chemscheme — Support for chemical schemes *

Joseph Wright †

Released 2008/07/17

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 v1.4, last revised 2008/07/17.

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

Contents

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

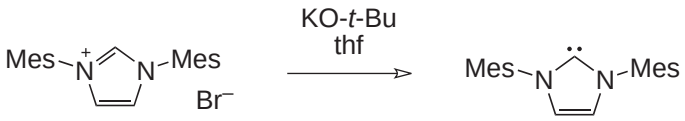
2.1 Basic use

`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

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 can be given the values `float`, `floatrow` or `memoir`:

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

or

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

or

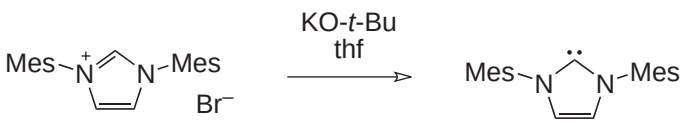
```
\usepackage[floats=memoir]{chemescheme}
```

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.

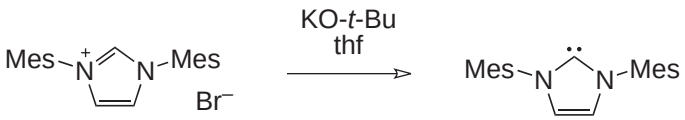
3 Horizontal positioning of all floats

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

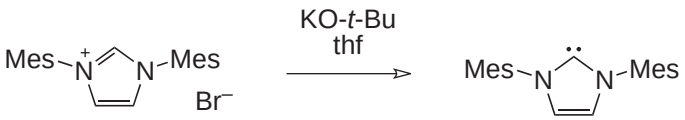
The LaTeX default is to position all float contents flush-left. There is no "hook" provided to alter this. The chemscheme packages therefore provides commands to align all float contents automatically. As the macro names make clear, `\floatcontentscentre` will make all floats centred (for users speaking 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 PDFLaTeX is possible: see Section 4.3.

`\chemschemeref` `\chemschemerefsub`

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

4.2 Usage

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 `\schemerefmarker`, 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.

`\schemerefmarker`

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

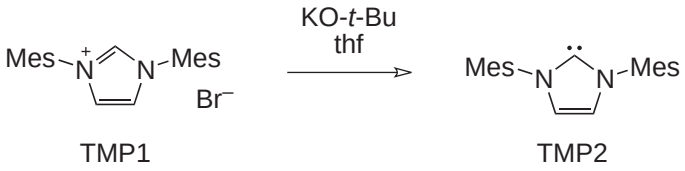
The value of `\schemerefmarker` may be altered as normal, so for example to use XXX as the indicator for replacement in all graphics, you would execute:

```
\renewcommand*{\chemeschemerefmarker}{XXX}
```

`\schemeref`

In the second step, the command `\schemeref` is used to indicate the mapping of the temporary markers to the automatically-managed numbering. The syntax of the command is `\chemschemeref[`⟨*temp-marker*⟩`]{`⟨*label*⟩`}`, where

Scheme 5: A scheme with temporary compound numbers.



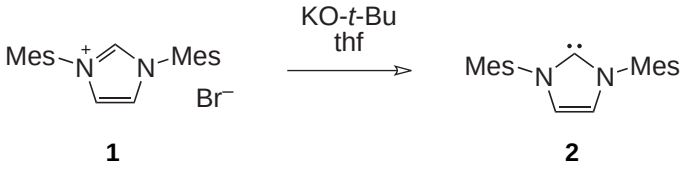
⟨*temp-marker*⟩ is the marker used in the graphic, and ⟨*label*⟩ is the name assigned to the compound by the user. By default, chemscheme will assume that ⟨*temp-marker*⟩ consists of the marker plus a number, beginning at 1 and incrementing by 1 for each additional structure inside one float. Each replacement 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]

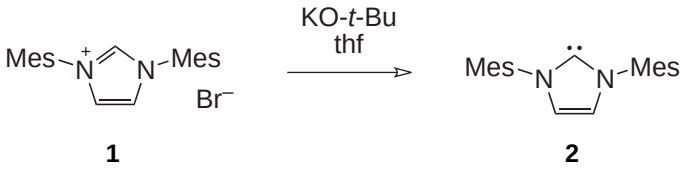
```
\begin{scheme}[ht]
  \schemeref[TMP1]{IMesHCl}
  \schemeref[TMP2]{IMes}
  \includegraphics{scheme-two}
  \caption{A scheme with explicitly numbered temporary labels.}
\end{scheme}
```

Notice that the new label is centred on the middle of the temporary marker, with 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 tracking of sub-labels (**1a**, **1b**, *etc.*). To allow use of these in schemes, the

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

Scheme 7: A scheme with explicitly numbered temporary labels.



`\schemerefsub` command is provided. This takes an additional argument `{`⟨*sub-label*⟩`}`, 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.

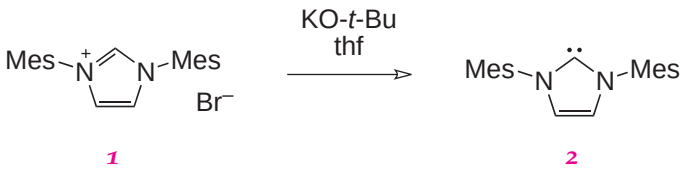
`\schemerefformat`

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 redefin-

Scheme 8: A scheme with altered label formatting.



ing `\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 PDFLaTeX

The automatic substitution of numbers in graphics relies on the the psfrag package. This works with PostScript files, and cannot therefore be used with PDFLaTeX.

However, all is not lost as the pst-pdf package provides a method for including PostScript files in a PDFLaTeX run. However, this is not automatic and some effort is needed by the user.[2] The example below shows a example for using PDFLaTeX with chemscheme.

```
\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage{graphicx,chemscheme}
% Remove 'inactive' after the first LaTeX run
\usepackage[inactive,final]{pst-pdf}
\begin{document}
\floatcontentscentre
An example file for PDFLaTeX use.
\begin{scheme}
  \schemeref{IMesHCl}
  \schemeref{IMes}
  \includegraphics{scheme-two}
  \caption{A scheme with automated compound numbers.}
  \label{sc:scheme-one}
\end{scheme}
\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.

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

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

[4] The 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.

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 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 LaTeX, 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/Ti*k*z. 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 typesetting of chemical structures in (La)TeX, using specialised commands: XyMTeX and ppchTeX. Recent versions of the XyMTeX package have not been made available on CTAN, and the version held there is therefore considered to be obsolete. On the other hand, the ppchTeX system, developed originally for ConTeXt, 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-formattd output (for example, aligning structures).

[5] Thanks 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.

One which deserves mention for the TEX user is TpX. This is a general purpose Windows graphics program specifically aimed at producing TEX-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).

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 stings 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.

```
\NeedsTeXFormat{LaTeX2e}
\def\csh@id$#1: #2.#3 #4 #5-#6-#7 #8 #9${%
    #5/#6/#7\space v1.4\space}
\ProvidesPackage{chemscheme}
  [\csh@id $Id: chemscheme.dtx 36 2008-07-17 07:39:53Z joseph $
   Support for chemical schemes]
```

\ifcsh@bpchem Package options are handled using the key–value method, with kvoptions providing 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.

```
\RequirePackage{kvoptions,psfrag,iflang}
\SetupKeyvalOptions{
  family=csh,
  prefix=csh@}
\newif\ifcsh@bpchem
\define@key{csh}{tracking}
  {\csh@bpchemtrue
   \lowercase{\edef\@tempa{#1}}%
   \def\@tempb{bpchem}%
   \ifx\@tempa\@tempb\else
     \def\@tempb{chemcompounds}%
     \ifx\@tempa\@tempb
       \csh@bpchemfalse
     \else
       \PackageError{chemscheme}
         {Unknown value '#1' for option numbers}
         {The 'tracking' option accepts values 'bpchem'
          and 'chemcompounds'}%
     \fi
   \fi}
\DeclareVoidOption{chemcompounds}{\csh@bpchemfalse}
```

```
\DeclareVoidOption{bpchem}{\csh@bpchemtrue}
\DeclareVoidOption{chapter}
  {\PackageInfo{chemscheme}
    {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.

```
\define@key{csh}{floats}
  {\lowercase{\renewcommand*{\csh@fltpkg}{#1}}%
   \def\@tempa{floatrow}%
   \ifx\csh@fltpkg\@tempa\else
     \def\@tempa{float}%
     \ifx\csh@fltpkg\@tempa\else
       \def\@tempa{memoir}%
       \ifx\csh@fltpkg\@tempa
         \@ifclassloaded{memoir}{}
           {\PackageWarning{chemscheme}
            {You asked for floats to be created using memoir,
             \MessageBreak but have used a different document
             class\MessageBreak Using floatrow instead}%
            \renewcommand*{\csh@fltpkg}{floatrow}}%
       \else
         \PackageError{chemscheme}
           {Unknown value '#1' for option numbers}
           {The 'floats' option accepts values 'float',
            'floatrow' and 'memoir'}%
       \fi
     \fi
   \fi}
\@ifpackageloaded{floatrow}
  {\newcommand*{\csh@fltpkg}{floatrow}%
   \define@key{csh}{floats}
     {\PackageInfo{chemscheme}{Package floatrow already
      loaded\MessageBreak Option 'floats' disabled}}}
  {\@ifpackageloaded{float}
     {\newcommand*{\csh@fltpkg}{float}%
      \define@key{csh}{floats}
        {\PackageInfo{chemscheme}{Package float already
         loaded\MessageBreak Option 'floats' disabled}}}
     {\@ifclassloaded{memoir}
        {\newcommand*{\csh@fltpkg}{memoir}}
        {\newcommand*{\csh@fltpkg}{floatrow}}}}
\ProcessKeyvalOptions{csh}
```

\csh@load@memoir The necessary support is loaded by using an appropriately-named macro. For
\csh@load@float memoir, nothing actually has to be done.
\csh@load@floatrow

```
\newcommand*{\csh@load@memoir}{}
\newcommand*{\csh@load@float}{%
  \@ifpackageloaded{float}{}
    {\let\newfloat\@undefined}
  \RequirePackage{float,caption}}
\newcommand*{\csh@load@floatrow}{%
```

```
  \@ifpackageloaded{floatrow}{}
    {\let\newfloat\@undefined}
  \RequirePackage{floatrow,caption}}
\csname csh@load@\csh@fltpkg\endcsname
```

7.2 Support for schemes

\schemename
\listschemename

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.

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

scheme
\listofschemes
\csh@makesch@memoir
\csh@mksch@memoir
\csh@makesch@float
\csh@mksch@float
\csh@makesch@floatrow

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.

```
\newcommand*{\csh@makesch@memoir}{%
  \AtBeginDocument{\csh@mksch@memoir}}
\newcommand*{\csh@mksch@memoir}{%
  \newfloat[chapter]{scheme}{los}{\schemename}
  \kill@lastcounter{losdepth}
  \renewcommand*{\thescheme}{\thechapter.\@arabic\c@scheme}
  \addtodef{\@smemfront}{}{\counterwithout{scheme}{chapter}}
  \addtodef{\@smemmain}{}{%
    \ifartopt\else
      \counterwithin{scheme}{chapter}
    \fi}
  \addtodef{\backmatter}{}{%
    \ifartopt\else
      \counterwithout{scheme}{chapter}%
      \setcounter{scheme}{0}%
    \fi}
  \ifartopt
    \counterwithout{scheme}{chapter}%
  \fi
  \newlistof{listofschemes}{los}{\listschemename}
  \kill@lastcounter{losdepth}
  \newlistentry[chapter]{scheme}{los}{0}
  \cftsetindents{scheme}{0em}{2.3em}
  \addtodef{\insertchapterspace}{}%
    {\addtocontents{los}{\protect\addvspace{10pt}}}
  \@ifundefined{c@losdepth}%
    {\newcounter{losdepth}\setcounter{losdepth}{1}}{}}
\newcommand*{\csh@makesch@float}{%
  \AtBeginDocument{\csh@mksch@float}}
\newcommand*{\csh@mksch@float}{%
  \@ifundefined{chapter}
    {\newfloat{scheme}{tbp}{los}}
    {\newfloat{scheme}{tbp}{los}[chapter]%
     \csh@fixchapter}%
  \floatname{scheme}{\schemename}
```

```
  \newcommand*{\listofschemes}{%
    \listof{scheme}{\listschemename}}}
\newcommand*{\csh@makesch@floatrow}{%
  \DeclareNewFloatType{scheme}
    {fileext=los,placement=tbp,name=Scheme}
  \@ifundefined{chapter}{}
    {\floatsetup[scheme]{within=chapter}%
     \csh@fixchapter}%
  \newcommand*{\listofschemes}{%
    \listof{scheme}{\listschemename}}}
```

\csh@fixchapter
\csh@chapter
\@chapter

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

```
\newcommand*{\csh@fixchapter}{%
  \@ifundefined{KOMAScriptVersion}
    {\renewcommand*{\thescheme}{%
       \ifnum\c@chapter>\z@ \thechapter.\fi \@arabic\c@scheme}}
    {}%
    \let\csh@chapter\@chapter
    \renewcommand{\@chapter}{%
      \addtocontents{los}{\protect\addvspace{10\p@}}%
      \csh@chapter}}
```

The appropriate creation macro is now called.

```
\csname csh@makesch@\csh@fltpkg\endcsname
```

7.3 Positioning float contents

\floatcontentscentre
\floatcontentscenter
\floatcontentsleft
\floatcontentsright

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.

```
\def\@tempa{floatrow}
\ifx\@tempa\csh@fltpkg
  \newcommand*{\floatcontentscentre}{%
    \floatsetup{objectset=centering}}
  \newcommand*{\floatcontentscenter}{%
    \floatsetup{objectset=centering}}
  \newcommand*{\floatcontentsright}{%
    \floatsetup{objectset=raggedleft}}
  \newcommand*{\floatcontentsleft}{%
    \floatsetup{objectset=raggedright}}
```

\csh@floatboxreset
\@floatboxreset
\csh@everyfloat

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

```
\else
  \let\csh@floatboxreset\@floatboxreset
  \renewcommand*{\@floatboxreset}{%
    \csh@everyfloat
    \csh@floatboxreset}
  \newcommand*{\floatcontentscentre}{%
    \let\csh@everyfloat\centering}
  \newcommand*{\floatcontentscenter}{%
    \let\csh@everyfloat\centering}
```

```
  \newcommand*{\floatcontentsleft}{%
    \let\csh@everyfloat\relax}
  \newcommand*{\floatcontentsright}{%
    \let\csh@everyfloat\raggedleft}
\fi
\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.

```
\newif\ifcsh@babel
\@ifpackageloaded{babel}
  {\csh@babeltrue}
  {}
```

\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*!

```
\@ifclassloaded{memoir}{}
  {\@ifpackageloaded{babel}
     {\IfLanguageName{french}
        {\let\floatc@plain\FB@makecaption}
        {\IfLanguageName{frenchb}
          {\let\floatc@plain\FB@makecaption}
          {}}}
  {}}
```

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.

```
\AtBeginDocument{
  \@ifpackageloaded{babel}
    {\addto{\captionsngerman}{%
       \renewcommand*{\schemename}{Schema}}
     \addto{\captionsngerman}{%
       \renewcommand*{\listschemename}{Schemenverzeichnis}}
     \addto{\captionsgerman}{%
       \renewcommand*{\schemename}{Schema}}
     \addto{\captionsgerman}{%
       \renewcommand*{\listschemename}{Schemenverzeichnis}}
```

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

```
     \addto{\captionsfrench}{%
       \renewcommand*{\schemename}{\scshape Sch\'eme}}
     \addto{\captionsfrench}{%
       \renewcommand*{\listschemename}{Table des sch\'emes}}
     \addto{\captionsfrenchb}{%
       \renewcommand*{\schemename}{\scshape Sch\'eme}}
```

```
    \addto{\captionsfrenchb}{%
      \renewcommand*{\listschemename}{Table des sch\'emes}}
```

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

```
    \expandafter\selectlanguage\expandafter{\languagename}
```

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.

```
    \@ifclassloaded{memoir}{}
      {\IfLanguageName{french}
         {\ifcsh@babel\else
            \let\floatc@plain\FB@makecaption
          \fi}
         {\IfLanguageName{frenchb}
           {\ifcsh@babel\else
              \let\floatc@plain\FB@makecaption
            \fi}
         {}}}
     {}}{}}
```

7.5 Reference numbers in graphics

`\schemerefmarker` `\schemerefformat`

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

```
\newcommand*{\schemerefmarker}{TMP}
\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.

```
\ifcsh@bpchem
  \RequirePackage{bpchem}
  \let\csh@label\CNlabel
\else
  \RequirePackage{chemcompounds}
  \let\csh@label\compound
\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.

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

`\schemeref`

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

```
\newcommand*{\schemeref}{%
  \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.

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

\csh@schemeref 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.

```
\def\csh@schemeref[#1]#2{%
  \psfrag{#1}[b][b]
    {\schemerefformat{\csh@label{#2}}}}
```

\csh@schemerefsub For sub-referencing using bpchem, a similar macro is needed using the \CNlabelsub command. This is defined here, with the user macro later.

```
\def\csh@schemerefsub[#1]#2#3{%
  \psfrag{#1}[b][b]
    {\schemerefformat{\CNlabelsub{#2}{#3}}}}
```

\chemschemeref The \chemschemeref is now a backward-compatibility wrapper for \schemeref.

```
\newcommand*{\chemschemeref}[1]{\schemeref[#1]}
```

\schemerefsub \chemschemerefsub The user sub-referencing macro is defined in a package-dependent manner. If bpchem is loaded, then \schemerefsub works in the same way as \schemeref, but calling the sub version of the internal macro.

```
\ifcsh@bpchem
  \newcommand*{\schemerefsub}{%
    \advance\csh@num\@ne
    \@ifnextchar[%]
      {\csh@schemerefsub}
      {\csh@schemerefsub[\schemerefmarker\the\csh@num]}}
  \newcommand*{\chemschemerefsub}[1]{\schemerefsub[#1]}
```

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.

```
\else
  \newcommand*{\schemerefsub}[3][]{%
    \advance\csh@num\@ne
    \PackageWarning{chemscheme}
      {'chemcompound' option active\MessageBreak
       Command \string\chemschemerefsub\space not used
       \MessageBreak Please alter your source to \MessageBreak
       \string\chemschemeref}}
  \newcommand*{\chemschemerefsub}{\schemerefsub}
\fi
```

8 Change History

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.

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.