
auxi User Manual

Release 0.2.0

0.2.0

April 25, 2016

1	Introduction	1
2	Getting Started	3
2.1	auxi Installation	3
2.2	Importing auxi Components	3
2.3	Getting Help	4
3	Structure	5
3.1	Modelling Frameworks	5
3.2	Tools	40
4	auxi Reference	53
4.1	auxi package	53
5	Indices and tables	57

Introduction

auxi is a toolkit to help metallurgical process engineers with their day-to-day tasks. Many of the calculations that we do require things like molar masses, conversion of one compound to another using stoichiometry, heat transfer calculations, mass balances, energy balances, etc. It is usually quite time consuming to get started with these calculations in a tool like Excel. auxi aims to save you time by making many of these calculations available from within python.

We hope that auxi will help you spend less time focusing on searching for formulas and data, and setting up calculations, and more on thinking about the problems that you need to solve with these calculations. Enjoy!

For video tutorials on using auxi visit [auxi youtube](#) .

Getting Started

2.1 auxi Installation

auxi runs on both Linux and Windows.

2.1.1 Installation

To install auxi:

```
* On Linux: sudo pip install auxi
* On Windows: pip install auxi
```

To uninstall auxi:

```
* On Linux: sudo pip uninstall auxi
* On Windows: pip uninstall auxi
```

2.2 Importing auxi Components

If you want to use auxi in one of your python modules, you need to import its components in the same way that you do for any other python package. For example, to use the stoichiometry tool, you will have to do the following:

```
from auxi.tools.chemistry import stoichiometry
```

The same method is used for all modules, functions and classes in auxi. Here are a few more import examples:

```
from auxi.tools.chemistry.stoichiometry import molar_mass
from auxi.tools.chemistry.stoichiometry import molar_mass as mm
from auxi.tools.chemistry.stoichiometry import convert_compound
from auxi.tools.chemistry.stoichiometry import convert_compound as cc
```

```
from auxi.tools.chemistry import thermochemistry
from auxi.tools.chemistry.thermochemistry import Compound
```

2.3 Getting Help

You can use Python's standard help function on any of auxi's components. For example:

```
import auxi
help(auxi)

from auxi.tools.chemistry import stoichiometry

help(stoichiometry)
help(stoichiometry.molar_mass)
help(stoichiometry.convert_compound)
```

All the help information that you are able to access in this way are also available through auxi's HTML documentation that is included in the auxi Python package distribution.

Structure

auxi is a python package that packages modules containing functions and classes that the engineers will ultimately use to help with their day-to-day tasks. auxi consists of python packages that are divided into two packages, the modelling framework package and the tools package.

3.1 Modelling Frameworks

This package contains modules, functions and classes for developing different types of computational models.

3.1.1 Process Modelling

The auxi.modelling.process package contains modules, functions and classes for performing process modelling. Specifically, the package contains a module called “materials” for modelling materials.

Material Modelling

The auxi.modelling.process sub-package provides you with modules, classes and functions to model process. The process modelling package contains material models.

Material Modelling

The auxi.modelling.process.materials package contains modules, functions and classes that help perform material-related calculations and build process models. Specifically, the package contains a module called “chemistry” for a material and material package that can perform chemistry calculations, another called “psd” for a particulate material and material package that can do size distribution as well as a material and material package that can, in addition, perform slurry calculations, and another called “thermochemistry” that contains a material consisting of multiple chemical compounds, having the ability to do thermochemical calculations.

Material Models

The `auxi.modelling.process.materials` sub-package provides you with modules, classes and functions to do material modelling. The material models are divided into chemical, psd and thermochemistry.

Concepts

Motivation The material, material assay and material package concepts used in `auxi.modelling.process.materials` may initially seem somewhat foreign to new users. These concepts were developed to assist process engineers when doing metallurgical calculations, and while developing process models. It aims to reduce the complexity and time involved in performing these important but sometimes tedious tasks. Once these concepts have been mastered, they become incredibly powerful in the hands of a metallurgical process engineer.

Materials, Material Assays and Material Packages `auxi.modelling.process.materials` includes a number of different representations of materials, material assays and material packages, each of which is contained in a separate Python module. The different modules cater for different situations as follows:

- `auxi.modelling.process.materials.psd` describes materials using particle size distributions. It can be used for processes in which particle size is the most important material property, such as a comminution circuit.
- `auxi.modelling.process.materials.slurry` adds water to `psd`. It can describe the solid and liquid portion of a particulate process such as a comminution circuit.
- `auxi.modelling.process.materials.chem` can be used for doing mass balances in chemically reactive processes such as leaching, precipitation, direct reduction and smelting. Its material class describes a material using its chemical composition. This module cannot perform any energy balance calculations.
- `auxi.modelling.process.materials.thermo` adds thermochemistry to `chem`. It can be used to do mass and energy balances in chemically reactive system such as smelting furnaces, direct reduction kilns, etc.

The `auxi.modelling.process.materials.thermo` module will be used to illustrate the concepts here.

Material A Material class is used to represent a “type of material”. Examples are ilmenite, iron ore, coal, ferrochrome alloy, etc. These terms are fairly abstract and generic, since they don’t refer to something specific. The `thermo` module’s Material class uses a list of specific phases of chemical compounds to describe a “type of material”. Here are some examples:

Material		Material	
Name		Name	
Ilmenite		Coal	
Compound		Compound	
Al2O3[S1]		C[S1]	
CaO[S]		H2[G]	
Cr2O3[S]		O2[G]	
Fe2O3[S1]		N2[G]	
Fe3O4[S1]		S[S1]	
FeO[S]		Al2O3[S1]	
K2O[S]		CaO[S]	
MgO[S]		Fe2O3[S1]	
MnO[S]		MgO[S]	
Na2O[S1]		SiO2[S1]	
P4O10[S]			
SiO2[S1]			
TiO2[S1]			
V2O5[S]			

With the Ilmenite material we are specifying that, in our model or calculation, ilmenites will consist of the 14 compounds included in the first list. In the case of Coal, different coals will consist of the 10 compounds listed in the second list.

Material Assay When we need to develop a model or do some calculations, it is not sufficient to simply know that a “type of material”, such as ilmenite or coal, can consist of a specified list of compound phases. We need to know what the composition of a “specific material” is. With this composition we will be able to get started on some calculations. This is where material assays come in. In the next example, assays were added to the two materials defined above:

Material			
Name			
Ilmenite			
Composition Details (mass fractions)			
Compound	IlmeniteA	IlmeniteB	IlmeniteC
Al2O3[S1]	1.16000000e-02	1.55000000e-02	9.41000000e-03
CaO[S]	2.20000000e-04	1.00000000e-05	1.70000000e-04
Cr2O3[S]	8.00000000e-05	2.20000000e-04	1.10000000e-04
Fe2O3[S1]	2.02000000e-01	4.73000000e-01	4.96740000e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	2.79000000e-01	1.91000000e-01	0.00000000e+00
K2O[S]	4.00000000e-05	1.00000000e-05	5.00000000e-05
MgO[S]	1.04000000e-02	5.80000000e-03	1.09000000e-02
MnO[S]	5.40000000e-03	4.80000000e-03	5.25000000e-03

Na2O[S1]	7.000000000e-05	5.000000000e-05	3.100000000e-04
P4O10[S]	1.000000000e-05	3.200000000e-04	1.500000000e-04
SiO2[S1]	8.500000000e-03	4.900000000e-03	1.744000000e-02
TiO2[S1]	4.770000000e-01	2.940000000e-01	4.594900000e-01
V2O5[S]	3.600000000e-03	8.000000000e-03	0.000000000e+00
=====			
=====			
Material			
=====			
Name	Coal		

Composition Details (mass fractions)			
Compound	ReductantA	ReductantB	

C[S1]	8.40973866e-01	1.000000000e+00	
H2[G]	1.37955186e-02	0.000000000e+00	
O2[G]	4.94339606e-02	0.000000000e+00	
N2[G]	6.09802120e-03	0.000000000e+00	
S[S1]	2.04933390e-03	0.000000000e+00	
Al2O3[S1]	1.20884160e-03	0.000000000e+00	
CaO[S]	2.94179980e-03	0.000000000e+00	
Fe2O3[S1]	7.85955656e-02	0.000000000e+00	
MgO[S]	1.41179360e-03	0.000000000e+00	
SiO2[S1]	3.49129950e-03	0.000000000e+00	
=====			

Our Ilmenite material now has three assays associated with it, and they are named IlmeniteA, IlmeniteB and IlmeniteC. Ilmenite therefore refers to a “type of material”, and IlmeniteA, IlmeniteB and IlmeniteC refer to “specific materials”.

Two assays were added to our Coal material. The first, ReductantA, refers to a coal with 84 % carbon and roughly 8.5 % ash. Reductant B is pure graphite.

Material Packages Using `auxi.modelling.process` we can now create a certain quantity of a “specific material” that is identified by a material and material assay. When we do this with the `thermo Material` class, we also have to specify pressure and temperature. The result of creating 1000 kg of IlmeniteB at 1 atm pressure and 500 °C temperature is the following:

=====	
MaterialPackage	
=====	
Material	Ilmenite
Mass	1.00000000e+03 kg
Amount	9.81797715e+00 kmol
Pressure	1.00000000e+00 atm
Temperature	5.00000000e+02 °C
Enthalpy	-1.87069549e+03 kWh

Compound Details:			
Formula	Mass	Mass Fraction	Mole Fraction
-----	-----	-----	-----
Al ₂ O ₃ [S1]	1.55371337e+01	1.55371337e-02	1.55207829e-02
CaO[S]	1.00239573e-02	1.00239573e-05	1.82066196e-05
Cr ₂ O ₃ [S]	2.20527060e-01	2.20527060e-04	1.47782739e-04
Fe ₂ O ₃ [S1]	4.74133178e+02	4.74133178e-01	3.02416515e-01
Fe ₃ O ₄ [S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	1.91457584e+02	1.91457584e-01	2.71429867e-01
K ₂ O[S]	1.00239573e-02	1.00239573e-05	1.08388880e-05
MgO[S]	5.81389521e+00	5.81389521e-03	1.46923993e-02
MnO[S]	4.81149948e+00	4.81149948e-03	6.90848565e-03
Na ₂ O[S1]	5.01197863e-02	5.01197863e-05	8.23650657e-05
P ₄ O ₁₀ [S]	3.20766632e-01	3.20766632e-04	1.15084949e-04
SiO ₂ [S1]	4.91173906e+00	4.91173906e-03	8.32630400e-03
TiO ₂ [S1]	2.94704343e+02	2.94704343e-01	3.75840583e-01
V ₂ O ₅ [S]	8.01916581e+00	8.01916581e-03	4.49078466e-03
=====	=====	=====	=====

In the above result some of the useful work that `auxi.modelling.process.materials` does behind the scenes is already evident. The amount in kmol and the enthalpy in kWh of the material package was calculated, as were the masses and mole fractions of the compounds. You will notice that the mass fractions in the material package is slightly different from those in the IlmeniteB material assay. This is because the assay was automatically normalised to add up to 1.0. You can switch of normalisation if that is more appropriate.

Summary The `auxi.modelling.process.materials` concepts described above can be summarised as follows:

- A material provides a list of properties that describes a “type of material”.
- A material assay describes a “specific material” by providing values for the listed properties.
- A material package describes a “specific quantity of material” belonging to a certain “type of material”.

You may be wondering what the use of all this is. Why go through all the effort of defining materials, material assays and material packages? The next section demonstrates the power of these concepts.

Material Package Calculations The use of materials and material packages are demonstrated here through the use of code snippets and the results produce by that code. We will be using ilmenite in the example. Firstly, let us import the `py:class:auxi.modelling.process.materials.thermo.Material` class, create a material object and print it out:

```
from auxi.modelling.process.materials.thermo import Material
```

```
ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
print(ilmenite)
```

The material looks as follows:

=====			
Material			
=====			
Name	Ilmenite		

Composition Details	(mass fractions)		
Compound	IlmeniteA	IlmeniteB	IlmeniteC

Al2O3[S1]	1.16000000e-02	1.55000000e-02	9.41000000e-03
CaO[S]	2.20000000e-04	1.00000000e-05	1.70000000e-04
Cr2O3[S]	8.00000000e-05	2.20000000e-04	1.10000000e-04
Fe2O3[S1]	2.02000000e-01	4.73000000e-01	4.96740000e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	2.79000000e-01	1.91000000e-01	0.00000000e+00
K2O[S]	4.00000000e-05	1.00000000e-05	5.00000000e-05
MgO[S]	1.04000000e-02	5.80000000e-03	1.09000000e-02
MnO[S]	5.40000000e-03	4.80000000e-03	5.25000000e-03
Na2O[S1]	7.00000000e-05	5.00000000e-05	3.10000000e-04
P4O10[S]	1.00000000e-05	3.20000000e-04	1.50000000e-04
SiO2[S1]	8.50000000e-03	4.90000000e-03	1.74400000e-02
TiO2[S1]	4.77000000e-01	2.94000000e-01	4.59490000e-01
V2O5[S]	3.60000000e-03	8.00000000e-03	0.00000000e+00
=====			

Creating, Adding and Extracting Next we can use the material object (called ilmenite) to create a material package using each of the ilmenite assays:

```
ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
print(ilma_package)
ilmb_package = ilmenite.create_package("IlmeniteB", 500.0, 1.0, 750.0)
print(ilmb_package)
ilmc_package = ilmenite.create_package("IlmeniteC", 250.0, 1.0, 1200.0)
print(ilmc_package)
```

Different masses were created from each assay (300 kg of IlmeniteA, 500.0 kg of IlmeniteB and 250.0 kg of IlmeniteC). All three packages were assigned a pressure of 1 atm, which is of no consequence. The packages were assigned temperatures of 25, 750 and 1200 °C respectively. In three short lines of code, `auxi.modelling.process.materials` did the following for us:

- Normalise the specified assay so that the mass fractions add up to 1.0. (We can choose not to do this.)
- Calculate the mass of each compound by multiplying the component mass fraction by the total package mass.
- Calculate the mass fraction of each compound.

- Calculate the mole fraction of each compound.
- Calculate the total amount (in kmol) of components in the package.
- Calculate the total enthalpy of the package by calculating the enthalpy of each compound and adding it together.

The result is as follows:

=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	3.00000000e+02	kg	
Amount	3.52817004e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	2.50000000e+01	°C	
Enthalpy	-6.87812118e+02	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	3.48725349e+00	1.16241783e-02	9.69390473e-03
CaO[S]	6.61375661e-02	2.20458554e-04	3.34280337e-04
Cr2O3[S]	2.40500241e-02	8.01667468e-05	4.48486990e-05
Fe2O3[S1]	6.07263107e+01	2.02421036e-01	1.07784066e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	8.38744589e+01	2.79581530e-01	3.30892788e-01
K2O[S]	1.20250120e-02	4.00833734e-05	3.61829148e-05
MgO[S]	3.12650313e+00	1.04216771e-02	2.19865404e-02
MnO[S]	1.62337662e+00	5.41125541e-03	6.48625791e-03
Na2O[S1]	2.10437710e-02	7.01459035e-05	9.62343053e-05
P4O10[S]	3.00625301e-03	1.00208434e-05	3.00142421e-06
SiO2[S1]	2.55531506e+00	8.51771685e-03	1.20540764e-02
TiO2[S1]	1.43398268e+02	4.77994228e-01	5.08901291e-01
V2O5[S]	1.08225108e+00	3.60750361e-03	1.68652807e-03
=====			
=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	5.00000000e+02	kg	
Amount	4.90898858e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	7.50000000e+02	°C	
Enthalpy	-9.05451326e+02	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	7.76856687e+00	1.55371337e-02	1.55207829e-02
CaO[S]	5.01197863e-03	1.00239573e-05	1.82066196e-05

Cr2O3[S]	1.10263530e-01	2.20527060e-04	1.47782739e-04
Fe2O3[S1]	2.37066589e+02	4.74133178e-01	3.02416515e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	9.57287918e+01	1.91457584e-01	2.71429867e-01
K2O[S]	5.01197863e-03	1.00239573e-05	1.08388880e-05
MgO[S]	2.90694760e+00	5.81389521e-03	1.46923993e-02
MnO[S]	2.40574974e+00	4.81149948e-03	6.90848565e-03
Na2O[S1]	2.50598931e-02	5.01197863e-05	8.23650657e-05
P4O10[S]	1.60383316e-01	3.20766632e-04	1.15084949e-04
SiO2[S1]	2.45586953e+00	4.91173906e-03	8.32630400e-03
TiO2[S1]	1.47352172e+02	2.94704343e-01	3.75840583e-01
V2O5[S]	4.00958290e+00	8.01916581e-03	4.49078466e-03
=====			
=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	2.50000000e+02	kg	
Amount	2.40014670e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	1.20000000e+03	°C	
Enthalpy	-5.25247309e+02	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	2.35245295e+00	9.40981180e-03	9.61275553e-03
CaO[S]	4.24991500e-02	1.69996600e-04	3.15758164e-04
Cr2O3[S]	2.74994500e-02	1.09997800e-04	7.53824179e-05
Fe2O3[S1]	1.24182516e+02	4.96730065e-01	3.24003606e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
K2O[S]	1.24997500e-02	4.99990000e-05	5.52880254e-05
MgO[S]	2.72494550e+00	1.08997820e-02	2.81687499e-02
MnO[S]	1.31247375e+00	5.24989500e-03	7.70863128e-03
Na2O[S1]	7.74984500e-02	3.09993800e-04	5.20968045e-04
P4O10[S]	3.74992500e-02	1.49997000e-04	5.50346434e-05
SiO2[S1]	4.35991280e+00	1.74396512e-02	3.02328445e-02
TiO2[S1]	1.14870203e+02	4.59480810e-01	5.99250982e-01
V2O5[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
=====			

We can now add these three packages of ilmenite together:

```
total_package = ilma_package + ilmb_package + ilmc_package
print(total_package)
```

In one line of code we did the following:

- Calculate the total mass of each component by adding up the component masses from the three original packages.

- Calculate the mass fraction of each compound.
- Calculate the mole fraction of each compound.
- Calculate the total amount (in kmol) of compounds in the package.
- Calculate the total enthalpy of the package by adding up the enthalpies from the three original packages.
- Calculate the temperature of the new package.

This new package (total_package) looks like this:

```
=====
MaterialPackage
=====
Material          Ilmenite
Mass              1.05000000e+03 kg
Amount           1.08373053e+01 kmol
Pressure         1.00000000e+00 atm
Temperature      6.61513374e+02 °C
Enthalpy         -2.11851075e+03 kWh
-----
Compound Details
Formula          Mass              Mass Fraction    Mole Fraction
-----
Al2O3[S1]       1.36082733e+01  1.29602603e-02  1.23153413e-02
CaO[S]          1.13648695e-01  1.08236852e-04  1.87005885e-04
Cr2O3[S]        1.61813004e-01  1.54107623e-04  9.82371950e-05
Fe2O3[S1]       4.21975416e+02  4.01881349e-01  2.43833300e-01
Fe3O4[S1]       0.00000000e+00  0.00000000e+00  0.00000000e+00
FeO[S]          1.79603251e+02  1.71050715e-01  2.30674699e-01
K2O[S]          2.95367407e-02  2.81302292e-05  2.89340215e-05
MgO[S]          8.75839623e+00  8.34132975e-03  2.00516825e-02
MnO[S]          5.34160012e+00  5.08723821e-03  6.94823498e-03
Na2O[S1]        1.23602114e-01  1.17716299e-04  1.84018059e-04
P4O10[S]        2.00888819e-01  1.91322685e-04  6.52958859e-05
SiO2[S1]        9.37109739e+00  8.92485465e-03  1.43915687e-02
TiO2[S1]        4.05620643e+02  3.86305374e-01  4.68638424e-01
V2O5[S]         5.09183399e+00  4.84936570e-03  2.58325918e-03
=====
```

We can easily extract a part of a material package into a new one. Let us remove 30 kg from the new package and store it in a new package:

```
dust_package = total_package.extract(30.0)
print(dust_package)
print(total_package)
```

By using one line of code we subtracted 30 kg of material from the original package and created a new one containing the subtracted 30 kg. All the other properties (e.g component masses, total amount and enthalpy) of the two packages were also recalculated. The extracted 30 kg package looks like this:

=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	3.00000000e+01	kg	
Amount	3.09637295e-01	kmol	
Pressure	1.00000000e+00	atm	
Temperature	6.61513374e+02	°C	
Enthalpy	-6.05288787e+01	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	3.88807809e-01	1.29602603e-02	1.23153413e-02
CaO[S]	3.24710557e-03	1.08236852e-04	1.87005885e-04
Cr2O3[S]	4.62322868e-03	1.54107623e-04	9.82371950e-05
Fe2O3[S1]	1.20564405e+01	4.01881349e-01	2.43833300e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	5.13152145e+00	1.71050715e-01	2.30674699e-01
K2O[S]	8.43906876e-04	2.81302292e-05	2.89340215e-05
MgO[S]	2.50239892e-01	8.34132975e-03	2.00516825e-02
MnO[S]	1.52617146e-01	5.08723821e-03	6.94823498e-03
Na2O[S1]	3.53148898e-03	1.17716299e-04	1.84018059e-04
P4O10[S]	5.73968055e-03	1.91322685e-04	6.52958859e-05
SiO2[S1]	2.67745640e-01	8.92485465e-03	1.43915687e-02
TiO2[S1]	1.15891612e+01	3.86305374e-01	4.68638424e-01
V2O5[S]	1.45480971e-01	4.84936570e-03	2.58325918e-03
=====			

The original package, which now contains 30 kg less, now looks like this:

=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	1.02000000e+03	kg	
Amount	1.05276680e+01	kmol	
Pressure	1.00000000e+00	atm	
Temperature	6.61513374e+02	°C	
Enthalpy	-2.05798187e+03	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	1.32194655e+01	1.29602603e-02	1.23153413e-02
CaO[S]	1.10401589e-01	1.08236852e-04	1.87005885e-04
Cr2O3[S]	1.57189775e-01	1.54107623e-04	9.82371950e-05
Fe2O3[S1]	4.09918976e+02	4.01881349e-01	2.43833300e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	1.74471729e+02	1.71050715e-01	2.30674699e-01
K2O[S]	2.86928338e-02	2.81302292e-05	2.89340215e-05
MgO[S]	8.50815634e+00	8.34132975e-03	2.00516825e-02

MnO [S]	5.18898297e+00	5.08723821e-03	6.94823498e-03
Na2O [S1]	1.20070625e-01	1.17716299e-04	1.84018059e-04
P4O10 [S]	1.95149139e-01	1.91322685e-04	6.52958859e-05
SiO2 [S1]	9.10335175e+00	8.92485465e-03	1.43915687e-02
TiO2 [S1]	3.94031481e+02	3.86305374e-01	4.68638424e-01
V2O5 [S]	4.94635301e+00	4.84936570e-03	2.58325918e-03
=====			

Summary All the other capabilities of the `auxi.modelling.process.materials.thermometer` class are not demonstrated here, since the purpose of this section is simply to introduce you to the material, material assay and material package concepts in `auxi.modelling.process.materials`. For full details on how to use the different Material and MaterialPackage classes and objects, refer to the following section:

- `section_chemistry_material_calculations`
- `section_psd_material_calculations`
- `section_psd_slurry_material_calculations`
- *thermochemistry material Calculations*

The final point to make is that the classes in `auxi.modelling.process.materials` can assist you in performing large numbers of metallurgical calculations with very few lines of code. The purpose of this is to focus you on the process concepts rather than entangle you in the detail of tens or hundreds of stoichiometry and thermochemical calculations. This should keep your code clean and your mind clear, getting the job done well in a short space of time.

thermochemistry material Calculations The purpose of this section is to explain a number of concepts and demonstrate the use of the Material and MaterialPackage classes in the `auxi.modelling.process.materials.thermo` module.

Material Description Files You need to create one or more material description files (MDFs) before you can create a material object in Python. Material description data are stored in simple text files with “.txt” extensions. The most simple format of such a file is the “mix.txt” file shown here:

```
Compound
Al2O3 [S1]
C [S1]
CaO [S]
Cr2O3 [S]
Fe2O3 [S1]
Fe3O4 [S1]
FeO [S]
H2 [G]
K2O [S]
MgO [S]
MnO [S]
```

```
Na2O[S1]
N2[G]
O2[G]
P4O10[S]
S[S1]
SiO2[S1]
TiO2[S1]
V2O5[S]
```

The file contains a header row, which in this case only contains the word “Compound”. All subsequent rows contain chemical compound phases. For example, the second line contains the S1 phase of Al₂O₃. When you consult FactSage, you will see that S1 refers to the gamma phase. The third line contains the graphite phase of carbon, and so forth. The purpose of this file is to tell auxi that materials based on this MDF will contain these compound phases, and nothing else.

Material description files can also contain material assays. The content of the “ilmenite.txt” MDF is shown here:

Compound	IlmeniteA	IlmeniteB	IlmeniteC
Al ₂ O ₃ [S1]	0.01160	0.01550	0.00941
CaO[S]	0.00022	0.00001	0.00017
Cr ₂ O ₃ [S]	0.00008	0.00022	0.00011
Fe ₂ O ₃ [S1]	0.20200	0.47300	0.49674
Fe ₃ O ₄ [S1]	0.00000	0.00000	0.00000
FeO[S]	0.27900	0.19100	0.00000
K ₂ O[S]	0.00004	0.00001	0.00005
MgO[S]	0.01040	0.00580	0.01090
MnO[S]	0.00540	0.00480	0.00525
Na ₂ O[S1]	0.00007	0.00005	0.00031
P ₄ O ₁₀ [S]	0.00001	0.00032	0.00015
SiO ₂ [S1]	0.00850	0.00490	0.01744
TiO ₂ [S1]	0.47700	0.29400	0.45949
V ₂ O ₅ [S]	0.00360	0.00800	0.00000

The first row still contains the word “Compound” as header for the list of compound phases. The subsequent words in the first row are assay names. **An assay name may not contain space or tab characters.** If it does, it will be interpreted as more than one name.

The first column of the file has the same meaning as the single column in the “mix.txt” file. It is a list of chemical compound phases that are allowed in materials based on this MDF. All subsequent columns contain assay information. Generally the numbers are mass fractions of the different component phases for the respective material assays. If you will be normalising your assays, the numbers can be masses, percentages or mass fractions, since they will be converted to mass fractions by normalisation.

There is more more twist in the MDF tale. You can add your own custom material properties to the file. The “ilmenite.txt” file was modified to include prices for the different ilmenites:

Compound	IlmeniteA	IlmeniteB	IlmeniteC
Al ₂ O ₃ [S1]	0.01160	0.01550	0.00941
CaO[S]	0.00022	0.00001	0.00017

Cr2O3[S]	0.00008	0.00022	0.00011
Fe2O3[S1]	0.20200	0.47300	0.49674
Fe3O4[S1]	0.00000	0.00000	0.00000
FeO[S]	0.27900	0.19100	0.00000
K2O[S]	0.00004	0.00001	0.00005
MgO[S]	0.01040	0.00580	0.01090
MnO[S]	0.00540	0.00480	0.00525
Na2O[S1]	0.00007	0.00005	0.00031
P4O10[S]	0.00001	0.00032	0.00015
SiO2[S1]	0.00850	0.00490	0.01744
TiO2[S1]	0.47700	0.29400	0.45949
V2O5[S]	0.00360	0.00800	0.00000
#			
Price[USD/ton]	47.5000	32.2300	45.1400

The name of the property in this case is “Price” and its units are “USD/ton”. **There may be no spaces in the string containing the property name and units.** In this case the string is “Price[USD/ton]”, which serves the purpose of describing the custom property clearly.

Be careful not to leave empty lines at the end of your material description file. It tends to cause problems.

Materials Now that we have created a few material description files, we can create material objects in Python.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 print(ilmenite)
5
6 reductant = Material("Reductant", "./materials/reductant.txt")
7 print(reductant)
8
9 mix = Material("Mix", "./materials/mix.txt")
10 print(mix)

```

The Material class is imported on line 1. On line 3 a Material object is created, specifying the name of the object as the first parameter, and the location and name of the material description file as the second parameter. Two more Material objects are created on lines 6 and 9. The materials are preted out after creation, with the following result:

=====			
Material			
=====			
Name	Ilmenite		

Composition Details (mass fractions)			
Compound	IlmeniteA	IlmeniteB	IlmeniteC

Al2O3[S1]	1.16000000e-02	1.55000000e-02	9.41000000e-03
CaO[S]	2.20000000e-04	1.00000000e-05	1.70000000e-04

Cr2O3[S]	8.00000000e-05	2.20000000e-04	1.10000000e-04
Fe2O3[S1]	2.02000000e-01	4.73000000e-01	4.96740000e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	2.79000000e-01	1.91000000e-01	0.00000000e+00
K2O[S]	4.00000000e-05	1.00000000e-05	5.00000000e-05
MgO[S]	1.04000000e-02	5.80000000e-03	1.09000000e-02
MnO[S]	5.40000000e-03	4.80000000e-03	5.25000000e-03
Na2O[S1]	7.00000000e-05	5.00000000e-05	3.10000000e-04
P4O10[S]	1.00000000e-05	3.20000000e-04	1.50000000e-04
SiO2[S1]	8.50000000e-03	4.90000000e-03	1.74400000e-02
TiO2[S1]	4.77000000e-01	2.94000000e-01	4.59490000e-01
V2O5[S]	3.60000000e-03	8.00000000e-03	0.00000000e+00

Custom Properties:			

Price[USD/ton]	4.75000000e+01	3.22300000e+01	4.51400000e+01
=====			
=====			
Material			
=====			
Name	Reductant		

Composition Details (mass fractions)			
Compound	ReductantA	ReductantB	

C[S1]	8.40973866e-01	1.00000000e+00	
H2[G]	1.37955186e-02	0.00000000e+00	
O2[G]	4.94339606e-02	0.00000000e+00	
N2[G]	6.09802120e-03	0.00000000e+00	
S[S1]	2.04933390e-03	0.00000000e+00	
Al2O3[S1]	1.20884160e-03	0.00000000e+00	
CaO[S]	2.94179980e-03	0.00000000e+00	
Fe2O3[S1]	7.85955656e-02	0.00000000e+00	
MgO[S]	1.41179360e-03	0.00000000e+00	
SiO2[S1]	3.49129950e-03	0.00000000e+00	
=====			
=====			
Material			
=====			
Name	Mix		

Compound			

Al2O3[S1]			
C[S1]			
CaO[S]			
Cr2O3[S]			
Fe2O3[S1]			
Fe3O4[S1]			

```

FeO[S]
H2[G]
K2O[S]
MgO[S]
MnO[S]
Na2O[S1]
N2[G]
O2[G]
P4O10[S]
S[S1]
SiO2[S1]
TiO2[S1]
V2O5[S]
=====

```

The material objects are now ready to create material packages.

Material Packages

Creating Empty Packages The simplest way to create material packages is to create empty ones.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 empty_ilmenite_package = ilmenite.create_package()
8 print(empty_ilmenite_package)
9
10 empty_reductant_package = reductant.create_package()
11 print(empty_reductant_package)
12
13 empty_mix_package = mix.create_package()
14 print(empty_mix_package)

```

The empty packages are created by calling the “create_package” method of the Material objects without passing any parameters.

```

=====
MaterialPackage
=====
Material          Ilmenite
Mass              0.00000000e+00 kg
Amount           0.00000000e+00 kmol
Pressure         1.00000000e+00 atm
Temperature      2.50000000e+01 °C
Enthalpy         0.00000000e+00 kWh
-----

```

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
CaO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Cr2O3[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Fe2O3[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
K2O[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
MgO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
MnO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Na2O[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
P4O10[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
SiO2[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
TiO2[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
V2O5[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
=====			
=====			
MaterialPackage			
=====			
Material	Reductant		
Mass	0.00000000e+00	kg	
Amount	0.00000000e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	2.50000000e+01	°C	
Enthalpy	0.00000000e+00	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

C[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
H2[G]	0.00000000e+00	0.00000000e+00	0.00000000e+00
O2[G]	0.00000000e+00	0.00000000e+00	0.00000000e+00
N2[G]	0.00000000e+00	0.00000000e+00	0.00000000e+00
S[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Al2O3[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
CaO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Fe2O3[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
MgO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
SiO2[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
=====			
=====			
MaterialPackage			
=====			
Material	Mix		
Mass	0.00000000e+00	kg	
Amount	0.00000000e+00	kmol	
Pressure	1.00000000e+00	atm	

Temperature	2.50000000e+01	°C	
Enthalpy	0.00000000e+00	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
C[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
CaO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Cr2O3[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Fe2O3[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
H2[G]	0.00000000e+00	0.00000000e+00	0.00000000e+00
K2O[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
MgO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
MnO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Na2O[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
N2[G]	0.00000000e+00	0.00000000e+00	0.00000000e+00
O2[G]	0.00000000e+00	0.00000000e+00	0.00000000e+00
P4O10[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
S[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
SiO2[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
TiO2[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
V2O5[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
=====			

Creating Filled Packages It is just as easy to create packages that contain some mass. Let's do that with ilmenite.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8 print(ilma_package)

```

The parameters to the “create_package” method are:

1. material assay name, “IlmeniteA”
2. mass, 300 kg
3. pressure, 1 atm
4. temperature, 25 °C

We therefore created 300 kg based on the composition specified by the IlmeniteA assay, at 1 atm pressure and 25 °C temperature. The resulting package is shown here.

```
=====
MaterialPackage
=====
```

```
Material          Ilmenite
Mass              3.00000000e+02 kg
Amount           3.52817004e+00 kmol
Pressure          1.00000000e+00 atm
Temperature       2.50000000e+01 °C
Enthalpy         -6.87812118e+02 kWh
-----
```

Compound Details

Formula	Mass	Mass Fraction	Mole Fraction
Al2O3[S1]	3.48725349e+00	1.16241783e-02	9.69390473e-03
CaO[S]	6.61375661e-02	2.20458554e-04	3.34280337e-04
Cr2O3[S]	2.40500241e-02	8.01667468e-05	4.48486990e-05
Fe2O3[S1]	6.07263107e+01	2.02421036e-01	1.07784066e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	8.38744589e+01	2.79581530e-01	3.30892788e-01
K2O[S]	1.20250120e-02	4.00833734e-05	3.61829148e-05
MgO[S]	3.12650313e+00	1.04216771e-02	2.19865404e-02
MnO[S]	1.62337662e+00	5.41125541e-03	6.48625791e-03
Na2O[S1]	2.10437710e-02	7.01459035e-05	9.62343053e-05
P4O10[S]	3.00625301e-03	1.00208434e-05	3.00142421e-06
SiO2[S1]	2.55531506e+00	8.51771685e-03	1.20540764e-02
TiO2[S1]	1.43398268e+02	4.77994228e-01	5.08901291e-01
V2O5[S]	1.08225108e+00	3.60750361e-03	1.68652807e-03

```
=====
```

Adding Material to a Package - Another Package Now we create another ilmenite package with a different composition, mass and temperature, and add it to the first:

```
1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8 ilmb_package = ilmenite.create_package("IlmeniteB", 500.0, 1.0, 750.0)
9
10 ilma_package += ilmb_package
11 print(ilma_package)
12 print(ilmb_package)
```

This changes the original “ilma_package”, but the second “ilmb_package” remains the same. This is quite a powerful action, since one line of code does all of the following:

- Calculate the total mass of each component by adding up the component masses from the two packages.

- Calculate the mass fraction of each compound.
- Calculate the mole fraction of each compound.
- Calculate the total amount (in kmol) of compounds in the package.
- Calculate the total enthalpy by adding up the enthalpies of the two original packages.
- Calculate the temperature of the new package.

The resulting two packages are shown below:

=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	8.00000000e+02	kg	
Amount	8.43715862e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	4.88474167e+02	°C	
Enthalpy	-1.59326344e+03	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	1.12558204e+01	1.40697755e-02	1.30841549e-02
CaO[S]	7.11495448e-02	8.89369310e-05	1.50379294e-04
Cr2O3[S]	1.34313554e-01	1.67891942e-04	1.04738770e-04
Fe2O3[S1]	2.97792900e+02	3.72241125e-01	2.21026985e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	1.79603251e+02	2.24504063e-01	2.96295501e-01
K2O[S]	1.70369907e-02	2.12962383e-05	2.14370100e-05
MgO[S]	6.03345073e+00	7.54181341e-03	1.77425932e-02
MnO[S]	4.02912637e+00	5.03640796e-03	6.73192250e-03
Na2O[S1]	4.61036642e-02	5.76295802e-05	8.81647712e-05
P4O10[S]	1.63389569e-01	2.04236961e-04	6.82149359e-05
SiO2[S1]	5.01118458e+00	6.26398073e-03	9.88514810e-03
TiO2[S1]	2.90750440e+02	3.63438050e-01	4.31482633e-01
V2O5[S]	5.09183399e+00	6.36479248e-03	3.31812755e-03
=====			
=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	5.00000000e+02	kg	
Amount	4.90898858e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	7.50000000e+02	°C	
Enthalpy	-9.05451326e+02	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	7.76856687e+00	1.55371337e-02	1.55207829e-02
CaO[S]	5.01197863e-03	1.00239573e-05	1.82066196e-05
Cr2O3[S]	1.10263530e-01	2.20527060e-04	1.47782739e-04
Fe2O3[S1]	2.37066589e+02	4.74133178e-01	3.02416515e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	9.57287918e+01	1.91457584e-01	2.71429867e-01
K2O[S]	5.01197863e-03	1.00239573e-05	1.08388880e-05
MgO[S]	2.90694760e+00	5.81389521e-03	1.46923993e-02
MnO[S]	2.40574974e+00	4.81149948e-03	6.90848565e-03
Na2O[S1]	2.50598931e-02	5.01197863e-05	8.23650657e-05
P4O10[S]	1.60383316e-01	3.20766632e-04	1.15084949e-04
SiO2[S1]	2.45586953e+00	4.91173906e-03	8.32630400e-03
TiO2[S1]	1.47352172e+02	2.94704343e-01	3.75840583e-01
V2O5[S]	4.00958290e+00	8.01916581e-03	4.49078466e-03
=====			

Adding Material to a Package - A Compound Mass Sometimes you need to add material to a package, one compound at a time.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 ilma_package += ("TiO2[S1]", 150.0)
10 print(ilma_package)

```

This adds 150 kg of TiO2[S1] to ilma_package. The temperature of the added material is assumed to be the same as that of the original package, which means that ilma_package's temperature does not change. Here is the result:

=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	4.50000000e+02	kg	
Amount	5.40632064e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	2.50000000e+01	°C	
Enthalpy	-1.18069622e+03	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	3.48725349e+00	7.74945219e-03	6.32625154e-03
CaO[S]	6.61375661e-02	1.46972369e-04	2.18151669e-04
Cr2O3[S]	2.40500241e-02	5.34444979e-05	2.92683040e-05

Fe2O3[S1]	6.07263107e+01	1.34947357e-01	7.03399852e-02
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	8.38744589e+01	1.86387686e-01	2.15940951e-01
K2O[S]	1.20250120e-02	2.67222489e-05	2.36130050e-05
MgO[S]	3.12650313e+00	6.94778473e-03	1.43484374e-02
MnO[S]	1.62337662e+00	3.60750361e-03	4.23293814e-03
Na2O[S1]	2.10437710e-02	4.67639357e-05	6.28026001e-05
P4O10[S]	3.00625301e-03	6.68056224e-06	1.95873232e-06
SiO2[S1]	2.55531506e+00	5.67847790e-03	7.86650184e-03
TiO2[S1]	2.93398268e+02	6.51996152e-01	6.79508511e-01
V2O5[S]	1.08225108e+00	2.40500241e-03	1.10062984e-03
=====			

Adding Material to a Package - A Compound Mass with Specified Temperature We can also add a certain mass of a specified compound at a temperature different from the original package.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 ilma_package += ("TiO2[S1]", 150.0, 1000.0)
10 print(ilma_package)

```

This action calculates a new total mass, component masses, mass fractions and mole fractions, as well as a new enthalpy and temperature.

=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	4.50000000e+02	kg	
Amount	5.40632064e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	3.84927151e+02	°C	
Enthalpy	-1.14449836e+03	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	3.48725349e+00	7.74945219e-03	6.32625154e-03
CaO[S]	6.61375661e-02	1.46972369e-04	2.18151669e-04
Cr2O3[S]	2.40500241e-02	5.34444979e-05	2.92683040e-05
Fe2O3[S1]	6.07263107e+01	1.34947357e-01	7.03399852e-02
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	8.38744589e+01	1.86387686e-01	2.15940951e-01

K2O[S]	1.20250120e-02	2.67222489e-05	2.36130050e-05
MgO[S]	3.12650313e+00	6.94778473e-03	1.43484374e-02
MnO[S]	1.62337662e+00	3.60750361e-03	4.23293814e-03
Na2O[S1]	2.10437710e-02	4.67639357e-05	6.28026001e-05
P4O10[S]	3.00625301e-03	6.68056224e-06	1.95873232e-06
SiO2[S1]	2.55531506e+00	5.67847790e-03	7.86650184e-03
TiO2[S1]	2.93398268e+02	6.51996152e-01	6.79508511e-01
V2O5[S]	1.08225108e+00	2.40500241e-03	1.10062984e-03
=====			

Adding Packages of Different Materials Together We very often need to add packages from different materials together. For example, ilmenite and reductant can be added together so that reduction reactions can be modelled.

```
1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8 reda_package = reductant.create_package("ReductantA", 35.0, 1.0, 25.0)
9
10 new_package = ilma_package + reda_package
11 print(new_package)
```

This, however, does not work. See the last two lines of the error message below.

```
Traceback (most recent call last):
File "test.py", line 10, in <module>
    new_package = ilma_package + reda_package
File "thermochemistry.material.py", line 430, in __add__
    self.material.name + ".")
Exception: Packages of 'Reductant' cannot be added to packages of 'Ilmenite'.
The compound 'C[S1]' was not found in 'Ilmenite'.
```

Let's try it by swapping the two material packages around.

```
1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8 reda_package = reductant.create_package("ReductantA", 35.0, 1.0, 25.0)
9
10 new_package = reda_package + ilma_package
11 print(new_package)
```

```

Traceback (most recent call last):
File "test.py", line 10, in <module>
    new_package = reda_package + ilma_package
File "thermochemistry.material.py", line 430, in __add__
    self.material.name + "'.")
Exception: Packages of 'Ilmenite' cannot be added to packages of 'Reductant'.
The compound 'Cr2O3[S]' was not found in 'Reductant'.

```

Still no luck. These packages cannot be added together because their materials are not compatible. We need to use an intermediate material package from a compatible material that will allow us to add ilmenite and reductant together. This is the purpose of the “mix” material that we created early on.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8 reda_package = reductant.create_package("ReductantA", 35.0, 1.0, 25.0)
9
10 new_package = mix.create_package()
11 new_package += ilma_package
12 new_package += reda_package
13 print(new_package)

```

Success at last! The mix material package is able to receive all the compound masses from both the ilmenite and reductant packages.

```

=====
MaterialPackage
=====
Material          Mix
Mass              3.35000000e+02 kg
Amount            6.30500835e+00 kmol
Pressure          1.00000000e+00 atm
Temperature       2.50000000e+01 °C
Enthalpy          -6.92925041e+02 kWh
-----
Compound Details
Formula           Mass           Mass Fraction   Mole Fraction
-----
Al2O3[S1]         3.52956294e+00   1.05360088e-02   5.49034965e-03
C[S1]             2.94340853e+01   8.78629412e-02   3.88683906e-01
CaO[S]            1.69100559e-01   5.04777788e-04   4.78268203e-04
Cr2O3[S]          2.40500241e-02   7.17911166e-05   2.50965308e-05
Fe2O3[S1]         6.34771555e+01   1.89484046e-01   6.30462073e-02
Fe3O4[S1]         0.00000000e+00   0.00000000e+00   0.00000000e+00
FeO[S]            8.38744589e+01   2.50371519e-01   1.85161693e-01
H2[G]             4.82843151e-01   1.44132284e-03   3.79888138e-02

```

K2O[S]	1.20250120e-02	3.58955583e-05	2.02473128e-05
MgO[S]	3.17591590e+00	9.48034598e-03	1.24977222e-02
MnO[S]	1.62337662e+00	4.84590037e-03	3.62959406e-03
Na2O[S1]	2.10437710e-02	6.28172270e-05	5.38509982e-05
N2[G]	2.13430742e-01	6.37106693e-04	1.20838199e-03
O2[G]	1.73018862e+00	5.16474215e-03	8.57578913e-03
P4O10[S]	3.00625301e-03	8.97388957e-06	1.67954337e-06
S[S1]	7.17266865e-02	2.14109512e-04	3.54772799e-04
SiO2[S1]	2.67751054e+00	7.99256877e-03	7.06780432e-03
TiO2[S1]	1.43398268e+02	4.28054533e-01	2.84772072e-01
V2O5[S]	1.08225108e+00	3.23060025e-03	9.43750984e-04
=====			

Adding Material Together - Package + Package In the above three sections we demonstrated how material can be added to an existing package. Here we will add material together to create a new package.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8 ilmb_package = ilmenite.create_package("IlmeniteB", 500.0, 1.0, 750.0)
9
10 new_package = ilma_package + ilmb_package
11 print(new_package)

```

This action performs all the calculations to create a new package with properties based on the two original packages. Specifically note that the temperature was automatically calculated.

=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	8.00000000e+02	kg	
Amount	8.43715862e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	4.88474167e+02	°C	
Enthalpy	-1.59326344e+03	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	1.12558204e+01	1.40697755e-02	1.30841549e-02
CaO[S]	7.11495448e-02	8.89369310e-05	1.50379294e-04
Cr2O3[S]	1.34313554e-01	1.67891942e-04	1.04738770e-04
Fe2O3[S1]	2.97792900e+02	3.72241125e-01	2.21026985e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00

FeO[S]	1.79603251e+02	2.24504063e-01	2.96295501e-01
K2O[S]	1.70369907e-02	2.12962383e-05	2.14370100e-05
MgO[S]	6.03345073e+00	7.54181341e-03	1.77425932e-02
MnO[S]	4.02912637e+00	5.03640796e-03	6.73192250e-03
Na2O[S1]	4.61036642e-02	5.76295802e-05	8.81647712e-05
P4O10[S]	1.63389569e-01	2.04236961e-04	6.82149359e-05
SiO2[S1]	5.01118458e+00	6.26398073e-03	9.88514810e-03
TiO2[S1]	2.90750440e+02	3.63438050e-01	4.31482633e-01
V2O5[S]	5.09183399e+00	6.36479248e-03	3.31812755e-03
=====			

Adding Material Together - Package + Compound Mass Now we add a package and specific mass of a compound together to produce a new package.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 new_package = ilma_package + ("TiO2[S1]", 150.0)
10 print(new_package)

```

The added compound mass is assumed to be at the same temperature as the original package. This results in the new package having the same temperature as the original package.

=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	4.50000000e+02	kg	
Amount	5.40632064e+00	kmol	
Pressure	1.00000000e+00	atm	
Temperature	2.50000000e+01	°C	
Enthalpy	-1.18069622e+03	kWh	

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	3.48725349e+00	7.74945219e-03	6.32625154e-03
CaO[S]	6.61375661e-02	1.46972369e-04	2.18151669e-04
Cr2O3[S]	2.40500241e-02	5.34444979e-05	2.92683040e-05
Fe2O3[S1]	6.07263107e+01	1.34947357e-01	7.03399852e-02
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	8.38744589e+01	1.86387686e-01	2.15940951e-01
K2O[S]	1.20250120e-02	2.67222489e-05	2.36130050e-05
MgO[S]	3.12650313e+00	6.94778473e-03	1.43484374e-02
MnO[S]	1.62337662e+00	3.60750361e-03	4.23293814e-03
Na2O[S1]	2.10437710e-02	4.67639357e-05	6.28026001e-05

P4O10[S]	3.00625301e-03	6.68056224e-06	1.95873232e-06
SiO2[S1]	2.55531506e+00	5.67847790e-03	7.86650184e-03
TiO2[S1]	2.93398268e+02	6.51996152e-01	6.79508511e-01
V2O5[S]	1.08225108e+00	2.40500241e-03	1.10062984e-03
=====			

Adding Material Together - Package + Compound Mass at Specified Temperature Now we add the same compound mass as in the previous section, but at a different temperature.

```

1 from auxi.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 new_package = ilma_package + ("TiO2[S1]", 150.0, 1000.0)
10 print(new_package)

```

The new package now has a different temperature, which is calculated based on the enthalpy of the original package and the enthalpy of the added compound mass.

```

=====
MaterialPackage
=====
Material          Ilmenite
Mass              4.50000000e+02 kg
Amount            5.40632064e+00 kmol
Pressure          1.00000000e+00 atm
Temperature       3.84927151e+02 °C
Enthalpy          -1.14449836e+03 kWh
-----
Compound Details
Formula           Mass           Mass Fraction   Mole Fraction
-----
Al2O3[S1]         3.48725349e+00   7.74945219e-03   6.32625154e-03
CaO[S]            6.61375661e-02   1.46972369e-04   2.18151669e-04
Cr2O3[S]          2.40500241e-02   5.34444979e-05   2.92683040e-05
Fe2O3[S1]         6.07263107e+01   1.34947357e-01   7.03399852e-02
Fe3O4[S1]         0.00000000e+00   0.00000000e+00   0.00000000e+00
FeO[S]            8.38744589e+01   1.86387686e-01   2.15940951e-01
K2O[S]            1.20250120e-02   2.67222489e-05   2.36130050e-05
MgO[S]            3.12650313e+00   6.94778473e-03   1.43484374e-02
MnO[S]            1.62337662e+00   3.60750361e-03   4.23293814e-03
Na2O[S1]          2.10437710e-02   4.67639357e-05   6.28026001e-05
P4O10[S]          3.00625301e-03   6.68056224e-06   1.95873232e-06
SiO2[S1]          2.55531506e+00   5.67847790e-03   7.86650184e-03
TiO2[S1]          2.93398268e+02   6.51996152e-01   6.79508511e-01
V2O5[S]           1.08225108e+00   2.40500241e-03   1.10062984e-03
=====

```

Extract Material from a Package - Mass When we need to create a new package by extracting material from an existing material, we use the “extract” method. First of all we can simply specify the total mass to be extracted.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 new_package = ilma_package.extract(75.0)
10 print(ilma_package)
11 print(new_package)

```

This removes 75 kg from the original package, and produces a new package of 75 kg. The new package has the same composition, temperature and pressure as the original one.

```

=====
MaterialPackage
=====
Material          Ilmenite
Mass              2.25000000e+02 kg
Amount           2.64612753e+00 kmol
Pressure         1.00000000e+00 atm
Temperature      2.50000000e+01 °C
Enthalpy        -5.15859089e+02 kWh
-----
Compound Details
Formula          Mass          Mass Fraction    Mole Fraction
-----
Al2O3[S1]       2.61544012e+00    1.16241783e-02    9.69390473e-03
CaO[S]          4.96031746e-02    2.20458554e-04    3.34280337e-04
Cr2O3[S]        1.80375180e-02    8.01667468e-05    4.48486990e-05
Fe2O3[S1]       4.55447330e+01    2.02421036e-01    1.07784066e-01
Fe3O4[S1]       0.00000000e+00    0.00000000e+00    0.00000000e+00
FeO[S]          6.29058442e+01    2.79581530e-01    3.30892788e-01
K2O[S]          9.01875902e-03    4.00833734e-05    3.61829148e-05
MgO[S]          2.34487734e+00    1.04216771e-02    2.19865404e-02
MnO[S]          1.21753247e+00    5.41125541e-03    6.48625791e-03
Na2O[S1]        1.57828283e-02    7.01459035e-05    9.62343053e-05
P4O10[S]        2.25468975e-03    1.00208434e-05    3.00142421e-06
SiO2[S1]        1.91648629e+00    8.51771685e-03    1.20540764e-02
TiO2[S1]        1.07548701e+02    4.77994228e-01    5.08901291e-01
V2O5[S]         8.11688312e-01    3.60750361e-03    1.68652807e-03
=====

=====
MaterialPackage
=====
Material          Ilmenite

```

Mass	7.50000000e+01	kg
Amount	8.82042511e-01	kmol
Pressure	1.00000000e+00	atm
Temperature	2.50000000e+01	°C
Enthalpy	-1.71953030e+02	kWh

Compound Details		
Formula	Mass	Mass Fraction Mole Fraction

Al2O3[S1]	8.71813372e-01	1.16241783e-02 9.69390473e-03
CaO[S]	1.65343915e-02	2.20458554e-04 3.34280337e-04
Cr2O3[S]	6.01250601e-03	8.01667468e-05 4.48486990e-05
Fe2O3[S1]	1.51815777e+01	2.02421036e-01 1.07784066e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
FeO[S]	2.09686147e+01	2.79581530e-01 3.30892788e-01
K2O[S]	3.00625301e-03	4.00833734e-05 3.61829148e-05
MgO[S]	7.81625782e-01	1.04216771e-02 2.19865404e-02
MnO[S]	4.05844156e-01	5.41125541e-03 6.48625791e-03
Na2O[S1]	5.26094276e-03	7.01459035e-05 9.62343053e-05
P4O10[S]	7.51563252e-04	1.00208434e-05 3.00142421e-06
SiO2[S1]	6.38828764e-01	8.51771685e-03 1.20540764e-02
TiO2[S1]	3.58495671e+01	4.77994228e-01 5.08901291e-01
V2O5[S]	2.70562771e-01	3.60750361e-03 1.68652807e-03
=====		

Extract Material from a Package - Compound We can also extract all the mass of a single compound from an existing package into a new one.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 new_package = ilma_package.extract("TiO2[S1]")
10 print(ilma_package)
11 print(new_package)

```

This modifies the original package's composition and enthalpy, and creates a new package of the same temperature consisting purely of the specified compound.

=====		
MaterialPackage		
=====		
Material	Ilmenite	
Mass	1.56601732e+02	kg
Amount	1.73267975e+00	kmol
Pressure	1.00000000e+00	atm
Temperature	2.50000000e+01	°C

Enthalpy	-2.16620609e+02 kWh		

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	3.48725349e+00	2.22682946e-02	1.97392185e-02
CaO[S]	6.61375661e-02	4.22329724e-04	6.80678509e-04
Cr2O3[S]	2.40500241e-02	1.53574445e-04	9.13231864e-05
Fe2O3[S1]	6.07263107e+01	3.87775474e-01	2.19475361e-01
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	8.38744589e+01	5.35590878e-01	6.73780610e-01
K2O[S]	1.20250120e-02	7.67872226e-05	7.36774791e-05
MgO[S]	3.12650313e+00	1.99646779e-02	4.47701043e-02
MnO[S]	1.62337662e+00	1.03662751e-02	1.32076460e-02
Na2O[S1]	2.10437710e-02	1.34377640e-04	1.95957154e-04
P4O10[S]	3.00625301e-03	1.91968057e-05	6.11165160e-06
SiO2[S1]	2.55531506e+00	1.63172848e-02	2.45451193e-02
TiO2[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
V2O5[S]	1.08225108e+00	6.91085003e-03	3.43419366e-03
=====			
=====			
MaterialPackage			
=====			
Material	Ilmenite		
Mass	1.43398268e+02 kg		
Amount	1.79549029e+00 kmol		
Pressure	1.00000000e+00 atm		
Temperature	2.50000000e+01 °C		
Enthalpy	-4.71191509e+02 kWh		

Compound Details			
Formula	Mass	Mass Fraction	Mole Fraction

Al2O3[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
CaO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Cr2O3[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Fe2O3[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Fe3O4[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
FeO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
K2O[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
MgO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
MnO[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
Na2O[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
P4O10[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
SiO2[S1]	0.00000000e+00	0.00000000e+00	0.00000000e+00
TiO2[S1]	1.43398268e+02	1.00000000e+00	1.00000000e+00
V2O5[S]	0.00000000e+00	0.00000000e+00	0.00000000e+00
=====			

Extract Material from a Package - Compound Mass We may not want to extract all the mass of a specific compound. In this case we can specify the mass to extract.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 new_package = ilma_package.extract(("TiO2[S1]", 110.0))
10 print(ilma_package)
11 print(new_package)

```

The existing package is modified appropriately and a new package containing only the specified mass of the required compound is produced.

```

=====
MaterialPackage
=====
Material          Ilmenite
Mass              1.90000000e+02 kg
Amount           2.15085961e+00 kmol
Pressure         1.00000000e+00 atm
Temperature      2.50000000e+01 °C
Enthalpy        -3.26363778e+02 kWh
-----
Compound Details
Formula          Mass          Mass Fraction    Mole Fraction
-----
Al2O3[S1]       3.48725349e+00  1.83539657e-02  1.59014304e-02
CaO[S]          6.61375661e-02  3.48092453e-04  5.48337915e-04
Cr2O3[S]        2.40500241e-02  1.26579074e-04  7.35677195e-05
Fe2O3[S1]       6.07263107e+01  3.19612162e-01  1.76803968e-01
Fe3O4[S1]       0.00000000e+00  0.00000000e+00  0.00000000e+00
FeO[S]          8.38744589e+01  4.41444520e-01  5.42781136e-01
K2O[S]          1.20250120e-02  6.32895370e-05  5.93527704e-05
MgO[S]          3.12650313e+00  1.64552796e-02  3.60656982e-02
MnO[S]          1.62337662e+00  8.54408749e-03  1.06397557e-02
Na2O[S1]        2.10437710e-02  1.10756690e-04  1.57858278e-04
P4O10[S]        3.00625301e-03  1.58223842e-05  4.92339666e-06
SiO2[S1]        2.55531506e+00  1.34490266e-02  1.97729462e-02
TiO2[S1]        3.33982684e+01  1.75780360e-01  1.94424522e-01
V2O5[S]         1.08225108e+00  5.69605833e-03  2.76650220e-03
=====

=====
MaterialPackage
=====
Material          Ilmenite
Mass              1.10000000e+02 kg

```

Amount	1.37731044e+00	kmol
Pressure	1.00000000e+00	atm
Temperature	2.50000000e+01	°C
Enthalpy	-3.61448340e+02	kWh

Compound Details		
Formula	Mass	Mass Fraction Mole Fraction

Al2O3[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
CaO[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
Cr2O3[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
Fe2O3[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
Fe3O4[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
FeO[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
K2O[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
MgO[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
MnO[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
Na2O[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
P4O10[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
SiO2[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
TiO2[S1]	1.10000000e+02	1.00000000e+00 1.00000000e+00
V2O5[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
=====		

Extract Material from a Package - Material We may need to extract all the compounds that appear in a specific material into a new package.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 new_package = ilma_package.extract(reductant)
10 print(ilma_package)
11 print(new_package)

```

The existing package loses all the masses of components that appear in the specified material. The new package contains these masses and have the same temperature and pressure as the original material.

=====	
MaterialPackage	
=====	
Material	Ilmenite
Mass	2.30038480e+02 kg
Amount	2.99240730e+00 kmol
Pressure	1.00000000e+00 atm

Temperature	2.50000000e+01	°C
Enthalpy	-5.62518853e+02	kWh

Compound Details		
Formula	Mass	Mass Fraction Mole Fraction

Al2O3[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
CaO[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
Cr2O3[S]	2.40500241e-02	1.04547831e-04 5.28784420e-05
Fe2O3[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
Fe3O4[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
FeO[S]	8.38744589e+01	3.64610559e-01 3.90136068e-01
K2O[S]	1.20250120e-02	5.22739153e-05 4.26611298e-05
MgO[S]	0.00000000e+00	0.00000000e+00 0.00000000e+00
MnO[S]	1.62337662e+00	7.05697857e-03 7.64756215e-03
Na2O[S1]	2.10437710e-02	9.14793518e-05 1.13464164e-04
P4O10[S]	3.00625301e-03	1.30684788e-05 3.53880134e-06
SiO2[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
TiO2[S1]	1.43398268e+02	6.23366440e-01 6.00015342e-01
V2O5[S]	1.08225108e+00	4.70465238e-03 1.98848527e-03
=====		
=====		
MaterialPackage		
=====		
Material	Reductant	
Mass	6.99615200e+01	kg
Amount	5.35762740e-01	kmol
Pressure	1.00000000e+00	atm
Temperature	2.50000000e+01	°C
Enthalpy	-1.25293265e+02	kWh

Compound Details		
Formula	Mass	Mass Fraction Mole Fraction

C[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
H2[G]	0.00000000e+00	0.00000000e+00 0.00000000e+00
O2[G]	0.00000000e+00	0.00000000e+00 0.00000000e+00
N2[G]	0.00000000e+00	0.00000000e+00 0.00000000e+00
S[S1]	0.00000000e+00	0.00000000e+00 0.00000000e+00
Al2O3[S1]	3.48725349e+00	4.98453077e-02 6.38374820e-02
CaO[S]	6.61375661e-02	9.45342042e-04 2.20134358e-03
Fe2O3[S1]	6.07263107e+01	8.67995875e-01 7.09792759e-01
MgO[S]	3.12650313e+00	4.46888965e-02 1.44788444e-01
SiO2[S1]	2.55531506e+00	3.65245789e-02 7.93799718e-02
=====		

Multiplying a Package by a Scalar It may sometimes be useful to multiply a package by a scalar.


```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 ilma_package *= 2.0
10 print(ilma_package)

```

This doubles the package mass and enthalpy. Temperature, pressure and composition remain the same, since these are intensive properties.

```

=====
MaterialPackage
=====
Material          Ilmenite
Mass              6.00000000e+02 kg
Amount           7.05634009e+00 kmol
Pressure         1.00000000e+00 atm
Temperature      2.50000000e+01 °C
Enthalpy         -1.37562424e+03 kWh
-----
Compound Details
Formula          Mass          Mass Fraction    Mole Fraction
-----
Al2O3[S1]       6.97450697e+00    1.16241783e-02    9.69390473e-03
CaO[S]          1.32275132e-01    2.20458554e-04    3.34280337e-04
Cr2O3[S]        4.81000481e-02    8.01667468e-05    4.48486990e-05
Fe2O3[S1]       1.21452621e+02    2.02421036e-01    1.07784066e-01
Fe3O4[S1]       0.00000000e+00    0.00000000e+00    0.00000000e+00
FeO[S]          1.67748918e+02    2.79581530e-01    3.30892788e-01
K2O[S]          2.40500241e-02    4.00833734e-05    3.61829148e-05
MgO[S]          6.25300625e+00    1.04216771e-02    2.19865404e-02
MnO[S]          3.24675325e+00    5.41125541e-03    6.48625791e-03
Na2O[S1]        4.20875421e-02    7.01459035e-05    9.62343053e-05
P4O10[S]        6.01250601e-03    1.00208434e-05    3.00142421e-06
SiO2[S1]        5.11063011e+00    8.51771685e-03    1.20540764e-02
TiO2[S1]        2.86796537e+02    4.77994228e-01    5.08901291e-01
V2O5[S]         2.16450216e+00    3.60750361e-03    1.68652807e-03
=====

```

Setting Package Temperature Using the “T” property of a MaterialPackage object, it is easy to set the temperature of a package to a new value.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")

```

```

5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8
9 ilma_package.T = 1000.0
10 print(ilma_package)

```

This results in the temperature to be updated, as well as the package's enthalpy.

```

=====
MaterialPackage
=====
Material          Ilmenite
Mass              3.00000000e+02 kg
Amount           3.52817004e+00 kmol
Pressure          1.00000000e+00 atm
Temperature       1.00000000e+03 °C
Enthalpy          -6.18986580e+02 kWh
-----
Compound Details
Formula           Mass              Mass Fraction    Mole Fraction
-----
Al2O3[S1]         3.48725349e+00    1.16241783e-02    9.69390473e-03
CaO[S]            6.61375661e-02    2.20458554e-04    3.34280337e-04
Cr2O3[S]          2.40500241e-02    8.01667468e-05    4.48486990e-05
Fe2O3[S1]         6.07263107e+01    2.02421036e-01    1.07784066e-01
Fe3O4[S1]         0.00000000e+00    0.00000000e+00    0.00000000e+00
FeO[S]            8.38744589e+01    2.79581530e-01    3.30892788e-01
K2O[S]            1.20250120e-02    4.00833734e-05    3.61829148e-05
MgO[S]            3.12650313e+00    1.04216771e-02    2.19865404e-02
MnO[S]            1.62337662e+00    5.41125541e-03    6.48625791e-03
Na2O[S1]          2.10437710e-02    7.01459035e-05    9.62343053e-05
P4O10[S]          3.00625301e-03    1.00208434e-05    3.00142421e-06
SiO2[S1]          2.55531506e+00    8.51771685e-03    1.20540764e-02
TiO2[S1]          1.43398268e+02    4.77994228e-01    5.08901291e-01
V2O5[S]           1.08225108e+00    3.60750361e-03    1.68652807e-03
=====

```

Setting Package Enthalpy We can use the “H” property of a MaterialPackage object to add or subtract enthalpy, or to set it to a new value. This is very useful when building an energy balance.

```

1 from auxi.modelling.process.materials.thermo import Material
2
3 ilmenite = Material("Ilmenite", "./materials/ilmenite.txt")
4 reductant = Material("Reductant", "./materials/reductant.txt")
5 mix = Material("Mix", "./materials/mix.txt")
6
7 ilma_package = ilmenite.create_package("IlmeniteA", 300.0, 1.0, 25.0)
8

```

```

9  ilma_package.H = ilma_package.H + 1.0
10 print(ilma_package)

```

This updates the package's enthalpy and automatically re-calculates its temperature.

```

=====
MaterialPackage
=====
Material          Ilmenite
Mass              3.00000000e+02 kg
Amount           3.52817004e+00 kmol
Pressure         1.00000000e+00 atm
Temperature      4.22166385e+01 °C
Enthalpy         -6.86812118e+02 kWh
-----
Compound Details
Formula          Mass          Mass Fraction    Mole Fraction
-----
Al2O3[S1]       3.48725349e+00    1.16241783e-02    9.69390473e-03
CaO[S]          6.61375661e-02    2.20458554e-04    3.34280337e-04
Cr2O3[S]        2.40500241e-02    8.01667468e-05    4.48486990e-05
Fe2O3[S1]       6.07263107e+01    2.02421036e-01    1.07784066e-01
Fe3O4[S1]       0.00000000e+00    0.00000000e+00    0.00000000e+00
FeO[S]          8.38744589e+01    2.79581530e-01    3.30892788e-01
K2O[S]          1.20250120e-02    4.00833734e-05    3.61829148e-05
MgO[S]          3.12650313e+00    1.04216771e-02    2.19865404e-02
MnO[S]          1.62337662e+00    5.41125541e-03    6.48625791e-03
Na2O[S1]        2.10437710e-02    7.01459035e-05    9.62343053e-05
P4O10[S]        3.00625301e-03    1.00208434e-05    3.00142421e-06
SiO2[S1]        2.55531506e+00    8.51771685e-03    1.20540764e-02
TiO2[S1]        1.43398268e+02    4.77994228e-01    5.08901291e-01
V2O5[S]         1.08225108e+00    3.60750361e-03    1.68652807e-03
=====

```

3.1.2 Business Modelling

The purpose of this section is to explain a number of concepts and demonstrate the use of the Entity, Component, Activity classes in the auxi.modelling.business module.

Basic Activity

A basic activity periodically create a transaction between two specified accounts.

To create an basic activity, import the 'BasicActivity' and the create a 'BasicActivity'

```

from auxi.modelling.business.basic import BasicActivity

basic_activity = BasicActivity("NameA",
                               description="DescriptionA",

```

```
dt_account="Bank\Default",  
cr_account="Sales\Default",  
amount=5000,  
start=datetime(2016, 2, 1),  
end=datetime(2017, 2, 1),  
interval=3)
```

3.2 Tools

This package contains modules, functions and classes used to provide tools to aid the engineer with calculations.

3.2.1 Chemistry Tools

The `auxi.tools.chemistry` package contains modules, functions and classes for doing chemical calculations. Specifically, the package contains a module called “stoichiometry” for doing stoichiometry calculations, and another called “thermochemistry” for doing thermochemical calculations.

3.2.2 Chemical Calculations

The `auxi.tools.chemistry` sub-package provides you with modules, classes and functions to do chemical calculations. The calculations are divided into the categories of stoichiometry and thermochemistry.

Stoichiometry Calculations

Warning

`auxi.tools.chemistry.stoichiometry` is not yet able to successfully parse compound formulae that contain parentheses. It is therefore suggested that a formula such as “Fe2(SO4)3” rather be expressed as “Fe2S3O12”.

Calculating Molar Mass

Determining the molar mass of a substance is done countless times in mass and energy balance models and other process models. It usually requires you to create your own little database or list that you look up the values from. Once you have that, you can perform the required calculations. The `auxi.tools.chemistry.stoichiometry` module provides the `molar_mass()` function for this purpose.

Standard Approach The normal way of getting the molar mass of one or more compounds is as follows:

```
from auxi.tools.chemistry import stoichiometry

molarmass_FeO = stoichiometry.molar_mass("FeO")
molarmass_CO2 = stoichiometry.molar_mass("CO2")
molarmass_FeCr2O4 = stoichiometry.molar_mass("FeCr2O4")

print("Molar mass of FeO      :", molarmass_FeO, "kg/kmol")
print("Molar mass of CO2      :", molarmass_CO2, "kg/kmol")
print("Molar mass of FeCr2O4:", molarmass_FeCr2O4, "kg/kmol")
```

The result of this should be:

```
Molar mass of FeO      : 71.8444 kg/kmol
Molar mass of CO2      : 44.0095 kg/kmol
Molar mass of FeCr2O4: 223.8348 kg/kmol
```

Compact Approach One of Python's strengths is its ability to make code very compact. You may not always want to use a lengthy "stoichiometry.molar_mass" reference to the function. Rather than importing the stoichiometry module, we can import the molar_mass() function directly, and give it another name. Here is how you can make it short and sweet:

```
from auxi.tools.chemistry.stoichiometry import molar_mass as mm

molarmass_FeO = mm("FeO")

print("Molar mass of FeO      :", molarmass_FeO, "kg/kmol")
print("Molar mass of CO2      :", mm("CO2"), "kg/kmol")
print("Molar mass of FeCr2O4:", mm("FeCr2O4"), "kg/kmol")
```

The result is still the same:

```
Molar mass of FeO      : 71.8444 kg/kmol
Molar mass of CO2      : 44.0095 kg/kmol
Molar mass of FeCr2O4: 223.8348 kg/kmol
```

More Examples Here are some more examples of molar mass calculations:

```
from auxi.tools.chemistry.stoichiometry import molar_mass as mm

def print_molar_mass(compound):
    print("Molar mass of", compound, "is", mm(compound), "kg/kmol.")

print_molar_mass("FeO1.5")
print_molar_mass("Fe2O3")
print_molar_mass("FeOTiO2")
print_molar_mass("FeTiO3")
print_molar_mass("Fe2(CO3)3")
```

```
print_molar_mass("Fe2C3O9")
print_molar_mass("H2O")
print_molar_mass("H")
print_molar_mass("He")
print_molar_mass("Au")
```

And the results are:

```
Molar mass of FeO1.5 is 79.8441 kg/kmol.
Molar mass of Fe2O3 is 159.6882 kg/kmol.
Molar mass of FeOTiO2 is 151.7102 kg/kmol.
Molar mass of FeTiO3 is 151.7102 kg/kmol.
Molar mass of Fe2(CO3)3 is 291.7167 kg/kmol.
Molar mass of Fe2C3O9 is 291.7167 kg/kmol.
Molar mass of H2O is 18.01528 kg/kmol.
Molar mass of H is 1.00794 kg/kmol.
Molar mass of He is 4.002602 kg/kmol.
Molar mass of Au is 196.96655 kg/kmol.
```

Calculating Compound Amount

Sometimes you need to convert the mass of a compound (kg) to the equivalent amount (kmol). The stoichiometry module provides the amount () function to do this.

The amount is calculated as follows:

$$n_{\text{compound}} = \frac{m_{\text{compound}}}{mm_{\text{compound}}}$$

where

- n_{compound} is the compound amount in kmol.
- m_{compound} is the compound mass in kg.
- mm_{compound} is the compound molar mass in kg/kmol.

Standard Approach The normal way of calculating the amount of a compound is as follows:

```
from auxi.tools.chemistry import stoichiometry

m_FeO = 10.0
n_FeO = stoichiometry.amount("FeO", m_FeO)
print("There is", n_FeO, "kmol of FeO in", m_FeO , "kg of the compound.")

m_CO2 = 12.3
n_CO2 = stoichiometry.amount("CO2", m_CO2)
print("There is", n_CO2, "kmol of CO2 in", m_CO2 , "kg of the compound.")

m_FeCr2O4 = 453.0
n_FeCr2O4 = stoichiometry.amount("FeCr2O4", m_FeCr2O4)
```

```
print("There is", n_FeCr2O4, "kmol of FeCr2O4 in",
      m_FeCr2O4 , "kg of the compound.")
```

The result of this should be:

```
There is 0.1391896932815919 kmol of FeO in 10.0 kg of the compound.
There is 0.2794851111691794 kmol of CO2 in 12.3 kg of the compound.
There is 2.0238139913900786 kmol of FeCr2O4 in 453.0 kg of the compound.
```

Compact Approach To make the code more compact, we can import the function instead of the module and get the same result like this:

```
from auxi.tools.chemistry.stoichiometry import amount

m_FeO = 10.0
n_FeO = amount("FeO", m_FeO)
print("There is", n_FeO, "kmol of FeO in", m_FeO , "kg of the compound.")

m_CO2 = 12.3
n_CO2 = amount("CO2", m_CO2)
print("There is", n_CO2, "kmol of CO2 in", m_CO2 , "kg of the compound.")

m_FeCr2O4 = 453.0
n_FeCr2O4 = amount("FeCr2O4", m_FeCr2O4)
print("There is", n_FeCr2O4, "kmol of FeCr2O4 in",
      m_FeCr2O4 , "kg of the compound.")
```

The result is still the same:

```
There is 0.1391896932815919 kmol of FeO in 10.0 kg of the compound.
There is 0.2794851111691794 kmol of CO2 in 12.3 kg of the compound.
There is 2.0238139913900786 kmol of FeCr2O4 in 453.0 kg of the compound.
```

Calculating Compound Mass

You often have the amount (kmol) of a compound and then need to calculate its mass. The `stoichiometry` module provides the `mass()` function for this. The mass is calculate with this formula:

$$m_{\text{compound}} = n_{\text{compound}} \cdot mm_{\text{compound}}$$

where

- m_{compound} is the compound mass in kg.
- n_{compound} is the compound amount in kmol.
- mm_{compound} is the compound molar mass in kg/kmol.

From this point forward the standard and compact approaches are not both demonstrated. Only the standard method, which imports the module, is used below since it is more explicit:

```
from auxi.tools.chemistry import stoichiometry

n_FeO = 10.0
m_FeO = stoichiometry.mass("FeO", n_FeO)
print("There is", m_FeO, "kg of FeO in", n_FeO , "kmol of the compound.")

m_CO2 = 12.3
n_CO2 = stoichiometry.mass("CO2", m_CO2)
print("There is", m_CO2, "kg of CO2 in", n_CO2 , "kmol of the compound.")

m_FeCr2O4 = 453.0
n_FeCr2O4 = stoichiometry.mass("FeCr2O4", m_FeCr2O4)
print("There is", m_FeCr2O4, "kg of FeCr2O4 in",
      n_FeCr2O4 , "kmol of the compound.")
```

The results are:

```
There is 718.444 kg of FeO in 10.0 kmol of the compound.
There is 12.3 kg of CO2 in 541.31685 kmol of the compound.
There is 453.0 kg of FeCr2O4 in 101397.1644 kmol of the compound.
```

Identifying Elements in Compounds

The list of elements present in one or more compounds can be used when calculating element balances. Determining this list is often done manually. `stoichiometry` has the `elements()` function to automate this task. This is how you use it:

```
from auxi.tools.chemistry import stoichiometry

elements_Fe2O3 = stoichiometry.elements(["Fe2O3"])
print("Fe2O3 contains these elements:", elements_Fe2O3)

elements_CO2 = stoichiometry.elements(["CO2"])
print("CO2 contains these elements:", elements_CO2)

elements_Fe2Cr2O4 = stoichiometry.elements(["Fe2Cr2O4"])
print("Fe2Cr2O4 contains these elements:", elements_Fe2Cr2O4)

elements_Al2S3O12 = stoichiometry.elements(["Al2(SO4)3"])
print("Al2(SO4)3 contains these elements:", elements_Al2S3O12)

elements_all = stoichiometry.elements(["Fe2O3", "CO2", "Fe2Cr2O4", "Al2(SO4)3"])
print("Fe2O3, CO2, Fe2Cr2O4 and Al2(SO4)3 contain these elements:",
      elements_all)
```

Here are the results:

```
Fe2O3 contains these elements: {'Fe', 'O'}
CO2 contains these elements: {'O', 'C'}
Fe2Cr2O4 contains these elements: {'Fe', 'O', 'Cr'}
```



```
Al2(SO4)3 contains these elements: {'Al', 'O', 'S'}
Fe2O3, CO2, Fe2Cr2O4 and Al2(SO4)3 contain these elements:
{'Al', 'Fe', 'O', 'C', 'S', 'Cr'}
```

Calculating Stoichiometry Coefficients

The `stoichiometry_coefficient()` and `stoichiometry_coefficients()` functions in `stoichiometry` determine the stoichiometry coefficients of elements in chemical compounds automatically. If we are only interested in the coefficient for a single element, we use `stoichiometry_coefficient()` like this:

```
from auxi.tools.chemistry import stoichiometry

coeff_Fe2O3_Fe = stoichiometry.stoichiometry_coefficient("Fe2O3", "Fe")
print("Stoichiometry coefficient of Fe in Fe2O3:", coeff_Fe2O3_Fe)

coeff_Fe2O3_O = stoichiometry.stoichiometry_coefficient("Fe2O3", "O")
print("Stoichiometry coefficient of O in Fe2O3:", coeff_Fe2O3_O)

coeff_Fe2O3_C = stoichiometry.stoichiometry_coefficient("Fe2O3", "C")
print("Stoichiometry coefficient of C in Fe2O3:", coeff_Fe2O3_C)
```

The results are:

```
Stoichiometry coefficient of Fe in Fe2O3: 2.0
Stoichiometry coefficient of O in Fe2O3: 3.0
Stoichiometry coefficient of C in Fe2O3: 0.0
```

We can determine the coefficients for a list of elements using the `stoichiometry_coefficients()` function:

```
from auxi.tools.chemistry import stoichiometry

elements = ["Fe", "O", "C", "Ar"]
st_Fe2O3 = stoichiometry.stoichiometry_coefficients("Fe2O3", elements)
print("Stoichiometry coefficient of", elements, "in Fe2O3:",
      st_Fe2O3)

elements = ["Al", "Ca", "Fe", "Si", "O", "C", "H"]
st_Lawsonite = stoichiometry.stoichiometry_coefficients("CaAl2Si2O7(OH)2·H2O",
                                                         elements)
print("Stoichiometry coefficient of", elements,
      "in Lawsonite (CaAl2(Si2O7)(OH)2·H2O):", st_Lawsonite)
```

This produces these results:

```
Stoichiometry coefficient of ['Fe', 'O', 'C', 'Ar'] in Fe2O3:
[2.0, 3.0, 0.0, 0.0]
Stoichiometry coefficient of ['Al', 'Ca', 'Fe', 'Si', 'O', 'C', 'H']
in Lawsonite (CaAl2(Si2O7)(OH)2·H2O):
[2.0, 1.0, 0.0, 2.0, 10.0, 0.0, 4.0]
```

Calculating Element Mass Fractions

Another two useful tools in the calculation of element balances are the `element_mass_fraction()` and `element_mass_fractions()` functions in `stoichiometry`. They are similar to the stoichiometry coefficient functions, but calculate the mass fraction of an element or list of elements in a chemical compound. The calculations are done with the following equation:

$$y_{\text{compound,element}} = \frac{n_{\text{compound,element}} \cdot mm_{\text{element}}}{mm_{\text{compound}}}$$

where

- $y_{\text{compound,element}}$ is the mass fraction of the specified element in the compound.
- $n_{\text{compound,element}}$ is the stoichiometry coefficient of the specified element in the compound.
- mm_{element} is the element's molar mass in kg/kmol.
- mm_{compound} is the compound's molar mass in kg/kmol.

For determining the mass fraction of a single element we can use `element_mass_fraction()` as follows:

```
from auxi.tools.chemistry import stoichiometry

y_Fe2O3_Fe = stoichiometry.element_mass_fraction("Fe2O3", "Fe")
print("Mass fraction of Fe in Fe2O3:", y_Fe2O3_Fe)

y_Fe2O3_O = stoichiometry.element_mass_fraction("Fe2O3", "O")
print("Mass fraction of O in Fe2O3:", y_Fe2O3_O)

y_Fe2O3_C = stoichiometry.element_mass_fraction("Fe2O3", "C")
print("Mass fraction of C in Fe2O3:", y_Fe2O3_C)
```

This produces these results:

```
Mass fraction of Fe in Fe2O3: 0.699425505453753
Mass fraction of O in Fe2O3: 0.300574494546247
Mass fraction of C in Fe2O3: 0.0
```

Similarly, we can use `element_mass_fractions()` to perform the calculation for a list of elements:

```
from auxi.tools.chemistry import stoichiometry

elements = ["Fe", "O", "C", "Ar"]
y_Fe2O3 = stoichiometry.element_mass_fractions("Fe2O3", elements)
print("Mass fractions of", elements, "in Fe2O3:", y_Fe2O3)

elements = ["Al", "Ca", "Fe", "Si", "O", "C", "H"]
y_Lawsonite = stoichiometry.element_mass_fractions("CaAl2Si2O7O2H2H2O", elements)
print("Mass fractions of", elements,
      "in Lawsonite (CaAl2(Si2O7)(OH)2·H2O):",
      y_Lawsonite)
```

This results in:

```
Mass fractions of ['Fe', 'O', 'C', 'Ar'] in Fe2O3:
[ 0.69942551  0.30057449  0.0  0.0 ]
Mass fractions of ['Al', 'Ca', 'Fe', 'Si', 'O', 'C', 'H']
in Lawsonite (CaAl2(Si2O7)(OH)2·H2O):
[ 0.17172686  0.12754034  0.0  0.17875314  0.50914938  0.0  0.01283028 ]
```

Converting Compounds

Sometimes it is needed to convert the mass of one compound to an equivalent mass of another compound. For example, how much Fe will I get when I reduce a certain mass of Fe2O3? stoichiometry has the `convert_compound()` function to help out. The function calculates the result as follows:

$$m_{\text{target}} = m_{\text{source}} \cdot \frac{y_{\text{source,element}}}{y_{\text{target,element}}}$$

where

- m_{target} is the target compound mass in kg.
- m_{source} is the source compound mass in kg.
- $y_{\text{target,element}}$ is the mass fraction of the specified base element in the target compound.
- $y_{\text{source,element}}$ is the mass fraction of the specified base element in the source compound.

Here are some simple examples of how to use `convert_compound()`:

```
from auxi.tools.chemistry import stoichiometry

m_Fe2O3 = 10.0
m_Fe = stoichiometry.convert_compound(m_Fe2O3, "Fe2O3", "Fe", "Fe")
print("From", m_Fe2O3, "kg of Fe2O3,", m_Fe ,
      "kg of Fe can be produced.")

m_Fe = 10.0
m_Fe2O3 = stoichiometry.convert_compound(m_Fe, "Fe", "Fe2O3", "Fe")
print("When", m_Fe, "kg of Fe is oxidised completely,", m_Fe2O3 ,
      "kg of Fe2O3 will be produced.")
```

The results are:

```
From 10.0 kg of Fe2O3, 6.994255054537531 kg of Fe can be produced.
When 10.0 kg of Fe is oxidised completely, 14.297448294386246 kg of
Fe2O3 will be produced.
```

Thermochemical Calculations

Preparing Thermochemical Data

The `auxi.tools.chemistry.thermochemistry` module provides a number of useful functions for doing thermochemical calculations that would otherwise have been quite cumbersome to do. To make these calculations possible, some thermochemical data is needed. The `auxi` distribution package currently contains data for around 80 compounds. This may, however, not be sufficient for your process calculations. FactSage data can be converted into auxi thermochemical data using the `convert_fact_file_to_auxi_thermo_file()` function.

To prepare your own compound data files with FactSage, follow these steps:

- Open FactSage.
- Click on the “View Data” button.
- Select the “Compound” option, NOT “Solution”.
- Select the database that you want to use. “FactPS” should be OK.
- Type the formula of the compound you need in the box at the bottom.
- Click OK.
- Click on the “Cp(T)” tab.
- Select “File” from the menu and then “Save As ...”.
- Select the folder where you want to store all your thermochemical data files.
- The filename must have a specific format. Taking “Ar” as an example, use “Compound_Ar.txt” for the file name.
- Click “Save”.

You will have to repeat this procedure for all the compounds that you need to include in your calculations.

To convert the factsage file to an auxi thermochemical file use the following code:

```
from auxi.tools.chemistry import thermochemistry as thermo
thermo.convert_fact_file_to_auxi_thermo_file("path/to/factsage_file", "path/to/n
```

Loading Thermochemical Data

If you are going to use the default set of data provided with `auxi`, you do not need to do anything. The entire data set will be available by default. You can obtain a list of all the compounds and their phases by using the following code:

```
from auxi.tools.chemistry import thermochemistry as thermo

thermo.list_compounds()
```

Here are the first few lines of the result:

```
Compounds currently loaded in the thermochemistry module:
Ag ['G', 'L', 'S']
Ag2O ['S']
Al ['G', 'L', 'S']
Al2O3 ['G', 'L', 'S1', 'S2', 'S3', 'S4']
Al4C3 ['S1']
C ['G', 'S1', 'S2']
C2H2 ['G']
CH4 ['Aq', 'G']
CO ['G']
CO2 ['G']
...
```

The result lists all the compounds with the phases for which data are available. Taking the compound SiO₂ as an example, data are available for eight solid phases (S1 to S8), for the liquid phase and for the gas phase.

If you have decided to create your own data folder, you can force `auxi` to use the data in that folder. Here is the code for this:

```
from auxi.tools.chemistry import thermochemistry as thermo

thermo.load_data('/home/someuser/thermodata')
thermo.list_compounds()
```

This example data folder only contains a small selection of files:

```
Compounds currently loaded in the thermo module:
Ag ['G', 'L', 'S']
CaO ['G', 'L', 'S']
Cr2O3 ['L', 'S']
Cu ['G', 'L', 'S']
CuO ['G', 'S']
```

Calculating Heat Capacity

The `Cp()` function in the `auxi.tools.chemistry.thermochemistry` module can be used to calculate the heat capacity at constant pressure for a compound. This can be done as follows:

```
from auxi.tools.chemistry import thermochemistry as thermo

Cp_H2O = thermo.Cp("H2O[L]", 70.0)
print("The Cp of 1 kg of water at 70 °C is", Cp_H2O, "kWh/K.")
```

```
Cp_H2O = thermo.Cp("H2O[G]", 70.0)
print("The Cp of 1 kg of water vapour at 70 °C is", Cp_H2O, "kWh/K.")

m_ZrO2 = 2.34
Cp_ZrO2 = thermo.Cp("ZrO2[S1]", 893.5, m_ZrO2)
print("The Cp of 2.34 kg of ZrO2[S1] at 893.5 °C is", Cp_ZrO2, "kWh/K.")
```

Here are the results:

```
The Cp of 1 kg of water at 70 °C is 0.0011634065724223574 kWh/K.
The Cp of 1 kg of water vapour at 70 °C is 0.0005217114220395267 kWh/K.
The Cp of 2.34 kg of ZrO2[S1] at 70 °C is 0.0004084615851157184 kWh/K.
```

The first parameter to the function must specify both the compound's formula and phase. If the phase is not specified it is impossible to calculate a result. The heat capacity of water is clearly significantly different from that of water vapour.

The last parameter of the `Cp()` is mass and it is optional. If no value is specified, it is taken to be 1 kg. This was the case for the first two calculations above. A mass of 2.34 kg was specified in the last `Cp` calculation.

Calculating Enthalpy

The `H()` function in `thermochemistry` is used to calculate the enthalpy of a compound. This can be done as follows:

```
from auxi.tools.chemistry import thermochemistry as thermo

H_H2O = thermo.H("H2O[L]", 70.0)
print("The enthalpy of 1 kg of water at 70 °C is", H_H2O, "kWh.")

H_H2O = thermo.H("H2O[G]", 70.0)
print("The enthalpy of 1 kg of water vapour at 70 °C is", H_H2O, "kWh.")

m_ZrO2 = 2.34
H_ZrO2 = thermo.H("ZrO2[S1]", 893.5, m_ZrO2)
print("The enthalpy of 2.34 kg of ZrO2[S1] at 893.5 °C is", H_ZrO2, "kWh.")
```

Here are the results:

```
The enthalpy of 1 kg of water at 70 °C is -4.35495670039936 kWh.
The enthalpy of 1 kg of water vapour at 70 °C is -3.7054553712406264 kWh.
The enthalpy of 2.34 kg of ZrO2[S1] at 893.5 °C is -5.463105585819936 kWh.
```

The parameters to the `H()` function works the same as that of the `Cp()` function. Both formula and phase are required in the first parameter, the second is temperature in °C and the third is mass, which is optional with a default value of 1 kg.

Calculating Entropy

The `S()` function in `thermochemistry` is used to calculate the entropy of a compound. This can be done as follows:

```
from auxi.tools.chemistry import thermochemistry as thermo

S_H2O = thermo.S("H2O[L]", 70.0)
print("The entropy of 1 kg of water at 70 °C is", S_H2O, "kWh/K.")

S_H2O = thermo.S("H2O[G]", 70.0)
print("The entropy of 1 kg of water vapour at 70 °C is", S_H2O, "kWh/K.")

m_ZrO2 = 2.34
S_ZrO2 = thermo.S("ZrO2[S1]", 893.5, m_ZrO2)
print("The entropy of 2.34 kg of ZrO2[S1] at 893.5 °C is", S_ZrO2, "kWh/K.")
```

Here are the results:

```
The entropy of 1 kg of water at 70 °C is 0.0012418035680941087 kWh/K.
The entropy of 1 kg of water vapour at 70 °C is 0.0029829908763826032 kWh/K.
The entropy of 2.34 kg of ZrO2[S1] at 893.5 °C is 0.000762164298048799 kWh/K.
```

The parameters to the `S()` function works the same as that of the `Cp()` function. Both formula and phase are required in the first parameter, the second is temperature in °C and the third is mass, which is optional with a default value of 1 kg.

Calculating Gibbs Free Energy

The `G()` function in `thermochemistry` is used to calculate the Gibbs free energy of a compound. This can be done as follows:

```
from auxi.tools.chemistry import thermochemistry as thermo

G_H2O = thermo.G("H2O[L]", 70.0)
print("The Gibbs free energy of 1 kg of water at 70 °C is", G_H2O,
      "kWh.")

G_H2O = thermo.G("H2O[G]", 70.0)
print("The Gibbs free energy of 1 kg of water vapour at 70 °C is", G_H2O,
      "kWh.")

m_ZrO2 = 2.34
G_ZrO2 = thermo.G("ZrO2[S1]", 893.5, m_ZrO2)
print("The Gibbs free energy of 2.34 kg of ZrO2[S1] at 893.5 °C is", G_ZrO2,
      "kWh.")
```

Here are the results:

```
The Gibbs free energy of 1 kg of water at 70 °C is
-4.781081594790853 kWh.
The Gibbs free energy of 1 kg of water vapour at 70 °C is
-4.729068690471317 kWh.
The Gibbs free energy of 2.34 kg of ZrO2[S1] at 893.5 °C is
-6.352284564138569 kWh.
```

The parameters to the `G()` function works the same as that of the `Cp()` function. Both formula and phase are required in the first parameter, the second is temperature in °C and the third is mass, which is optional with a default value of 1 kg.

auxi Reference

4.1 auxi package

4.1.1 auxi.modelling package

auxi.modelling.process package

auxi.modelling.process.materials package

chemistry material module

psd material module

psd slurry material module

thermochemistry material module

auxi.modelling.financial package

auxi.modelling.financial.des module

AccountType enum

GeneralLedgerAccount class

Transaction class

TransactionTemplate class

GeneralLedgerStructure class

GeneralLedger class

auxi.modelling.financial.reporting module

ReportType enum

Report class

GeneralLedgerStructure class

TransactionList class

BalanceSheet class

IncomeStatement class

auxi.modelling.business package

auxi.modelling.business.basic module

BasicActivity class

BasicLoanActivity class

auxi.modelling.business.models module

TimeBasedModel class

auxi.modelling.business.structure module

Activity class

Component class

Entity class

4.1.2 auxi tools package

auxi tools chemistry package

auxi tools chemistry stoichiometry module

Element class

auxi tools chemistry thermochemistry module

CpRecord class

Phase class

Compound class

Indices and tables

- `genindex`
- `modindex`
- `search`