137E-01	1.000E+00
4	PB:liquidus	2.54172E-02	2.60000E-02	1.0000E-02	-5.82789E-02	1.000E+00
5	PB:liquidus	7.84687E-02	7.90000E-02	1.0000E-02	-5.31350E-02	1.000E+00
6	PB:liquidus	1.20051E-01	1.23000E-01	1.0000E-02	-2.94886E-01	1.000E+00
7	PB:liquidus	1.93897E-01	2.02000E-01	1.0000E-02	-8.10273E-01	1.000E+00
8	PB:liquidus	2.60476E-01	2.72000E-01	1.0000E-02	-1.15236E+00	1.000E+00
9	PB:liquidus	3.51968E-01	3.63000E-01	1.0000E-02	-1.10316E+00	1.000E+00
10	PB:liquidus	6.20791E-01	6.06000E-01	1.0000E-02	1.47910E+00	1.000E+00
11	PB:liquidus	6.88784E-01	6.67000E-01	1.0000E-02	2.17839E+00	1.000E+00
12	PB:eutectic	7.03753E-01	6.88000E-01	1.0000E-02	1.57533E+00	5.000E+00
13	PB:liquidus	7.44976E-01	7.24000E-01	1.0000E-02	2.09762E+00	1.000E+00
14	PB:liquidus	8.45722E-01	8.23000E-01	1.0000E-02	2.27224E+00	1.000E+00
15	PB:liquidus	9.04020E-01	8.90000E-01	1.0000E-02	1.40198E+00	1.000E+00
16	Activity:800	8.88463E-03	1.40000E-02	2.0000E-02	-2.55769E-01	1.000E+00
17	Activity:800	3.83371E-02	4.20000E-02	2.0000E-02	-1.83147E-01	1.000E+00
18	Activity:800	1.00678E-01	9.50000E-02	2.0000E-02	2.83901E-01	1.000E+00
19	Activity:900	9.27908E-03	1.70000E-02	2.0000E-02	-3.86046E-01	1.000E+00
20	Activity:900	3.90124E-02	4.90000E-02	2.0000E-02	-4.99382E-01	1.000E+00
21	Activity:900	1.01102E-01	1.08000E-01	2.0000E-02	-3.44877E-01	1.000E+00
22	Activity:900	1.99475E-01	2.02000E-01	2.0000E-02	-1.26246E-01	1.000E+00
23	Activity:900	4.71964E-01	4.92000E-01	2.0000E-02	-1.00180E+00	1.000E+00
24	Activity:900	7.60795E-01	7.94000E-01	2.0000E-02	-1.66024E+00	1.000E+00
25	Teutectic	4.28139E+02	4.27000E+02	1.0000E+00	1.13928E+00	5.000E+00
26	Teutectic	7.68912E+02	7.67000E+02	1.0000E+00	1.91238E+00	5.000E+00
27	DeltaHf	-3.13361E+04	-3.24700E+04	2.5000E+03	4.53558E-01	1.000E+00
28	Hmelt	5.49836E+04	4.94600E+04	1.0000E+03	5.52355E+00	1.000E+00
29	Tmelt	1.01381E+00	.00000E+00	5.0000E+00	2.02761E-01	5.000E+00
Mean weighted square of errors of selected experiments (omitting any undefined values) = 2.75865D+00						

Typical output from *list result experiment* is shown above.

SYSTEM takes the values ELEMENTS / COMPONENTS / SUBSTANCES
/ PHASES / UNARIES / SETTINGS / ALL

The SYSTEM parameter is used to LIST the current calculation SETtings and selected attributes of the system under study to the screen. Information accessible in this way is important in other commands, CLASSIFY and SET for example, where COMPONENTS, PHASES and SUBSTANCES are referred to by their numerical position in the appropriate SYSTEM list.

ELEMENTS

lists the ELEMENTS in the currently DEFINED system. This is particularly useful if the COMPONENTS are non-elemental.

COMPONENTS

lists the COMPONENTS of the currently DEFINED system, their status (NORMAL, ABSENT, FREE or PARA, see CLASSIFY), any associated reference phases (see CLASSIFY), overall amounts (see SET) and, in the case of FREE components, incremental amounts, DELTA, used as a basis for iteration towards equilibrium constraints. COMPONENTS may be elements or compounds.

SUBSTANCES

lists the individual SUBSTANCES in the currently DEFINED system and their status (NORMAL or ABSENT, see CLASSIFY).

PHASES

lists the PHASES in the currently DEFINED system, displays their status (NORMAL, ABSENT, number of MISCIBILITY gaps, see CLASSIFY) and indicates how they are modelled, or how interactions between substances in the same phase are treated.

UNARIES

lists the UNARIES present in the currently DEFINED system, their SOURCE database (see DEFINE) and the temperature range over which their thermodynamic data apply.

Unaries and substances are often identical. The most common case where they are not is that of phases modelled using different sublattices. Substance lists for such phases contain separate entries for each species (atom or ion, for example) on each sublattice. Unary lists, comprise combinations of such species, one on each sublattice. It is to such combinations that thermodynamic data are assigned.

SETTINGS

lists the current calculation SETtings (see SET) including the system temperature, system pressure or volume, total system amount or mass, amounts and masses of individual components, any temperature, pressure or composition RANGES established, any equilibrium constraints and any USER VARIABLE values.

ALL

lists the COMPONENTS, SUBSTANCES, ELEMENTS, PHASES, UNARIES in the currently DEFINED system as well as the current calculation SETTINGS.

list system all !

is equivalent to

list system comp subs elem phas unar sett !

VALUE takes the values <experimental value > / NONE

The VALUE parameter of the LIST command is used to associate a target experimental value with a calculated PROPERTY accessed using the LIST PROPERTY command. This is usually done in an EXPERIMENTAL_FILE (see DEFINE) rather than interactively. Differences between calculated properties and experimental VALUES, scaled using a stated UNCERTAINTY, contribute to the sum of squares of errors which is minimised during optimisation runs.

list property h value -125000 uncertainty 5000 !

A previously specified experimental VALUE can be cancelled by entering a new VALUE of NONE.

UNCERTAINTY takes the values <experimental uncertainty> / NONE

The UNCERTAINTY parameter of the LIST command is used to associate an UNCERTAINTY with an experimental VALUE entered using LIST VALUE, corresponding to PROPERTY accessed using LIST PROPERTY. This is usually done in an EXPERIMENTAL_FILE (see DEFINE) rather than interactively. The stated UNCERTAINTY is used to scale differences between calculated properties and experimental VALUES before evaluating contributions to the sum of squares of errors, minimised during optimisation runs. Scaling allows errors associated with different types of PROPERTY (enthalpies and mole fractions, for instance) to be compared on a common basis.

list prop w(SiO2<LIQUID_OXIDE>) value 0.534 uncertainty 0.01 !

A previously specified UNCERTAINTY can be cancelled by entering a new value of NONE.

WEIGHT takes the values <weight for experiment> / NONE

The WEIGHT parameter of the LIST command allows each calculated PROPERTY, associated with an experimental VALUE and UNCERTAINTY, to be assigned a WEIGHT for use during optimisation runs. Entering a WEIGHT of 2 for a particular PROPERTY in an EXPERIMENTAL_FILE (see DEFINE) would double the contribution of that PROPERTY to the calculated sum of squares of errors. It would be as if the complete LIST PROPERTY

command had been entered in the EXPERIMENTAL_FILE twice. If no WEIGHT is specified a WEIGHT of 1 is assumed.

list prop act(O2) value 0.238 uncertainty 0.01 weight 5 !

A previously specified WEIGHT can be cancelled by entering a new value of NONE.

General comments:

The parameters of the LIST command can be divided into two categories, those used to gain information (SYSTEM, PARAMETER, RESULT) and those used to set up calculations to be undertaken during optimisation runs (PROPERTY, VALUE, UNCERTAINTY, WEIGHT, ABSCISSA, LABEL_OF_EXPERIMENT).

Typically the latter parameters are used together within a single LIST command, accessing a calculated result and supplying the additional information necessary to incorporate that result in an optimisation run, such as a target value and experimental uncertainty. An EXPERIMENTAL_FILE, used to control optimisation runs, might contain many such LIST commands interspersed with CLASSIFY, RANGE, SET and COMPUTE commands, these being used to set up and perform individual calculations, making results available for LIST PROPERTY.

For the sake of clarity it is often useful to separate the ASSESSMENT module commands used in an EXPERIMENTAL_FILE from the numeric data provided. This can be done by making repeated calls to smaller sub-macros, which prompt for information. An example of such a macro, named b2o3act, designed to calculate B₂O₃ activities in Na₂O-B₂O₃ melts, is shown below:

```

classify reference(2) LIQUID !
set w 1 w(2) *****
t *****
! compute !
list property act(B2O3)
value *****
uncertainty *****
weight *****
abscissa *****
label *****
!
```

The use of macros such as b2o3act enables EXPERIMENTAL_FILES to be written which take the form of simple tables of data. An example is given below:

; macro	w(2)	T/K	act(B2O3)	uncert.	weight	abscissa	label
'b2o3act'	0.7	1123	0.05	0.02	1	x(B2O3)	'act:76Green'
'b2o3act'	0.8	1123	0.21	0.02	1	x(B2O3)	'act:76Green'
'b2o3act'	0.9	1123	0.84	0.02	1	x(B2O3)	'act:76Green'
'b2o3act'	0.6	1273	0.02	0.01	1	x(B2O3)	'act:76White'
'b2o3act'	0.8	1273	0.28	0.01	1	x(B2O3)	'act:76White'
'b2o3act'	0.8	1273	0.31	0.02	1	x(B2O3)	'act:76Brown'

4.3 CLASSIFY

Function: The CLASSIFY command is used to remove individual components, phases or substances from consideration in subsequent calculations or to reintroduce those previously removed, to establish reference states for the thermodynamic properties of components, to flag "PARA" components, the relative amounts of which are to be fixed in all phases in which they appear, and to indicate potential miscibility gaps in phases. It is also important in setting up open system calculations in that it allows appropriate computational parameters to be specified and the amounts of individual components to be set FREE to change.

Parameters:

ABSENT	DELTA(*)
FREE	MISCIBILITY(*)
NORMAL	PARA
REFERENCE(*)	

Defaults:

DELTA(*) = determined by the program

MISCIBILITY(*) = as flagged in the current data (.mpi) file

NORMAL = all components, phases and substances unless flagged as absent in the current data (.mpi) file

REFERENCE(*) = DEFAULT_REF_STATE
(that used in the current data (.mpi) file)

Examples:

```

classify absent phase(3) 1 2 7 !
classify normal component(1) !
classify normal substance(CO2<g>) 12 46 !
classify misc(9) = 1 free component(3) !
classify reference(Cr) bcc !
classify normal p(*) !
classify normal substance(all) !
classify absent substance(13,15-17) !
classify para component(1) component(2) !

```

Parameter values:

ABSENT, NORMAL take the values COMPONENT(*) / SUBSTANCE (*)
PHASE(*) / <number>

The ABSENT parameter is used to remove selected COMPONENTs, SUBSTANCEs or PHASEs from consideration in subsequent phase equilibrium calculations. The NORMAL parameter is used to reintroduce COMPONENTs, SUBSTANCEs or PHASEs previously classified ABSENT. Although the default status of all COMPONENTs, SUBSTANCEs and PHASEs is NORMAL, certain SUBSTANCEs and PHASEs may be classified ABSENT automatically on reading thermodynamic data from a .mpi file (see DEFINE). SUBSTANCEs will be classified ABSENT, for example, if they are constrained never to form in the system under study. PHASEs will be classified ABSENT if their data is deemed to be incomplete.

COMPONENT(*), SUBSTANCE(*), PHASE(*)

removes from consideration or reintroduces those COMPONENTs, SUBSTANCEs and PHASEs listed in brackets using either their names or their numbers from the appropriate SYSTEM list (see LIST SYSTEM). An asterisk in brackets (*) is used to indicate all COMPONENTs, SUBSTANCEs or PHASEs in the currently DEFINED system. Substances may also be indicated by specifying their numbers without the SUBSTANCE keyword.

DELTA(*) takes the values <increment> / UNDEFINED

The DELTA(*) parameter is used to specify a step size for the amount of a FREE component, to be used as a basis for iteration towards an equilibrium constraint. This is effective only in BOTH_LARGE or STAGE_1 calculations (see COMPUTE). The component name or its number on the SYSTEM list should be given in brackets after the DELTA keyword. To cancel a previously established step size the value UNDEFINED should be entered.

In most cases it is best to allow the program to select its own step size. It will do this if DELTA is UNDEFINED.

FREE takes the value COMPONENT(*)

The FREE parameter is used to indicate that the total amount of a particular component is to be varied by the program in order to meet equilibrium constraints imposed on substances containing that component. The component name or its number on the SYSTEM list should be given in brackets after the COMPONENT keyword.

If oxygen, for example, as component number 2, were known to be present at equilibrium with an $O_2(g)$ partial pressure of 1×10^{-9} then the following commands might be entered:

```
set lgt_p(O2<g>) = -9 ! classify free comp(2) !
```

Care should be taken when setting equilibrium constraints since certain values might not have unique solutions in terms of the FREE component amounts required to achieve them. A warning is issued if an equilibrium constraint is SET without a corresponding FREE component classification.

A FREE classification can be cancelled by a subsequent NORMAL classification.

MISCIBILITY(*) takes the value <number> / NONE

The MISCIBILITY parameter provides a mechanism for flagging phases which may exhibit miscibility gaps. The phase name or its number on the SYSTEM list should be given in brackets after the MISCIBILITY keyword, then the number of possible miscibility gaps

should be entered. Possible values are 1, 2 or NONE, which cancels any previous MISCIBILITY classification giving the specified phase the status NORMAL. For example:

```
classify miscibility(9) = 1 miscibility(1) = none !
class misc(BCC) 2 !
```

MISCIBILITY classifications may appear automatically if data are read from a data input (.mpi) file which contains miscibility gap flags.

PARA takes the value COMPONENT(*)

CLASSIFY PARA provides a mechanism for flagging “PARA” components, the relative amounts of which are to be fixed in all phases in which they appear. The component name or its number on the SYSTEM list should be given in brackets after the COMPONENT keyword. Each phase in the system under study must either contain all of the components classified as PARA or none of them.

As an example of a para-equilibrium calculation, C might be assumed to diffuse rapidly during transformations from ferrite (BCC_A2) to austenite (FCC_A1) in the Fe-Cr-C system, while Fe and Cr do not diffuse at all. Fe and Cr would then be classified as PARA components.

REFERENCE[(*)] takes the values DEFAULT_REF_STATE/ <phase number> / <phase name> where [] indicates an optional entry

The REFERENCE parameter allows a reference phase to be established for each component present in the system under study. This allows thermodynamic properties such as Gibbs energy, enthalpy, chemical potential and activity to be referred to components in chosen phases. The component name or its number on the SYSTEM list should be given in brackets after the REFERENCE keyword. The phase name or its number on the SYSTEM list should then be provided as a value. For example:

```
classify refer(1) halite ref(SiO2) 7 !
classify reference liquid !
```

If no component name or number is given, the reference phase of every component is changed. The DEFAULT_REF_STATE is that implied in the data input (.mpi) file from which the thermodynamic data for the system under study were taken.

4.4 MODIFY_PARAMETER

Function: The MODIFY command is used to select thermodynamic parameters from the current data (.mpi) file the values of which are to be FIXED, OPTIMISED, LINKed or STEPPED, to assign new absolute or relative (OFFSET) values to parameters, to guide the optimisation process by providing SCALES for any changes made to parameters (also used as

step sizes during parameter stepping) and to LINK parameters together so that any changes made to one will be reflected in its partner(s).

Parameters:	INDEX	LINK
	OFFSET	SCALE
	TYPE	VALUE

Defaults: SCALE = determined by the program
 TYPE = fixed
 VALUE = parameter value from the current data (.mpi) file
 (with no offset)

Examples: *modify index 123 type optimise offset -10000 !*
mod ind 12 value -154749.35 !
modify index 15 link 9 !
modify index 86 type opt offset -5000 scale 1000 !
modify index 85 type fixed offset -10000 !
modify index 19 type unchanged !

Parameter values:

INDEX takes the value <number>

MODIFY INDEX is used to select a thermodynamic parameter from the DEFINED data input (.mpi) file. The value of this parameter, or its intended status during subsequent optimisation and stepping runs, can be changed using other keywords entered as part of the same MODIFY command (TYPE, VALUE, LINK, OFFSET, SCALE). The INDEX keyword should be followed by the number of the selected parameter obtained using the SHOW command which lists parameters available for change and their corresponding numbers.

LINK takes the value <number>

MODIFY LINK is used to set up groups of LINKed thermodynamic parameters such that changes made to one parameter in a group are reflected in the others. This is useful, for example, in ensuring that equivalent parameters appropriate to different temperature ranges within a single data set are treated identically during optimisation runs. The number of the parameter for which a LINK is to be established should be given after the INDEX keyword and the number of the parameter it is to be LINKed to, the target parameter, should be given after the LINK keyword. For example:

modify index 17 link 11 !

LINKs parameter 17 to parameter 11. Parameter numbers can be viewed using the SHOW command. The parameter with the lowest number within a LINKed group should act as the target for the others. More than one parameter can be LINKed to the same target, but LINKed parameters cannot, themselves, act as targets. Target

parameters must be OPTIMISED, STEPPED or FIXED (see MODIFY TYPE).

OFFSET takes the value <user supplied value>

MODIFY OFFSET is used to change the value of a thermodynamic parameter, selected using MODIFY INDEX, by applying a specified OFFSET to the number found in the data input (.mpi) file. According to the TYPE of the parameter in question (see MODIFY TYPE) this OFFSET might be FIXED or subject to automatic change during optimisation (OPTIMISED) or stepping (STEPPED). The OFFSET keyword should be followed by the numeric value of the required OFFSET. For example:

modify index 24 offset -10000 !

SCALE takes the value <user supplied value>

MODIFY SCALE is used to associate a SCALE with a thermodynamic parameter selected from the current data input (.mpi) file using MODIFY INDEX. A parameter's SCALE is used as a basis for determining the magnitude of changes to be made to its VALUE during optimisation and stepping. The SCALE required should be entered as a positive number after the keyword SCALE. For example:

modify index 36 value -10000 scale 1000 !

It is often best to let the program determine its own SCALE for changes to thermodynamic parameters made during optimisation. This is done by default or if a parameter's SCALE is set to zero.

In the output produced during optimisation runs (see OPTIMISE OUTPUT_TYPE) changes in parameter values are expressed as fractions or multiples of corresponding parameter SCALES. This allows changes associated with different types of parameter to be compared on common basis.

TYPE takes the values FIXED / STEPPED / OPTIMISED / UNCHANGED

MODIFY TYPE is used to indicate the intended status during stepping or optimisation runs of thermodynamic parameters selected using MODIFY INDEX. Parameters can be assigned fixed values (FIXED), flagged for automatic change during stepping runs (STEPPED) or flagged for automatic change during optimisation runs (OPTIMISED). Previous TYPE settings can be cancelled using the UNCHANGED keyword.

FIXED

indicates that the value of the selected thermodynamic parameter should remain FIXED during subsequent stepping and optimisation runs. It is often used in combination with MODIFY VALUE to assign a new fixed value to a chosen parameter. For example:

modify index 31 type fixed value -18456 !

STEPPED

flags the selected thermodynamic parameter such that, when the STEP_PARAMETER command is issued, its value is STEPPED automatically between limits determined by its SCALE, to determine the effect on calculated properties (see LIST PROPERTY) from an EXPERIMENTAL_FILE (see DEFINE). No more than two parameters can be STEPPED at one time.

modify index 7 type stepped scale 5000 !

OPTIMISED

flags the selected thermodynamic parameter for automatic change during future optimisation runs. For example:

modify index 21 type optimised !

UNCHANGED

negates the effects of previous OPTIMISED, STEPPED or FIXED entries for the selected thermodynamic parameter. The value of the parameter is returned to that in the data input (.mpi) file.

VALUE take the value <user supplied value>

MODIFY VALUE is used to specify a new value for a thermodynamic parameter selected from the current data input (.mpi) file using MODIFY INDEX. According to the TYPE of the parameter in question (see MODIFY TYPE) this VALUE might be FIXED or subject to automatic change during optimisation (OPTIMISED) or stepping (STEPPED). The VALUE keyword should be followed by the new number required. For example:

modify index 24 value 1427346.34 !

4.5 RANGE

Function: The RANGE command is used to specify temperature, pressure or composition ranges for consideration in subsequent phase boundary searches along with temperature, pressure or composition tolerance values indicating the accuracy to which individual boundaries are to be found. Currently only TEMPERATURE can be RANGED.

Parameters: LGT_PRESSURE PRESSURE
 TEMPERATURE
 or
 <"start group">, <"final group">, <composition tolerance>

Defaults: No default temperature, pressure or composition ranges apply.

Examples: *range temperature 1000 2000 0.1 !*

```
range lgt_pressure -10 0 0.01 !
range "start" "end" 0.001 !
```

Parameter values:

LGT_PRESSURE takes the values <lower limit>, <upper limit>, <tolerance>

RANGE LGT_PRESSURE will be used to specify, in logarithmic terms, a pressure range to be searched for phase boundaries and an accuracy for the determination of boundary pressures. Currently LGT_PRESSURE cannot be RANGED within the ASSESSMENT module.

PRESSURE takes the values <lower limit>, <upper limit>, <tolerance>

RANGE PRESSURE will be used to specify, a pressure range to be searched for phase boundaries and an accuracy for the determination of boundary pressures. Currently PRESSURE cannot be RANGED within the ASSESSMENT module.

TEMPERATURE takes the values <lower limit>, <upper limit>, <tolerance>

RANGE TEMPERATURE is used to specify a temperature RANGE to be searched for phase boundaries and an accuracy for the determination of boundary temperatures. The TEMPERATURE keyword should be followed by numbers representing, in turn, the lower temperature limit for the search, the upper limit temperature limit for the search and the accuracy required (usually 1.0 or 0.1 K). For example:

```
range t 1000 2000 0.1 !
```

Searching is initiated by use of the COMPUTE command once a temperature RANGE has been specified. The calculated temperature of the nth boundary found can then be accessed using LIST PROPERTY TEMPERATURE(n).

<"start group">, <"final group">, <composition tolerance>

Composition ranges for phase boundary searches will be specified in terms of named start and end composition groups. Currently composition cannot be RANGED within the ASSESSMENT module.

4.6 SET

Function: The SET command is most commonly used to fix the system temperature, system pressure or volume and initial amounts or masses of components, individually or in total, prior to a calculation. It can also be used to fix partial pressures, molalities, amounts or masses of individual species at equilibrium (see CLASSIFY FREE), to associate group names with system compositions or to assign values to named USER_VARIABLES.

Parameters:

LGT_MOLALITY(*)	LGT_PRESSURE(*)
MOLALITY(*)	N(*)
PRESSURE(*)	TEMPERATURE
VOLUME	W(*)
X(*)	USER_VARIABLE(*)
<"group name">	

Defaults:

PRESSURE = 101325 Pa
 TEMPERATURE = UNDEFINED
 VOLUME = UNDEFINED
 N = UNDEFINED (all components and system amount)
 W = UNDEFINED (all components and system mass)

Examples:

```

set temperature = 1010 pressure = 1e5 n(3) = 0.25 !
set pressure(23) = 1.2E-5 !
set lgt_pressure(CO<gas>) = -7.0 !
set v 0.01 !
set n(SiO2) = 0.1 !
set n(Fe<liquid>) = 0.12 !
set x(1) = 0.123 x(SiO2<LIQUID>) = 0.5 !
set w = 1 w(1) = 0.1 w(3) = 0.5 !
set user_variable(hold) = h !
set temperature = user_variable(told) !
set user_variable(g1) = g(Cr) !
set "start" 0.1 0.2 0.7 "end" 0.7 0.6 0.1 !
  
```

Parameter values:

MOLALITY(*) takes the values <user supplied value> / UNDEFINED

LGT_MOLALITY(*) takes the values <user supplied value> / UNDEFINED

The MOLALITY and LGT_MOLALITY parameters are used to assign a value to the equilibrium molality of a chosen substance in aqueous solution. In one case the required value itself is entered, in the other case its base ten logarithm. The substance in question is identified in brackets using its number or formula plus phase label as displayed by the command LIST SYSTEM SUBSTANCES. To cancel a previously established SETting, the keyword UNDEFINED should be entered in place of a numeric value.

PRESSURE[(*)] takes the values <user supplied value> / UNDEFINED

LGT_PRESSURE(*) takes the values <user supplied value> / UNDEFINED
 where [] indicates an optional entry

The PRESSURE parameter is used, with no substance name or formula appended in brackets, to set the total equilibrium pressure of the system under study. Values should be entered in pascals.

```
set pressure 101325 !
```

The PRESSURE and LGT_PRESSURE parameters are used, with a substance number or substance formula plus phase name appended

in brackets, to assign an equilibrium partial pressure value to a selected substance from the gas phase. In one case the required value itself is entered, in the other case its base ten logarithm.

```
set pressure(15) 101325*0.21 !
set lgt_press(O2<g>) -4 !
```

To cancel a previously established SETting, the keyword UNDEFINED should be entered in place of a numeric value.

N[*] takes the values <user supplied value> / UNDEFINED
where [] indicates an optional entry

The parameter N, with no brackets appended, is used to set a value for the sum of the amounts (in moles) of all components in the currently DEFINED system.

```
set n 100 !
```

Where brackets are appended and contain either a valid component number, as displayed by LIST SYSTEM COMPONENTS, or a substance formula without a phase label, the parameter N is used to set amounts (in moles) of individual components. A substance formula can only be entered if the system components are elements, in which case amounts for each component are calculated from the substance amount specified and used to supplement existing SETtings. If a component number is given, the amount of that component is set to the value entered.

```
set n(2) 0.567 !
set n(CO2) 0.2 !
```

Where brackets are appended and contain a substance formula with a phase label attached, as displayed by LIST SYSTEM SUBSTANCES, the parameter N is used to SET the equilibrium amount of the chosen substance. Equilibrium amounts cannot be SET for gaseous substances if the total system pressure has been SET already.

```
set n(O2<g>) 1.64E-5 !
```

To cancel a previously established SETting, the keyword UNDEFINED should be entered in place of a numeric value.

TEMPERATURE takes the values <user supplied value> / USER_VARIABLE(*)
/ UNDEFINED

The TEMPERATURE parameter is used to set the TEMPERATURE of the currently defined system to a specified value or to a value stored in a named USER_VARIABLE. In the latter case, TEMPERATURE should be followed by the keyword USER_VARIABLE with the appropriate USER_VARIABLE name appended in brackets.

TEMPERATURE values should be given in kelvin. Entry of the keyword UNDEFINED cancels any previously established SETtings.

```
set temperature 1273 !
set temp user_variable(temp7) !
```

VOLUME takes the values <user supplied value> / UNDEFINED

The VOLUME parameter is used to set the gas VOLUME of the currently defined system to a specified value, in metres cubed. Entry of the keyword UNDEFINED cancels any previously established SETtings.

W[*] takes the values <user supplied value> / UNDEFINED
where [] indicates an optional entry

The parameter W, with no brackets appended, is used to set a value for the sum of the masses (in kilograms) of all components in the currently DEFINED system.

```
set w 1 !
```

With brackets appended, containing a valid component number, as displayed by LIST SYSTEM COMPONENTS, the parameter W is used to SET the overall mass (in kilograms) of a particular component.

```
set w(1) 53.9 !
```

To cancel a previously established SETting, the keyword UNDEFINED should be entered in place of a numeric value.

X(*) takes the values <user supplied value> / UNDEFINED

The X parameter is used to assign a value to the equilibrium mole fraction of a chosen substance. The substance in question is identified in brackets using its number or formula plus phase label as displayed by the command LIST SYSTEM SUBSTANCES. To cancel a previously established SETting, the keyword UNDEFINED should be entered in place of a numeric value.

USER_VARIABLE(*) takes the values H / GIBBS_ENERGY[*] / PRESSURE[*]
/ TEMPERATURE[*] / VOLUME / X(*) / W(*) / ACTIVITY(*)
/ USER_VARIABLE(*) / SUM(*:*) / DIFFERENCE(*:*) / PRODUCT(*:*)
/ RATIO(*:*) / LOG10(*) / LN(*) / <user supplied value>
where [] indicates an optional entry

SET USER_VARIABLE is used to update the value of an existing USER_VARIABLE, named in brackets, or to establish a new USER_VARIABLE and assign it a value, if the name given in brackets is not already in use. Values can be calculated properties, accessed using keywords identical to those described in connection with LIST

PROPERTY, copied from other USER_VARIABLES or entered explicitly as numbers.

H

assigns the named USER_VARIABLE a value equal to the total enthalpy of the system under consideration.

GIBBS_ENERGY(*)

assigns the named USER_VARIABLE a value equal to the total Gibbs energy of the system under consideration or the partial Gibbs energy of a particular component named in brackets. The CLASSIFY command should be used to establish appropriate reference states.

```
set user_variable(gtot) = gibbs_energy !
set user(g1) = gibb(Cr) !
```

TEMPERATURE(*)

assigns the named USER_VARIABLE a value equal to the temperature of the system under consideration or, in the case of a phase boundary calculation, the temperature of the nth phase boundary within a specified temperature RANGE, the required boundary number n being supplied in brackets. For example:

```
set user_variable(tsys) = temperature !
set user_var(bound2) = temperature(2) !
```

PRESSURE(*)

assigns the named USER_VARIABLE a value equal to the total pressure of the system under consideration or the partial pressure of a particular gaseous species named in brackets. For example:

```
set user_variable(ptot) = pressure !
set user_var(po2) = press(O2<g>) !
```

VOLUME

assigns the named USER_VARIABLE a value equal to the total volume of the system under consideration.

X(*)

assigns the named USER_VARIABLE a value equal to the mole fraction of a component, named in brackets, either in a particular phase (if a phase name in angular brackets is appended to the component name) or in the system as a whole. For example:

```
set user_variable(x2) = x(Si) !
set user(xcao) = x(CaO<LIQUID_OXIDE>) !
```

W(*)

assigns the named USER_VARIABLE a value equal to the mass fraction of a component, named in brackets, either in a particular phase (if a phase name in angular brackets is appended to the component name) or in the system as a whole. For example:

```
set user_v(w1) w(CaCl2) !
set_user_variable(wfe) w(Fe<BCC_A2>) !
```

ACTIVITY(*)

assigns the named USER_VARIABLE a value equal to the activity of a particular component named in brackets. The CLASSIFY command should be used to establish appropriate reference states.

USER_VARIABLE(*)

assigns the named USER_VARIABLE a value equal to that of another named USER_VARIABLE. For example:

```
set user_var(tnew) = user_var(told) !
```

SUM(*:*)

assigns the named USER_VARIABLE a value equal to the result of adding one calculated property, user variable or constant to another. The property names, user variable names or numbers should be entered in brackets, separated by a colon, following the SUM keyword. For example:

```
set user_variable(gtot) = sum(gibbs_energy:user_v(g12)) !
set u_v(xsum) = sum(x(Cr<LIQUID>):x(Ni<LIQUID>)) !
```

DIFFERENCE(*:*)

assigns the named USER_VARIABLE a value equal to the result of subtracting one calculated property, user variable or constant from another. The property names, user variable names or numbers should be entered in brackets, separated by a colon, following the DIFFERENCE keyword. That specified second is subtracted from that specified first. For example:

```
set user_var(hdiff) = difference(h:user_v(h1)) !
set user_v(tcel) = diff(temperature:273.15) !
```

PRODUCT(*:*)

assigns the named USER_VARIABLE a value equal to the result of multiplying one calculated property, user variable or constant by another. The property names, user variable names or numbers should be entered in brackets, separated by a colon, following the PRODUCT keyword. For example:

```
set user_variable(gnew) = product(5:gibbs_energy) !
```

RATIO(*:*)

assigns the named USER_VARIABLE a value equal to the result of dividing one calculated property, user variable or constant by another. The property names, user variable names or numbers should be entered in brackets, separated by a colon, following the RATIO keyword. That specified first is divided by that specified second. For example:

```
set user_var(halfg) = ratio(gibbs_energy:2) !
set us_var(foox) = rat(x(FeO<LIQUID>:x(Fe2O3<LIQUID>)) !
```

LOG10(*)

assigns the named USER_VARIABLE a value equal to the base ten logarithm of a calculated property, user variable or constant. The property name, user variable name or number should be given in brackets following the LOG10 keyword.

LN(*)

assigns the named USER_VARIABLE a value equal to the natural logarithm of a calculated property, user variable or constant. The property name, user variable name or number should be given in brackets following the LN keyword.

<user supplied value>

assigns the named USER_VARIABLE a specified numeric value.

<"group name">

The <"group name"> option is used to assign a name to a particular set of component amounts or masses, for later reference within a RANGE command. The chosen group name, in quotes, should be followed by one number for each component in the currently DEFINED system. These numbers are interpreted as masses in kilograms if the total system mass is SET to one kilogram or amounts in moles otherwise. For example:

```
set "start" 0.1 0.2 0.7 "end" 0.7 0.2 0.1 !
```

General comments:

The best method to use in setting up a system composition will depend on the system being studied and the type of calculation being undertaken. In general different methods should not be used at the same time. Compositions may be set:

- 1) By entering the amounts (or masses) of numbered components individually. For example:

```
set n(1) 0.2 n(2) 0.5 n(3) 0.3 !
set w(1) 10 w(2) 30 w(3) 56 w(4) 4 !
```

- 2) By entering a blank component number, then the required component amounts (or masses) in turn within a single SET command. For example:

```
set n( ) 0.2 0.5 0.3 !
set w( ) 10 30 56 4 !
```

- 3) By entering the total system amount (or mass) and the individual amounts (or masses) of all but one of the components (the other component amount (or mass) being UNDEFINED). For example:

```
set n 1 n(2) 0.5 n(3) 0.3 !
set w 100 w(1) 10 w(2) 30 w(4) 4 !
```

- 4) By entering amounts of substances specified by formula without a phase label. For example:

```
set n(C2H6) 1 n(N2) 80 n(O2) 20 !
```

If an amount is entered for a specific substance, identified by number or formula and phase label, it is interpreted as the amount of that substance required at equilibrium. Setting equilibrium constraints upon substances (amounts or partial pressures, for example) as well as overall amounts for each system component, will generally produce an insoluble problem. Consequently, if an equilibrium constraint is applied to a particular substance, it is necessary to FREE the amounts of one or more of the components contained by that substance, or allow those component amounts to be determined by the program. For example, if the equilibrium partial pressure of O2<g> was fixed it would be necessary to CLASSIFY FREE either oxygen or another component containing oxygen.

4.7 STEP_PARAMETER

Function: The STEP_PARAMETER command is used to initiate calculations in which the values of one or two selected thermodynamic parameters are stepped automatically above and below their current values and a comparison is made between calculated and experimental properties from the DEFINED experimental data file at each stage. This is sometimes a useful prelude to a full optimisation run. The number of steps to be taken for each parameter must be specified as part of the complete STEP_PARAMETER command. The step limits are based upon SCALES established using the MODIFY_PARAMETER command.

Parameters: NUMBER_OF_STEPS

Defaults: NUMBER_OF_STEPS = UNDEFINED

Examples: *step number 20 !*

Parameter values:

NUMBER_OF_STEPS takes the value <number>

NUMBER_OF_STEPS is used to specify the number of different values to be considered for each STEPPED parameter (see MODIFY TYPE) during calculations initiated by the STEP_PARAMETER command. The keyword NUMBER_OF_STEPS should be followed by a positive integer. For example:

```
step_parameter number_of_steps 20 !
```

4.8 SHOW

Function: The SHOW command is used to discover which index (number) should be used to refer to a particular thermodynamic parameter in other commands such as MODIFY. It lists to the screen the names, current values and indices of thermodynamic parameters from the current data input (.mpi) file available to be FIXED, STEPPED, LINKED or OPTIMISED (see MODIFY). Parameters can be selected by type (unary or interaction) or by phase.

Parameters: ALL_DATA INTERACTION_DATA
UNARY DATA

Defaults: ALL DATA = ALL PHASES

Examples: *show all_data liquid !*
 show interaction all_phases !
 show unary bcc !

Parameter values:

ALL DATA takes the values <phase name> / ALL PHASES

SHOW ALL_DATA is used to list all thermodynamic parameters from the currently DEFINED data input (.mpi) file available to be FIXED, STEPPED, LINKED or OPTIMISED (see MODIFY). Parameters can be listed for a named phase or for ALL_PHASES in the currently DEFINED system.

```
show all_data liquid !
show all_data all_phases !
```

The ALL_DATA list includes all parameters from the UNARY_DATA and INTERACTION_DATA lists.

INTERACTION DATA takes the values <phase name> / ALL PHASES

SHOW INTERACTION_DATA is used to list those thermodynamic parameters appropriate to items of interaction data in the currently DEFINED data input (.mpi) file that are available to be FIXED, STEPPED, LINKED or OPTIMISED (see MODIFY). Parameters can be listed for a named phase or for ALL_PHASES in the currently DEFINED system.

```
show interaction_data fcc !
show interaction_data all phases !
```

Typical output obtained using the `SHOW INTERACTION_DATA` command can be examined below.

Index	T(high) /K	Current Value	Parameter Name	
62	3000.00	-3.74500E+04	CaCl2,ClK<LIQUID>	*xi*xj
63	3000.00	9.84500E+00	CaCl2,ClK<LIQUID>	*xi*xj*T
64	3000.00	8.19000E+03	CaCl2,ClK<LIQUID>	*xi*xj*(xi-xj)
65	3000.00	-5.74000E+00	CaCl2,ClK<LIQUID>	*xi*xj*(xi-xj)*T
66	3000.00	-9.25000E+02	CaCl2,ClK<LIQUID>	*xi*xj*(xi-xj)**2
67	3000.00	1.92900E+00	CaCl2,ClK<LIQUID>	*xi*xj*(xi-xj)**2*T
68	3000.00	3.14000E+03	CaCl2,Cl2Zn<LIQUID>	*xi*xj
69	3000.00	-2.09000E+03	CaCl2,Cl2Zn<LIQUID>	*xi*xj*(xi-xj)
70	3000.00	-8.27400E+04	ClK,Cl2Zn<LIQUID>	*xi*xj
71	3000.00	1.36940E+01	ClK,Cl2Zn<LIQUID>	*xi*xj*T
72	3000.00	-4.00800E+04	ClK,Cl2Zn<LIQUID>	*xi*xj*(xi-xj)
73	3000.00	1.88070E+01	ClK,Cl2Zn<LIQUID>	*xi*xj*(xi-xj)*T
74	3000.00	2.39100E+04	ClK,Cl2Zn<LIQUID>	*xi*xj*(xi-xj)**2
75	3000.00	.00000E+00	ClK,Cl2Zn<LIQUID>	*xi*xj*(xi-xj)**2*T
76	3000.00	1.58800E+04	ClK,Cl2Zn<LIQUID>	*xi*xj*(xi-xj)**3
77	3000.00	.00000E+00	ClK,Cl2Zn<LIQUID>	*xi*xj*(xi-xj)**3*T

UNARY_DATA takes the values <phase name> / ALL_PHASES

SHOW UNARY_DATA is used to list those thermodynamic parameters appropriate to items of unary data in the currently DEFINEd data input (.mpi) file that are available to be FIXED, STEPPED, LINKED or OPTIMISED (see MODIFY). Parameters can be listed for a named phase or for ALL_PHASES in the currently DEFINEd system.

show unary_data bcc !

show unary_data all_phases !

Index	T(high) /K	Current Value	Parameter Name	
85	700.00	-4.52490E+05	ClK<HALITE>	*xi
86	700.00	2.63150E+02	ClK<HALITE>	*xi*T
87	700.00	-5.12948E+01	ClK<HALITE>	*xi*T*ln(T)
88	700.00	-1.40523E-03	ClK<HALITE>	*xi*T**2
89	700.00	-1.73100E-06	ClK<HALITE>	*xi*T**3
90	700.00	7.67320E+04	ClK<HALITE>	*xi/T
91	800.00	-4.87176E+05	ClK<HALITE>	*xi
92	800.00	7.62308E+02	ClK<HALITE>	*xi*T
93	800.00	-1.27777E+02	ClK<HALITE>	*xi*T*ln(T)
94	800.00	7.29682E-02	ClK<HALITE>	*xi*T**2
95	800.00	-1.51909E-05	ClK<HALITE>	*xi*T**3
96	800.00	3.00201E+06	ClK<HALITE>	*xi/T
97	1045.00	-7.29641E+05	ClK<HALITE>	*xi
98	1045.00	3.63572E+03	ClK<HALITE>	*xi*T
99	1045.00	-5.53395E+02	ClK<HALITE>	*xi*T*ln(T)
100	1045.00	4.04611E-01	ClK<HALITE>	*xi*T**2
101	1045.00	-6.35871E-05	ClK<HALITE>	*xi*T**3
102	1045.00	2.88679E+07	ClK<HALITE>	*xi/T
103	1100.00	-9.29276E+06	ClK<HALITE>	*xi
104	1100.00	8.37326E+04	ClK<HALITE>	*xi*T
105	1100.00	-1.19459E+04	ClK<HALITE>	*xi*T*ln(T)
106	1100.00	7.09895E+00	ClK<HALITE>	*xi*T**2
107	1100.00	-7.95735E-04	ClK<HALITE>	*xi*T**3
108	1100.00	1.22924E+09	ClK<HALITE>	*xi/T
109	2000.00	-4.69544E+05	ClK<HALITE>	*xi
110	2000.00	4.29820E+02	ClK<HALITE>	*xi*T
111	2000.00	-7.33994E+01	ClK<HALITE>	*xi*T*ln(T)


```
plot errors experiment(1,3,5,10,12) !
plot errors exp !
```

Part of a LABEL can be entered to select results with different LABELs beginning with the same group of letters.

Typical output generated using *plot errors experiment* is shown on the next page.

VALUES takes the value EXPERIMENT[(*)] where [] indicates an optional entry

The VALUES parameter is used to generate graphs in which calculated PROPERTY values accessed using LIST PROPERTY are plotted as ordinate and a variable specified using LIST ABSCISSA, common to each PROPERTY, is plotted as abscissa. Symbols representing experimental data, entering using LIST VALUE, with error bars based upon uncertainties specified using LIST UNCERTAINTY, are superimposed. All or selected LIST PROPERTY values may be included.

The LIST commands associating an experimental VALUE, UNCERTAINTY and an ABSCISSA variable with each calculated PROPERTY usually appear in an EXPERIMENTAL_FILE (see DEFINE).

EXPERIMENT(*)

is used to select results for inclusion in a plot. This is done by appending a list of PROPERTY numbers or LABELs (see LIST PROPERTY) to the EXPERIMENT keyword, in brackets. The same ABSCISSA variable should be associated with each selected PROPERTY (see LIST ABSCISSA). If the EXPERIMENT keyword is entered without brackets, all available results are selected. For example:

```
plot values experiment(1,3,5,10,12) !
plot values exp !
```

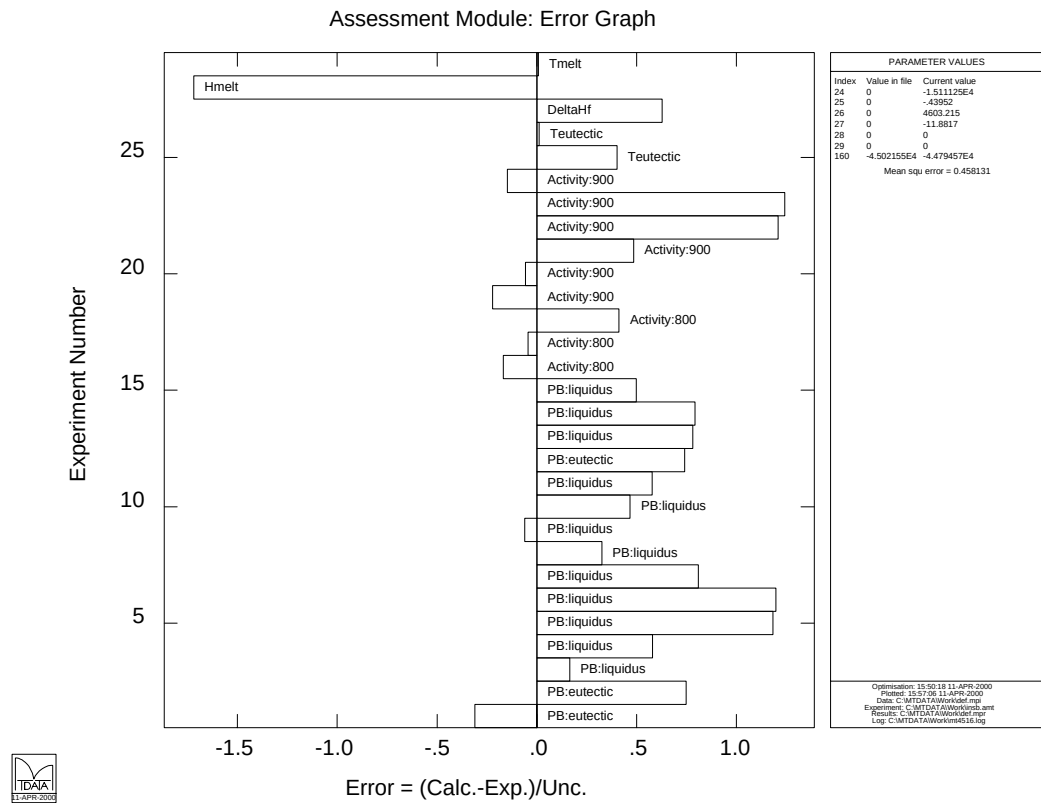
Part of a LABEL can be entered to select results with different LABELs beginning with the same group of letters.

Typical output generated using *plot values experiment* is shown on the next page.

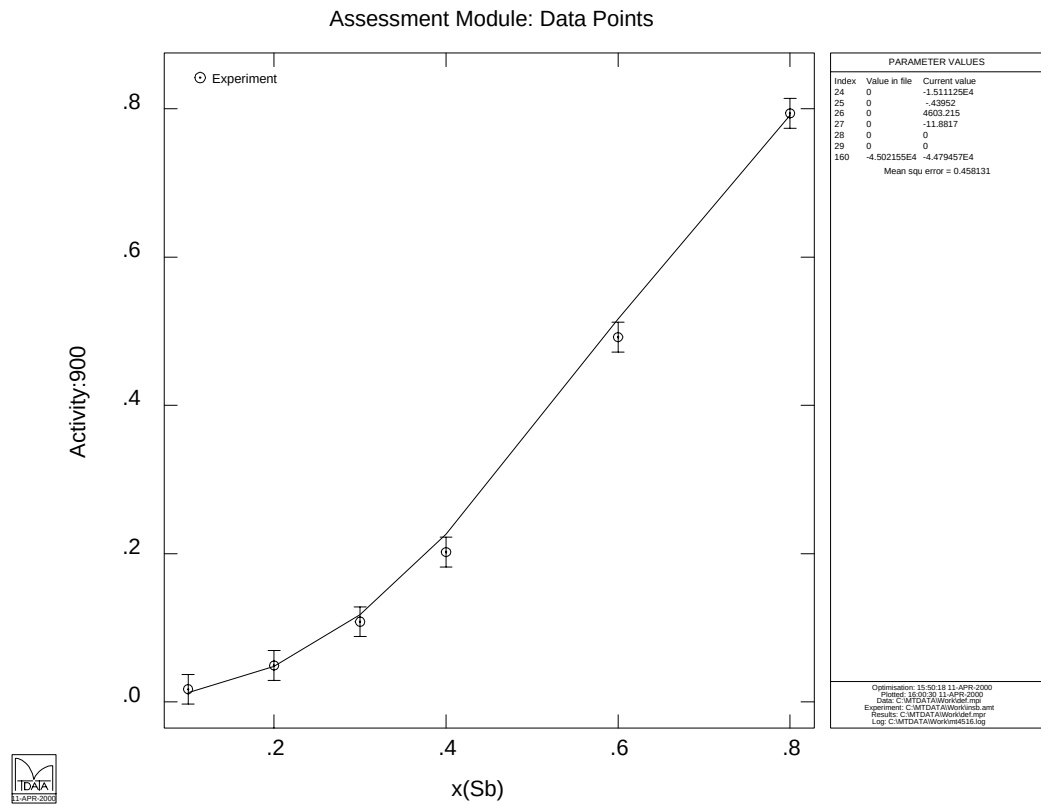
4.10 SAVE

Function: The SAVE command is used during the optimisation process to save the current values of selected FIXED, LINKED, STEPPED or OPTIMISED PARAMETERS to an in-memory store, to save data sets incorporating the current values of FIXED, LINKED, STEPPED and OPTIMISED parameters to a named DATA_LOADING_FILE ready for loading into a database or to write a SET_UP macro to be used at a later date to recreate the exact state of the current optimisation in

Typical output from the command *plot errors experiment* !



Typical output from the command *plot values experiment* !



terms of the data input filename, experimental data file name and the status and value of individual parameters.

Parameters: DATA_LOADING_FILE
PARAMETER
SET_UP

Defaults: PARAMETER = ALL

The default SAVED value for each parameter is the value taken from the current data input (.mpi) file.

Examples: *save data_loading_file 'newdata.loa' !*
save parameter 35 !
save parameter all !
save set_up 'crfeass.mac' !

Parameter values:

DATA_LOADING_FILE takes the value <"file name">

SAVE DATA_LOADING_FILE is used to save data sets incorporating the current values of all FIXED, LINKED, STEPPED and OPTIMISED parameters (see MODIFY) to a named MTDATA database loading file. The DATA_LOADING_FILE keyword should be followed by a valid file name in quotes. For example:

save data "optdata.loa" !

The resulting file can be used to create a new database or update an existing database by means of the UTILITY module's LOAD_DATA command.

PARAMETER takes the values <number> / ALL

SAVE PARAMETER is used to save the current values of ALL or selected FIXED, LINKED, STEPPED and OPTIMISED PARAMETERS (see MODIFY) to an in-memory store. The stored values can be inspected using the LIST PARAMETER VALUE command and restored using the RESTORE command, overwriting any new values current at the time.

To save an individual parameter the PARAMETER keyword should be followed by the appropriate parameter INDEX (see MODIFY and SHOW). To save all FIXED, LINKED, STEPPED and OPTIMISED parameters the keyword ALL should be used. For example:

save parameter = 25 !
save param all !

SET_UP takes the value <"file name">

SAVE SET_UP is used to write a SET_UP macro in which the exact state of the current optimisation, in terms of the data input (.mpi) file name, EXPERIMENTAL_FILE name and the status and value of individual parameters, is stored. Running the macro from the WHICH MODULE ? prompt during a subsequent MTDATA session will regenerate all of the SAVED settings. The commands UNDO or RESTORE, however, will not work until further OPTIMISE or SAVE PARAMETER commands, respectively, have been issued.

The SET_UP keyword should be followed by a valid file name in quotes. For example:

```
save set_up "mysetup.mac" !
```

4.11 COMPUTE

Function: This command is used to initiate a phase equilibrium calculation based upon the current settings (see SET) and classifications (see CLASSIFY). The required type of calculation (STAGES_REQUIRED and INITIAL_POINT_OPTION) can be specified. Note that this command does not initiate an optimisation run (see OPTIMISE).

Parameters: INITIAL_POINT_OPTION STAGES_REQUIRED

Defaults: INITIAL_POINT_OPTION = AUTOMATIC
STAGES_REQUIRED = 1_ONLY

Examples: *compute !*
compute stages = both_small initial = last !

Parameter values:

INITIAL_POINT_OPTION takes the values AUTOMATIC / LOWER_BOUNDS
/ PREVIOUS_SOLUTION / LAST_ACTIVE_SET

The INITIAL_POINT_OPTION parameter is used to specify how initial values for properties such as equilibrium substance amounts are to be estimated. These initial values are altered by the program during the course of a phase equilibrium calculation to achieve the lowest possible overall Gibbs energy.

Usually it is best to accept the default INITIAL_POINT_OPTION, namely AUTOMATIC, and let the program choose its own initial values. Sometimes, however, it is beneficial to use results from one calculation to provide initial values for another. This is done by entering INITIAL_POINT LAST_ACTIVE_SET or INITIAL_POINT PREVIOUS_SOLUTION. A typical use of these options is to speed up calculations in a closely related series.

AUTOMATIC

specifies that the program should select its own initial values.

LOWER_BOUNDS

specifies that initial amounts should be set at their lower bounds, effectively zero.

PREVIOUS_SOLUTION

specifies that the equilibrium substance amounts produced by one calculation should be used as initial values in the next.

LAST_ACTIVE_SET

similar to PREVIOUS_SOLUTION but also makes available intermediate variables from a preceding STAGE_1 calculation.

STAGES_REQUIRED takes the values 1_ONLY / STAGE_1_PLUS / 2_ONLY / ROBUST_STAGE_2 / BOTH_LARGE / BOTH_SMALL

The STAGES_REQUIRED parameter is used to select the algorithm to be used in performing phase equilibrium calculations.

1_ONLY

selects the STAGE_1 Gibbs energy minimisation algorithm, which gives equilibrium compositions to six decimal places in amounts. STAGE_1 is the only algorithm available for certain thermodynamic models, including the compound energy model with multiple sublattices. It should not be used when the amounts at equilibrium of more than two substances are constrained and is not suitable for calculating amounts of less than 10^{-6} moles.

STAGE_1_PLUS

selects the STAGE_1_PLUS algorithm which effectively carries out a highly robust STAGE_1 calculation followed by a STAGE_2 calculation for certain phase types, such as gas. This produces a refinement of the composition of such phases.

2_ONLY

selects the STAGE_2 algorithm, which is not as robust as STAGE_1 and can fail unless used in conjunction with STAGE_1 (BOTH_LARGE, BOTH_SMALL, ROBUST_STAGE_2) or, with an appropriate INITIAL_POINT_OPTION, following a preceding calculation with very similar settings. STAGE_2 gives equilibrium compositions to six decimal places in logarithms of amounts. Its applicability is limited to certain models.

ROBUST_STAGE_2

selects the ROBUST_STAGE_2 algorithm which is an intelligent combination of STAGE_1 and STAGE_2 involving repeated STAGE_2 calculations interspersed with STAGE_1 calculations if difficulties arise. This is useful for performing sets of related calculations in systems with large numbers of species, but should not be used if the number of solution phases at equilibrium is liable to change.

BOTH LARGE

selects an algorithm involving the automatic use of STAGE_1 followed by STAGE_2, any equilibrium constraints being applied in both cases. Its use is appropriate where equilibrium constraints on amounts are of the order of 10^{-5} or above and two in number, at most. Otherwise BOTH_SMALL should be used. STAGE_2 limitations on models apply.

BOTH SMALL

selects an algorithm involving the automatic use of STAGE_1 followed by STAGE_2 with any equilibrium constraints being applied just for STAGE_2. Its use is appropriate where equilibrium constraints on amounts are of the order of 10^{-5} or below or number more than two. STAGE_2 limitations on models apply.

4.12 OPTIMISE

Function: The OPTIMISE command is used to initiate an optimisation run in which thermodynamic parameters marked as OPTIMISED or LINKED are adjusted automatically to minimise the differences between calculated properties and experimental values (see LIST) as defined in an experimental data file (see DEFINE). Options allow different ALGORITHMS, and TOLERANCES to be selected for the optimisation process, the overall number of comparisons made between calculation and experiment (MAXIMUM_FUNCTIONS) to be limited and a method for monitoring progress (by means of TEXT or GRAPHICAL output) to be specified. In the case of GRAPHICAL monitoring a frequency for updating the output can be specified.

Parameters:

ALGORITHM	MAXIMUM_FUNCTIONS
OUTPUT_TYPE	TOLERANCE
UPDATE_GRAPHICS	

Defaults: ALGORITHM = LCQNDF
 MAXIMUM_FUNCTIONS = 100
 OUTPUT_TYPE = GRAPHICAL
 TOLERANCE = 0.01
 UPDATE_GRAPHICS = 1

Examples: *optimise !*
 optimise algorithm = upolyt toler = 0.001 out = graph !
 opt alg = lsqndn max = 500 !
 optimise max = 0 !

Parameter values:

ALGORITHM takes the values LCQNDF / UPOLYT / LSQNDN

ALGORITHM is used to select the algorithm to be used in subsequent optimisation runs, that is to say the mathematical method employed to determine the best possible values for OPTIMISED parameters.

Options are selected by entering the keywords LCQNDF, UPOLYT and LSQNDN. The default, if no other choice is made, is LCQNDF.

The LCQNDF and LSQNDN algorithms work best when the parameters being changed are reasonably close to their final values. UPOLYT is often a better choice when the parameters being changed are far from their final values.

LCQNDF

requests use of the LCQNDF (Linearly Constrained Quasi Newton Function only) algorithm. This algorithm is used by default.

UPOLYT

requests use of the UPOLYT (Unconstrained POLYTope) algorithm. This algorithm is recommended for general use rather than LCQNDF or LSQNDN.

LSQNDN

requests use of the LSQNDN (Least Squares No Derivatives Needed) algorithm. This works in a similar way to LCQNDF but also makes available a detailed statistical analysis of each optimisation run.

MAXIMUM_FUNCTIONS takes the value <number>

MAXIMUM_FUNCTIONS is used to specify a maximum number of times for the sum of squares of errors, minimised during an optimisation run, to be evaluated. The default value is 100. Setting too low a value may not allow enough scope for the best possible set of thermodynamic parameters to be found. Setting too high a value, especially with a small TOLERANCE, may lead to unnecessary calculations being performed to try and improve an already adequate result. The chosen number, a positive integer, should be entered after the MAXIMUM_FUNCTIONS keyword. For example:

optimise max_fun = 500 !

During an optimisation run results are reported, graphically or in text form, whenever a new lowest value for the sum of squares of errors is found, that is to say when a new iteration begins. Each iteration may involve more than one evaluation of the sum of squares of errors.

OUTPUT_TYPE takes the values GRAPHICAL / TEXT

OUTPUT_TYPE is used to select whether the progress of subsequent optimisation runs is to be monitored graphically or by means of tables or intermediate parameter values output to the screen. It should be followed either by the keyword GRAPHICAL or by the keyword TEXT. GRAPHICAL is the default option. The tabular output generated by entering TEXT may be lengthy.

GRAPHICAL

generates graphs showing changes made to individual parameters during the course of each optimisation run and the effects of these changes on the sum of squares of errors being minimised. Plots are updated with a frequency requested using UPDATE_GRAPHICS.

TEXT

generates tables showing changes made to individual parameters during the course of each optimisation run and the effects of these changes on the sum of squares of errors being minimised. New tables are output with a frequency requested using UPDATE_GRAPHICS. Output may be lengthy.

TOLERANCE takes the value <number>

TOLERANCE is used to specify a target accuracy to be achieved in optimisation runs. In the case of the LCQNDF and LSQNDN algorithms the value supplied is interpreted as a target accuracy for each thermodynamic parameter, expressed a fraction of its SCALE (see MODIFY). In the case of UPOLYT it is interpreted as a limiting value for the sum of squares of errors, minimised during each optimisation run.

The required TOLERANCE should be entered as a positive number immediately following the TOLERANCE keyword. For example:

optimise tolerance = 0.001 !

For most purposes there will be no need to change the default value of 10^{-5} adopted by the program. This is close to the lower limit of suitable values.

UPDATE_GRAPHICS takes the value <number>

UPDATE_GRAPHICS is used to specify a frequency for updating the output, text or graphics, used to monitor the progress of optimisation runs. This is expressed as a number of iterations to be completed before each update, entered as a positive integer after the UPDATE_GRAPHICS keyword. For example:

optimise update 5 !

An iteration is completed each time a new lowest value for the sum of squares of errors, minimised during optimisation runs, is found. By default output is updated at the end of every iteration, that is to say UPDATE_GRAPHICS = 1.

4.13 LASER

Function: The LASER command is used on selected MTDATA platforms to request hard copies of those graphs plotted most recently but not

previously sent to a printer. This command is not used in MTDATA for Windows 9x and NT.

Parameters: No parameters.

Defaults: Not applicable.

Example: *laser*

4.14 RESTORE

Function: The RESTORE command is used to overwrite the current values of selected FIXED, LINKed, STEPPED or OPTIMISED (see MODIFY) PARAMETERS with those previously SAVED in an in-memory store.

Parameters: PARAMETER

Defaults: PARAMETER = ALL

The default SAVED value for each parameter is the value taken from the current data input (.mpi) file.

Examples: *restore parameter 23 !*
restore parameter all !

Parameter values:

PARAMETER takes the values <number> / ALL

RESTORE PARAMETER is used to recover the values of individual or complete sets of thermodynamic parameters previously SAVED to an in-memory store (see SAVE PARAMETER). Existing values are overwritten.

To restore a single value, PARAMETER should be followed by the appropriate parameter INDEX (see MODIFY and SHOW). To recover all SAVED parameters, the keyword ALL should be used. For example:

restore parameter = 25 !
restore param all !

LIST PARAMETER VALUE can be used to inspect STOREd values.

4.15 UNDO_OPTIMISATION

Function: The UNDO_OPTIMISATION command negates the effects of the last optimisation run performed before its issue. New values for thermodynamic parameters generated during an optimisation run are, by default, adopted as current when the run ends. The UNDO_OPTIMISATION command overwrites any new values by those current at the start of the last optimisation run.

Parameters: No parameters.

Defaults: Not applicable.

Example: *undo_optimisation*

4.16 RETURN

Function: The RETURN command is used to leave the ASSESSMENT module ready to begin work in another module or to leave MTDATA altogether. The current optimisation SET_UP will be lost unless previously SAVED.

Parameters: No parameters

Defaults: Not applicable.

Example: *return*

4.17 SWAP

Function: The SWAP command is used to leave the ASSESSMENT module and enter another module, such as BINARY or MULTIPHASE, retaining the current optimisation SET_UP (provided the DEFINE command is not used in the new module). This allows calculations to be performed in modules other than ASSESSMENT using the current set of OPTIMISED (but not necessarily SAVED) parameters. The command SWAP ASSESSMENT is used to return to the ASSESSMENT module.

Parameters: <the name of another MTDATA module>

Defaults: Not applicable.

Examples: *swap multiphase*
swap binary

4.18 END_OF_EXPERIMENTS

Function: The END_OF_EXPERIMENTS command is used to mark the end of the contents of an experimental file (see DEFINE). It should only be used in an experimental file and never entered interactively.

Parameters: No parameters.

Defaults: Not applicable.

Example: *end_of_experiments*

4.19 <"MACRO NAME">

Function: A macro, or list of MTDATA commands stored in a file, can be run by simply specifying the name of the file in single or double quotes.

Operating system commands can be executed by prefixing them with "\$ or just \$.

Parameters: A list of parameters and values appropriate to the particular macro being run may be supplied on the same line of input after the quote(s) following the macro name.

Defaults: Not applicable.

Examples: *'mymacro.mac*
'mt-transn_t.mac' 20 0 300 3000 0.1
\$dir

5 QUICK REFERENCE SECTION

DEFINE

Function: The DEFINE command is used to obtain thermodynamic data for a system to be studied, either from listed databases, in which case the system must be specified in terms of its components, or from a previously created data input (.mpi) file. Data found during a database search are written automatically to a named or default data output file using the .mpi format. The DEFINE command is also used to specify the file (macro) which contains calculation instructions and experimental data for planned optimisation runs.

Parameters:

data_input_file =	<"file name"> / default
experimental_file =	<"file name"> / default
output_data / results =	<"file root"> / default
source =	<database list> / <"database path"> / none_but<database list> / all
system =	<"list of component names">

Defaults:

data_input_file =	"def.mpi"
output_data / results =	"def" (for example "def.mpr")
source =	first database from the database menu

No default system or experimental_file.

LIST

Function: The LIST command is used to display the elements, components, substances, unaries and phases in the system under study, their classifications (see CLASSIFY) and any initial or equilibrium settings (or RANGED values) established by the user. It also provides the basic tools for setting up an optimisation run in that it allows target experimental values and uncertainties to be associated with calculated properties or combinations of properties. Appropriate labels, weights for use during optimisation runs and abscissa settings for use during the plotting of results can be assigned to individual calculations or "experiments". In addition, the LIST command allows the differences between calculated and experimental results or the identities and values of FIXED, STEPPED, LINKed and OPTIMISED parameters (see MODIFY) to be listed to the screen.

Parameters:

abscissa =	temperature / user_variable(u) / x(c) / w(c)
label_of_experiment =	<"main label [: additional label]">
parameter =	name[(p)] / offset[(p)] / value[(p)]
property =	h / gibbs_energy[(c)] / temperature[(b)] / pressure[(s)] / volume / x(cp) / w(cp) / activity(c) / user_variable(u) / sum(v1:v2) / difference(v1:v2) / product(v1:v2) / ratio(v1:v2) / log10(v1) / ln(v1)

c = <component number>
 ce = <component name>, <component number>, *, all, or list
 p = <phase name> or <phase number>
 pe = <phase name>, <phase number>, *, all, or list
 se = <substance name>, <substance number>, *, all, or list

Defaults: delta(c) = determined by the program
 miscibility(p) = as flagged in the current data (.mpi) file
 normal = all components, phases and substances unless
 flagged as absent in the current data (.mpi) file
 reference[(c)] = default_ref_state
 (that used in the current data (.mpi) file)

MODIFY_PARAMETER

Function: The MODIFY command is used to select thermodynamic parameters from the current data (.mpi) file the values of which are to be FIXED, OPTIMISED, LINKed or STEPPED, to assign new absolute or relative (OFFSET) values to parameters, to guide the optimisation process by providing SCALES for any changes made to parameters (also used as step sizes during parameter stepping) and to LINK parameters together so that any changes made to one will be reflected in its partner(s).

Parameters: index = <number> / unchanged
 link = <number> / unchanged
 offset = <number> / unchanged
 scale = <number> / unchanged
 type = fixed / stepped / optimised / unchanged
 value = <number> / unchanged

Defaults: scale = determined by the program
 type = fixed
 value = parameter value from the current data (.mpi) file
 (with no offset)

RANGE

Function: The RANGE command is used to specify temperature, pressure or composition ranges for consideration in subsequent phase boundary searches along with temperature, pressure or composition tolerance values indicating the accuracy to which individual boundaries are to be found. Currently only TEMPERATURE can be RANGED.

Parameters: lgt_pressure = <lower limit>, <upper limit>, <tolerance>
 pressure = <lower limit>, <upper limit>, <tolerance>
 temperature = <lower limit>, <upper limit>, <tolerance>
 or
 <"start group">, <"final group">, <composition tolerance>

Defaults: No default temperature, pressure or composition ranges.

SET

Function: The SET command is most commonly used to fix the system temperature, system pressure or volume and initial amounts or masses of components, individually or in total, prior to a calculation. It can also be used to fix partial pressures, molalities, amounts or masses of individual species at equilibrium (see CLASSIFY FREE), to associate group names with system compositions or to assign values to named USER_VARIABLES.

Parameters:

lgt_molality(s) =	<log ₁₀ (equilibrium molality of s)> / undefined
lgt_pressure(s) =	<log ₁₀ (equilibrium partial pressure of s)> / undefined
molality(s) =	<equilibrium molality of s> / undefined
n[(cse)] =	<system amount> / <amount of c or s> / <equilibrium amount of s> / undefined
pressure[(s)] =	<system pressure> / undefined / <equilibrium partial pressure of s>
temperature =	<temperature> / user_variable(u) / undefined
volume =	<system volume> / undefined
w[(cs)] =	<system mass> / <mass of c> / <equilibrium mass of s> / undefined
x(s) =	<equilibrium mole fraction of s> / undefined
user_variable(u1) =	h / gibbs_energy[(c)] / temperature[(b)] / pressure[(sn)] / volume / x(cp) / w(cp) / activity(c) / user_variable(u2) / sum(v1:v2) / difference(v1:v2) / product(v1:v2) / ratio(v1:v2) / log10(v1) / ln(v1)
<"group name">	<amount / mass of component 1>, <amount / mass of component 2>, ... , <amount / mass of final component>

[] indicates an optional entry

b =	<phase boundary number>
c =	<component name>
cp =	<component name, phase name in angular brackets> or <component name>
cs =	<component number> to define component mass, or <substance formula with phase label> to define equilibrium substance mass
cse =	<component number> or <substance formula, no phase label> to define component amount, or <substance formula with phase label> to define equilibrium substance amount
s =	<substance number> or <substance formula with phase label>
sn =	<substance name>
u1, u2 =	<user variable name>
v1, v2 =	<number> or any item from the property menu up to and

Defaults: pressure = 101325 Pa
 temperature = undefined
 volume = undefined
 n = undefined (all components and system amount)
 w = undefined (all components and system mass)

STEP_PARAMETER

Function: The STEP_PARAMETER command is used to initiate calculations in which the values of one or two selected thermodynamic parameters are stepped automatically above and below their current values and a comparison is made between calculated and experimental properties from the DEFINED experimental data file at each stage. This is sometimes a useful prelude to a full optimisation run. The number of steps to be taken for each parameter must be specified as part of the complete STEP_PARAMETER command. The step sizes are based upon SCALES established using the MODIFY_PARAMETER command.

Parameters: number_of_steps = <number of steps>

Defaults: number_of_steps = undefined

SHOW

Function: The SHOW command is used to discover which index (number) should be used to refer to a particular thermodynamic parameter (see MODIFY). It lists to the screen the names, current values and indices of thermodynamic parameters from the current data input (.mpi). Parameters can be selected by type (unary or interaction) or by phase.

Parameters: all_data = <phase name> / all_phases
 interaction_data = <phase name> / all_phases
 unary_data = <phase name> / all_phases

Defaults: all_data = all_phases

PLOT

Function: The PLOT command is used to obtain diagrams comparing the values of properties calculated using the LIST PROPERTY command with experimental data entered using the LIST VALUE command. Two types of diagram can be plotted, bar charts of ERRORS (differences between calculated and experimental results divided by corresponding experimental uncertainties entered using the LIST UNCERTAINTY command) or lines joining calculated VALUES associated with the same abscissa variable (specified using the LIST ABSCISSA command) overlaid with symbols representing experimental data. The various LIST commands necessary to make PLOTS possible are usually provided in an experimental data file (see DEFINE).

Parameters: errors = experiment[(en)]
 values = experiment[(el)]

[] indicates an optional entry

en = <experiment number> or <list of experiment numbers>
 el = <label or part of label shared by experiments selected>

Defaults: All “experiments” (property and experimental data combinations) established using a LIST PROPERTY command are plotted, unless specific experiment numbers or labels, as appropriate, are given.

SAVE

Function: The SAVE command is used during the optimisation process to save the current values of selected FIXED, LINKed, STEPPED or OPTIMISED PARAMETERS to an in-memory store, to save the current values of all FIXED, LINKed, STEPPED or OPTIMISED parameters to a named DATA_LOADING_FILE file ready for loading into a database or to write a SET_UP macro to be used at a later date to recreate the exact state of the current optimisation in terms of the data input filename, experimental data file name and the status and value of individual parameters.

Parameters: data_loading_file = <“file name”>
 parameter = <parameter index> / all
 set_up = <“file name”>

Defaults: parameter = all

The default SAVED value for each parameter is the value taken from the current data input (.mpi) file.

COMPUTE

Function: This command is used to initiate a phase equilibrium calculation based upon the current settings (see SET) and classifications (see CLASSIFY). The required type of calculation (STAGES_REQUIRED and INITIAL_POINT_OPTION) can be specified. Note that this command does not initiate an optimisation run (see OPTIMISE).

Parameters: initial_point_option = automatic / previous_solution /
 lower_bounds / last_active_set
 stages_required = 1_only / stage_1_plus / robust_stage_2
 / 2_only / both_large / both_small

Defaults: initial_point_option = automatic
 stages_required = 1_only

OPTIMISE

Function: The OPTIMISE command is used to initiate an optimisation run in which thermodynamic parameters marked as OPTIMISED or LINKED are adjusted automatically to minimise the differences between calculated properties and experimental values (see LIST) as defined in an experimental data file (see DEFINE). Options allow different ALGORITHMs, and TOLERANCES to be selected for the optimisation process, the overall number of comparisons made between calculation and experiment (MAXIMUM_FUNCTIONS) to be limited and a method for monitoring progress (by means of TEXT or GRAPHICAL output) to be specified. In the case of GRAPHICAL monitoring a frequency for updating the output can be specified.

Parameters:

algorithm =	lcqndf / upolyt / lsqndn
maximum_functions =	<number>
output_type =	graphical / text
tolerance =	<number>
update_graphics =	<frequency of update>

Defaults:

algorithm =	lcqndf
maximum_functions =	100
output_type =	graphical
tolerance =	0.01
update_graphics =	1

LASER

Function: The LASER command is used on selected MTDATA platforms to request hard copies of those graphs plotted most recently but not previously sent to a printer. This command is not used in MTDATA for Windows 9x and NT.

Parameters: No parameters.

Defaults: Not applicable.

RESTORE

Function: The RESTORE command is used to overwrite the current values of selected FIXED, LINKed, STEPPED or OPTIMISED (see MODIFY) PARAMETERS with those previously SAVED in an in-memory store.

Parameters: parameter = <parameter index> / all

Defaults: parameter = all

The default SAVED value for each parameter is the value taken from the current data input (.mpi) file.

UNDO_OPTIMISATION

Function: The UNDO_OPTIMISATION command negates the effects of the last optimisation run performed before its issue. New values for thermodynamic parameters generated during an optimisation run are, by default, adopted as current when the run ends. The UNDO_OPTIMISATION command overwrites any new values by those current at the start of the last optimisation run.

Parameters: No parameters.

Defaults: Not applicable.

RETURN

Function: The RETURN command is used to leave the ASSESSMENT module ready to begin work in another module or to leave MTDATA altogether. The current optimisation SET_UP will be lost unless previously SAVED.

Parameters: No parameters

Defaults: Not applicable.

SWAP

Function: The SWAP command is used to leave the ASSESSMENT module and enter another module, such as BINARY or MULTIPHASE, retaining the current optimisation SET_UP (provided the DEFINE command is not used in the new module). This allows calculations to be performed in modules other than ASSESSMENT using the current set of OPTIMISED (but not necessarily SAVED) parameters. The command SWAP ASSESSMENT is used to return to the ASSESSMENT module.

Parameters: <the name of another MTDATA module>

Defaults: Not applicable.

END_OF_EXPERIMENTS

Function: The END_OF_EXPERIMENTS command is used to mark the end of the contents of an experimental file (see DEFINE). It should only be used in an experimental file and never entered interactively.

Parameters: No parameters.

Defaults: Not applicable.

<"MACRO NAME">

Function: A macro, or list of MTDATA commands stored in a file, can be run by simply specifying the name of the file in single or double quotes.

Operating system commands can be executed by prefixing them with "\$ or just \$.

Parameters: A list of parameters and values appropriate to the particular macro being run may be supplied on the same line of input after the quote(s) following the macro name.

Defaults: Not applicable.

6 QUICK REFERENCE SECTION

define	data_input_file =	<"file name">	!
		default	!
	experimental_file =	<"file name">	!
		default	!
	output_data / results =	<"file root">	!
		default	!
	source =	<database list>	!
		<"database path">	!
		none_but<database list>	!
		all	!
	system =	<"component list">	!
list	abscissa =	temperature	!
		user_variable(u)	!
		x(c)	!
		w(c)	!
	label_of_experiment = parameter =	<"experiment label">	!
		name[(p)]	!
		offset[(p)]	!
	property =	value[(p)]	!
		h	!
		gibbs_energy[(c)]	!
		temperature[(b)]	!
		pressure[(s)]	!
		volume	!
		x(cp)	!
		w(cp)	!
		activity(c)	!
		user_variable(u)	!
		sum(v1:v2)	!
		difference(v1:v2)	!
		product(v1:v2)	!
		ratio(v1:v2)	!
		log10(v1)	!
		ln(v1)	!
	result =	experiment[(e)]	!
	system =	elements	!
		components	!
		substances	!
		phases	!
		unaries	!
		settings	!
		all	!
	value =	<experimental value >	!
	uncertainty =	none	!
		<experimental uncertainty>	!
	weight =	none	!
		<weight for experiment>	!
		none	!

b =	<phase boundary number>	
c =	<component name>	
cp =	<component name, phase name in angular brackets> or <component name>	
e =	<experiment number > or <experiment number list>	
s =	<substance name>	
p =	<fixed, stepped or optimised parameter number>	
u =	<user variable name>	
v1, v2 =	<number> or any item from the property menu up to and including user_variable(u)	
[]	optional entry	
classify	absent =	component(ce) ! phase(pe) ! substance(se) ! <number> ! delta(c) = <increment> ! undefined ! free = component(ce) ! miscibility(p) = <number> ! none ! normal = component(ce) ! phase(pe) ! substance(se) ! <number> ! para = component(ce) ! reference[(c)] = <phase number> ! <phase name> ! default_ref_state !
c =	<component number>	
ce =	<component name>, <component number>, *, all, or list	
p =	<phase name> or <phase number>	
pe =	<phase name>, <phase number>, *, all, or list	
se =	<substance name>, <substance number>, *, all, or list	
[]	optional entry	
modify_parameter	index =	<number> ! unchanged ! link = <number> ! unchanged ! offset = <number> ! unchanged ! scale = <number> ! unchanged ! type = fixed ! stepped ! optimised ! unchanged ! value = <number> ! unchanged !

range	lgt_pressure =	<lower limit>, <upper limit>, <tolerance>	!
	pressure =	<lower limit>, <upper limit>, <tolerance>	!
	temperature =	<lower limit>, <upper limit>, <tolerance>	!
		or	
	<"start group">, <"final group">, <composition tolerance>		!

(Only range temperature is currently available)

set	lgt_molality(s) =	<log ₁₀ (equilibrium molality of s)>	!
		undefined	!
	lgt_pressure(s) =	<log ₁₀ (equilibrium partial pressure of s)>	!
		undefined	!
	molality(s) =	<equilibrium molality of s>	!
		undefined	!
	n[(cse)] =	<system amount>	!
		<amount of c or s>	!
		<equilibrium amount of s>	!
		undefined	!
	pressure[(s)] =	<system pressure>	!
		<equilibrium partial pressure of s>	!
		undefined	!
	temperature =	<temperature>	!
		user_variable(u)	!
		undefined	!
	volume =	<system volume>	!
		undefined	!
	w[(cs)] =	<system mass>	!
		<mass of c>	!
		<equilibrium mass of s>	!
		undefined	!
	x(s) =	<equilibrium mole fraction of s>	!
		undefined	!
	user_variable(u1) =	h	!
		gibbs_energy[(c)]	!
		temperature[(b)]	!
		pressure[(sn)]	!
		volume	!
		x(cp)	!
		w(cp)	!
		activity(c)	!
		user_variable(u2)	!
		sum(v1:v2)	!
		difference(v1:v2)	!
		product(v1:v2)	!
		ratio(v1:v2)	!
		log10(v1)	!
		ln(v1)	!
		<"group name">	
	<amount / mass of component 1>, <amount / mass of component 2>, ..., <amount / mass of final component>	!	

b =	<phase boundary number>		
c =	<component name>		
cp =	<component name, phase name in angular brackets> or <component name>		
cs =	<component number> to define component mass, or <substance formula with phase label> to define equilibrium substance mass		
cse =	<component number> or <substance formula, no phase label> to define component amount, or <substance formula with phase label> to define equilibrium substance amount		
s =	<substance number> or <substance formula with phase label>		
sn =	<substance name>		
u1, u2 =	<user variable name>		
v1, v2 =	<number> or any item from the property menu up to and including user_variable(u)		
[]	optional entry		
step_parameter	number_of_steps =	<number of steps>	!
show	all_data =	<phase name>	!
		all_phases	!
	interaction_data =	<phase name>	!
		all_phases	!
	unary_data =	<phase name>	!
		all_phases	!
plot	errors =	experiment[(en)]	!
	values =	experiment[(el)]	!
en =	<experiment number> or <list of experiment numbers>		
el =	<label or part of label shared by experiments selected>		
[]	optional entry		
save	data_loading_file =	<"file name">	!
	parameter =	<parameter index>	!
		all	!
	set_up =	<"file name">	!
compute	initial_point_option =	automatic	!
		previous_solution	!
		lower_bounds	!
		last_active_set	!
	stages_required =	1_only	!
		stage_1_plus	!
		robust_stage_2	!
		2_only	!
		both_large	!
		both_small	!

optimise	algorithm =	lcqndf	!
		upolyt	!
		lsqndn	!
	maximum_functions =	<number>	!
	output_type =	graphical	!
		text	!
	tolerance =	<number>	!
	update_graphics =	<frequency of update>	!
laser			
restore	parameter =	<parameter index>	!
		all	!
undo_optimisation			
return			
swap	<MTDATA module name>		
end_of_experiments			
<"macro name">			