

chemscheme — Support for chemical schemes ^{*}

Joseph Wright [†]

Released 2008/07/31

Abstract

The chemscheme package consists of two parts, both related to chemical schemes. The package adds a `scheme` float type to the \LaTeX default types `figure` and `table`. The `scheme` float type acts in the same way as those defined by the \LaTeX kernel, but is intended for chemical schemes. The package also provides a method for adding automatic chemical numbering to schemes.

^{*}This file describes version v1.4a, last revised 2008/07/31.

[†]E-mail: joseph.wright@morningstar2.co.uk

Contents

1	Introduction	3
2	Floating schemes	3
2.1	Basic use	3
2.2	Altering the defaults . .	4
2.3	babel support	4
3	Horizontal positioning of all floats	5
4	Reference numbers in graphics	6
4.1	Background	6
4.2	Usage	6
4.3	chemscheme and PDF \LaTeX	8
4.4	Managing chemical numbering	9
5	Generating chemical schemes	9
5.1	Overview	9
5.2	Macro-based methods .	10
5.3	Graphical methods . . .	10
6	Known issues	11
7	Implementation	11
7.1	Setup code	11
7.2	Support for schemes . .	13
7.3	Positioning float contents	14
7.4	babel support	15
7.5	Reference numbers in graphics	16
8	Change History	17
9	Index	19
10	References	21

1 Introduction

By default, \LaTeX defines two float types, `figure` and `table`. Synthetic chemists make heavy use of schemes, which need a `scheme` float type. This is provided by `chemscheme`, in a manner consistent with the kernel floats.

Synthetic chemists also number compounds for ease of reference. There are a number of \LaTeX packages which cover this area, most notably `bpchem` and `chemcompounds`. However, adding numbers automatically to schemes is not covered by any existing package. The `chemscheme` package seeks to rectify this.

2 Floating schemes

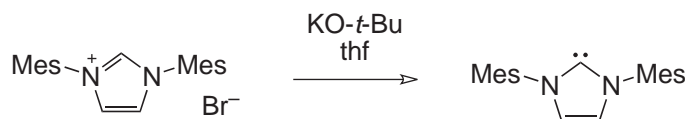
2.1 Basic use

`scheme` The package provides a new float type, `scheme`, accessed in the usual way.

```
\begin{scheme} [ht]
  \includegraphics{scheme-one}
  \caption{A scheme with no compound numbers.}
\end{scheme}
```

The `scheme` float is designed to behave in the same way as the standard

Scheme 1: A scheme with no compound numbers.



\LaTeX float environments `figure` and `table`. Thus schemes will be placed at the top of a page, where possible. As shown in the example, the use of positional modifiers is allowed. Labelling and referencing schemes also follows the \LaTeX conventions. `chemscheme` works hard to emulate the document class in use, and so the exact behaviour will depend on whether the standard classes, KOMA-Script or memoir are being used.

`\listofschemes`
`\listschemename`

To match the `\listoffigures` and `\listoftables` macros provided by the \LaTeX kernel, `chemscheme` provides a `\listofschemes` command. This works in the same way as the kernel commands, with the default text stored in the macro `\listschemename`. Users upgrading from version 1.1 should note the change of macro name (from `\listscheme`). This is to bring `chemscheme` into line with the \LaTeX kernel naming convention. Also notice that `\listofschemes` no longer accepts an optional argument. The standard output is illustrated below.

List of Schemes

1	A scheme with no compound numbers.	3
2	A scheme that is not a Scheme!	4
3	A flush-left scheme.	5

4	A flush-right scheme.	5
5	A scheme with temporary compound numbers.	6
6	A scheme with automated compound numbers.	7
7	A scheme with explicitly numbered temporary labels.	7
8	A scheme with altered label formatting.	8

2.2 Altering the defaults

`floats` There are a number of methods available to generate new float types. The `memoir` class can do this internally, or the `float` or `floatrow` packages can be loaded to provide the necessary system. When `memoir` is used, `chemscheme` uses the internal systems for generating new floats; for the standard classes, `floatrow` is used by default. The `floats` package option can be used to control which method is employed. This uses the key-value system, can be given the values `float`, `floatrow` or `memoir`:

```
\usepackage[floats=float]{chemscheme}
```

or

```
\usepackage[floats=floatrow]{chemscheme}
```

or

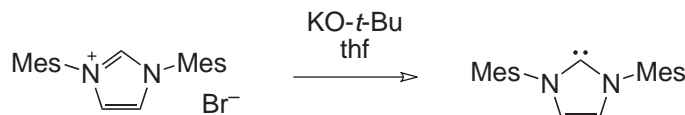
```
\usepackage[floats=memoir]{chemscheme}
```

The package will use the specified method if available (so `memoir` only works if using the `memoir` class!). If either `float` or `floatrow` is loaded before `chemscheme`, this forces the choice.

`\schemename` `\schemename` contains the text used in scheme captions (by default `Scheme`). This is used in the same manner as `\figurename` or `\tablename` to set up the text used in scheme captions.

```
\renewcommand*{\schemename}{Illustration}
\begin{scheme}[ht]
  \includegraphics{scheme-one}
  \caption{A scheme that is not a Scheme!}
\end{scheme}
```

Illustration 2: A scheme that is not a Scheme!



2.3 babel support

Schemes are provided with some support for `babel`. Currently, in addition to English, `chemscheme` provides alternatives for `\schemename` and

`\listschemename` in French and German. Users of other languages are encouraged to supply suitable translations for inclusion in future versions of chemscheme.

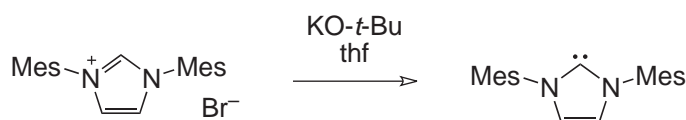
3 Horizontal positioning of all floats

`\floatcontentscentre`
`\floatcontentscenter`
`\floatcontentsleft`
`\floatcontentsright`

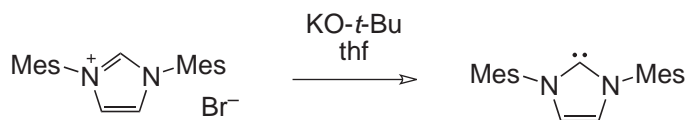
The \LaTeX default is to position all float contents flush-left. There is no “hook” provided to alter this. The chemscheme package therefore provides commands to align all float contents automatically. As the macro names make clear, `\floatcontentscentre` will make all floats centred (for users speaking U.S. English, the alternative spelling `\floatcontentscenter` is also available). The default behaviour is restored using the command `\floatcontentsleft`. Finally, `\floatcontentsright` is provided for use if needed. Notice that the float positioning commands should be given *outside* floating environments, and apply to all subsequent floats.

```
\floatcontentsleft
\begin{scheme} [ht]
  \includegraphics{scheme-one}
  \caption{A flush-left scheme.}
\end{scheme}
\floatcontentsright
\begin{scheme} [ht]
  \includegraphics{scheme-one}
  \caption{A flush-right scheme.}
\end{scheme}
\floatcontentscentre
```

Scheme 3: A flush-left scheme.



Scheme 4: A flush-right scheme.



It is important to note that the positioning mechanism used here relies on a low-level hack of the \LaTeX kernel. This has been tested with the standard \LaTeX classes, the memoir class and the KOMA-Script bundled. Other document classes may not give the desired behaviour.

4 Reference numbers in graphics

4.1 Background

There are a number of packages available on CTAN for tracking compound reference numbers. The two with the most up to date and comprehensive features are `bpchem` and `chemcompounds`. Both allow in-text numbering to be handled automatically. However, neither will allow the use of these numbers directly in schemes, figures, *etc.* Both leave it to the user to manually adapt schemes to match any changes in numbering.

The `chemscheme` package provides a mechanism for rectifying this issue. The package makes use of the `psfrag` package, which means that it can only directly produce DVI output (using \LaTeX). However, direct PDF output using \PDF\LaTeX is possible: see Section 4.3.

`\chemscheme`
`\chemscheme`
`\chemscheme`

Users upgrading from v1.2 should note that “chem” has been removed from the start of most macro names. The main referencing commands `\chemscheme` and `\chemscheme` are retained for backward compatibility.

4.2 Usage

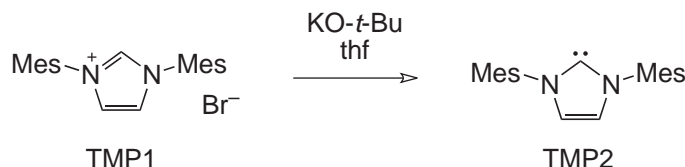
`\schemerefer`

Getting automated numbers into schemes is a two step procedure. In the first step, schemes (or other graphics) should be prepared as normal and saved as encapsulated postscript (EPS) files. The most popular chemistry drawing package, `CHEMDRAW`, is able to do this from the *Save As . . .* dialog. The positions where the auto-labels should be have to be marked in the EPS file. The marker should consist of an “indicator” that the text is to be replaced, followed by a reference number or letter. For automated substitution, the “indicator” text should be the same in all graphics; the value it is stored in `\schemerefer`, and has default value `TMP`. Thus the graphics should contain labels `TMP1`, `TMP2`, *etc.* A suitable unmodified graphic is shown in in the next example.

```
\begin{scheme} [ht]
  \includegraphics{scheme-two}
  \caption{A scheme with temporary compound numbers.}
\end{scheme}
```

The value of `\schemerefer` may be altered as normal, so for example

Scheme 5: A scheme with temporary compound numbers.



to use XXX as the indicator for replacement in all graphics, you would execute:

```
\renewcommand*{\schemerefer}{XXX}
```

`\schemerefer`

In the second step, the command `\schemerefer` is used to indicate the map-

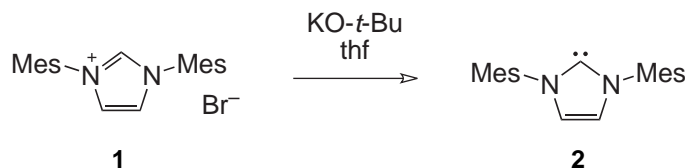
ping of the temporary markers to the automatically-managed numbering. The syntax of the command is `\chemschemeref[⟨temp-marker⟩]{⟨label⟩}`, where `⟨temp-marker⟩` is the marker used in the graphic, and `⟨label⟩` is the name assigned to the compound by the user. By default, `chemscheme` will assume that `⟨temp-marker⟩` consists of the marker plus a number, beginning at 1 and incrementing by 1 for each additional structure inside one float. Each replacement requires a separate `\chemschemeref`, all of which should appear before the relevant `\includegraphics` command.

An example will make usage clearer. In the example used in this document, the starting material is given label `IMesHCl` and the product is called `IMes`. As is shown in the next example, in the EPS file these are labelled `TMP1` and `TMP2`, respectively. The automated package defaults are used.

```
\begin{scheme} [ht]
  \schemeref{IMesHCl}
  \schemeref{IMes}
  \includegraphics{scheme-two}
  \caption{A scheme with automated compound numbers.}
\end{scheme}
```

With user-specified information on the text to be replaced, the entire text to

Scheme 6: A scheme with automated compound numbers.

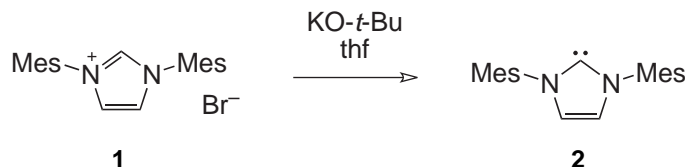


be matched must be given.¹

```
\begin{scheme} [ht]
  \schemeref[TMP1]{IMesHCl}
  \schemeref[TMP2]{IMes}
  \includegraphics{scheme-two}
  \caption{A scheme with explicitly numbered temporary labels.}
\end{scheme}
```

Notice that the new label is centred on the middle of the temporary marker, with

Scheme 7: A scheme with explicitly numbered temporary labels.



the same baseline. This should allow the user to obtain good alignment of labels and structures.

`\schemerefsub`

As described in Section 4.4, `chemscheme` supports `bpchem`, which allows

¹In this example, this is redundant as the automated system will work fine.

tracking of sub-labels (**1a**, **1b**, *etc.*). To allow use of these in schemes, the `\schemerefsb` command is provided. This takes an additional argument $\langle sub-label \rangle$, which is used to generate the appropriate text. When used with `chemcompounds`, this command will gobble its arguments and issue a warning; no substitution will take place.

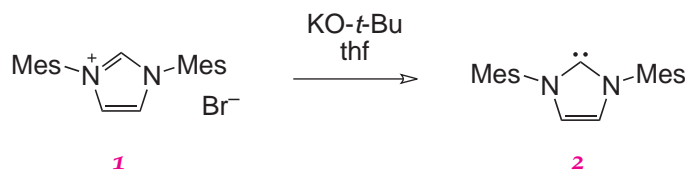
`\schemerefformat`

The format of chemical references is controlled by the underlying package, `bpchem` or `chemcompounds`. However, it is useful to be able to specify additional formatting for schemes. By default, `chemscheme` formats all reference numbers in a sans serif font. This is controlled by `\schemerefformat`.

```
% This needs the color or xcolor package loaded
\renewcommand*{\schemerefformat}
{ \color{magenta}\textit}
\begin{scheme} [ht]
  \schemeref{IMesHCl}
  \schemeref{IMes}
  \includegraphics{scheme-two}
  \caption{A scheme with altered label formatting.}
\end{scheme}
```

The additional formatting applied within schemes may be altered by redefi-

Scheme 8: A scheme with altered label formatting.



ning `\schemerefformat`. By careful choice of the font commands given here, good visual matching should be obtained between the automatically-generated labels and other text in the scheme. In this document, the `CHEMDRAW` source uses 10 point Arial, with the \LaTeX sans serif font provided by the `helvet` package, loaded scaled to 95 %, *i.e.*:

```
\usepackage[scaled=0.95]{helvet}
```

The `CHEMDRAW` file used to generate the example schemes is included with the package as `chemscheme.cdx`.

4.3 chemscheme and PDF \LaTeX

The automatic substitution of numbers in graphics relies on the `psfrag` package. This works with `POSTSCRIPT` files, and cannot therefore be used with `PDF \LaTeX` . However, all is not lost as the `pst-pdf` package provides a method for including `POSTSCRIPT` files in a `PDF \LaTeX` run. However, this is not automatic and some effort is needed by the user.² The example below shows a example for using `PDF \LaTeX` with `chemscheme`.

```
1 \documentclass{article}
2 \usepackage[T1]{fontenc}
```

²The rest of this section is based closely on an example by Stefan Pinnow.


```

3 \usepackage{graphicx,chemscheme}
4 % Remove 'inactive' after the first LaTeX run
5 \usepackage[inactive,final]{pst-pdf}
6 \begin{document}
7 \floatcontentscentre
8 An example file for PDFLaTeX use.
9 \begin{scheme}
10 \schemeref{IMesHCl}
11 \schemeref{IMes}
12 \includegraphics{scheme-two}
13 \caption{A scheme with automated compound numbers.}
14 \label{sc:scheme-one}
15 \end{scheme}
16 \end{document}

```

First you need to run the file through \LaTeX , so your package can make the replacements in the picture. Then you need another run through \LaTeX with the `inactive` option `pst-pdf` removed so that the modified pictures are extracted.³ Do not worry that you end up with a very odd looking DVI! Then you have to convert the extracted pictures to PDF by the following commands

```

dvips -o \jobname-pics.ps \jobname.dvi
ps2pdf \jobname-pics.ps

```

This converts the modified graphics into PDF format. After this, you can use \PDFLaTeX as normal for your schemes. Notice that you will have to repeat the process if you need to modify the schemes or numbering in any way.

4.4 Managing chemical numbering

bpchem
chemcompounds
tracking

The `chemscheme` package can use one of two packages for management of chemical numbering: `bpchem` and `chemcompounds`. As of v1.3, `chemcompounds` is the default package for managing reference numbers.⁴ Both packages have advantages: `bpchem` allows the tracking of sub-references (very common in organic chemistry), whilst `chemcompounds` has a very well thought-out interface. It is technically feasible to support both simultaneously, but this is unlikely to have wide application. For this reason, `chemscheme` loads only one package, and uses this package to provide numbering management. The `chemscheme` package recognises the `bpchem` and `chemcompounds` to control this; the key-value version `tracking=...` can also be given.

5 Generating chemical schemes

5.1 Overview

There are a number of ways of generating the graphical content of schemes. The choice of method will depend on what is available to the user, and how complex

³You can simply do the first \LaTeX run without loading `pst-pdf` at all, if you prefer.

⁴The change from `bpchem` is due to issues with `hyperref` support. The method used by `bpchem` to generate compound labels means that they are made into hyperlinks by `hyperref`, and therefore end up coloured when using the `colorlinks` option. This is unlikely to be the desired effect, and `chemcompounds` does not behave in this way.

the schemes desired are. In this section, an overview of several approaches is given.⁵ The package author, who is a research worker in a university, favours using CHEMDRAW as it is regarded by many synthetic chemists as the best tool for this job. However, this is clearly overkill for users requiring a single diagram on a one-off basis. CHEMDRAW is also a commercial package running only under Windows and the MacOS. The following is necessarily somewhat brief and selective. For a thorough overview of graphics in L^AT_EX, see Goossens *et al.* [1].

5.2 Macro-based methods

At the most basic, a chemical scheme is simply a collection of lines and symbols, as with any vector diagram. Hence, it is possible to construct schemes directly using packages such as PSTricks or pgf/Tikz. This is a complex method, and cannot be recommended for anyone except the very experienced and brave.

At a more practical level, there are two packages available which allow type-setting of chemical structures in (La)T_EX, using specialised commands: XyMT_EX and ppchT_EX. Recent versions of the XyMT_EX package have not been made available on CTAN, and the version held there is therefore considered to be obsolete. On the other hand, the ppchT_EX system, developed originally for ConT_EXt, is available. Both systems suffer from the lack of chemical logic in the input: it is very hard to tell from the code what is being represented. Drawing items such as “curly arrows”, or making subtle alterations to positioning, is very challenging in purely macro-based systems. For these reasons, it is usually much more sensible to examine the available graphical methods.

5.3 Graphical methods

Moving to graphical systems, there is no reason that general-purpose vector drawing packages cannot be used for schemes. There are obviously several commercial (CORELDRAW, ADOBE ILLUSTRATOR, *etc.*) and freeware (for example the GIMP) drawing packages that can be used in this way. Simply rings and lines can easily be constructed, although in general-purpose programs the user has to watch that all bonds are the same length.

For producing a large number of complex schemes, the particular abilities of dedicated software become a necessity. As well as the already-mentioned CHEMDRAW, programs such as ISIS DRAW and CHEMSKETCH are available free for personal use;⁶ these programs are all Windows specific. In the open-source arena, there are a number of packages such as XDRAWCHEM and BKCHEM, which offer cross-platform functionality. The differences between the various packages are in the ease of use, and ability to generate well-formatted output (for example, aligning structures).

One which deserves mention for the T_EX user is TpX. This is a general purpose Windows graphics program specifically aimed at producing T_EX-friendly output (such as PSTricks and tikz code) from a graphical interface. TpX can accept clipboard data from other programs, so can be used to produce EPS files from programs which do not have native export facilities (such as ISIS DRAW).

⁵Thanks to Norwid-R. Behrnd for suggesting this section and giving a number of useful examples and tips.

⁶“Free” as in without charge, not as in open source.

6 Known issues

The interaction of the different document classes, with options, plus the babel system means ensuring every possibility is covered is impossible. Users are asked to report any problems with compatibility with other packages or emulation of the standard float types. Additional babel strings are also welcome, as are improvements to those already provided.

7 Implementation

7.1 Setup code

```
\csh@id The initial code goes through the usual steps of identifying the package.
17 \NeedsTeXFormat{LaTeX2e}
18 \def\csh@id$#1: #2.#3 #4 #5-#6-#7 #8 #9${%
19   #5/#6/#7\space v1.4a\space}
20 \ProvidesPackage{chemscheme}
21   [\csh@id $Id: chemscheme.dtx 48 2008-07-31 19:22:43Z joseph $
22   Support for chemical schemes]

\ifcsh@bpchem Package options are handled using the key-value method, with kvoptions provi-
ding the back-end. Most of the code here is pretty basic; the aim is to set up the
correct handling of float-creation and compound tracking.
23 \RequirePackage{kvoptions,psfrag,iflang}
24 \SetupKeyvalOptions{
25   family=csh,
26   prefix=csh@}
27 \newif\ifcsh@bpchem
28 \define@key{csh}{tracking}
29   {\csh@bpchemtrue
30    \lowercase{\edef\@tempa{#1}}%
31    \def\@tempb{bpchem}%
32    \ifx\@tempa\@tempb\else
33      \def\@tempb{chemcompounds}%
34      \ifx\@tempa\@tempb
35        \csh@bpchemfalse
36      \else
37        \PackageError{chemscheme}
38          {Unknown value '#1' for option numbers}
39          {The 'tracking' option accepts values 'bpchem'
40           and 'chemcompounds'}%
41      \fi
42    \fi}
43 \DeclareVoidOption{chemcompounds}{\csh@bpchemfalse}
44 \DeclareVoidOption{bpchem}{\csh@bpchemtrue}
45 \DeclareVoidOption{chapter}
46   {\PackageInfo{chemscheme}
47     {Ignoring obsolete option 'chapter'}}

\csh@fltpkg The provision of floats can be handled by floatrow, float or memoir. The user is
given an option, but if one of the packages is already loaded then this has to be
taken into account.
```

```

48 \define@key{csh}{floats}
49   {\lowercase{\renewcommand*{\csh@fltpkg}{#1}}}%
50   \def\@tempa{floatrow}%
51   \ifx\csh@fltpkg\@tempa\else
52     \def\@tempa{float}%
53     \ifx\csh@fltpkg\@tempa\else
54       \def\@tempa{memoir}%
55       \ifx\csh@fltpkg\@tempa
56         \@ifclassloaded{memoir}{}
57         {\PackageWarning{chemscheme}
58           {You asked for floats to be created using memoir,
59             \MessageBreak but have used a different document
60             class\MessageBreak Using floatrow instead}%
61           \renewcommand*{\csh@fltpkg}{floatrow}}}%
62     \else
63       \PackageError{chemscheme}
64       {Unknown value '#1' for option numbers}
65       {The 'floats' option accepts values 'float',
66        'floatrow' and 'memoir'}%
67     \fi
68   \fi
69 \fi}
70 \@ifpackageloaded{floatrow}
71   {\newcommand*{\csh@fltpkg}{floatrow}%
72   \define@key{csh}{floats}
73     {\PackageInfo{chemscheme}{Package floatrow already
74       loaded\MessageBreak Option 'floats' disabled}}}
75   {\@ifpackageloaded{float}
76     {\newcommand*{\csh@fltpkg}{float}%
77     \define@key{csh}{floats}
78       {\PackageInfo{chemscheme}{Package float already
79         loaded\MessageBreak Option 'floats' disabled}}}
80     {\@ifclassloaded{memoir}
81       {\newcommand*{\csh@fltpkg}{memoir}}
82       {\newcommand*{\csh@fltpkg}{floatrow}}}}
83 \ProcessKeyvalOptions{csh}

```

\csh@load@memoir
\csh@load@float
\csh@load@floatrow

The necessary support is loaded by using an appropriately-named macro. For memoir, nothing actually has to be done.

```

84 \newcommand*{\csh@load@memoir}{}
85 \newcommand*{\csh@load@float}{%
86   \@ifpackageloaded{float}{}
87   {\let\newfloat\undefined}
88   \RequirePackage{float,caption}}
89 \newcommand*{\csh@load@floatrow}{%
90   \@ifpackageloaded{floatrow}{}
91   {\let\newfloat\undefined}
92   \RequirePackage{floatrow,caption}}
93 \csname csh@load@\csh@fltpkg\endcsname

```

7.2 Support for schemes

`\schemename` The default name for a scheme, and the default title for the list of schemes, are provided. Both command names follow the kernel conventions for figures and tables; unlike the kernel versions, however, these macros are not `\long`.

```
94 \newcommand*{\schemename}{Scheme}
95 \newcommand*{\listschemename}{List of Schemes}
```

`scheme` As with package loading, three different macros are defined here to create the new float type. As well as the basics of creating the float, some work is needed as adjustments must be made to match the style of the kernel floats. For the `floatrow` package, things have to be done in the preamble, whereas for the others everything happens later.

```
\listofschemes
\csh@makesch@memoir
\csh@mksch@memoir
\csh@makesch@float
\csh@mksch@float
\csh@makesch@floatrow
96 \newcommand*{\csh@makesch@memoir}{%
97   \AtBeginDocument{\csh@mksch@memoir}}
98 \newcommand*{\csh@mksch@memoir}{%
99   \newfloat[chapter]{scheme}{los}{\schemename}
100   \kill@lastcounter{losdepth}
101   \renewcommand*{\thescheme}{\thechapter.\@arabic\c@scheme}
102   \addtocontents{smemfront}{\counterwithout{scheme}{chapter}}
103   \addtocontents{smemmain}{}%
104   \ifartopt\else
105     \counterwithin{scheme}{chapter}
106   \fi
107   \addtocontents{backmatter}{}%
108   \ifartopt\else
109     \counterwithout{scheme}{chapter}%
110     \setcounter{scheme}{0}%
111   \fi
112   \ifartopt
113     \counterwithout{scheme}{chapter}%
114   \fi
115   \newlistof{listofschemes}{los}{\listschemename}
116   \kill@lastcounter{losdepth}
117   \newlistentry[chapter]{scheme}{los}{0}
118   \cftsetindents{scheme}{0em}{2.3em}
119   \addtocontents{insertchapterspace}{}%
120   {\addtocontents{los}{\protect\addvspace{10pt}}}
121   \ifundefined{c@losdepth}%
122     {\newcounter{losdepth}\setcounter{losdepth}{1}}
123 \newcommand*{\csh@makesch@float}{%
124   \AtBeginDocument{\csh@mksch@float}}
125 \newcommand*{\csh@mksch@float}{%
126   \@ifundefined{chapter}
127     {\newfloat{scheme}{tbp}{los}}
128     {\newfloat{scheme}{tbp}{los}[chapter]%
129     \csh@fixchapter}%
130   \floatname{scheme}{\schemename}
131   \newcommand*{\listofschemes}{%
132     \listof{scheme}{\listschemename}}}
```

Using `floatrow`, the definition depends on whether chapters are in use.

```
133 \newcommand*{\csh@makesch@floatrow}{%
134   \@ifundefined{chapter}
```

```

135     {\DeclareNewFloatType{scheme}
136      {fileext=los,placement=tbp,name=Scheme}}
137     {\DeclareNewFloatType{scheme}
138      {fileext=los,placement=tbp,name=Scheme,within=chapter}%
139      \csh@fixchapter}%
140     \newcommand*{\listofschemes}{%
141      \listof{scheme}{\listschemename}}

```

\csh@fixchapter When not using memoir, chapters may or may not be defined. This means a little extra work.

```

\csh@chapter
\@chapter
142 \newcommand*{\csh@fixchapter}{%
143   \@ifundefined{KOMAScriptVersion}
144   {\renewcommand*{\thescheme}{%
145     \ifnum\c@chapter>\z@ \thechapter.\fi \@arabic\c@scheme}}
146   {}%
147   \let\csh@chapter\@chapter
148   \renewcommand{\@chapter}{%
149     \addtocontents{los}{\protect\addvspace{10\p@}}%
150     \csh@chapter}}

```

The appropriate creation macro is now called.

```

151 \csname csh@makesch@\csh@fltpkg\endcsname

```

7.3 Positioning float contents

\floatcontentscentre When using floatrow, centring all floats is easy. This is not the case for other methods, and so some macros are provided. The definitions depend on whether floatrow is in use.

```

\floatcontentscenter
\floatcontentsleft
\floatcontentsright
152 \def\@tempa{floatrow}
153 \ifx\@tempa\csh@fltpkg
154   \newcommand*{\floatcontentscentre}{%
155     \floatsetup{objectset=centering}}
156   \newcommand*{\floatcontentscenter}{%
157     \floatsetup{objectset=centering}}
158   \newcommand*{\floatcontentsright}{%
159     \floatsetup{objectset=raggedleft}}
160   \newcommand*{\floatcontentsleft}{%
161     \floatsetup{objectset=raggedright}}

```

\csh@floatboxreset When floatrow is not in use, then a low-level hack of \@floatboxtreset is used.

```

\@floatboxreset
\csh@everyfloat
162 \else
163   \let\csh@floatboxreset\@floatboxreset
164   \renewcommand*{\@floatboxreset}{%
165     \csh@everyfloat
166     \csh@floatboxreset}
167   \newcommand*{\floatcontentscentre}{%
168     \let\csh@everyfloat\centering}
169   \newcommand*{\floatcontentscenter}{%
170     \let\csh@everyfloat\centering}
171   \newcommand*{\floatcontentsleft}{%
172     \let\csh@everyfloat\relax}
173   \newcommand*{\floatcontentsright}{%

```

```

174 \let\csh@everyfloat\raggedleft}
175 \fi
176 \floatcontentsleft

```

7.4 babel support

`\ifcsh@babel` To make the new float type work with `babel`, some alternative text is provided for non-English users. First, we need to know if `babel` is loaded *before* `chemscheme` or after, as this affects one if the hacks needed for French.

```

177 \newif\ifcsh@babel
178 \@ifpackageloaded{babel}
179 {\csh@babeltrue}
180 {}

```

`\floatc@plain` There is then one hack to be made for French which has to be made before the beginning of the document *if* `babel` is loaded before `chemscheme`. This is only needed when `memoir` is not in use. So there is a run of tests to see if the conditions for the hack are met. `babel` support for French needs two name checks, *zut alors!*

```

181 \@ifclassloaded{memoir}{}
182 {\@ifpackageloaded{babel}
183   {\IfLanguageName{french}
184     {\let\floatc@plain\FB@makecaption}
185     {\IfLanguageName{frenchb}
186       {\let\floatc@plain\FB@makecaption}
187       {}}}
188 {}

```

The definition of new stings is delayed until the beginning of the document, so that things work if `babel` is loaded after `chemscheme`. The multiple language names mean quite a bit of repetition here.

```

189 \AtBeginDocument{
190   \@ifpackageloaded{babel}
191     {\addto{\captionsngerman}{%
192       \renewcommand*{\schemename}{Schema}}
193     \addto{\captionsngerman}{%
194       \renewcommand*{\listschemename}{Schemenverzeichnis}}
195   \addto{\captionsngerman}{%
196     \renewcommand*{\schemename}{Schema}}
197   \addto{\captionsngerman}{%
198     \renewcommand*{\listschemename}{Schemenverzeichnis}}

```

French settings. `babel` also changes some style parameters here, so `chemscheme` tries to match this.

```

199   \addto{\captionsfrench}{%
200     \renewcommand*{\schemename}{\scshape Sch\`eme}}
201   \addto{\captionsfrench}{%
202     \renewcommand*{\listschemename}{Table des sch\`emes}}
203   \addto{\captionsfrenchb}{%
204     \renewcommand*{\schemename}{\scshape Sch\`eme}}
205   \addto{\captionsfrenchb}{%
206     \renewcommand*{\listschemename}{Table des sch\`emes}}

```

To make these changes, `babel` has to be instructed to reload the current language.

```
207 \expandafter\selectlanguage\expandafter{\language}
```

If `babel` was loaded after `chemscheme`, and the document is in French, then the non-memoir patch for floats is needed here. This is the same one described earlier, which cannot be delayed to here if `babel` is loaded first.

```
208 \@ifclassloaded{memoir}{}
209 {\IfLanguageName{french}
210   {\ifcsh@babel\else
211     \let\floatc@plain\FB@makecaption
212     \fi}
213   {\IfLanguageName{frenchb}
214     {\ifcsh@babel\else
215       \let\floatc@plain\FB@makecaption
216       \fi}
217   {}}}
218 {}{}{}}
```

7.5 Reference numbers in graphics

`\schemerefmarker` The two macros `\schemerefmarker` and `\schemerefformat` are used to allow customisation of the behaviour of the package. Here defaults are provided.

```
219 \newcommand*{\schemerefmarker}{TMP}
220 \newcommand*{\schemerefformat}{\textsf}
```

`\csh@label` Depending on the user options provided, either `bpchem` or `chemcompounds` is loaded to manage chemical citations. The macro `\chemsch@label` is defined as the labelling command of the appropriate package.

```
221 \ifcsh@bpchem
222   \RequirePackage{bpchem}
223   \let\csh@label\CNlabel
224 \else
225   \RequirePackage{chemcompounds}
226   \let\csh@label\compound
227 \fi
```

`\csh@num` A counter is needed for automatic substitution of reference numbers. This has to be a \TeX count, rather than a \LaTeX counter, as it has to be local.

```
228 \newcount\csh@num
```

`\schemeref` The user macro for referencing a compound in a graphic. First the automatic counter is incremented.

```
229 \newcommand*{\schemeref}{%
230   \advance\csh@num\@ne
```

Now the presence of an optional argument is tested for. If there is not one, the value of the automatic counter is used.

```
231   \@ifnextchar[%]
232     {\csh@schemeref}
233     {\csh@schemeref[\schemerefmarker\the\csh@num]}}
```


\csh@schemeref	<p>The internal command to substitute the temporary marker by the automatic label text. The use of [b] [b] ensures that the new text is centred on the position of the marker.</p> <pre> 234 \def\csh@schemeref[#1]#2{% 235 \psfrag{#1}[b][b] 236 {\schemerefformat{\csh@label{#2}}}}</pre>
\csh@schemerefsup	<p>For sub-referencing using bpchem, a similar macro is needed using the \CNlabelsub command. This is defined here, with the user macro later.</p> <pre> 237 \def\csh@schemerefsup[#1]#2#3{% 238 \psfrag{#1}[b][b] 239 {\schemerefformat{\CNlabelsub{#2}{#3}}}}</pre>
\chemschemeref	<p>The \chemschemeref is now a backward-compatibility wrapper for \schemeref.</p> <pre> 240 \newcommand*\chemschemeref[1]{\schemeref[#1]}</pre>
\schemerefsup \chemschemerefsup	<p>The user sub-referencing macro is defined in a package-dependent manner. If bpchem is loaded, then \schemerefsup works in the same way as \schemeref, but calling the sub version of the internal macro.</p> <pre> 241 \ifcsh@bpchem 242 \newcommand*\schemerefsup{% 243 \advance\csh@num\@ne 244 \@ifnextchar[%] 245 {\csh@schemerefsup} 246 {\csh@schemerefsup[\schemerefmarker\the\csh@num]}} 247 \newcommand*\chemschemerefsup[1]{\schemerefsup[#1]}</pre> <p>If chemcompounds is in use, then the sub-reference commands gobble the arguments given. The automatic counter is still incremented, so that things stay (reasonably) logical.</p> <pre> 248 \else 249 \newcommand*\schemerefsup[3][[]]{% 250 \advance\csh@num\@ne 251 \PackageWarning{chemscheme} 252 {'chemcompound' option active\MessageBreak 253 Command \string\chemschemerefsup\space not used 254 \MessageBreak Please alter your source to \MessageBreak 255 \string\chemschemeref}} 256 \newcommand*\chemschemerefsup{\schemerefsup} 257 \fi</pre>

8 Change History

v1.0	
General: Initial public release	1
v1.1	
General: Added float centring code	1
v1.2	
General: Improved emulation of standard float types	1
Removed chapter option	1
\listschemename: Replaces \listschemes	13
v1.2a	
General: Documentation for PDF use added	1
License changed from GPL to LPPL	1
v1.3	
General: Improved French and Ger- man babel support	1
\chemschemeref: Converted to a wrapper for \schemeref	17
\chemschemerefsup: Now a wrapper for \schemerefsup	17
\schemeref: New macro	16
\schemerefformat: Name change	16
\schemerefmarker: Name change	16
\schemerefsup: New macro	17
v1.4	
General: Added support for floatrow	1
Float creation rewritten	12
Option handling completely re- written	11
v1.4a	
\csh@makesch@floatrow: Fixed bug with float definition, floa- trow and chapters	13

9 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols

\@chapter	142
\@floatboxreset	162
\@smemfront	102
\@smemmain	103
\`	200, 202, 204, 206

B

bpchem (option)	9
-----------------------	---

C

\c@scheme	101, 145
chemcompounds (option)	9
\chemschemeref	6, 240, 255
\chemschemerefsup	6, 241
\csh@babeltrue	179
\csh@bpchemfalse	35, 43
\csh@bpchemtrue	29, 44
\csh@chapter	142
\csh@everyfloat	162
\csh@fixchapter	129, 139, 142
\csh@floatboxreset	162
\csh@fltpkg	48, 93, 151, 153
\csh@id	17
\csh@label	221, 236
\csh@load@float	84
\csh@load@floatrow	84
\csh@load@memoir	84
\csh@makesch@float	96
\csh@makesch@floatrow	96
\csh@makesch@memoir	96
\csh@mksch@float	96
\csh@mksch@memoir	96
\csh@num ..	228, 230, 233, 243, 246, 250
\csh@schemeref	232, 233, 234
\csh@schemerefsup	237, 245, 246

E

environments:

scheme	3, 96
--------------	-------

F

\floatc@plain	181, 211, 215
\floatcontentscenter	5, 152
\floatcontentscentre	5, 7, 152
\floatcontentsleft	5, 152
\floatcontentsright	5, 152
floats (option)	4

I

\ifcsh@babel	177, 210, 214
\ifcsh@bpchem	23, 221, 241

L

\listofschemes	3, 96
\listschemename	3, 20
94, 115, 132, 141, 194, 198, 202, 206	

O

options:

bpchem	9
chemcompounds	9
floats	4
tracking	9

10 References

- [1] Michel Goossens, Frank Mittelbach, Sebastian Rahtz, and Denis Roegel. *The LaTeX Graphics Companion*. Tools and Techniques for Computer Typesetting. Addison Wesley, 2 edition, 2007.