chemmacros v1.1

2011/06/22

Clemens Niederberger

http://www.mychemistry.eu/
contact@mychemistry.eu

'chemmacros' is a collection of macros and commands which are intended to make typesetting chemistry documents with LaTeX 2$_\varepsilon$ faster and more convenient. Coverage includes some nomenclature commands, oxidation numbers, thermodynamic data, newman projections, *etc*.

Contents

1 Licence, Requirements

'chemmacros' v1.1 underlies the The LaTeX project public license (http://www.latex-project.org/lppl.txt).

'chemmacros' internally loads the packages 'amsmath'[1], 'ifthen'[2], 'siunitx'[3], 'xparse'[4] and 'tikz' (Ti*k*Z = 'pgf'[5]) as well as the tikzlibrary `calc`. If they're missing it will cause an error.

'siunitx' needs LaTeX3 support as provided in the 'expl3'[6] and 'xpackages'[7] bundles. 'xparse' is part of the 'xpackages' bundle. This means, that 'chemmacros' also needs LaTeX3 support.

The definition of some commands depends on which packages else have been loaded. Some commands are only defined if a certain package has been loaded. This concerns the packages 'bpchem'[8], 'chemstyle'[9] and 'mhchem'[10].It is mentioned explicitly in the documentation, if a command has a definition depending on one of these packages.

The package option `bpchem` (section 2) needs the package 'bpchem' to be available.

The package option `xspace` (section 2) needs the package 'xspace'[11] to be available.

If the user loads 'mhchem', the packages 'mathtools'[12] and 'environ'[13] are needed.

Please take notice, that the package options have changed with version 1.1.

1) http://www.ctan.org/pkg/amsmath
2) http://www.ctan.org/pkg/ifthen
3) http://www.ctan.org/pkg/siunitx
4) http://www.ctan.org/pkg/xparse
5) http://www.ctan.org/pkg/pgf
6) http://www.ctan.org/pkg/expl3
7) http://www.ctan.org/pkg/xpackages
8) http://www.ctan.org/pkg/bpchem
9) http://www.ctan.org/pkg/chemstyle
10) http://www.ctan.org/pkg/mhchem
11) http://www.ctan.org/pkg/xspace
12) http://www.ctan.org/pkg/mathtools
13) http://www.ctan.org/pkg/environ

2 Package Options

NEW v.1.1

'chemmacros' has four package options:

`bpchem` With this option first the package 'bpchem' is loaded and second the appearance of the \NMR command is changed to match the 'bpchem' commands \HNMR and \CNMR. Without option: ^{1}H-NMR: δ; with option: ^{1}H-NMR: δ;

`circled` Some chemists – like me – prefer circled charge symbols to have a clear distinction between charge and math symbols. In 'chemmacros's default behaviour they're *not* circled (+ und −). With the option `circled` all commands of 'chemmacros' use the circled ones ($\oplus$ und $\ominus$).

`german` This option changes \pKa from pK_A into pK_S. Also the phase identifiers \solid and \liquid are changed from $_{(s)}$ and $_{(l)}$ into $_{(f)}$ and $_{(fl)}$.

`xspace` With this option, the following commands get a \xspace: \Hpl \HtO \water \Hyd \HtO \pH \pOH \pKa \pKb \cis \trans \insitu \abinitio \mech \NMR $\mathrm{H}^{\oplus}$ $\mathrm{H_3O}^{\oplus}$ $\mathrm{H_2O}$ $\mathrm{OH}^{\ominus}$ $\mathrm{H_3O}^{\oplus}$ $p\mathrm{H}$ $p\mathrm{OH}$ pK_A pK_B *cis trans in situ ab initio* $\mathrm{S_N}$ ^{1}H-NMR: δ
The two commands \cis and \trans are also defined by the 'bpchem' package. If you load that package, they are redefined by 'chemmacros'. In the definition of 'bpchem' they *always* have a \xspace, with 'chemmacros' only with option `xspace`. Apart from that they're identical.

3 Particles, Ions and a Symbol

Some simple macros for displaying often needed particles and a symbol. Please note, that they're displayed differently depending on the package options used, see section 2.

- \Hpl $\mathrm{H}^{\oplus}$ (proton)
- \Hyd $\mathrm{OH}^{\ominus}$ (hydroxide)

NEW v.1.1

- \HtO $\mathrm{H_3O}^{\oplus}$ (oxonium) (**H** three **O**)

NEW v.1.1

- \water $\mathrm{H_2O}$
- \el $e^{\ominus}$ (electron)

- `\prt` $p^{\oplus}$ (proton)
- `\ntr` n^{0} (neutron)

NEW v.1.1

- `\Nu` $\mathrm{Nu}^{\ominus}$ (nucleophile)

NEW v.1.1

- `\El` $\mathrm{E}^{\oplus}$ (electrophile)
- `\transitionstatesymbol` ≠ transition state symbol (uses 'Ti$\boldsymbol{k}$Z')

These commands are working both in text mode and math mode. Depending on wether 'mhchem' has been loaded, atoms are defined with the `\cf{}` command or with `\mbox{}`.

NEW v.1.1

There is another command which allows to typeset radicals with charges and subscripts.

- `\R[<sign>]{<subscript>}` *e.g.* `\R[+]{tert}` `\R[-]{sek}` `\R{prim}` $\mathrm{R}^{\oplus}_{\mathrm{tert}}$ $\mathrm{R}^{\ominus}_{\mathrm{sek}}$ $\mathrm{R}_{\mathrm{prim}}$

4 Stereo Descriptors, Nomenclature, Latin Phrases

4.1 Stereo Descriptors and Nomenclature

The following macros are intended to make the writing of IUPAC names more convenient:

- Cahn-Ingold-Prelog:
  - `\Rcip` *(R)*
  - `\Scip` *(S)*
  - `\cip{<conf>}` e. g.: `\cip{R,S}` *(R,S)*
- Fischer:
  - `\Dfi` D
  - `\Lfi` L

- cis/trans & zusammen/entgegen:
  - `\Z` *(Z)*
  - `\E` *(E)*
  - `\cis` *cis* (This command is also defined by the package 'bpchem'. 'chemmacros' redefines it, see section 2.)
  - `\trans` *trans* (This command is also defined by the package 'bpchem'. 'chemmacros' redefines it, see section 2.)
- ortho/meta/para:
  - `\ortho` *o*
  - `\meta` *m*
  - `\para` *p*

absolute configuration (uses 'Ti***k***Z'):

- `\Rconf[<letter>]` `\Rconf`:  `\Rconf[]`: 
- `\Sconf[<letter>]` `\Sconf`:  `\Sconf[]`: 

Examples:

```
\Dfi-Tartaric Acid = \cip{2S,3S}-Tartaric Acid
```

D-Tartaric Acid = *(2S,3S)*-Tartaric Acid

```
\Dfi-($-$)-Threose = \cip{2S,3R}-($-$)-2,3,4-
   Trihydroxybutanal
```

D-(−)-Threose = *(2S,3R)*-(−)-2,3,4-Trihydroxybutanal

```
\cis-2-Buten = \Z-2-Butene, \cip{2E,4Z}-Hexadiene
```

cis-2-Buten = *(Z)*-2-Butene, *(2E,4Z)*-Hexadiene

```
\meta-Xylol = 1,3-Dimethylbenzene
```

m-Xylol = 1,3-Dimethylbenzene

```
% with 'bpchem' command \IUPAC:
\IUPAC{\Dfi-Tar\|taric Acid} = \IUPAC{\cip{2S,3S}-Tar\|
   taric Acid}, \IUPAC{\Dfi-($-$)-Threose} = \IUPAC{\cip
   {2S,3R}-($-$)-2,3,4\-Tri\|hydroxy\|butanal}
```

D-Tartaric Acid = *(2S,3S)*-Tartaric Acid, D-(−)-Threose = *(2S,3R)*-(−)-2,3,4-Trihydroxybutanal

The last example uses the \IUPAC command, which is provided by the 'bpchem' package.

Of course the appearance depends on the font you chose:

```
\cip{2S,3R} \E \Z \Dfi \Lfi \par
\fontfamily{ptm}\selectfont
\cip{2S,3R} \E \Z \Dfi \Lfi \par
\fontfamily{ppl}\selectfont
\cip{2S,3R} \E \Z \Dfi \Lfi
```

*(2S,3R) (E)(Z)*DL

*(2S,3R) (E)(Z)*DL

*(2S,3R) (E)(Z)*DL

4.2 Latin Phrases

NEW v.1.1

At last there are two commands for common latin phrases.

- \insitu *in situ*
- \abinitio *ab initio*

If the package 'chemstyle' has been loaded, too[1)], they are defined using 'chemstyle's \latin command. This means that then the appearance depends on 'chemstyle's option abbremph:

```
\insitu, \abinitio\\
\cstsetup{abbremph=false}
\insitu, \abinitio
```

1) 'chemstyle' defines other similar commands like *et al.*, *in vacuo*.

in situ, *ab initio*
in situ, ab initio

If ‘chemstyle’ hasn’t been loaded, they’re always in *italics*.

5 Units with ‘siunitx’

NEW v.1.1

In chemistry some non-SI units are very common. ‘siunitx’ provides the command `\DeclareSIUnit{<command>}{<unit>}` to add arbitrary units. ‘chemmacros’ uses that command to provide some units. Like all ‘siunitx’ units they’re only valid inside `\SI{<num>}{<unit>}` and `\si{<unit>}`.

- `\atmosphere` atm
- `\atm` atm
- `\calory` cal
- `\cal` cal
- `\cmc` cm^3 [1)]
- `\molar` $\mathrm{mol\,dm^{-3}}$ [1)]
- `\moLar` $\mathrm{mol\,L^{-1}}$
- `\Molar` M [1)]
- `\MolMass` $\mathrm{g\,mol^{-1}}$
- `\normal` N
- `\torr` torr

By the way: `\mmHg` mmHg is already defined by ‘siunitx’.

[1)] These units are also defined by ‘chemstyle’. They are only defined by ‘chemmacros’, if ‘chemstyle’ is not loaded.

6 Acid/Base

NEW v.1.1

Easy representation of pH, pK_A ...

- `\pH` pH
- `\pOH` pOH
- `\pKa[<num>]` `\pKa` pK_A, `\pKa[1]` pK_A1
- `\pKb[<num>]` `\pKb` pK_B, `\pKb[1]` pK_B1

These commands can be used both in text and in math mode as well as inside the `\ce` command of the 'mhchem' package. The command `\pKa` depends on the package option `german`, see section 2.

7 Oxidation Numbers and (real) Charges

7.1 Ion Charges

Simple displaying of charges:

- `\pch[<number>]` positive charge (**p**lus + **ch**arge): `\pch` $^{\oplus}$, `Na\pch` $\mathrm{Na}^{\oplus}$, `Ca\pch[2]` $\mathrm{Ca}^{2\oplus}$
- `\mch[<number>]` negative charge (**m**inus + **ch**arge): `\mch` $^{\ominus}$, `F\mch` $\mathrm{F}^{\ominus}$, `S\mch[2]` $\mathrm{S}^{2\ominus}$

7.2 Oxidation Numbers

Typesetting oxidation numbers:

- `\ox{<number>,<atom>}` places `<number>` above `<atom>`; `\ox{+1,Na}`, `\ox{+I,Na}`, `\ox{-2,S}`, `\ox{-II,S}` $\overset{+1}{\mathrm{Na}}$, $\overset{+\mathrm{I}}{\mathrm{Na}}$, $\overset{-2}{\mathrm{S}}$, $\overset{-\mathrm{II}}{\mathrm{S}}$

If the package 'mhchem' has been loaded `<atom>` is set inside the `\ce` command: `\ox{+II,Ca}\ox{-I,F2}` $\overset{+\mathrm{II}}{\mathrm{Ca}}\overset{-\mathrm{I}}{\mathrm{F}_2}$. Without 'mhchem' this isn't working this way ($\overset{+\mathrm{II}}{\mathrm{Ca}}\overset{-\mathrm{I}}{\mathrm{F2}}$) and

you need to use the math way: `\ox{+II,Ca}\ox{-I,F$_2$}` $\overset{+\mathrm{II}}{\mathrm{Ca}}\overset{-\mathrm{I}}{\mathrm{F}_2}$.

7.3 Partial Charges and similar Stuff

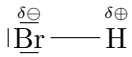
The next ones probably are seldomly needed but nevertheless useful:

- `\delp` $\delta\oplus$ (**del**ta + **p**lus)
- `\delm` $\delta\ominus$ (**del**ta + **m**inus)

These macros for example can be used with the `\ox` command or with the 'chemfig'[1] package:

```
\ox{\delp,H}\ox{\delm,Cl}\par
\chemfig{\chemabove[3pt]{\lewis{246,Br}}{\delm}-\
    chemabove[3pt]{H}{\delp}}
```

$\overset{\delta\oplus}{\mathrm{H}}\overset{\delta\ominus}{\mathrm{Cl}}$

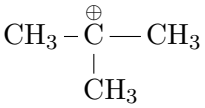


The following macros are useful together with 'chemfig', too.

- `\scrp` $\oplus$(**scr**iptstyle + **p**lus)
- `\scrm` $\ominus$(**scr**iptstyle + **m**inus)

For example:

```
{\setatomsep{1.8em}\chemfig{CH_3-\chemabove{C}{\scrp
    }(-[6]C|H_3)-\vphantom{H_3}CH_3}}
```



[1] http://www.ctan.org/pkg/chemfig

8 Reaction Mechanisms

NEW v.1.1

With the command

```
\mech[<type>]
```

one can specify the most common reaction mechanisms. `<type>` can have one of the following values:

- `<type>=` (empty, no opt. argument) nucleophilic substitution `\mech` S_N
- `<type>=1` unimolecular nucleophilic substitution `\mech[1]` S_N1
- `<type>=2` bimolecular nucleophilic substitution `\mech[2]` S_N2
- `<type>=se` electrophilic substitution `\mech[se]` S_E
- `<type>=1e` unimolecular electrophilic substitution `\mech[1e]` S_E1
- `<type>=2e` bimolecular electrophilic substitution `\mech[2e]` S_E2
- `<type>=ar` electrophilic aromatic substitution `\mech[ar]` Ar-S_E
- `<type>=` elimination `\mech[e]` E (probably never to be used)
- `<type>=e1` unimolecular elimination `\mech[e1]` E1
- `<type>=e2` bimolecular elimination `\mech[e2]` E2
- `<type>=cb` unimolecular elimination "conjugated base", i. e. via carbanion `\mech[cb]` $E1_{cb}$

This command can also be used in math mode and inside the `\ce` command of the 'mhchem' package.

```
$\mech[cb]$ \ce{\mech[2]} \ce{\mech[ar]}
```

$E1_{cb}$ S_N2 Ar-S_E

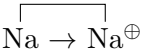
9 Redox Reactions

'chemmacros' provides two commands[1)] to visualize the transfer of electrons in redox reactions. Both commands are using 'Ti*k*Z'.

```
\OX{<name>,<atom>}
\redox(<name1>,<name2>)[<tikz>][<dim>]{<text>}
```

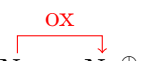
`\OX` places `<atom>` into a node, which is named with `<name>`. If you have set two `\OX`, they can be connected with a line using `\redox`. To do so the names of the two nodes that are to be connected are written in the round braces. Since `\redox` draws a tikzpicture with options `remember picture,overlay`, the document needs to be *compiled at least two times*.

```
\OX{a,Na} $\rightarrow$ \OX{b,Na}\pch\redox(a,b){
    oxidation}
```



This line can be customized using 'Ti*k*Z' keys in `<tikz>`:

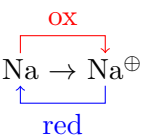
```
\OX{a,Na} $\rightarrow$ \OX{b,Na}\pch\redox(a,b)[->,red]{
    ox}
```



NEW v.1.1

`<dim>` can be used to adjust the length of the *vertical parts* of the line. The default length is `.6em`. This length is multiplied with `<dim>`. If you use a negative value the line is placed *below* the text.

```
\OX{a,Na} $\rightarrow$ \OX{b,Na}\pch
\redox(a,b)[->,red]{ox}
\redox(a,b)[<-,blue][-1]{red}
```

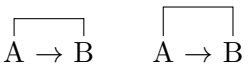


The default length of the vertical lines can be customized with `\setredoxdist{<length>}`:

```
\OX{a,A} $\rightarrow$ \OX{b,B}
\redox(a,b){}
\bigskip
```

1) Thanks to Peter Cao who suggested this feature.

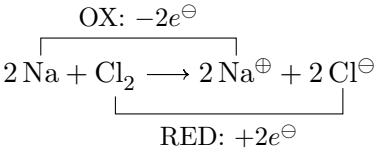
```
\setredoxdist{1em}
\OX{a,A} $\rightarrow$ \OX{b,B}
\redox(a,b){}
```



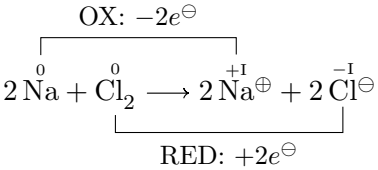
An empty argument resets the length to the default value.

Both commands also can be used with the ‘mhchem’ command \ce and with the \ox command (section 7.2).

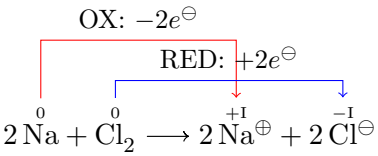
```
\ce{ 2 \OX{o1,Na} + \OX{r1,Cl2} -> 2 \OX{o2,Na}\pch + 2 \
   OX{r2,Cl}\mch }
\redox(o1,o2){\small OX: $- 2\el$}
\redox(r1,r2)[][-1]{\small RED: $+ 2\el$}
```



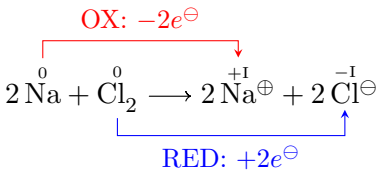

```
\ce{ 2 \OX{o1,\ox{0,Na}} + \OX{r1,\ox{0,Cl2}} -> 2 \OX{o
   2,\ox{+I,Na}}\pch + 2 \OX{r2,\ox{-I,Cl}}\mch }
\redox(o1,o2){\small OX: $- 2\el$}
\redox(r1,r2)[][-1]{\small RED: $+ 2\el$}
```




```
\ce{ 2 \OX{o1,\ox{0,Na}} + \OX{r1,\ox{0,Cl2}} -> 2 \OX{o
   2,\ox{+I,Na}}\pch + 2 \OX{r2,\ox{-I,Cl}}\mch }
\redox(o1,o2)[draw=red,->][3.33]{\small OX: $- 2\el$}
\redox(r1,r2)[draw=blue,->]{\small RED: $+ 2\el$}
```




```
\ce{ 2 \OX{o1,\ox{0,Na}} + \OX{r1,\ox{0,Cl2}} -> 2 \OX{o
   2,\ox{+I,Na}}\pch + 2 \OX{r2,\ox{-I,Cl}}\mch }
\redox(o1,o2)[green,-stealth]{\small OX: $- 2\el$}
\redox(r1,r2)[purple,-stealth][-1]{\small RED: $+ 2\el$}
```



10 (Standard) State, Thermodynamics

10.1 Thermodynamic Variables

The following commands use 'siunitx':

- `\Enthalpy[<sub>,<sup>,<unit>,<subscript pos>]{<value>}`
- `\Entropy[<sub>,<sup>,<unit>,<subscript pos>]{<value>}`
- `\Gibbs[<sub>,<sup>,<unit>,<subscript pos>]{<value>}`

Their usage is pretty much self-explaining:
`\Enthalpy{-123.4}` gives $\Delta H^{\ominus} = -123.4\,\mathrm{kJ\,mol^{-1}}$.
If you want to specify what kind of enthalpy (reaction, formation, ...) is meant, you can use the first optional argument `<sub>`:
`\Enthalpy[r]{-123.4}` $\Delta_\mathrm{r} H^{\ominus} = -123.4\,\mathrm{kJ\,mol^{-1}}$
NEW v.1.1 `\Enthalpy[vapor.,,,right]{123}` $\Delta H^{\ominus}_{\mathrm{vapor.}} = 123\,\mathrm{kJ\,mol^{-1}}$

In the last example you could see the usage of the fourth optional argument (`<subscript pos>`). It is used to specify whether the subscript is placed to the right or to the left of the main symbol. It can have the values `left` (or empty) or `right`.

The standard state symbol can be replaced by the second optional argument `<sup>`...
`\Enthalpy[,\transitionstatesymbol]{-123.4}` $\Delta H^{\ddagger} = -123.4\,\mathrm{kJ\,mol^{-1}}$
... and else depends if the package 'chemstyle' has been loaded, see section 10.2.

The third optional argument `<unit>` can be used to change the unit:
`\Enthalpy[,,\kilo\calory\per\mole]{-123.4}` $\Delta H^{\ominus} = -123.4\,\mathrm{kcal\,mol^{-1}}$
The unit is set corresponding to the rules of 'siunitx' and depends on its settings:

```
1 \Enthalpy{-1234.56e3}\par
2 \sisetup{per-mode=symbol,exponent-product=\cdot,output-
    decimal-marker={,},group-four-digits=true}
3 \Enthalpy{-1234.56e3}
```

$\Delta H^{\ominus} = -1234.56 \times 10^{3}\,\mathrm{kJ\,mol^{-1}}$

$\Delta H^\ominus = -1\,234{,}56 \cdot 10^3\,\mathrm{kJ/mol}$

The other two commands work exactly the same way.

```
\Entropy{12.3}, \Gibbs{-12.3}.
```

$S^\ominus = 12.3\,\mathrm{J\,K^{-1}\,mol^{-1}}$, $\Delta G^\ominus = -12.3\,\mathrm{kJ\,mol^{-1}}$.

10.1.1 Create New Variables

You can use the command

```
\setnewstate[<standard sup>,<Delta symbol>,<subscript pos
   >]{<name>}{<symbol>}{<unit>}
```

to create new corresponding commands:

```
\setnewstate{Helmholtz}{A}{\kilo\joule\per\mole}
\setnewstate[ ,,right]{ElPot}{E}{\volt}
\Helmholtz{123.4} \ElPot{-1.1} \ElPot[\ce{Sn}|\ce{Sn \pch
   [2]}||\ce{Pb \pch[2]}|\ce{Pb},0]{0.01}
```

$\Delta A^\ominus = 123.4\,\mathrm{kJ\,mol^{-1}}$ $\Delta E = -1.1\,\mathrm{V}$ $\Delta E^0_{\mathrm{Sn|Sn^{2\oplus}||Pb^{2\oplus}|Pb}} = 0.01\,\mathrm{V}$

As you can see, \ElPot has its subscript by definition on the right as default behaviour. Of course you can still place it on the left by using the option \ElPot[r,,,left]{0.12} $\Delta_\mathrm{r} E = 0.12\,\mathrm{V}$ (even if this example might not make much sense).

Indeed, the commands

```
\Enthalpy, \Entropy, \Gibbs
```

are defined as follows:

```
\setnewstate{Enthalpy}{H}{\kilo\joule\per\mole}
\setnewstate[, ]{Entropy}{S}{\joule\per\kelvin\per\mole}
\setnewstate{Gibbs}{G}{\kilo\joule\per\mole}
```

10.1.2 Redefine Variables

NEW v.1.1

With

```
\renewstate[<standard sup>,<Delta symbol>,<subscript pos
    >]{<name>}{<symbol>}{<unit>}
```

you can redefine the already existing commands:

```
\renewstate{Enthalpy}{h}{\joule}
\Enthalpy[f]{12.5}
```

$\Delta_\mathrm{f} h^\ominus = 12.5\,\mathrm{J}$

So – for following thermodynamic conventions – one could define a molar and an absolute variable:

```
\setnewstate[ ]{enthalpy}{h}{\kilo\joule\per\mole}% molar
\renewstate[ ]{Enthalpy}{H}{\kilo\joule}% absolute
\enthalpy{-12.3} \Enthalpy{-12.3}
```

$\Delta h = -12.3\,\mathrm{kJ\,mol^{-1}}$ $\Delta H = -12.3\,\mathrm{kJ}$

10.2 State

The commands presented in section 10.1 internally all use the command

```
\State[<superscript>,<Delta symbol>,<subscript pos>]{<
    symbol>}{<subscript>}
```

It can be used to write the thermodynamic variables without value and unit. Please note that `{<subscript>}` is an *optional* argument.

Examples:

```
\State{A}, \State{G}{f}, \State[ ,,right]{E}{\ce{Na}}, \
    State[\SI{1000}{\celsius}]{H}
```

$\Delta A^\ominus$, $\Delta_\mathrm{f} G^\ominus$, ΔE_Na, $\Delta H^{1000\,^\circ\mathrm{C}}$

I admit: not in every case it is easier or more convenient to use this command instead of the direct typing, for example `$\Delta E_\ce{Na}$` ΔE_Na. The examples only are intended to show how the command works and what it *can* be used for. The first example surely is typed faster than `$\Delta_\mathrm{f}G^\ominus$` $\Delta_\mathrm{f} G^\ominus$.

The standard state symbol ⦵ is only used, if the package 'chemstyle' is loaded, which provides it with the command \standardstate. Else the symbol \circ ∘ is used. \State{A}{b}: with 'chemstyle' $\Delta_\mathrm{b}A^{⦵}$, without $\Delta_\mathrm{b}A^{\circ}$.

NEW v.1.1

With the command

```
\setstatesubscript{<subscript pos>}
```

one can change the predefined value of the subscript postition. You probably have noticed, that its default value is left.

```
\State{A}{b}\\
\setstatesubscript{right}
\State{A}{b}
```

$\Delta_\mathrm{b}A^{⦵}$
$\Delta A^{⦵}_\mathrm{b}$

This command does *not* change the behaviour of \setnewstate and \renewstate.

11 Spectroscopy

NEW v.1.1

If substances are examined wether they are what they're supposed to, one often needs NMR spectroscopy. Measured results then are written in a way like: ^{1}H-NMR (400 MHz, $CDCl_3$): $\delta = 1.59\ldots$ 'chemmacros' provides a command which simplifies writing this down (uses 'siunitx').

```
\NMR{<num>,<elem>}(<num>,<unit>)[<solvent>]
\NMR*{<num>,<elem>}(<num>,<unit>)[<solvent>]
```

All arguments are optional! Without arguments we get:

- \NMR ^{1}H-NMR: δ (very much like the 'bpchem' command \HNMR)
- \NMR* ^{1}H-NMR (without : δ)

With the first argument you can specify the kind of NMR:

- \NMR{13,C} ^{13}C-NMR: δ
- \NMR*{13,C} ^{13}C-NMR

With the second argument the frequency (in MHz) can be specified:

- `\NMR(400)` ^{1}H-NMR (400 MHz): δ
- `\NMR*(400)` ^{1}H-NMR (400 MHz)

You also can change the unit:

- `\NMR(4e8,\hertz)` ^{1}H-NMR (4×10^{8} Hz): δ
- `\NMR*(4e8,\hertz)` ^{1}H-NMR (4×10^{8} Hz)

Please note that the setup of 'siunitx' also has an impact on this command:

- `\sisetup{exponent-product=\cdot}\NMR(4e8,\hertz)` ^{1}H-NMR ($4 \cdot 10^{8}$ Hz): δ
- `\sisetup{exponent-product=\cdot}\NMR*(4e8,\hertz)` ^{1}H-NMR ($4 \cdot 10^{8}$ Hz)

And finally with the third argument the solvent can be specified:

- `\NMR[CDCl3]` ^{1}H-NMR ($CDCl_3$): δ
- `\NMR*[CDCl3]` ^{1}H-NMR ($CDCl_3$)

Depending on wether you use 'mhchem' or not, the solvent is written inside the `\ce` command. If you don't use 'mhchem', the subscript isn't recognized automatically and you need the use the math mode:

- `\NMR[CDCl$_3$]` ^{1}H-NMR ($CDCl_3$): δ
- `\NMR*[CDCl$_3$]` ^{1}H-NMR ($CDCl_3$)

All arguments can be combined freely, the command can also be used inside math mode.

If you want the appearance to match the ones of 'bpchem' (compare 'bpchem' command `\HNMR` ^{1}H-NMR: δ to 'chemmacros' command `\NMR` ^{1}H-NMR: δ), you can use the package option `bpchem` (see section 2).

Examples:

```
\NMR{13,C}(100) \\
\NMR*{13,C}(100) \\
\NMR*{19,F}[CFCl3] \\
```

```
\NMR*{19,F}(285)[CFCl3] \\
\NMR(400)[CDCl3] = \num{1.59} (q, 1H, \textit{J} = \SI
   {11.6}{\hertz}, H-4)
```

^{13}C-NMR (100 MHz): δ
^{13}C-NMR (100 MHz)
^{19}F-NMR ($CFCl_3$)
^{19}F-NMR (285 MHz, $CFCl_3$)
^{1}H-NMR (400 MHz, $CDCl_3$): $\delta = 1.59$ (q, 1H, $J = 11.6$ Hz, H-4)

12 Commands for 'mhchem'

There are some commands which are meant for the use with 'mhchem'. They are defined if 'mhchem' is loaded. Before they're described some words on using commands inside the \ce and \cee commands. Probably due to the way these commands are processed there can be difficulties especially when using commands with arguments.

Often you have to leave blank spaces:

```
\ce{Na\pch}\\      % no problem
\ce{Ca\pch[2]}\\   % displayed wrong
\ce{Ca \pch[2]}\\ % displayed right
\ce{Ca$\pch[2]$}   % displayed right
```

$Na^{\oplus}$
$Ca^{\oplus}[_2]$
$Ca^{2\oplus}$
$Ca^{2\oplus}$

You also need to put curly braces at the end of commands:

```
\ce{\mch OMe}\\    % displayed wrong
\ce{\mch{} OMe}\\ % displayed right
\ce{$\mch$OMe}     % displayed right
```

$^{\ominus}OMe$
$^{\ominus}$OMe
$^{\ominus}$OMe

This is *not* only true for 'chemmacros' commands!

```
\ce{A \quad B} \ce{Na2\textbf{O}}\\     % displayed wrong
\ce{A \quad{} B} \ce{Na2 \textbf{O}}\\ % displayed right
\ce{A $\quad$ B} \ce{Na2 \textbf{O}}   % displayed right
```

A $\quad$ *B* Na_2O
A $\quad{}$ B $Na_2\mathbf{O}$
A $\quad$ B $Na_2\mathbf{O}$

As you can see in most cases instead of using blank spaces or curly braces you can also put the according command between `$ $`.

12.1 Reaction Environments

12.1.1 Defined by 'chemmacros'

You can use these environments for numbered...

```
\begin{reaction}
  <mhchem code>
\end{reaction}
\begin{reactions}
  <mhchem code>
\end{reactions}
```

...and their starred versions for unnumbered reactions.

```
\begin{reaction*}
  <mhchem code>
\end{reaction*}
\begin{reactions*}
  <mhchem code>
\end{reactions*}
```

With them you can create (un)numbered reaction equations similar to mathematical equations.

The environments `reaction`/`reaction*` use the `equation`/`equation*` environments and the environments `reactions`/`reactions*` use the `align`/`align*` environments to display the reactions.

Reaction with counter:

```
\begin{reaction}
 A -> B
\end{reaction}
```

$$\mathrm{A} \longrightarrow \mathrm{B} \tag{1}$$

Reaction without counter:

```
\begin{reaction*}
 C -> D
\end{reaction*}
```

$$\mathrm{C} \longrightarrow \mathrm{D}$$

Several aligned reactions with counter:

```
\begin{reactions}
 A     &-> B + C \\
 D + E &-> F
\end{reactions}
```

$$\begin{aligned} \mathrm{A} &\longrightarrow \mathrm{B} + \mathrm{C} && \{2\} \\ \mathrm{D} + \mathrm{E} &\longrightarrow \mathrm{F} && \{3\} \end{aligned}$$

Several aligned reactions without counter:

```
\begin{reactions*}
 G     &-> H + I \\
 J + K &-> L
\end{reactions*}
```

$$\begin{aligned} \mathrm{G} &\longrightarrow \mathrm{H} + \mathrm{I} \\ \mathrm{J} + \mathrm{K} &\longrightarrow \mathrm{L} \end{aligned}$$

If you want to change the layout of the counter tags, you can use `\renewtagform{<tagname>}[<format>]{<right delim>}{<left delim>}`[1)].

```
\renewtagform{CMreaction}[R \textbf]{[}{]}
\begin{reaction}
 H2O + CO2 <<=> H2CO3
\end{reaction}
```

$$\mathrm{H_2O} + \mathrm{CO_2} \leftharpoondown\!\rightharpoonup \mathrm{H_2CO_3} \qquad [\mathrm{R}\ \mathbf{4}]$$

1) Provided by the 'mathtools' package.

12.1.2 Own Reactions

NEW v.1.1

You can create new types of reactions with the command:

```
\newreaction{<name>}{<math name>}
```

`<name>` will be the name of the new environment. `<math name>` is the used math environment.

The command has variants. The first one is `\newreaction*`, which will also define a starred version of the new environment, if the starred math environment exists. If it doesn't exist, this will cause an error.

The second one is `\newreaction+`, which is used to define an environment with a mandatory argument. Of course this only works, if the used math environment has a mandatory argument.

You can also use the combined version `\newreaction*+`.

The predefined environments are defined via

```
\newreaction*{reaction}{equation}
\newreaction*{reactions}{align}
```

Let's suppose, you'd like to have the alignment behaviour of the `alignat` environment for 'mhchem' reactions. You could do the following:

```
\newreaction*+{reactionsat}{alignat}
```

With this the `reactionsat` environment is defined.

```
\begin{reactionsat}{3}
 A      &-> B      &&-> C      &&-> D \\
 aaaaa  &-> bbbbb  &&-> ccccc  &&-> ddddd
\end{reactionsat}
\begin{reactionsat*}{2}
 A      &-> B      & C              &-> D   \\
 aaaaa  &-> bbbbb  &$\quad$ ccccc  &-> ddddd
\end{reactionsat*}
```

$$\begin{aligned} \mathrm{A} &\longrightarrow \mathrm{B} && \longrightarrow \mathrm{C} && \longrightarrow \mathrm{D} && \{5\} \\ \mathrm{aaaaa} &\longrightarrow \mathrm{bbbbb} && \longrightarrow \mathrm{ccccc} && \longrightarrow \mathrm{ddddd} && \{6\} \end{aligned}$$

$$\begin{array}{ll} \mathrm{A} \longrightarrow \mathrm{B} & \mathrm{C} \longrightarrow \mathrm{D} \\ \mathrm{aaaaa} \longrightarrow \mathrm{bbbbb} & \mathrm{ccccc} \longrightarrow \mathrm{ddddd} \end{array}$$

12.2 Phases

NEW v.1.1

These commands are intended to indicate the phase of a compound. Although these commands were intended for the use with 'mhchem' they can be used without it as well and are also defined if 'mhchem' isn't loaded.

```
\solid[<anything>]
\liquid[<anything>]
\gas
\aq % dissolved in water
```

I always found it tedious to type out phase indicators:

```
\ce{C_{(s)} + 2 H2O_{(l)} -> CO2_{(g)} + 2 H2_{(g)}}
```

The same result now can be achieved with:

```
\ce{C\solid{} + 2 H2O\liquid{} -> CO2\gas{} + 2 H2\gas}
```

$\mathrm{C_{(s)} + 2\,H_2O_{(l)} \longrightarrow CO_{2(g)} + 2\,H_{2(g)}}$

To make it complete: `NaCl\aq` gives $\mathrm{NaCl_{(aq)}}$.

With the package option `german` (see section 2) you get:
$\mathrm{C_{(f)} + 2\,H_2O_{(fl)} \longrightarrow CO_{2(g)} + 2\,H_{2(g)}}$

You can get the same result without the package option by using the arguments:

```
\ce{C \solid[f] + 2 H2O \liquid[fl] -> CO2\gas{} + 2 H2\
    gas}
```

$\mathrm{C_{(f)} + 2\,H_2O_{(fl)} \longrightarrow CO_{2(g)} + 2\,H_{2(g)}}$

If you looked closely, you have probably noticed that the german `\liquid` isn't identical to `\liquid[fl]` but to `\liquid[f\/l]`. This makes the subscript more readable.

Of course it doesn't matter which command with optional argument you use. Both of them just write a subscript with braces. `\solid[f]` is identical to `\liquid[f]`.

One can think of other uses, too:

```
C\solid[graphite]
```

$\mathrm{C}_{(\mathrm{graphite})}$

12.3 Text Under Compounds

NEW v.1.1

It has always been a bit laborious to write something under a molecule with 'mhchem'. 'chemmacros' provides a command for that:

```
\mhName[<width>][<textattr>]{<mhchem code>}{<name>}
```

For example:

```
\ce{4 C2H5Cl + Pb / Na -> \mhName{Pb(C2H5)4}{former
    antiknock additive} + NaCl}
```

$$4\,\mathrm{C_2H_5Cl} + \mathrm{Pb/Na} \longrightarrow \underset{\substack{\text{former antiknock}\\ \text{additive}}}{\mathrm{Pb(C_2H_5)_4}} + \mathrm{NaCl}$$

As you can see the text is set centered and `\tiny`, while the molecule uses it's normal space. With the first optional argument you can choose the width, that the molecule uses:

```
\ce{4 C2H5Cl + Pb / Na -> \mhName[3cm]{Pb(C2H5)4}{former
    antiknock additive} + NaCl}
```

$$4\,\mathrm{C_2H_5Cl} + \mathrm{Pb/Na} \longrightarrow \underset{\text{former antiknock additive}}{\mathrm{Pb(C_2H_5)_4}} + \mathrm{NaCl}$$

The text attributes have `\centering\tiny` as default. Using other attributes can overwrite them in certain circumstances:

```
\ce{4 C2H5Cl + Pb / Na -> \mhName[3cm]{Pb(C2H5)4}{\small
    former antiknock additive} + NaCl}\\
\ce{4 C2H5Cl + Pb / Na -> \mhName[3cm]{Pb(C2H5)4}{\
    raggedright\color{red}\bfseries former antiknock
    additive} + NaCl}
```

$$4\,\mathrm{C_2H_5Cl} + \mathrm{Pb/Na} \longrightarrow \underset{\substack{\text{former antiknock}\\ \text{additive}}}{\mathrm{Pb(C_2H_5)_4}} + \mathrm{NaCl}$$

$$4\,\mathrm{C_2H_5Cl} + \mathrm{Pb/Na} \longrightarrow \underset{\substack{\textbf{former antiknock}\\ \textbf{additive}}}{\mathrm{Pb(C_2H_5)_4}} + \mathrm{NaCl}$$

Using the second optional argument will overwrite them in any case:

```
\ce{4 C2H5Cl + Pb / Na -> \mhName[3cm][\small]{Pb(C2H5)
   4}{former antiknock additive} + NaCl}\\
\ce{4 C2H5Cl + Pb / Na -> \mhName[3cm][\raggedright\color
   {red}\bfseries]{Pb(C2H5)4}{former antiknock additive}
   + NaCl}
```

$$4\,C_2H_5Cl + Pb/Na \longrightarrow \underset{\substack{\text{former}\\ \text{additive}}}{Pb(C_2H_5)_4} \underset{\text{antiknock}}{} + NaCl$$

$$4\,C_2H_5Cl + Pb/Na \longrightarrow \underset{\textbf{former antiknock additive}}{Pb(C_2H_5)_4} + NaCl$$

You can change the default values globally by using

```
\setmhName{<textattr>}
```

With this command you can change the predefined settings as you like:

```
\setmhName{\centering\footnotesize\color{blue}}
\ce{4 C2H5Cl + Pb / Na -> \mhName{Pb(C2H5)4}{former
   antiknock additive} + NaCl}
```

$$4\,C_2H_5Cl + Pb/Na \longrightarrow \underset{\substack{\text{former}\\ \text{antiknock}\\ \text{additive}}}{Pb(C_2H_5)_4} + NaCl$$

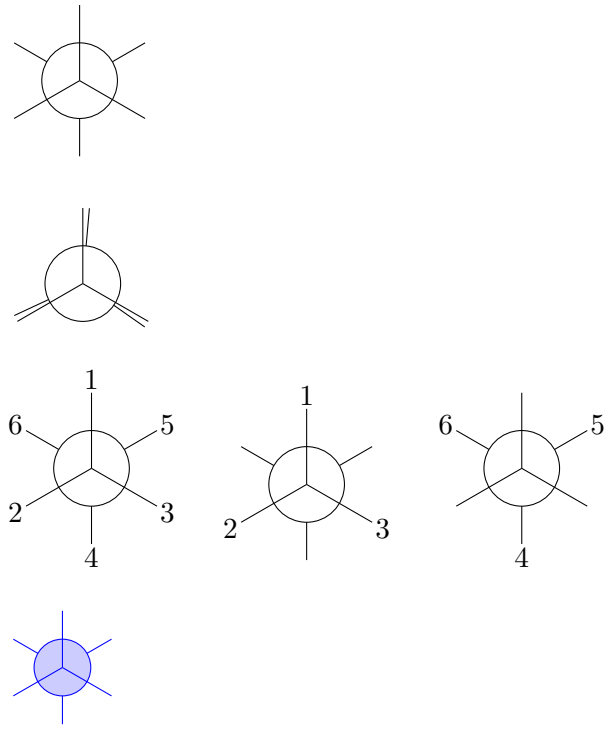
13 Newman Projections

The command

```
\newman[<angle>,<scale>,<tikz>]{<1>,<2>,<3>,<4>,<5>,<6>}
```

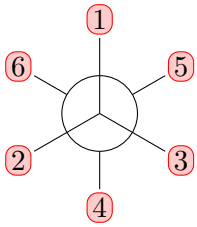
allows you to create newman projections (uses 'Ti*k*Z'). Examples:

```
\newman{}\par% default: staggered
\newman[175]{}\par% rotated by 175 degrees => eclipsed
\newman{1,2,3,4,5,6} \newman{1,2,3} \newman{,,,4,5,6}\par
   % with atoms
\newman[,.75,draw=blue,fill=blue!20]{}% scaled and
   customized with TikZ
```



Another option allows you to customize the nodes within which the atoms are placed:

```
\newman[][<tikz nodes>]{<1>,<2>,<3>,<4>,<5>,<6>}
% example:
\newman[][draw=red,fill=red!20,inner sep=2pt,rounded
    corners]{1,2,3,4,5,6}
```

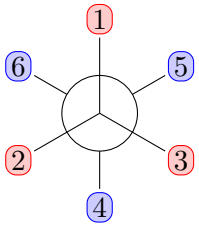


If you want to display the "front" atoms differently from the "back" atoms, you can use a third option:

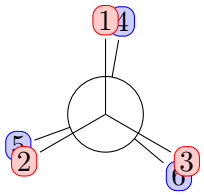
```
\newman[][<tikz front nodes>][<tikz back nodes
    >]{<1>,<2>,<3>,<4>,<5>,<6>}
```

Examples:

```
\newman[][draw=red,fill=red!20,inner sep=2pt,rounded
    corners][draw=blue,fill=blue!20,inner sep=2pt,rounded
    corners]{1,2,3,4,5,6}
```




```
\newman[170][draw=red,fill=red!20,inner sep=2pt,rounded
    corners][draw=blue,fill=blue!20,inner sep=2pt,rounded
    corners]{1,2,3,4,5,6}
```



14 p-Orbitals

'chemmacros' provides commands to visualize p-orbitals.

```
\porb[<size factor>,<color>,<angle>]
\phorb[<size factor>,<color>,<angle>]
\setorbheight{<length>}
```

This displays a horizontal orbital or one rotated by `<angle>`: `\porb` `\qquad` `\porb[,,30]` 

`\phorb` only displays one half orbital: `\phorb[,red,90]` 

The size of the orbitals depends on an internal length that can be set with `\setorbheight{<length>}`. It's default value is `1em`.

```
\porb\par
\setorbheight{2em}\porb
```

The size of an orbital can also be changed directly using the optional argument `<size factor>`.

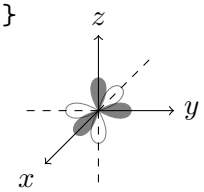
```
\porb\par
\porb[2]\par
\porb[.5]
```

There are shortcuts for the x-, y- and z-orbitals:

```
\pzorb \qquad \pyorb \qquad \pxorb
```

Since the orbitals are drawn in a tikzpicture with the option `overlay`, they are set all at the same spot, if you don't shift them:

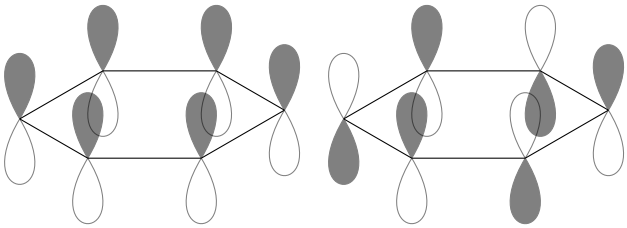
```
\hspace{2cm}\pxorb\pyorb\pzorb
\tikz[overlay]{
  \draw[->](0,0)--(1,0)node[right]{$y$};
  \draw[dashed](0,0)--(-1,0);
  \draw[->](0,0)--(0,1)node[above]{$z$};
  \draw[dashed](0,0)--(0,-1);
  \draw[->](0,0)--(-.707,-.707)node[below left]{$x$};
  \draw[dashed](0,0)--(.707,.707);
}
```



The orbitals also can be used together with 'chemfig':

```
\setorbheight{2em}\setbondoffset{0pt}
\chemfig{?\pzorb-[,1.3]\pzorb-[:30,1.1]\pzorb-[:150,.9]\
   pzorb-[4,1.3]\pzorb-[:-150,1.1]\pzorb?}\qquad
```

```
\chemfig{?\pzorb-[,1.3]{\porb[,,-90]}-[:30,1.1]\pzorb
  -[:150,.9]{\porb[,,-90]}-[4,1.3]\pzorb-[:-150,1.1]{\
  porb[,,-90]}?}
```



15 List of Commands

\el, \prt, \ntr, \HtO, \water, \Hpl, \Hyd, \Nu, \El, \transitionstatesymbol, \R	section 3: Particles, Ions and a Symbol
\cip, \Rcip, \Scip, \Dfi, \Lfi, \E, \Z, \cis, \trans, \Rconf, \Sconf, \ortho, \meta, \para, \insitu, \abinitio	section 4: Stereo Descriptors, Nomenclature, Latin Phrases
\pH, \pOH, \pKa, \pKb	section 6: Acid/Base
\delm, \delp, \mch, \pch, \ox, \scrm, \scrp	section 7: Oxidation Numbers and (real) Charges
\mech	section 8: Reaction Mechanisms
\redox, \OX	section 9: Redox Reactions
\Enthalpy, \Entropy, \Gibbs, \setnewstate, \renewstate, \State, \setstatesubscript	section 10: (Standard) State, Thermodynamics
\NMR	section 11: Spectroscopy
\begin{reaction}, \begin{reaction*}, \begin{reactions}, \begin{reactions*}, \newreaction, \solid, \liquid, \gas, \mhName, \setmhName	section 12: Commands for 'mhchem'
\newman	section 13: Newman Projections
\phorb, \porb, \pxorb, \pyorb, \pzorb, \setorbheight	section 14: p-Orbitals