popparsing Documentation

Release 0.1

Brandon Bocklund, Joel Veneracion, Saraubh Bajaj

Sep 18, 2018

Contents

The popparsing package is used to turn a file with POP commands into a JSON file that contains all information about each equilibrium set

CHAPTER 1

Installation and Usage

Prerequisites for the popparsing package include the **pyparsing** and **sympy** packages. To install, open up a terminal or command prompt and enter `pip install popparsing`. To generate a JSON file from a POP file, enter in the command prompt `popparsing [input_file.pop] [output_file.json]` where `input_file.pop` is a file with POP commands and extension .pop and `output_file.json` is the json file to write to.

CHAPTER 2

Input

The popparsing package supports POP files with the following commands below. The subsequent subsections are details of supported syntaxes of **only commands that affect the output of the pop file parser**.

Affects the output?	
YES	NO
CHANGE_STATUS PHASE	ADVANCED_OPTIONS
CREATE_NEW_EQUILIBRIUM	CHANGE_STATUS COMPONENT
ENTER_SYMBOL CONSTANT	COMMENT
EXPERIMENT	DEFINE_COMPONENTS
LABEL_DATA	ENTER_SYMBOL FUNCTION
SET_CONDITION	ENTER_SYMBOL TABLE
SET_REFERENCE_STATE	EVALUATE_FUNCTIONS
TABLE_END	EXPORT
TABLE_VALUES	EXTERNAL
	FLUSH_BUFFER
	IMPORT
	SAVE
	SAVE_WORKSPACE
	SAVE_WORKSPACES
	SET_ALL_START_VALUES
	SET_ALTERNATE_CONDITION
	SET_NUMERICAL_LIMITS
	SET_START_VALUE
	SET_WEIGHT
	TABLE_HEAD

2.1 CHANGE_STATUS PHASE

The CHANGE_STATUS PHASE command must have two arguments separated by an '='. The first argument to the left of the equal sign is a list of phase names separated by commas or spaces. The second argument is either just a status string or both a status string and a floating point value separated by a space. A supported abbreviation of the command is CH P. Status phase names FIXED, DORMANT, and ENTERED are supported and can be abbreviated to any shorter length. However, phase names abbreviations are currently not supported and must be typed exactly how it should appear on the equilibrium data set.

Valid Examples:

```
CHANGE_STATUS PHASE BCC HCP DEL=DORMANT
CHANGE_STATUS PHASE BCC HCP DEL=FIXED 1
CHANGE_STATUS PHASE BCC,HCP,DEL=FIXED 1
CHANGE_STATUS PHASE BCC,HCP,DEL=FIX 1
CHANGE_STATUS PHASE BCC, HCP, DEL = FIX 1
CHANGE_STATUS PHASE BCC HCP DEL = FIX 1
CH P BCC HCP DEL=F 1
```

2.2 CREATE_NEW_EQUILIBRIUM

The CREATE_NEW_EQUILIBRIUM command must have two arguments. The first argument is unused by the pop file parser. The second argument is an integer, which is an initialization code that tells the parser how to initialize the equilibrium set. A supported abbreviated version of the command is C-N.

Valid Examples:

```
CREATE_NEW_EQUILIBRIUM 108, 1
CREATE_NEW_EQUILIBRIUM 108,1
CREATE_NEW_EQUILIBRIUM 108 1
C-N 108 1
```

Note: Only initialization code 1 is supported at the moment.

2.3 ENTER_SYMBOL CONSTANT

The ENTER_SYMBOL CONSTANT command has one or multiple arguments separated by either space or commas. Each argument has the same syntax and assigns a constant value to a variable name, with the variable name to the left of the equal sign and the numerical value to the right. There must be no space between the variable name and the equal sign or between the equal sign and the numerical value for each argument. The supported abbreviation of the command is E-SYM CON.

Valid Examples:

```
ENTER_SYMBOL CONSTANT P0=1E5 R0=8.314
ENTER_SYMBOL CONSTANT P0=1E5,R0=8.314
E-SYM CON P0=1E5 R0=8.314
```

2.4 EXPERIMENT

The EXPERIMENT command has one or more arguments separated by spaces or commas. The arguments only have one specific syntax with four parts in a specific order. The first part is the variable name, and the second part is either an '=', '<', or '>'. The third part is either a numerical value or an '@' followed by an integer indicating the column number of the table created by the TABLE_VALUES command the property or variable name corresponds to. (See *TABLE_VALUES and TABLE_END* for more details) Finally, a ':' and a numerical value representing the error is placed after. The error value can optionally be followed by a % sign. A supported abbreviation of the command is EXP.

Valid Examples:

```
$ Note: '@2' indicates that ACR(MG)
$ corresponds to a list of values in the second column of
$ the created table.
EXPERIMENT ACR(MG)=@2:5% X(HCP_A3,ZR)=0.988:1E-2
EXPERIMENT ACR(MG)=@2:5%,X(HCP_A3,ZR)=0.988:1E-2
EXPERIMENT T=1370:2
EXPERIMENT X(LIQ,NI)=0.803:5%
```

Note: The error value after the colon representing the is is currently not included in the output, but both the colon and error value are necessary parts of the syntax.

2.5 LABEL_DATA

The LABEL_DATA command has one argument which is just a string with alphanumeric characters. It is common to use the abbreviated version, LABEL.

Examples:

```
LABEL_DATA AHMR1
LABEL ACP
```

2.6 SET_CONDITION

The SET_CONDITION command has one or more arguments separated by commas or spaces. Each argument has the same specific syntax: a variable or property name, followed by an '=' sign, and then either a numerical value or an '@' followed by an integer indicating the table column number the property or variable corresponds to. (See *TABLE_VALUES and TABLE_END* for details) The supported abbreviation for the command is S-C.

```
$ Note: '@1' indicates that X(NI)
$ corresponds to a list of values in the first column of
$ the created table.
SET_CONDITION T=1073 P=P0 X(NI)=@1
SET_CONDITION T=1073,P=P0,X(NI)=@1
SET_CONDITION T=1073, P=P0, X(NI)=@1
$ Temperature variable corresponds to a list of values
$ on the second colun of the created table
S-C T=@2
```

2.7 SET_REFERENCE_STATE

The SET_REFERENCE_STATE command must have at least two arguments: the component name and the phase name respectively separated by either comma or space. The pop file parser does support additional arguments after the first two, but currently, those arguments will not affect the output data. The supported abbreviation of the command is S-R-S.

Valid Examples:

```
SET_REFERENCE_STATE U BCC_A2,,,,,,
SET_REFERENCE_STATE U,BCC_A2,,,,,,
SET_REFERENCE_STATE U BCC_A2
SET_REFERENCE_STATE U, BCC_A2
SET_REFERENCE_STATE U BCC_A2 * 100000
S-R-S U BCC_A2
```

2.8 TABLE_VALUES and TABLE_END

The TABLE_VALUES command marks the start of the table while the TABLE_END command marks the end of it. Each line between the two commands must contain the same number of numerical values, or in other words, each column in the constructed table must have the same number of elements. Additionally, these entered values are separated by spaces. The number of spaces before and after each numerical value is irrelevant. These values will be used by variables that are assigned with an '@' followed by an integer indicating which column of values in the table will be used.

Example:

```
$The values on the first column correspond to X(LIQUID,Ti)
$The values on the second column correspond to temperature(T).
SET_CONDITION T=@2
EXPERIMENT X(LIQUID,Ti)=@1:0.01
TABLE_VALUES
0.00          1406
0.0045        1420
0.01          1445
0.02          1446
0.03          1495
0.04          1563
0.05          1643
0.10          1957
0.15          2093
0.20          2117
0.30          2198
TABLE_END
```

CHAPTER 3

Output

3.1 Main Equilibrium Dictionary

The output of the popparsing command is a JSON file containing a list of dictionaries (one per equilibrium data set) with each dictionary containing the following keys and values. Each dictionary is created by a `CREATE_NEW_EQUILIBRIUM` command, and the command lines between that particular line and either the next line calling the `CREATE_NEW_EQUILIBRIUM` command or the end of the file determines the contents of that dictionary.

Key	Value
phases	A dictionary of phases and their statuses and values
components	A list of elements involved in the equilibrium set
conditions	A dictionary of conditions and their values (a single number or a list of values);
	Also contains the 'reference_states' key which refers to a dictionary of components as keys and phases as values
outputs	A list of names of data that correspond to the list the 'values' key point to
values	A list of data (either a single number or a list)
reference	The argument from the `LABEL` command

Example:

Input:

```
CREATE_NEW_EQUILIBRIUM @@,1
CHANGE_STATUS PHASE LIQUID=DORMANT
CREATE_NEW_EQUILIBRIUM @@,1
```

Output:

```
[
    #First C-N command creates a dictionary.
```

(continues on next page)

(continued from previous page)

```
    #CHANGE_STATUS PHASE command inserts a 'phases' key
    #in the first dictionary.
    {
        'phases' : {
            'LIQUID' : {
                'status' : 'DORMANT'
            }
        }
    },
    #Second C-N command creates an empty dictionary.
    {
    }
]
```

3.2 Phases Sub-dictionary

The phases dictionary contains phase names as keys and a sub-dictionary of statuses and/or values as values.

Dictionary for each phase	
Key	Value
status	One of the following strings: `'FIXED'`, `'ENTERED'`, `'DORMANT'`, `'SUSPEND'`
value	An specified numerical value for the phase (only if status is `FIXED` or `ENTERED`)

Example:

Input:

```
CREATE_NEW_EQUILIBRIUM @@,1
CHANGE_STATUS PHASE LIQUID=DORMANT
CHANGE_STATUS PHASE SPINEL=FIXED 1
```

Output:

```
[
    {
        'phases' : {
            'LIQUID' : {
                'status' : 'DORMANT'
            }
            'SPINEL' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        }
    }
]
```

3.3 Conditions Sub-dictionary

The content of the dictionary that the 'conditions' key refers to depends on the arguments of one or more `SET_CONDITION` commands. It will contain condition names as keys, and each value will either be a floating

point number or a list of floating point numbers. In addition to the condition names, the 'conditions' dictionary will always contain a 'reference_states' key which stores another dictionary. The contents of the 'reference_states' dictionary is determined by the arguments of each SET_REFERENCE_STATE command. It will have component names as keys and phase names as values. If the SET_REFERENCE_STATE command is not used for the equilibrium set, then the dictionary will be empty.

Example:

Input:

```
CREATE_NEW_EQUILIBRIUM @@,1
SET_CONDITION T=@1 P=10000
TABLE_VALUES
100
200
300
TABLE_END
```

Output:

```
[
    {
        'conditions' : {
            'T' : [ 100.0, 200.0, 300.0 ],
            'P' : 100000
        }
    }
]
```

3.4 Components List

The 'components' list consist of string elements of all components found in the POP commands for the created equilibrium set. The pop file parser finds these elements in the first arguments of SET_REFERENCE_STATE commands and property names such as X(NI) and X(LIQ,NI) in the SET_CONDITION and EXPERIMENT commands.

Example: The following input would yield ['MG', 'NI'] as a list of components because of X(NI) and ACR(MG).

Input:

```
CREATE_NEW_EQUILIBRIUM @@,1
SET_CONDITION X(NI)=0.1
EXPERIMENT ACR(MG)=@1:5
TABLE_VALUES
0.1
0.2
0.3
TABLE_END
```

3.5 Outputs and Values Lists

The 'outputs' and 'values' keys refer to two lists of the same size and are created by parsing the arguments of EXPERIMENT commands. Each n-th element in the output list corresponds to the n-th element of the values list.

The outputs list will simply contains string type elements that represent the name of the measured outputs. Each element in the values list will be one of three data types: floating point number, dictionary, or list.

Floating point values are used for equalities while dictionaries are used for inequalities entered. Specifically, if the element is a dictionary, it will have two keys: 'equality' and 'value'. The 'equality' key will determine whether the inequality is '<' or '>' while the 'value' key will store the numerical value. Finally, lists are used for variables that correspond to a list of numerical values in a specific column of a table created by the `TABLE_VALUES` and `TABLE_END` commands.

Example:

Input:

```
CREATE_NEW_EQUILIBRIUM @@,1
EXPERIMENT X(LIQUID,Ti)=@1:0.01
EXPERIMENT LPFCC=4.02:5% DGMR(DEL)<0:0.001
TABLE_VALUES
0.01
0.02
0.03
TABLE_END
```

Output:

```
[
    {
        'outputs' : [
            'X(LIQUID,Ti)',
            'LPFCC',
            'DGMR(DEL)'
        ]
        'values' : [
            [0.01, 0.02, 0.03],
            4.02,
            {
                'equality' : '<',
                'value' : 0.0
            }
        ]
    }
]
```

CHAPTER 4

Full Example

The following input file will generate the following output file.

Input:

```
$=================================================
$  BCC FORMATION ENTHALPIES X(TI)=0.1 TO 0.6
$=================================================
TABLE_HEAD 540
CREATE_NEW_EQUILIBRIUM @@,1
CHANGE_STATUS PHASE *=SUS
CHANGE_STATUS PHASE BCC_A2=FIXED 1
SET_REFERENCE_STATE U BCC_A2,,,,,,
SET_REFERENCE_STATE Ti BCC_A2,,,,,,
SET_CONDITION  P=102325 T=200
SET_CONDITION X(Ti)=@1
LABEL AHMR1
EXPERIMENT HMR(BCC_A2)=@2:500
TABLE_VALUES
$ X(Ti)         HMR
0.10           2450
0.20           3000
0.30           2990
0.40           2430
0.50           1400
0.60            -65
TABLE_END
$ -----------
```

Output:

```
[
    {
        'phases': {
            'BCC_A2' : {
                'status' : 'FIXED',
```

(continues on next page)

(continued from previous page)

```
                'value' : 1.0
            }
        }
        'components': [
            'Ti',
            'U'
        ]
        'conditions': {
            'P': 102325.0
            'T': 200.0
            'X(Ti)': [
                0.1,
                0.2,
                0.3,
                0.4,
                0.5,
                0.6
            ]
            'reference_states': {
                'U': 'BCC_A2',
                'Ti': 'BCC_A2'
            }
        }
        'outputs': [ 'HMR(BCC_A2)' ]
        'values': [
            [
                2450.0,
                3000.0,
                2990.0,
                2430.0,
                1400.0,
                -65.0
            ]
        ]
    }
]
```

For another example, see the *MgNi POP File Example*.

4.1 API Documentation

4.1.1 popparsing package

Subpackages

popparsing.tests package

Submodules

popparsing.tests.test_pop_units module

popparsing.tests.testing_data module

Module contents

Submodules

popparsing.cli module

popparsing.conversion module

popparsing.formatting module

popparsing.parsing module

popparsing.pop_keywords module

Module contents

4.2 MgNi POP File Example

4.2.1 Input: MgNi.POP

```
E-SYM CON P0=1E5 R0=8.314

@@ THERMOCHEMICAL DATA IN LIQUID
@@ 100-300

@@ Data Set 1
TABLE 100
C-N @@,1
CH P LIQ=F 1
S-R-S MG LIQ,,,,,
S-C T=1073, P=P0, X(NI)=@1
LABEL ALA1
EXPERIMENT ACR(MG)=@2:5%
TABLE_VALUE
 0.20230        0.771
 0.22600        0.722
```

(continues on next page)

(continued from previous page)

```
 0.25400        0.672
 0.28390        0.623
 0.18470        0.746
 0.20840        0.710
 0.23580        0.667
 0.25950        0.626
 0.28900        0.570
 0.21590        0.709
 0.24690        0.656
 0.28040        0.600
 0.19450        0.740
 0.22560        0.681
 0.25880        0.635
 0.28950        0.577
 0.32410        0.519
 0.15000        0.822
 0.17210        0.787
 0.19280        0.748
 0.21300        0.708
 0.23170        0.678
 0.25270        0.640
 0.27040        0.607
 0.29050        0.577
 0.31020        0.550
 0.34380        0.506
TABLE_END

@@ Data Set 2
EN-SYM FUN HMRT=HMR/1000;
TABLE 200
C-N @@,1
CH P LIQ=F 1
S-R-S MG LIQ,,,,,
S-R-S NI LIQ,,,,,
S-C T=1005, P=P0, X(NI)=@1
EXPERIMENT HMRT=@2:10%
LABEL ALA2
TABLE_VALUE
 0.04800       -2.880
 0.10200       -5.780
 0.15300       -8.010
 0.05300       -3.080
 0.09900       -5.490
 0.14600       -7.330
 0.19400       -9.170
 0.02700       -1.620
 0.05600       -3.200
 0.08500       -4.620
 0.05100       -3.220
 0.10500       -5.820
 0.15700       -7.940
TABLE_END

@@ Data Set 3
ent fun ph1=(hmr+hmr.x(ni)-x(ni)*hmr.x(ni))/1000;
TABLE 300
C-N @@,1
```

(continues on next page)

(continued from previous page)

```
CH P LIQ=F 1
S-R-S MG LIQ,,,,,
S-R-S NI LIQ,,,,,
S-C T=1005, P=P0, X(NI)=@1
EXPERIMENT PH1=@2:10%
LABEL ALA3
TABLE_VALUE
 0.02400      -60.360
 0.07500      -53.410
 0.12800      -44.960
 0.02700      -57.860
 0.07600      -52.630
 0.12200      -41.420
 0.17000      -39.540
 0.01300      -61.030
 0.04100      -54.390
 0.07000      -48.550
 0.02500      -63.440
 0.07800      -48.580
 0.13100      -42.170
TABLE_END

@@ WE NOW DEAL WITH 2 PHASE EQUILIBRIA
@@ LIQ-FCC, LIQ-HCP_A3
@@ REFERENCE
@@ 1000-

@@ Data Set 4
TABLE 1000
C-N @@,1
CH P LIQ HCP_A3=F 1
S-C T=@1, P=P0
EXPERIMENT X(LIQ,NI)=@2:5%
EXPERIMENT X(HCP_A3,NI)<0.01:1E-2
S-S-V X(HCP_A3,NI)=1E-3
LABEL ALHC
TABLE_VALUE
900.7 .0235
869.4 .052
836.8 .0741
812.1 .0938
781.0 .1129
TABLE_END

@@ Data Set 5
TABLE 1100
C-N @@,1
CH P LIQ FCC=F 1
S-C T=@1, P=P0
EXPERIMENT X(LIQ,NI)=@2:5%
EXPERIMENT X(FCC,NI)>0.98:1E-2
S-S-V X(FCC,NI)=0.9999
LABEL ALFC
TABLE_VALUE
1428 .8265
1545 .8872
1708 .9762
```

(continues on next page)

(continued from previous page)

```
TABLE_END


@@ Data Set 6
@@NOW DEAL WITH THE EUTECTIC POINT ON THE NI RICH END
C-N 2,1
CH P LIQ,MGNI2,FCC=F 1
S-C P=P0
EXPERIMENT T=1370:2
EXPERIMENT X(LIQ,NI)=0.803:5%
LABEL AIEU



@@ Data Set 7
@@THIS THEN DEALS WITH THE TWO PHASE EQUILIBRIA IN L+MGNI2
TABLE 2000
C-N @@,1
CH P LIQ MGNI2=F 1
S-C X(LIQ,NI)=@2, P=P0
EXPERIMENT T=@1:5
LABEL ALM2
TABLE_VALUE
1054.4 .3004
1140.4 .3298
1163.9 .3388
1345 .3832
1385 .4347
1412 .4914
1418 .554
1417 .6236
1418 .6536
1413 .7012
1370 .7349
TABLE_END

@@ Data Set 8
@@ THIS DEALS WITH THE PERITECTIC MG2NI REACTION
C-N 10,1
CH P LIQ,MGNI2,MG2NI=F 1
S-C P=P0
EXPERIMENT T=1033:2
EXPERIMENT X(LIQ,NI)=0.29:5%
LABEL APER

@@ Data Set 9
@@THIS THEN TAKES CARE OF THE EUTECTIC ON THE MG RICH END
C-N 11,1
CH P LIQ,HCP_A3,MG2NI=F 1
S-C P=P0
EXPERIMENT T=779:2
EXPERIMENT X(LIQ,NI)=0.113:5%
LABEL AEMG

@@ Data Set 10
@@THE FOLLOWING TABLE TAKES CARE OF THE LIQUID MG2NI TWO PHASE
@@EQUILIBIA
```

(continues on next page)

(continued from previous page)

```
TABLE 3000
C-N @@,1
CH P LIQ MG2NI=F 1
S-C X(LIQ,NI)=@2, P=P0
EXPERIMENT T=@1:5
LABEL AM2N
TABLE_VALUE
834.2 .1236
879.9 .1393
917.6 .1563
960.6 .1836
994.5 .2192
1012.7 .2395
1023.2 .2662
TABLE_END

@@ Data Set 11
@@ STABILITY EQUILIBRIA RESTRICTIONS
TABLE 4000
C-N @@,1
CH P FCC MGNI2=F 1
CH P MG2NI=D
S-C T=@1, P=P0
EXPERIMENT DGM(MG2NI)<0:1E-2
LABEL AST1
TABLE_VALUE
1300
1200
1100
1000
900
800
700
600
500
400
300
200
TABLE_END

@@ Data Set 12
TABLE 5000
C-N @@,1
CH P HCP_A3 MG2NI=F 1
CH P MGNI2=D
S-C T=@1, P=P0
EXPERIMENT DGM(MGNI2)<0:1E-2
LABEL AST2
TABLE_VALUE
700
600
500
400
300
200
TABLE_END
```

(continues on next page)

(continued from previous page)

```
@@ Data Set 13
E-SY FUNCTION GLDD=MU(NI).X(NI);
TABLE 6000
C-N @@,1
CH P LIQ=F 1
S-C T=2500, P=P0, X(NI)=@1
EXPERIMENT GLDD>0:1E-2
LABEL ALDD
TABLE_VALUE
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
TABLE_END

SAVE
```

4.2.2 Output

```
[
    # Data set 1
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['NI', 'MG'],
        'conditions' : {
            'T' : 1073,
            'P' : 100000,
            'X(NI)' : [
                0.20230, 0.22600, 0.25400,
                0.28390, 0.18470, 0.20840,
                0.23580, 0.25950, 0.28900,
                0.21590, 0.24690, 0.28040,
                0.19450, 0.22560, 0.25880,
                0.28950, 0.32410, 0.15000,
                0.17210, 0.19280, 0.21300,
                0.23170, 0.25270, 0.27040,
                0.29050, 0.31020, 0.34380
            ],
            'reference_states' : {
                'MG' : 'LIQ'
            }
        },
        'outputs' : ['ACR(MG)'],
        'values' : [[
            0.771, 0.722, 0.672,
```

(continues on next page)

(continued from previous page)

```
            0.623, 0.746, 0.710,
            0.667, 0.626, 0.570,
            0.709, 0.656, 0.600,
            0.740, 0.681, 0.635,
            0.577, 0.519, 0.822,
            0.787, 0.748, 0.708,
            0.678, 0.640, 0.607,
            0.577, 0.550, 0.506
        ]],
        'reference' : 'ALA1'
    },
    # Data set 2
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['MG', 'NI'],
        'conditions' : {
            'T' : 1005,
            'P' : 100000,
            'X(NI)' : [
                0.04800, 0.10200, 0.15300,
                0.05300, 0.09900, 0.14600,
                0.19400, 0.02700, 0.05600,
                0.08500, 0.05100, 0.10500,
                0.15700
            ],
            'reference_states' : {
                'MG' : 'LIQ',
                'NI' : 'LIQ'
            }
        },
        'outputs' : ['HMRT'],
        'values' : [[
            -2.880, -5.780, -8.010,
            -3.080, -5.490, -7.330,
            -9.170, -1.620, -3.200,
            -4.620, -3.220, -5.820,
            -7.940
        ]],
        'reference' : 'ALA2'
    },
    # Data set 3
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['MG', 'NI'],
        'conditions' : {
            'T' : 1005,
            'P' : 100000,
```

(continues on next page)

(continued from previous page)

```
            'X(NI)' : [
                0.02400, 0.07500, 0.12800,
                0.02700, 0.07600, 0.12200,
                0.17000, 0.01300, 0.04100,
                0.07000, 0.02500, 0.07800,
                0.13100
            ],
            'reference_states' : {
                'MG' : 'LIQ',
                'NI' : 'LIQ'
            }
        },
        'outputs' : ['PH1'],
        'values' : [[
            -60.360, -53.410, -44.960,
            -57.860, -52.630, -41.420,
            -39.540, -61.030, -54.390,
            -48.550, -63.440, -48.580,
            -42.170
        ]],
        'reference' : 'ALA3'
    },
    # Data set 4
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'HCP_A3' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['NI'],
        'conditions' : {
            'T' : [ 900.7, 869.4, 836.8, 812.1, 781.0 ],
            'P' : 100000,
            'reference_states' : {
            }
        },
        'outputs' : ['X(LIQ,NI)', 'X(HCP_A3,NI)'],
        'values' : [[ 0.0235, 0.052, 0.0741, 0.0938, 0.1129 ],
                    { 'equality' : '<', 'value' : 0.01 }],
        'reference' : 'ALHC'
    },
    # Data set 5
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'FCC' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
```

(continues on next page)

(continued from previous page)

```
        },
        'components' : ['NI'],
        'conditions' : {
            'T' : [ 1428, 1545, 1708 ],
            'P' : 100000,
            'reference_states' : {
            }
        },
        'outputs' : ['X(LIQ,NI)', 'X(FCC,NI)'],
        'values' : [[ 0.8265, 0.8872, 0.9762 ],
                    { 'equality' : '>', 'value' : 0.98 }],
        'reference' : 'ALFC'
    },
    # Data set 6
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MGNI2' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'FCC' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['NI'],
        'conditions' : {
            'P' : 100000,
            'reference_states' : {
            }
        },
        'outputs' : [ 'T', 'X(LIQ,NI)' ],
        'values' : [ 1370, 0.803 ],
        'reference' : 'AIEU'
    },
    # Data set 7
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MGNI2' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['NI'],
        'conditions' : {
            'X(LIQ,NI)' : [ 0.3004, 0.3298, 0.3388, 0.3832,
                            0.4347, 0.4914, 0.5540, 0.6236,
                            0.6536, 0.7012, 0.7349 ],
            'P' : 100000,
```

(continues on next page)

(continued from previous page)

```
            'reference_states' : {
            }
        },
        'outputs' : ['T'],
        'values' : [[ 1054.4, 1140.4, 1163.9, 1345, 1385,
                      1412, 1418, 1417, 1418, 1413, 1370 ]],
        'reference' : 'ALM2'
    },
    # Data set 8
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MGNI2' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MG2NI' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['NI'],
        'conditions' : {
            'P' : 100000,
            'reference_states' : {
            }
        },
        'outputs' : [ 'T', 'X(LIQ,NI)' ],
        'values' : [ 1033, 0.29 ],
        'reference' : 'APER'
    },
    # Data set 9
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'HCP_A3' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MG2NI' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['NI'],
        'conditions' : {
            'P' : 100000,
            'reference_states' : {
            }
        },
        'outputs' : [ 'T', 'X(LIQ,NI)' ],
```

(continues on next page)

(continued from previous page)

```
        'values' : [ 779, 0.113 ],
        'reference' : 'AEMG'
    },
    # Data set 10
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MG2NI' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['NI'],
        'conditions' : {
            'X(LIQ,NI)' : [ 0.1236, 0.1393, 0.1563,
                            0.1836, 0.2192, 0.2395,
                            0.2662 ],
            'P' : 100000,
            'reference_states' : {
            }
        },
        'outputs' : ['T'],
        'values' : [[ 834.2, 879.9, 917.6, 960.6,
                      994.5, 1012.7, 1023.2 ]],
        'reference' : 'AM2N'
    },
    # Data set 11
    {
        'phases' : {
            'FCC' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MGNI2' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MG2NI' : {
                'status' : 'DORMANT'
            }
        },
        'components' : [],
        'conditions' : {
            'T' : list(range(1300, 100, -100)),
            'P' : 100000,
            'reference_states' : {
            }
        },
        'outputs' : ['DGM(MG2NI)'],
        'values' : [ { 'equality' : '<', 'value' : 0 } ],
        'reference' : 'AST1'
    },
    # Data set 12
    {
```

(continues on next page)

(continued from previous page)

```
        'phases' : {
            'HCP_A3' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MG2NI' : {
                'status' : 'FIXED',
                'value' : 1.0
            },
            'MGNI2' : {
                'status' : 'DORMANT'
            }
        },
        'components' : [],
        'conditions' : {
            'T' : list(range(700, 100, -100)),
            'P' : 100000,
            'reference_states' : {
            }
        },
        'outputs' : ['DGM(MGNI2)'],
        'values' : [{ 'equality' : '<', 'value' : 0 }],
        'reference' : 'AST2'
    },
    # Data set 13
    {
        'phases' : {
            'LIQ' : {
                'status' : 'FIXED',
                'value' : 1.0
            }
        },
        'components' : ['NI'],
        'conditions' : {
            'T' : 2500,
            'P' : 100000,
            'X(NI)' : [ (x+1) / 10 for x in range(9) ],
            'reference_states' : {
            }
        },
        'outputs' : ['GLDD'],
        'values' : [ { 'equality' : '>', 'value' : 0 } ],
        'reference' : 'ALDD'
    }
]
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search