

---

# StoichTools: Tools for Doing Stoichiometry

## Table of Contents

What is StoichTools? .....	1
Parsing Chemical Formulas .....	3
Additional Parsing Examples .....	3
Chemical Abbreviations and Isotopes .....	4
Non-stoichiometric Formulas .....	4
From Atoms to Chemical Formulas .....	5
Hill Notation & Canonical Representations .....	5
Molecular Weight .....	6
Creating Molecular Weight Tables .....	6
Atomic Matrix .....	6
Atomic Matrix for Ionic Species .....	7
Balancing a Reaction .....	7
Stoichiometric Matrix .....	7
Multiple Independent Reactions .....	8
Further Examples of Complex Reactions .....	8
Chemical Equations with Ionic Charges .....	9
Chemical Half Equations .....	10
Nested Formulas .....	11
Version History .....	12

StoichTools comprises a set of Matlab functions for doing stoichiometric analysis. These functions parse standard chemical notation for a variety of stoichiometric calculations including finding molecular weights, balancing reactions for atom and charge conservation, finding independent reactions, and displaying formulas in Hill notation. The functions account for both charge and atomic balances so they can be used to balance ionic reactions and chemical half reactions.

StoichTools has extensive documentation including a set of worked homework problems demonstrating use of the functions.

These functions were developed to support introductory courses in Chemical Engineering.

Jeff Kantor  
December 18, 2010

## What is StoichTools?

StoichTools works with two types of data:

1. **Chemical formulas.** Each chemical formula is a string written in a nearly universal chemical notation. For example,  $\text{H}_2\text{SO}_4$  represents Sulfuric Acid. Grouping is allowed (e.g.,  $\text{CH}_3(\text{CH}_2)_6\text{CH}_3$  for octane) with either parentheses '()' or brackets '[]'. Charge is indicated by a trailing + or - followed by an optional number (e.g.,  $\text{Fe}^{+3}$  or  $\text{HSO}_4^-$ ). Phase information may be included as a terminal (aq), (l), (g), or (s). Cell arrays can be used in most places to work with multiple formulas at one time (e.g.,

```
{'H2SO4','H+','SO4-2'}).
```

2. **Atomic representation.** Many calculations require knowledge of the charge, and of number of atoms of each type in a chemical species. This is maintained in a Matlab structure where `r.C`, for example, is the number of carbon atoms. The symbol after the dot is the standard 1 or 2 character symbol for an element. The symbol `Q` is reserved to indicated charge. A Matlab structure array is used to store multiple atomic representations in a single variable.

StoichTools provides functions for the following types of chemical calculations:

### Working with Chemical Formulas

- `r = parse_formula(s)` processes a chemical formula to produce an atomic representation. This function is mainly used by other functions to process chemical formulas.
- `hillformula` processes a chemical formula or atomic representation to produce a chemical formula in standard Hill notation. The Hill notation widely used to represent species in chemical databases, such as the NIST Chemistry Webbook.

### Calculating Molecular Weights

- `mw = molweight(s)` computes the molecular weights of chemical compounds. Input can be a chemical formula, a cell array of chemical formulas, or an array of atomic representations. If no output is indicated, then a table of molecular weights is printed.

### Stoichiometry

- `[A,atoms,species] = atomic(s)` constructs the atomic matrix for a set of chemical compounds. Element `A(i,j)` is the number of `atoms{i}` in `species{j}`. Inputs may be chemical formula, a cell array of chemical formulas, If there are ionic species, then a special atom 'Q' is indicates the charge of the species. If no output is indicated, then the atomic matrix is displayed in tabular form.
- `V = stoich(s)` computes the stoichiometric matrix for a set of chemical compounds. The input is a cell array of chemical formulas, or an array of atomic representations. The columns of `V` correspond to independent chemical reactions satisfying atomic and charge balances. Element `V(j,k)` is the stoichiometric coefficient for species `j` in reaction `k`. A negative value denotes a reactant, a positive value denotes a product. If no output is indicated, then `disp_reaction` is used to display all independent reactions.
- `Vout = disp_reaction(V,s)` If no output is indicated, then format and displays the chemical reactions denoted by stoichiometric matrix `V` and the array of species `s`. The species may be cell array of formulas or an array of atomic representations. If feasible, the coefficients are scaled to integers. If integer coefficients are too long, then either rational or floating point coefficients are displayed. If an output is indicated, then `Vout` is a stoichiometric matrix with rescaled coefficients, and the reactions are not displayed.

### Homework Problems with Solutions

The StoichTools folder includes a number of worked homework problems. These are Matlab scripts with titles in the pattern `HW_xx.m`. Each script begins with a cell containing the problem statement. Subsequent cells demonstrate solution to the problem. The homework files can be viewed by using the

Matlab publishing function.

## Parsing Chemical Formulas

Given a set of chemical species, `r = parse_formula(s)` parses a cell array of chemical formulas to produce a structure array `r`. The value is the number of atoms of that element present in the corresponding formula. The structure array includes a field for each atomic element in the set of species. We call this the atomic representation of the species.

```
% Parsing methane
parse_formula('CH4')

ans =

    C: 1
    H: 4
```

## Additional Parsing Examples

```
ex{1} = 'NaHCO3';
ex{2} = 'KFe3(SO4)2(OH)6'; % Jorosite
ex{3} = 'KFe3(AsO4)2(HAsO4)2'; % Potassium-Iron-Arsenate
ex{4} = '(CH4)8(H2O)46'; % Methane Clathrate
ex{4} = 'HSO4-(aq)';

for k = 1:length(ex)
    disp(ex{k});
    parse_formula(ex{k})
end

NaHCO3

ans =

    Na: 1
    H: 1
    C: 1
    O: 3

KFe3(SO4)2(OH)6

ans =

    K: 1
    Fe: 3
    S: 2
    O: 14
    H: 6

KFe3(AsO4)2(HAsO4)2

ans =

    K: 1
    Fe: 3
    As: 4
```

```

O: 16
H: 2

HSO4-(aq)

ans =

H: 1
S: 1
O: 4
Q: -1

```

## Chemical Abbreviations and Isotopes

- Formulas may include D (Deuterium) or T (Tritium). These are treated as elements and included as distinct species in any atom balances.
- The common organic chemistry abbreviations Me (Methyl, CH<sub>3</sub>), Et (Ethyl, C<sub>2</sub>H<sub>5</sub>), Bu (Butyl, C<sub>4</sub>H<sub>9</sub>), Ph (Phenol, C<sub>6</sub>H<sub>5</sub>) may be included in formulas. These are replaced by their atomic formulas during the parsing process.
- The symbols M (any metal) and X (any halogen) may be used in formulas. Formulas containing the symbol M or X have unknown molecular weight.

```

parse_formula('D2O')
parse_formula('EtOH')
molweight({'H2O', 'D2O', 'T2O', 'EtOH', 'PhOH', 'TiO2', 'MO2'});

```

```
ans =
```

```

D: 2
O: 1

```

```
ans =
```

```

C: 2
H: 6
O: 1

```

<i>Species</i>	<i>Mol. Wt.</i>
-----	-----
H2O	18.02
D2O	20.03
T2O	22.03
EtOH	46.07
PhOH	94.11
TiO2	79.88
MO2	NaN

## Non-stoichiometric Formulas

Some applications of stoichiometry involve complex chemical compounds not easily described by simple chemical formulas. So-called 'non-stoichiometric' compounds can be also be parsed.

```
bacteria = 'CH1.8N0.24O0.36';  
parse_formula(bacteria);
```

## From Atoms to Chemical Formulas

Given a structure array of atomic representations, `|s = hillformula(r)|` constructs a cell array of corresponding chemical formulas.

```
% Formula for octane
```

```
octane.C = 8;  
octane.H = 18;  
hillformula(octane)
```

```
ans =  
  
    'C8H18'
```

## Hill Notation & Canonical Representations

The Hill notation is a commonly used system for writing chemical formulas in a standard form. `% hillformula(r)` produces a simple canonical representation of a chemical species. Note, however, that there may be many isomers for a given formula.

```
s = { 'Zr3B2', 'HBr', 'HCl', 'CH3(CH2)6CH3', 'NaCO3', 'CaC2', 'CH3OH', ...  
      'CH3COOH', 'HNO3', 'H2SO4', 'NH3', 'SnH4', 'CH3HgCH3', '(CH3CH2)4Pb', ...  
      '[Co(NH3)6]+3', '[B12H12]-2' };  
  
fprintf('\n%-15s %-15s\n-----\n', ...  
        'Formula', 'Hill Notation');  
for k = 1:length(s)  
    fprintf('%-15s %-15s\n', s{k}, char(hillformula(s{k})));  
end
```

<i>Formula</i>	<i>Hill Notation</i>
-----	-----
Zr3B2	B2Zr3
HBr	BrH
HCl	ClH
CH3(CH2)6CH3	C8H18
NaCO3	CNaO3
CaC2	C2Ca
CH3OH	CH4O
CH3COOH	C2H4O2
HNO3	HNO3
H2SO4	H2O4S
NH3	H3N
SnH4	H4Sn
CH3HgCH3	C2H6Hg
(CH3CH2)4Pb	C8H20Pb

```
[Co(NH3)6]+3      CoH18N6+3  
[B12H12]-2       B12H12-2
```

## Molecular Weight

```
mw = molweight(s)  
mw = molweight(r)
```

Given a cell array of chemical formulas, or a structure array of atomic representations, `molweight` computes a corresponding vector of molecular weights.

```
% Molecular Mass of Dimethyl Mercury
```

```
s = 'CH3HgCH3';  
mw = molweight('CH3HgCH3');  
fprintf('Molecular Weight of Dimethyl Mercury (%s) = %g\n',s,mw);
```

```
Molecular Weight of Dimethyl Mercury (CH3HgCH3) = 230.66
```

## Creating Molecular Weight Tables

If `molweight` as no output, then it prints a table of molecular weights.

```
molweight(s);
```

<i>Species</i>	<i>Mol. Wt.</i>
-----	-----
CH3HgCH3	230.66

## Atomic Matrix

```
[A,atoms,species] = atomic(s)  
[A,atoms,species] = atomic(r)
```

Given a cell array of chemical formulas `s`, or a structure array of atomic representations `r`, `atomic` computes the atomic matrix `A`. `atoms` is a cell array of the atomic elements, `species` is a cell array of species. `A(i,j)` is the number of atoms of element `atoms{i}` in species `species{j}`. % When called without an output argument, `atomic` displays the atomic matrix.

```
s = {'CH4','O2','H2O','CO2'};
```

```
atomic(s);  
A = atomic(s);  
disp(' ');  
disp('A = ');  
disp(A);
```

	<i>CH4</i>	<i>O2</i>	<i>H2O</i>	<i>CO2</i>
<i>C:</i>	1	0	0	1

```

H:      4      0      2      0
O:      0      2      1      2

A =
    1      0      0      1
    4      0      2      0
    0      2      1      2
    
```

## Atomic Matrix for Ionic Species

For ionic species an additional row is added, labeled by 'Q', indicating the net charge on each of the species included in the matrix.

```

s = { 'Fe+3', 'SO4-2', 'H+', 'OH-', 'H2O', 'Fe2(SO4)3' };
atomic(s);
    
```

```

      Fe+3    SO4-2    H+    OH-    H2O  Fe2(SO4)3
Fe:      1      0      0      0      0      2
H:      0      0      1      1      2      0
O:      0      4      0      1      1     12
S:      0      1      0      0      0      3
Q:      3     -2      1     -1      0      0
    
```

## Balancing a Reaction

Given a cell array of chemical formulas, or an array of atomic representations, `stoich(s)` computes stoichiometric coefficients that satisfy charge and atom balances. If no output is specified, then balanced reactions are displayed.

```

stoich({ 'NaPb', 'CH3CH2Cl', '(CH3CH2)4Pb', 'NaCl', 'Pb' });
stoich({ 'H+(aq)', 'OH-(aq)', 'H2O(l)' });
    
```

```

4 NaPb + 4 CH3CH2Cl <=> (CH3CH2)4Pb + 4 NaCl + 3 Pb
    
```

```

H+(aq) + OH-(aq) <=> H2O(l)
    
```

## Stoichiometric Matrix

Given a cell array of chemical formulas, or a structure array of atomic representations, `V = stoich(s)` computes the stoichiometric matrix `V`. `V(n,r)` is the stoichiometric coefficient of species `n` in reaction `r`. The atomic and stoichiometric matrices satisfies the relationship  $A \cdot V = 0$ .

```

s = { 'C8H18', 'O2', 'C', 'CO', 'CO2', 'H2O' };
V = stoich(s);
disp('Stoichiometric Matrix V = ');
disp(V);
    
```

```

Stoichiometric Matrix V =
    
```

```
-1    0    0
 0   -1    0
 0    0   -1
25   -2    2
-17    2   -1
 9     0    0
```

## Multiple Independent Reactions

```
V = stoich(s)
disp_reaction(V,s)
```

The columns of the stoichiometric matrix *V* represent independent reactions. The function `disp_reaction(V,s)` displays the reactions in a conventional human readable form.

```
s = {'C8H18', 'O2', 'C', 'CO', 'CO2', 'H2O'};
V = stoich(s);
disp_reaction(V,s);
```

```
C8H18 + 17 CO2 <=> 25 CO + 9 H2O
O2 + 2 CO <=> 2 CO2
C + CO2 <=> 2 CO
```

## Further Examples of Complex Reactions

Examples from <http://www.chemistryhelp.net/chemistry-calculator/chemical-equation-balancer>

```
stoich({'P2I4', 'P4', 'H2O', 'H3PO4', 'PH4I'});

stoich({'[Cr(N2H4CO)6]4[Cr(CN)6]3', 'KMnO4', 'H2SO4', 'K2Cr2O7', ...
        'MnSO4', 'CO2', 'KNO3', 'K2SO4', 'H2O'});

stoich({'Cu(s)', 'HNO3(aq)', 'Cu(NO3)2(aq)', 'NO(g)', 'H2O(l)'});

stoich({'Cu', 'HNO3', 'H2O', 'Cu(NO3)2', 'NO'});

stoich({'KMnO4', 'C3H5(OH)3', 'K2CO3', 'Mn2O3', 'CO2', 'H2O'});

stoich({'K2Cr2O7', 'FeCl2', 'HCl', 'KCl', ...
        'CrCl3', 'FeCl3', 'H2O'});

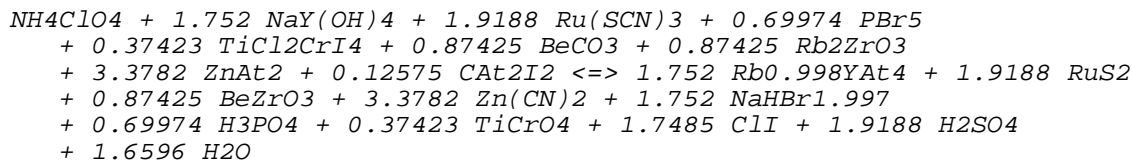
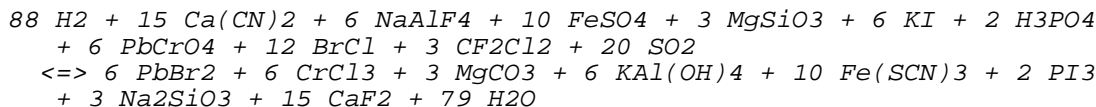
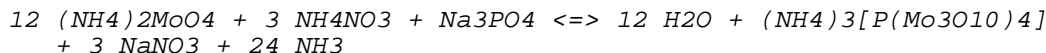
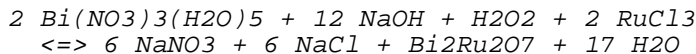
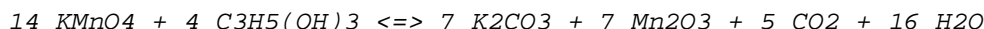
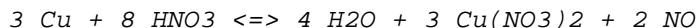
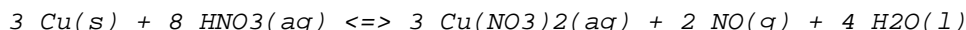
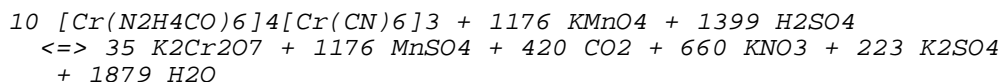
stoich({'Bi(NO3)3(H2O)5', 'NaOH', 'H2O2', 'RuCl3', ...
        'NaNO3', 'NaCl', 'Bi2Ru2O7', 'H2O'});

stoich({'(NH4)2MoO4', 'NH4NO3', 'Na3PO4', 'H2O', ...
        '(NH4)3[P(Mo3O10)4]', 'NaNO3', 'NH3'});

stoich({'H2', 'Ca(CN)2', 'NaAlF4', 'FeSO4', 'MgSiO3', 'KI', 'H3PO4', ...
        'PbCrO4', 'BrCl', 'CF2Cl2', 'SO2', 'PbBr2', 'CrCl3', 'MgCO3', ...
        'KAl(OH)4', 'Fe(SCN)3', 'PI3', 'Na2SiO3', 'CaF2', 'H2O'});

stoich({'NH4ClO4', 'NaY(OH)4', 'Ru(SCN)3', 'PBr5', 'TiCl2CrI4', 'BeCO3', ...
        'Rb2ZrO3', 'ZnAt2', 'Cat2I2', 'Rb0.998YAt4', 'RuS2', 'BeZrO3', 'Zn(CN)2', ...
        'NaHBr1.997', 'H3PO4', 'TiCrO4', 'ClI', 'H2SO4', 'H2O'});
```





## Chemical Equations with Ionic Charges

The charge on ionic species is indicated by + or - followed by an optional digit indicating the amount of charge. If ionic species are present, then a charge balance is included in the computation of the stoichiometric coefficients.

```
stoich({'ClO2+(aq)', 'H3O+(aq)', 'Cl2(g)', 'H2O(l)', 'ClO3-(aq)', 'ClO2(aq)'});
stoich({'Bi+3(aq)', 'HSnO2-(aq)', 'OH-(aq)', 'Bi(s)', 'H2O', 'SnO3-2(aq)'});
stoich({'CH3CH2OH', 'Cr2O7-2', 'H+', 'CH3COOH', 'Cr+3', 'H2O'});
stoich({'I-', 'I2', 'Mn+2', 'MnO4-', 'H+', 'H2O'});
stoich({'Cl2', 'Cl-', 'Fe+2', 'Fe+3'});
stoich({'Mn+2', 'BiO-3', 'H+', 'MnO4-', 'Bi3+', 'H2O'});
stoich({'NpO2+2', 'NpO2(OH)H2C2O4+', 'NpO2+', 'CO2', 'H+', 'O2'});
stoich({'H3PO4', '(NH4)6Mo7O24', 'H+', '(NH4)3PO4(MoO3)12', 'NH4+', 'H2O'});
```

```

4 ClO2+(aq) + Cl2(g) + 4 ClO3-(aq) <=> 10 ClO2(aq)
8 H3O+(aq) + Cl2(g) + 8 ClO3-(aq) <=> 12 H2O(l) + 10 ClO2(aq)

2 Bi+3(aq) + 3 HSnO2-(aq) + 9 OH-(aq) <=> 2 Bi(s) + 6 H2O
+ 3 SnO3-2(aq)

3 CH3CH2OH + 2 Cr2O7-2 + 16 H+ <=> 3 CH3COOH + 4 Cr+3 + 11 H2O

10 I- + 2 MnO4- + 16 H+ <=> 5 I2 + 2 Mn+2 + 8 H2O

Cl2 + 2 Fe+2 <=> 2 Cl- + 2 Fe+3

4 Mn+2 + 5 Bi3+ + 31 H2O <=> 15 BiO-3 + 62 H+ + 4 MnO4-

6 NpO2+2 + 2 NpO2(OH)H2C2O4+ <=> 8 NpO2+ + 4 CO2 + 6 H+ + O2

7 H3PO4 + 12 (NH4)6Mo7O24 + 51 H+ <=> 7 (NH4)3PO4(MoO3)12 + 51 NH4+
+ 36 H2O
    
```

## Chemical Half Equations

Include the bare electron 'e-' to balance chemical half reactions. In acidic solutions, if one of the main reactants contains oxygen, add 'H+' and 'H2O'. In basic solutions, if one of the main reactants contains oxygen then add 'OH-' and 'H2O'.

```

stoich({'Al+3(aq)', 'Al(s)', 'e-'});
stoich({'Cl-(aq)', 'Cl2(g)', 'e-'});

% Acidic Solutions

stoich({'MnO4-(aq)', 'Mn+2(aq)', 'H2O(l)', 'H+(aq)', 'e-'});
stoich({'O2(g)', 'H2O(l)', 'H+(aq)', 'e-'});
stoich({'Ag2O3', 'Ag+', 'H2O', 'H+', 'e-'});
stoich({'S2O3-2(aq)', 'S(s)', 'H2O(l)', 'H+(aq)', 'e-'});
stoich({'HOCCOOH(aq)', 'CO2(g)', 'H2O(l)', 'H+(aq)', 'e-'});

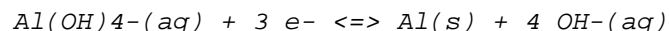
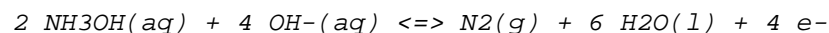
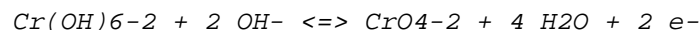
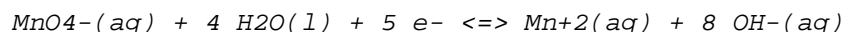
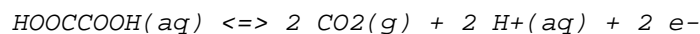
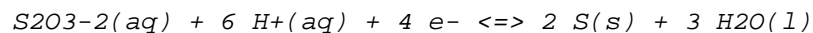
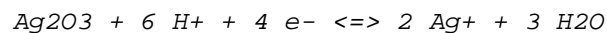
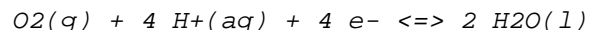
% Alkali Solutions

stoich({'MnO4-(aq)', 'Mn+2(aq)', 'H2O(l)', 'OH-(aq)', 'e-'});
stoich({'Cr(OH)6-2', 'CrO4-2', 'H2O', 'OH-', 'e-'});
stoich({'NH3OH(aq)', 'N2(g)', 'H2O(l)', 'OH-(aq)', 'e-'});
stoich({'Al(OH)4-(aq)', 'Al(s)', 'H2O(l)', 'OH-(aq)', 'e-'});
stoich({'ZrO(OH)2', 'Zr', 'H2O', 'OH-', 'e-'});
    
```

```
Al+3(aq) + 3 e- <=> Al(s)
```

```
2 Cl-(aq) <=> Cl2(g) + 2 e-
```

```
MnO4-(aq) + 8 H+(aq) + 5 e- <=> Mn+2(aq) + 4 H2O(l)
```



## Nested Formulas

Matlab regular expressions capabilities are used to parse chemical formulas. While this keeps StoichTools simple and fast, one of the drawbacks of regular expressions is the difficulty of matching nested expressions. Thus nesting is limited to bracketed expressions inside of parentheses, or parentheses inside of brackets. By this rule,  $[\text{Fe}_2(\text{SO}_4)_3]$  and  $(\text{Fe}_2[\text{SO}_4]_3)$  are allowed, but  $(\text{Fe}_2(\text{SO}_4)_3)$  and  $[\text{Fe}_2[\text{SO}_4]_3]$  are not. In practice, chemical formula rarely need more than two levels of nesting.

```
disp('These work fine.');
```

```
molweight({'[Fe2(SO4)3]', '(Fe2[SO4]3)'});
```

```
fprintf('\n\n');
```

```
try
```

```
    molweight({'(Fe2(SO4)3)', '[Fe2[SO4]3]'})
```

```
catch exception
```

```
    disp('But this does not.');
```

```
    disp(exception.message);
```

```
end
```

These work fine.

Species	Mol. Wt.
-----	-----
$[\text{Fe}_2(\text{SO}_4)_3]$	399.88
$(\text{Fe}_2[\text{SO}_4]_3)$	399.88

But this does not.  
 Could not parse formula:  
 $(\text{Fe}_2(\text{SO}_4)_3)$   
                   ^

## Version History

- 2010/12/18 Submitted to Matlab Central
- 2010/12/19 Updated documentation, added solved homeworks
- 2010/12/19 Put rows of the atomic matrix in Hill order
- 2010/12/19 Expanded regular expression parsing to include phases
- 2010/12/20 Enhanced parser to accept non-stoichiometric formulas
- 2010/12/20 Enhanced disp\_reaction for better coefficient formatting
- 2010/12/21 Parser to include common symbols D, T, Et, Me, Bu, Ph
- 2010/12/30 Fixed all mlint messages, reduced McCabe complexity
- 2010/12/30 Update to Matlab Central
- 2010/12/30 Further improvements to error handling (assert's)
- 2010/12/31 Fixed bug with NaN in molweight
- 2010/12/31 Renamed homework files so it makes more sense on MC
- 2010/12/31 Update to Matlab Central

### To Do's

- Add Generation/Consumption Analysis
- Add Extent of Reaction Analysis
- Include an electrochemistry homework example (battery?)
- Add a display feature for stoich
- Add webbook lookup for chemical property data

*Published with MATLAB® 7.11*