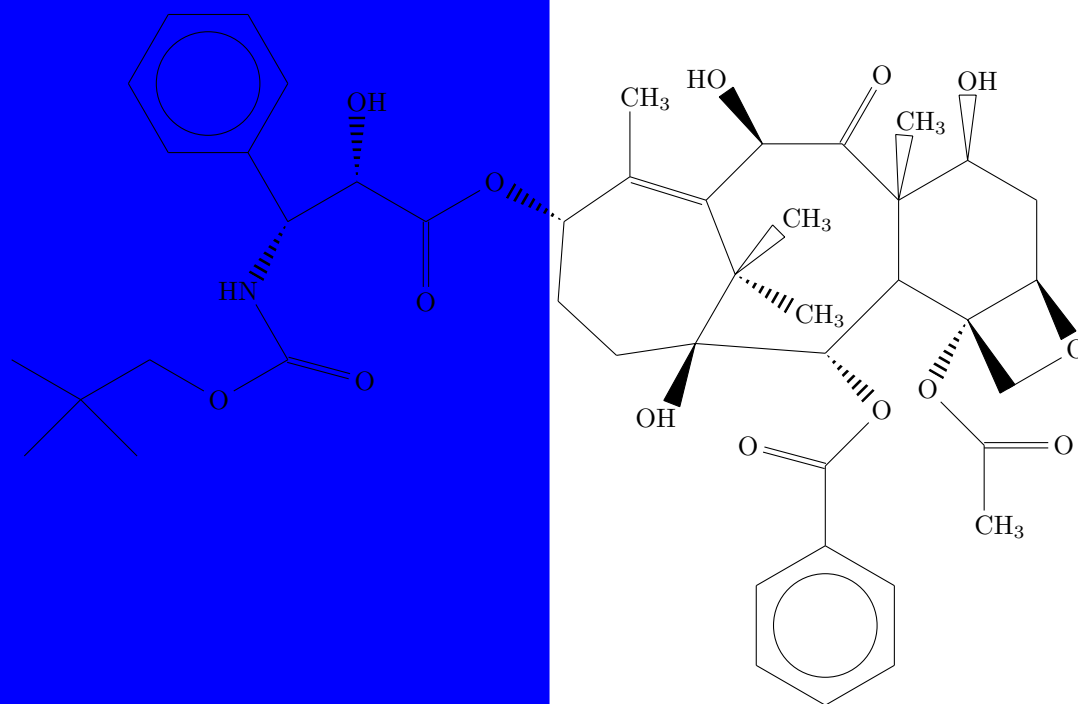


ChemFig v0.2

August 31, 2010
Christian Tellechea

A L^AT_EX package for drawing molecules



Taxotere

Contents

I	Introduction	3
1	Foreword	3
2	Presenting ChemFig	3
II	ChemFig for the impatient	5
1	Syntax	5
2	The different types of bonds	5
3	Different types of diagrams	6
3.1	Complete structural diagram	6
3.2	Condensed structural diagram	6
3.3	Cram representation	7
3.4	Skeleton diagram	7
3.5	Lewis diagrams	7
4	Branched molecules	7
5	Rings	8
6	Ions	9
7	Chemical equations	9
III	Operation of ChemFig	10
1	Groups of atoms	10
2	Different types of bonds	10
3	Bond angle	11
3.1	Predefined angles	11
3.2	Absolute angles	12
3.3	Relative angles	12
4	Length of a bond	13
5	Departure and arrival atoms	13
6	Customization of bonds	14
7	Default values	14
8	Branches	15
8.1	Principle	15
8.2	Nesting	16
8.3	Method	16
9	Connecting distant atoms	17

10 Rings	18
10.1 Syntax	18
10.2 Angular position	19
10.2.1 At the start	19
10.2.2 After a bond	20
10.3 Branches on a ring	20
10.4 Nested rings	21
10.5 Rings and groups of atoms	22
 IV Advanced usage	 23
1 Displaying atoms	23
2 Shifted double bonds	23
3 Delocalized double bonds	24
4 Saving a sub-molecule	24
5 Decorations	25
5.1 Lewis diagrams	25
5.2 Stacking characters	26
5.3 Chemical reactions	26
6 Using <code>\chemfig</code> in the <code>tikzpicture</code> environment	28
7 Vertical alignment	28
8 Beyond chemistry	29
9 Annotated examples	30
9.1 Ethanal	30
9.2 2-amino-4-oxohexanoic acid	31
9.2.1 Absolute angles	31
9.2.2 Relative angles	31
9.2.3 Ring	32
9.2.4 Nested rings	32
9.3 Glucose	32
9.3.1 Skeleton diagram	33
9.3.2 Fisher projection	33
9.3.3 “Chair” representation	34
9.3.4 Haworth projection	34
9.4 Adrenaline	35
9.4.1 Using one ring	35
9.4.2 Using two rings	36
9.5 Guanine	37
10 List of commands	38
 V Gallery	 39

PART I

Introduction

1 Foreword

This package has seen the light of day thanks to the assistance of Christophe CASSEAU, who had the idea after being confronted with the complexity of the syntax of the `ppchtex` package.

Throughout the writing of the code, he helped me find interesting features. He always encouraged me to write more advanced features even though I was sometimes (nearly always?) reluctant; if **ChemFig** has the features it does, it is in large part thanks to him. I thank him as well for his testing of beta versions of this package, and for his contributions to the writing of this manual.

*Experience shows that it has been difficult to combine drawing of molecules with the typographic quality of a program like \LaTeX , leaving little choice for the user who wishes to have a vector format for these drawings. After having abandoned the `ppchtex` package (developed for `conTeXt` and available under \LaTeX) because of the complexity of its syntax, I turned to the world of programs outside \LaTeX . The difficulty in this case is finding a compromise between quality and price. After many unsuccessful attempts I found the only option was a new package, and I would like to thank Christian TELLECHEA for bringing it to life. To meet my requirements, **ChemFig** needed to be easy to use but still have advanced features, something well nigh impossible. Yet he was able to put together a very flexible \TeX code which makes it a pleasure for me to write my molecules. I hope it will be the same for you readers looking for a package useful in the field of chemistry.*

Christophe CASSEAU

Finally, I wish to warmly thank Theo HOPMAN for offering to translate this manual into English.

2 Presenting ChemFig

To use this package, start by adding the following code to the preamble:

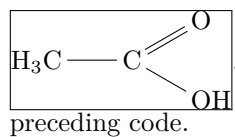
```
\usepackage{chemfig}
```

The most important command for drawing molecules is `\chemfig{<code>}`. The argument `code` is a set of characters describing the structure of the molecule according to the rules which are described in this manual.

Care has been taken to make it possible to draw the greatest possible number of molecular configurations, while maintaining a simple, flexible, and intuitive syntax. Despite this, the `<code>` which describes the 2D structure of the molecule increases in complexity in proportion to that of the molecule being drawn.

The command `\chemfig` draws a molecule using the commands provided by the `tikz` package, placed inside a `tikzpicture` environment. The choice of `tikz` implies that:

- the user has a choice of compilation method: \pdfLaTeX can be used equally well in dvi mode (`tex` \rightarrow `dvi` \rightarrow `ps` \rightarrow `pdf`) or in pdf mode (`tex` \rightarrow `pdf`). In effect `tikz`, via the underlying `pgf`, gives identical graphical results in the two modes;
- the bounding box is automatically calculated by `tikz` and the user need not worry about any overlap with the text. However, care must be taken with alignment when the molecule is drawn in a paragraph. In the following example, we have drawn the bounding box for the molecule:



. ChemFig always places the first atom of the molecule on the baseline of the

PART II

ChemFig for the impatient

This part is a non-exhaustive overview of the features of **ChemFig**. The goal is to introduce the basic ideas, allowing the user to get started drawing molecules as quickly as possible. This part does not go into detail; advanced use and a more formal approach to **ChemFig** commands will be discussed in the following parts.

1 Syntax

The command `\chemfig` is used in the following way:

```
\chemfig{<atom1><bond type>[<angle>,<coeff>,<n1>,<n2>,<tikz code>]<atom2>}
```

- `<angle>` is the bond angle between the two atoms;
- `<coeff>` is a coefficient multiplying the default bond length;
- `<n1>` and `<n2>` are the numbers of the departure and arrival atoms of the bond;
- `<tikz code>` is additional options concerning the colour or style of a bond.

Each bond takes optional arguments which are placed in square brackets. These arguments can adjust everything one needs for the bond. Each argument has a default value, so one can simply write:

<code>\chemfig{H-O-H}</code>		The water molecule	H — O — H
------------------------------	--	---------------------------	-----------

In all the examples, the grey line represents the baseline.

2 The different types of bonds

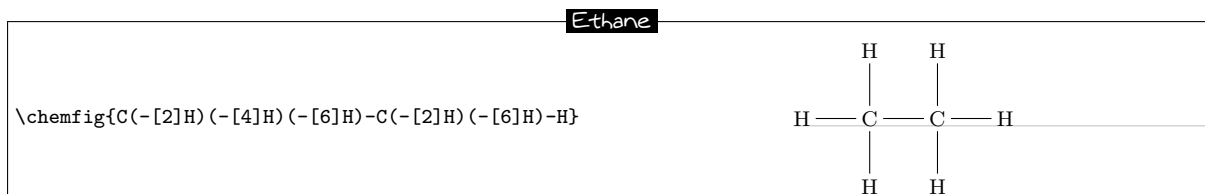
For **ChemFig**, bonds between two atoms are one of nine types, represented by the characters \square , \equiv , \sim , \triangleright , \triangleleft , $\triangleright\vdots$, $\triangleleft\vdots$, $\triangleright|$ and $\triangleleft|$:

Bond #	Code	Result	Bond type
1	<code>\chemfig{A-B}</code>	A — B	Single
2	<code>\chemfig{A=B}</code>	A = B	Double
3	<code>\chemfig{A\sim B}</code>	A ≡ B	Triple
4	<code>\chemfig{A>B}</code>	A ► B	right Cram, plain
5	<code>\chemfig{A<B}</code>	A ◄ B	left Cram, plain
6	<code>\chemfig{A>:B}</code>	A ►⋯ B	right Cram, dashed
7	<code>\chemfig{A<:B}</code>	A ◄⋯ B	left Cram, dashed
8	<code>\chemfig{A> B}</code>	A ► B	right Cram, hollow
9	<code>\chemfig{A< B}</code>	A ◄ B	left Cram, hollow

The command `\setdoublesep{<dim>}` adjusts the spacing between the lines in double or triple bonds. This spacing is 2pt by default.

3 Different types of diagrams

3.1 Complete structural diagram

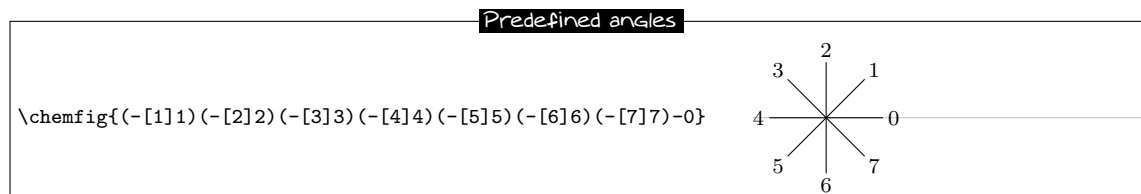


Each bond takes several optional arguments. The first optional argument defines the `<angle>` of the bond. Angles increase counterclockwise. If no angle is given then its default value is 0° .

Note: the parentheses allow multiple bonds from the same atom; see “Branched molecules”, page 7.

There are many ways of specifying the angle of a bond.

Predefined angles When the optional argument contains a whole number, this represents the angle that the bond makes with the horizontal, in multiples of 45° .



Absolute angle To give an angle in degrees relative to the horizontal, the optional argument must take this form: `[<absolute angle>]`. The `<absolute angle>` may be positive or negative, and may have decimal places. It is reduced to the interval $[0, 360)$.

Relative angle It is often useful to give the bond angle relative to the preceding bond. This syntax is used: `[<relative angle>]`. The `<relative angle>` may be positive or negative, and may have decimal places.

3.2 Condensed structural diagram



It is sometimes useful to change the length of a bond. In the preceding example, the first two bonds need to be lengthened; they are too short compared to the double bond. To do this, the `<coeff>` is needed in the optional arguments.

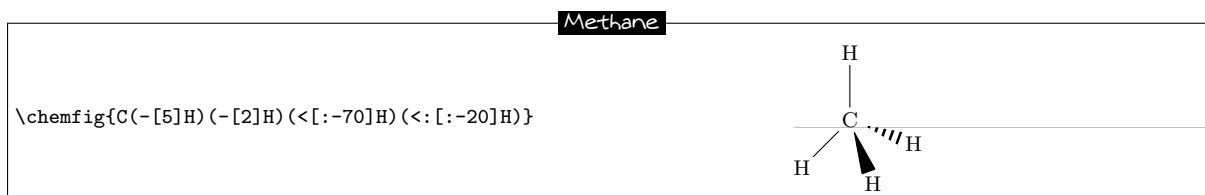


Notes:

- in the notation `[1.5]`, the comma indicates that the value placed in brackets corresponds to the second argument (`coeff`). To give a value for the fourth argument one would write `[,,2]`;
- characters between braces are not interpreted by **ChemFig**, which allows (for example) groupings of atoms to be written inside parentheses without having them treated as part of a branched molecule (see page 7).

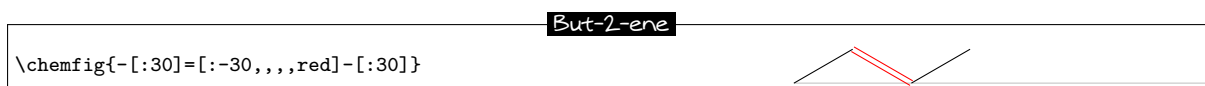
3.3 Cram representation

Another way of showing the bond angle between two atoms.

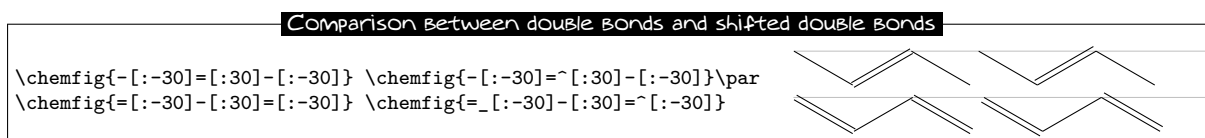


3.4 Skeleton diagram

The simplest input, only covalent bonds are listed with their possible settings.



The double bond lines are drawn on either side of where the single bond line would be. To keep graphical consistency between skeleton diagrams of various compounds it is useful to shift the second double bond line above or below the single bond line. Just follow the = symbol with ^ or _, as shown below:

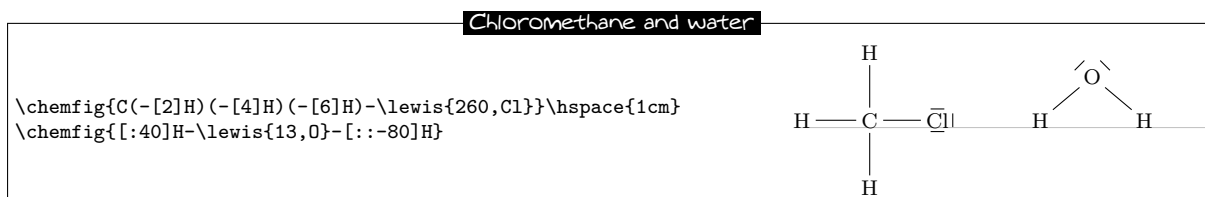


3.5 Lewis diagrams

The syntax is as follows:

`\lewis{<position index><electron state>,<atom>}`

The electron states can be a lone pair, an empty electron slot, or a single electron. By default the electron state is a lone pair. A single electron is represented by the character `\dot` and an empty slot by `\square`.



Note: the positions take whole number values between 0 and 7.



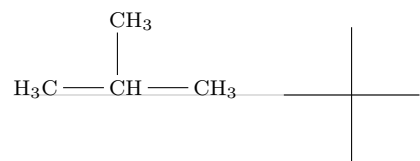
4 Branched molecules

To indicate a branch, simply follow the atom holding the branch with a `<code>` in parentheses. This `<code>` is the code of the submolecule which will be attached to the atom. Multiple branches can be

attached to the same atom, and these can be nested.

Alkanes

```
\chemfig{H_3C-CH(-[2]CH_3)-CH_3}
\hspace{.5cm}\chemfig{-(-[2])(-[6])}
```



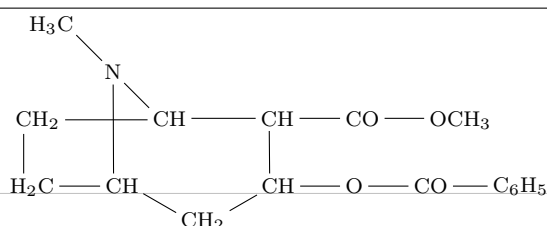
In this type of representation it is sometimes necessary to connect to distant atoms. To do this, the character ? is used, which creates a hook between two atoms. The function

`?[<name>,<bond>,<tikz>]`

takes three arguments: the name of the hook, the bond type, and tikz code. The following example shows how to make two different hooks.

Cocaine

```
\chemfig{H_2C(-C?[a]H-[:30]CH_2-[:30]C?[b]H-O-CO-C_6H_5)
-[2]CH_2-[1.7]CH(-[3]N?[a]-[3]H_3C)(-[1.35]C?[b]H-CO-OCH_3)}
```



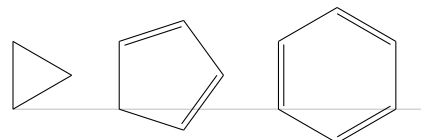
5 Rings

ChemFig can easily draw regular polygons. The syntax is the following:

`\chemfig{*n(<code for the molecule>)}`

Some rings

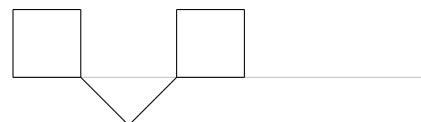
```
\chemfig{*3(---)}\hspace{.5cm}
\chemfig{*5(-----)}\hspace{.5cm}
\chemfig{*6(-----=)}
```



Branches are used in the same way as before.

Rings and Branches

```
\chemfig{*4(-(--[1]*4(----))---)}
```



Note: a ring does not start or finish with the atom or group of atoms with which one wants to close the ring. The chemical entity on which the ring is based must be outside the ring definition.

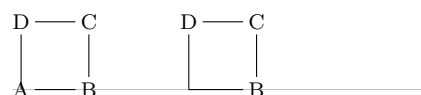
Proper coding

```
\chemfig{A*4(-B-C-D-)}
```



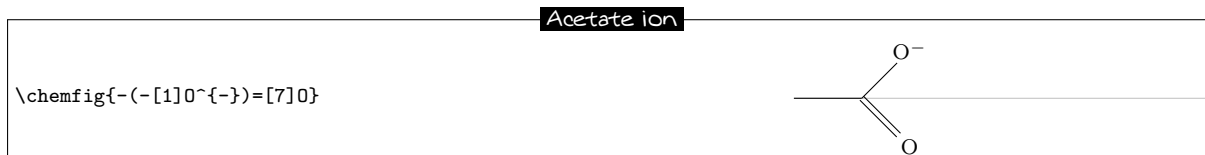
Bad coding

```
\chemfig{*4(A-B-C-D-)}\hspace{1cm}
\chemfig{*4(-B-C-D-A)}
```



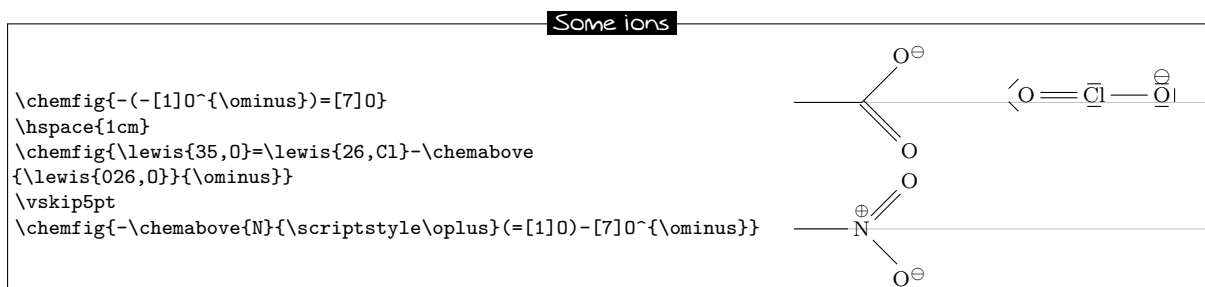
6 Ions

The `chemfig` commands enters the math mode¹ of T_EX, so it is very simple to write an ion. A negative charge (−) must always be enclosed in braces to avoid `ChemFig` confusing it with the symbol for a single bond.

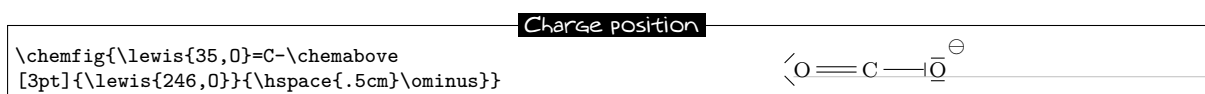


For purists it is possible to circle the charge of an ion by using the commands `\ominus` and `\oplus`.

To meet all requirements, there are two other commands `\chemabove` and `\chembelow` which allow placement of charges above or below the current atom.

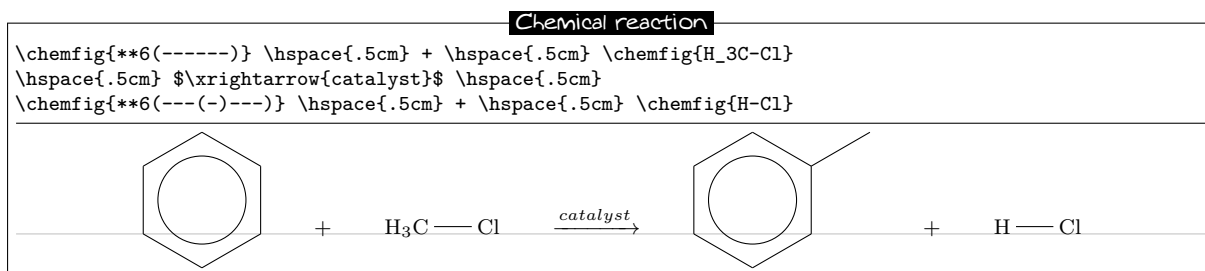


For those who are extremely picky, the commands `\chemabove` and `\chembelow` accept an optional argument which sets the distance between the charge and the atom.

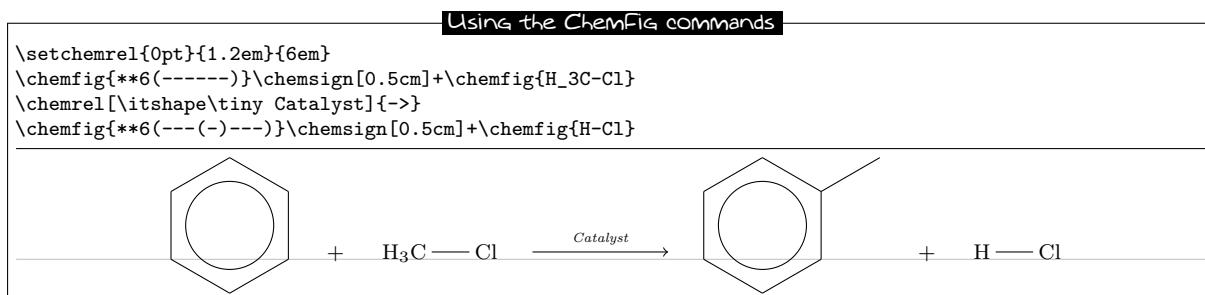


7 Chemical equations

Here is an example of a chemical reaction:



`ChemFig` adds two new commands `\chemsign` and `\chemrel` which slightly simplify the preceding syntax.



¹There is a problem with the placement of groups of atoms containing exponents or subscripts. See page 28.

PART III

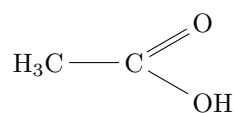
Operation of ChemFig

This part is devoted to describing the most common features of **ChemFig**. The outline of this description far exceeds that of “**ChemFig** for the impatient”, but the user will find here explanations sufficient to draw most molecules. The presentation of features is done from a theoretical angle, and the goal of this part is not to draw real molecules but to give the user a formal description of the functionality of **ChemFig**. The “Advanced usage”, page 23, will be more practical and will illustrate advanced features for the most demanding uses. It will also highlight methods of building real molecules, page 30. Finally, the last part will give examples of molecules and the code used to draw them.

1 Groups of atoms

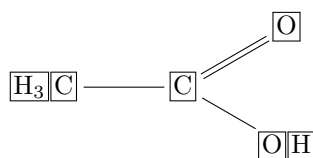
Drawing a molecule consists inherently of connecting groups of atoms with lines. Thus, in the molecule $\text{O}=\text{O}$, there are two groups of atoms, each consisting of a single atom “O”.

However, in this molecule



there are four groups of atoms: “H₃C”, “C”, “O” and “OH”. For reasons which we shall see later, **ChemFig** splits each group into single atoms. Each atom extends up to the next capital letter or one of these special characters: `[] { } ~ ! * < >`. **ChemFig** ignores all characters inside braces when splitting groups into atoms.

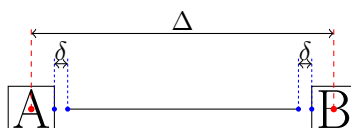
Therefore the first group of atoms “H₃C” is split into two atoms: `[H3]` and `[C]`. In terms of chemistry, of course, these are not real atoms; H₃, for example, consists of three hydrogen atoms. In what follows the word atom refers to **ChemFig**’s definition. Thus **ChemFig** sees the preceding molecule as follows:



2 Different types of bonds

As we have already seen (see page 5), bonds can be one of nine types, each corresponding to the characters `[] { } ~ ! * < >` and `[]`.

We must understand that when a bond is made between two atoms, these atoms are contained within invisible rectangular boxes. The centres of these two rectangles are separated by an adjustable distance Δ called the “interatomic distance”. Furthermore, bonds do not connect to the exact edges of the rectangles: a length δ , also adjustable, separates the edges of the rectangles and the beginning and end of the bond line. The rectangular boxes are made visible in the diagram below to help understanding.



The macro `\setatomsep{<dimension>}` adjusts the interatomic distance Δ . If the `<dimension>` is empty, it takes the default value of 3em. This command, like all other settings commands, affects all the following molecules.

Interatomic distance

```
\setatomsep{2em}\chemfig{A-B}\par
\setatomsep{50pt}\chemfig{A-B}
```

A — B
A ——— B

The command `\setbondoffset{<dimension>}` sets the spacing δ between the bond line and the atom:

Trimming Bonds

```
\setbondoffset{0pt}\chemfig{A-B}\par
\setbondoffset{5pt}\chemfig{A-B}
```

A — B
A — B

If one bond is followed immediately by another, then `ChemFig` inserts an empty group `{}`. Around this empty group the separation δ is zero:

Empty groups

```
\chemfig{A-B==C}\par
\chemfig{A>B><|<C}
```

A — B == C
A ► B ► C

By default, all atoms within groups of atoms are typeset in math mode (spaces are ignored). They may therefore contain math mode specific commands such as subscripts or superscripts²:

Math mode

```
\chemfig{A_1B^2-C_3^4}
```

A₁B² — C₃⁴

There are settings specifically for Cram bonds. This syntax is used:

```
\setcrambond{<dim1>}{<dim2>}{<dim3>}
```

Any empty argument takes its default value. The three arguments are:

- `<dim1>` is the size of the base of the triangle, and is 1.5pt by default;
- `<dim2>` is the thickness of the dots, and is 1pt by default;
- `<dim3>` is the spacing between the dots, and is 2pt by default.

Here is an example where the three dimensions are changed:

Modified Cram Bonds

```
\setcrambond{10pt}{0.4pt}{1pt}
\chemfig{A>B>:C>|D}
```

A ► B ||||| C ▷ D

3 Bond angle

Each bond takes an optional argument in brackets. This optional argument can adjust every aspect of a bond, and consists of five optional fields separated by commas. The first of these fields defines the bond angle. Angles increase counterclockwise, and are relative to the horizontal. If the angle field is empty, the angle takes its default value of 0°. We will see later how to change this default.

There are several ways of specifying the bond angle.

3.1 Predefined angles

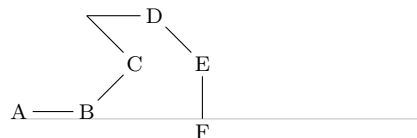
When the angle field contains an integer, this represents the angle the bond makes relative to the horizontal, in multiples of 45°. For example, [0] specifies an angle of 0°, [1] is 45°, and so on up to [7] which

²There is a problem with the placement of groups of atoms containing exponents or subscripts. See page 28.

specifies an angle of 315° . The integer may lie outside the interval $[0, 7]$, in which case the angle is reduced to the interval $[0, 360)$.

Predefined angles

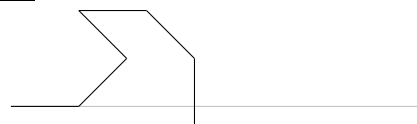
```
\chemfig{A-B-[1]C-[3]-D-[7]E-[6]F}
```



These angles remain valid if the atoms are empty, and this is the case for all the features we will see below:

Predefined angles with empty groups

```
\chemfig{--[1]-[3]--[7]-[6]}
```

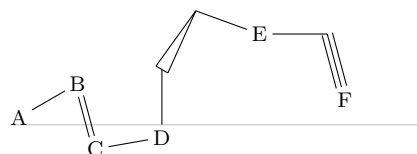


3.2 Absolute angles

If one wishes to specify an angle in degrees relative to the horizontal, then the optional angle field must take this form: `[<absolute angle>]`. If necessary, the `<absolute angle>` is reduced to the interval $[0, 360)$:

Absolute angles

```
\chemfig{A-[:30]B=[:75]C-[:10]D-[:90]>|[:60]-[:20]E-[:0]~[:75]F}
```



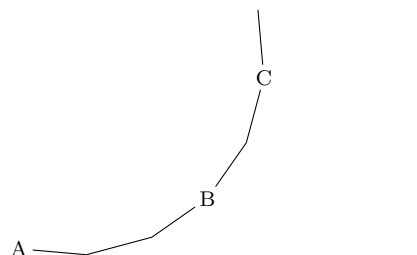
3.3 Relative angles

It is often useful to specify a bond angle relative to the preceding bond. This syntax must be then be used: `[::<relative angle>]`. The sign of the `<relative angle>` can be omitted if it is a `+`.

Here is a molecule where the first bond has an absolute angle of -5° , and the rest of the bond angles are incremented by 20° :

Result of relative angles

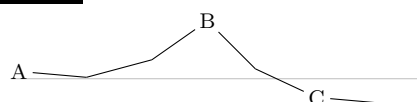
```
\chemfig{A-[:5]-[::+20]-[::20]B-[::+20]-[::20]C-[::20]}
```



One can “break” a chain of relative angles by putting an absolute or predefined angle where desired. Here, atom “B” is followed by a bond at an absolute angle of 315° .

Result of relative angles followed by absolute

```
\chemfig{A-[:5]-[::20]-[::20]B-[7]-[::20]C-[::20]}
```



4 Length of a bond

Rather than speaking of length of a bond, we should use the term interatomic spacing. If effect, only the interatomic spacing is adjustable with `\setatomsep` as we have seen on page 10. Once this parameter is set, the length of a bond depends on the content of atoms and, to a lesser extent, the angle the bond makes with the horizontal. It should be obvious that two “slimmer” atoms will have larger edge separations than two which are larger. This can be seen easily in the following example where an “I” atom is narrower than an “M” atom, which means that the bond between the “I” atoms is longer than that between the “M” atoms:

Influence of the size of atoms

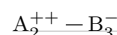
```
\chemfig{I-I}\par
\chemfig{M-M}
```



This aspect of the size of atoms becomes particularly acute when the atom involves subscripts or superscripts. In this example, the bond is extremely short, to the point of confusion with a negative sign —:

Too-short bond

```
\chemfig{A^{++}_{2}-B^{-}_{3}}
```

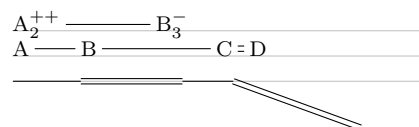


It is important to note that the exponent $-$ is *put inside braces*. If this were not done, **ChemFig** would stop the atom on this character, which is a bond character. The atom would then be “B \sim ”, which would lead to unexpected results.

We see in the example above that it is sometimes necessary to increase (or perhaps reduce) the interatomic distance associated with a bond. For this, the optional argument to bonds is actually made up of several comma-separated fields. As we have seen, the first field specifies the angle. The second field, if it is not empty, is a coefficient which multiplies the default interatomic distance Δ . Thus, writing `-[,2]` asks that this bond have the default angle (first field is empty) and that the atoms it connects be separated by twice the default distance.

Modified Bond length

```
\chemfig{A^{++}_{2}-[,2]B^{-}_{3}}\par
\chemfig{A-B-[,2]C=[,0.5]D}\par
\chemfig{= [,1.5]-[,0.75]=[:20,2]}
```



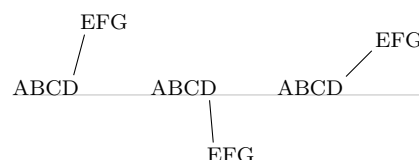
5 Departure and arrival atoms

A group of atoms can contain several atoms. Suppose we want to connect the group “ABCD” to the group “EFG” with a bond. **ChemFig** calculates which atom of the first group and which of the second group are to be connected by looking at the angle of bond relative to the horizontal. If the angle is between (but not including) -90° and 90° (modulo 360°) then the bond is made between the last atom of the first group and the first atom of the second group. In all other cases, the bond is made between the first atom of the first group and the last atom of the second group.

Here are some examples where the bond is in the interval $(-90, 90)$, and where the bond is made between D and E:

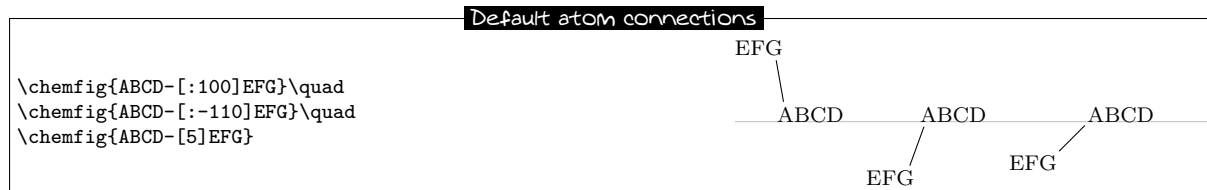
Default atom connections

```
\chemfig{ABCD-[:75]EFG}\quad
\chemfig{ABCD-[:85]EFG}\quad
\chemfig{ABCD-[1]EFG}
```



In the following examples, the angles are in the interval $[90, 270]$ and so the bond is made between A and

G:



One may sometimes want the bond partners to be atoms other than those determined by **ChemFig**. The departure and arrival atoms can be set with the optional bond argument by writing:

$$[,,<\text{integer } 1>,<\text{integer } 2>]$$

where **<integer 1>** and **<integer 2>** are the numbers of the desired departure and arrival atoms. These atoms must exist, otherwise an error message will be given.



6 Customization of bonds

There is a fifth and last optional argument for bonds which is found after the fourth comma:

$$[,,,,<\text{tikz code}>]$$

This **<tikz code>** is passed directly to **tikz** when the bond is drawn. There one can put characteristics such as colour (**red**), dash type (**dash pattern=on 2pt off 2pt**), thickness (**line width=2pt**), or even decoration if the **tikz** decoration library has been loaded. A bond can be made invisible by writing “**draw=none**”. To set several attributes, the syntax of **tikz** is used, separating them by a comma:



Numerous **tikz** decoration libraries are available. For example, one can use the “**pathmorphing**” library by putting `\usetikzlibrary{decorations.pathmorphing}` in the preamble in order to draw wavy bonds:



Cram bonds ignore thickness and dash settings.

7 Default values

At the beginning of each molecule, the default values for the optional arguments are initialized. They are:

- 0° for the bond angle;
- 1 for the length multiplication coefficient;
- **<empty>** for the numbers of the departure and arrival atoms, which lets **ChemFig** calculate these based on the bond angle;
- **<empty>** for the parameters passed to **tikz**.

These default values can be changed for the whole molecule by beginning the molecule code with

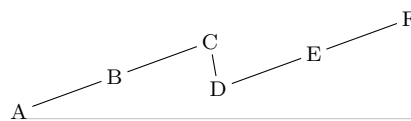
$$[<\text{angle}>,<\text{coeff}>,<\text{n1}>,<\text{n2}>,<\text{tikz code}>]$$

Thus, if the code of a molecule begins with `[:20,1.5]`, then all the bonds will be at angle of 20° by default, and the interatomic distances will have a length 1.5 times the default length. These default values

can be overridden at any time by giving an optional argument, such as for the bond which follows atom “C” in this example:

Overriding default values

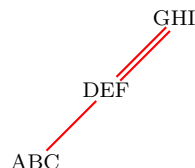
```
\chemfig{[:20,1.5]A-B-C-[:-80,0.7]D-E-F}
```



If something odd like `[1,1.5,2,2,red,thick]` is written, then unless otherwise indicated all the bonds will have an angle of 45° , the interatomic distances will be 1.5 times the default distance, the bonds will begin and end on the second atom of each group, and the bonds will be red and thick:

Default values

```
\chemfig{[1,1.5,2,2,red,thick]ABC-DEF=GHI}
```



8 Branches

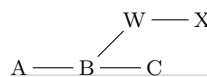
8.1 Principle

Up to now, all the molecules have been linear, which is rare. A sub-molecule can be attached to an atom by following the atom with `<code>` in parentheses. This `<code>` is the code of the submolecule which will be attached to the atom.

In this example, the sub-molecule “`-[1]W-X`” will be attached to atom “B”:

A Branch

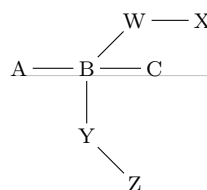
```
\chemfig{A-B(-[1]W-X)-C}
```



There can be several sub-molecules which are to be attached to the same atom. Just have several parentheses containing the code for each sub-molecule:

Multiple branches

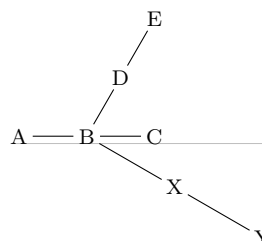
```
\chemfig{A-B(-[1]W-X)(-[6]Y-[7]Z)-C}
```



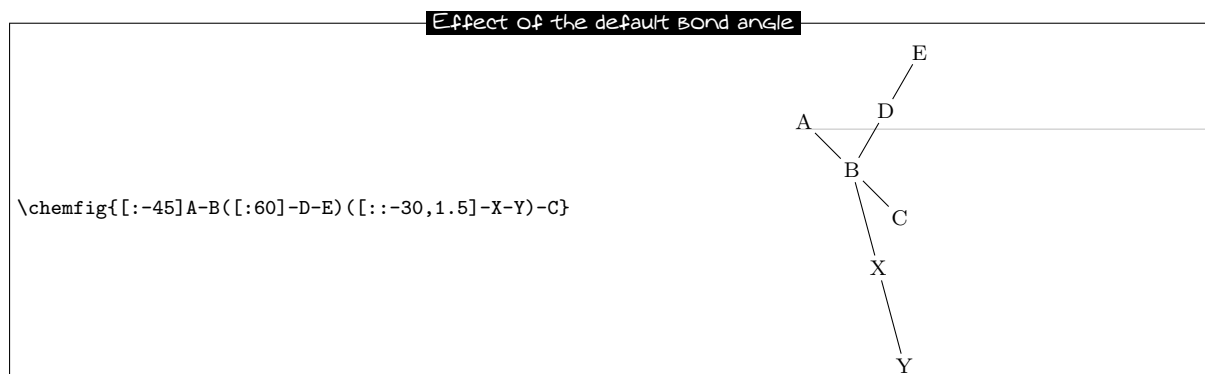
The code of each sub-molecule can define its own default values, which will be valid throughout the whole sub-molecule. Here a sub-molecule “`[:60]-D-E`” is attached to atom “B”, with a default angle of 60° absolute. A second sub-molecule “`[::-60,1.5]-X-Y`” is attached to “B” with a default bond angle 60° less than that of the preceding bond (which will be the one between “A” and “B”) and with an interatomic distance 1.5 times the default value:

Default values in branches

```
\chemfig{A-B[:60]-D-E)([::-60,1.5]-X-Y)-C}
```



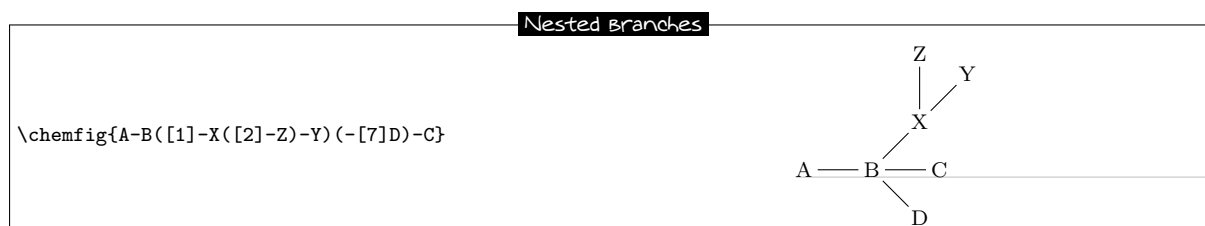
Observe what happens if, at the beginning of the main molecule, one writes “[:−45]”:



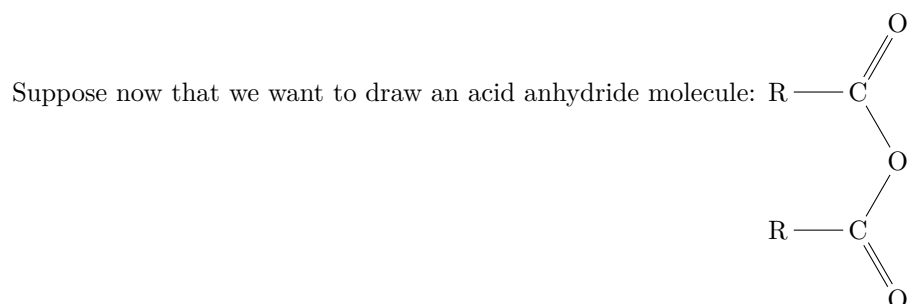
We see that the angle between the bond B-C and the bond B-X stays at 30° because it is a relative angle for the sub-molecule “-X-Y”. By contrast, the branch “-D-E” stays inclined at 60° to the horizontal, and does not follow the rotation given by the −45° angle at the beginning; this is expected because “-D-E” has an absolute angle. It is essential that all the angles be relative in order to rotate the whole molecule.

8.2 Nesting

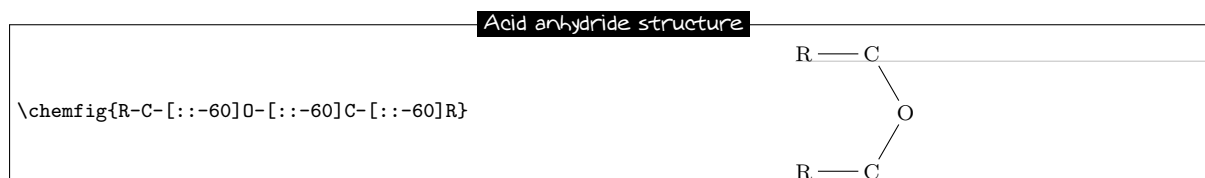
Sub-molecules may be nested, and the rules seen in the preceding paragraphs stay in force:



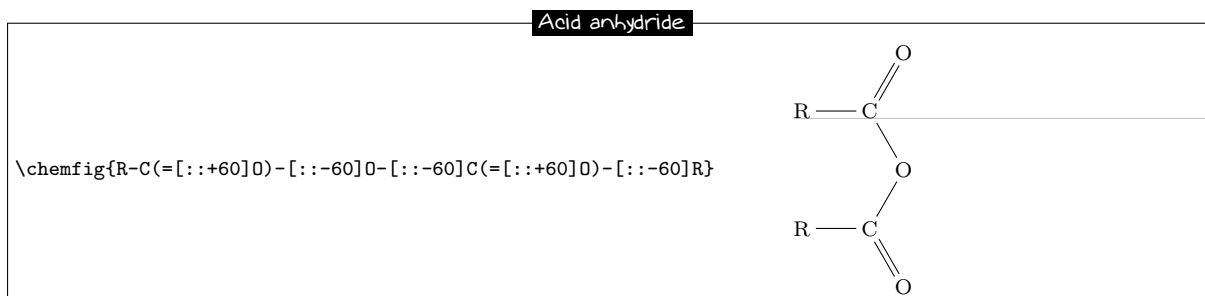
8.3 Method



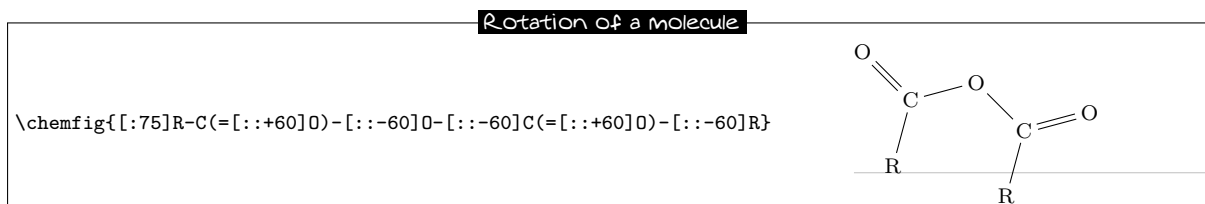
The best way to get this is to find the longest chain. Here, for example, we can draw the chain R-C-O-C-R taking into account angles and using only relative angles:



To this structure we just have to add two “=O” sub-molecules to each of the carbon atoms:



Because we used only relative angles, we can rotate this molecule by giving a default angle of e.g. 75°:



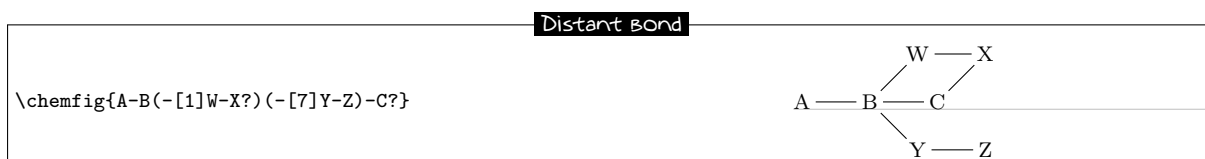
9 Connecting distant atoms

We have seen how to connect atoms *which are adjacent in the code*. It is often necessary to connect atoms which are not next to each other in the code. Let's call these particular bonds “distant bonds”.

Let's take this molecule:



and suppose that we want to connect the atoms X and C. In this case, **ChemFig** allows a “hook” to be placed *immediately* after the atom of interest. The character used for a hook is “?” because of its similarity to a hook. So, if one writes X? then the atom X will have a hook. Later in the code, all atoms followed by a ? will be connected to X:



We could connect other atoms to X by following them with ?. Here it's the atoms C and Z:



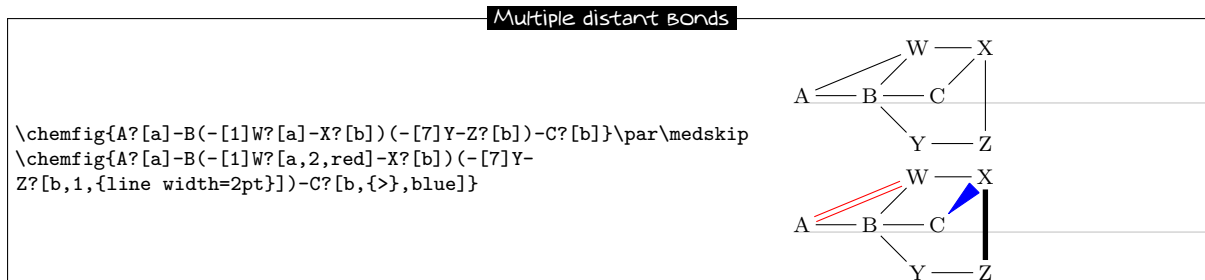
Now imagine if we were to leave the distant bonds X-C and X-Z while adding another: A-W. We must therefore ask for two *different* hooks, one on A and the other on X. Fortunately the character ? has an optional argument:

?[<name>,<bond>,<tikz>]

where each field takes its default value if it is empty:

- The `<name>` is the name of the hook: all alphanumeric characters (a...z, A...Z, 0...9) are allowed³. The name is `a` by default. In the first occurrence of the hook with this name, only this field is used.
- `<bond>` specifies how the atom with the current occurrence of the named hook is to be bonded to the atom with the first occurrence of the hook. There are two ways this can be done. First, this field can be an integer representing the desired bond type: 1=single bond, 2=double bond, etc. (See the table on page 5 for the bond codes.)
Second, the field can be one of the bond character codes, provided that this character is *between braces*.
- `<tikz>` will be passed directly to `tikz` as we have seen with regular bonds.

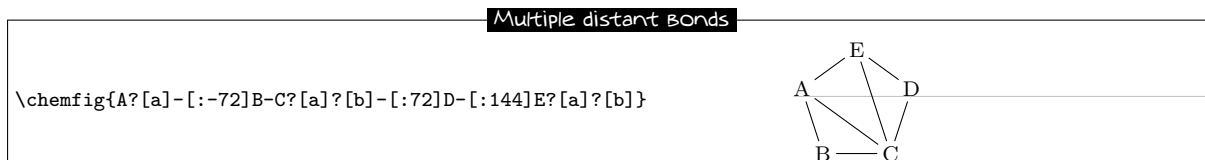
Here is our molecule with the required distant bonds, then with the bond A-W and X-C customized:



Several different hooks can be written after an atom. Suppose that in this unfinished pentagon, we wish to connect A-E, A-C and E-C:



Then we must do this:



10 Rings

The preceding example shows how to draw a regular polygon, but the method used is tedious because the angles depend on the number of sides of the polygon.

10.1 Syntax

ChemFig can easily draw regular polygons. The idea is to attach a ring to an `<atom>` outside the ring with this syntax:

`<atom>*<n>(<code>)`

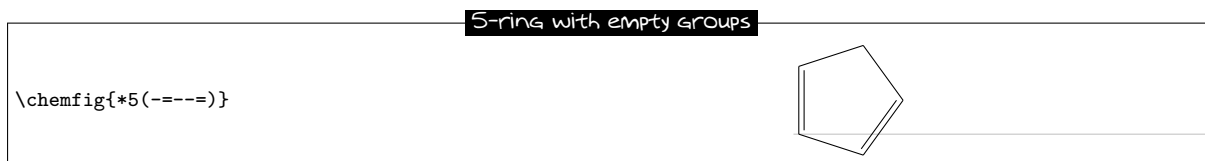
`<n>` is the number of sides of the polygon and the `<code>` describes the bonds and groups of atoms which make up its edges and vertices. This code *must* begin with a bond because the atom is outside the ring.

Here is a 5-ring, attached to the atom “A”:

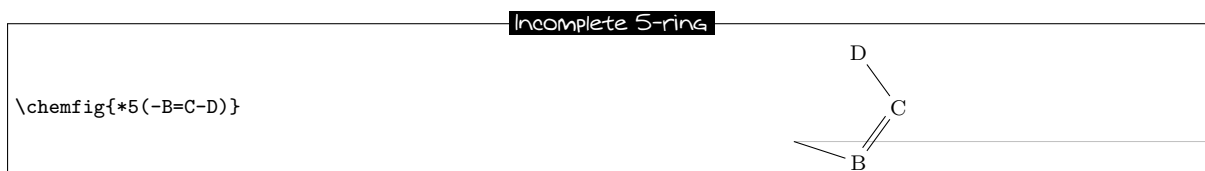


³This is not exactly right. Actually all the characters that can be put between `\csname...` and `\endcsname` are allowed.

A ring can also be drawn with one, several, or all the groups of atoms empty, as is the case for diagrams outside rings:



A ring can be incomplete:



If a ring has a code which contains too many bonds and atom groups for the given number of vertices, all the bonds and groups over the maximum allowed are ignored:

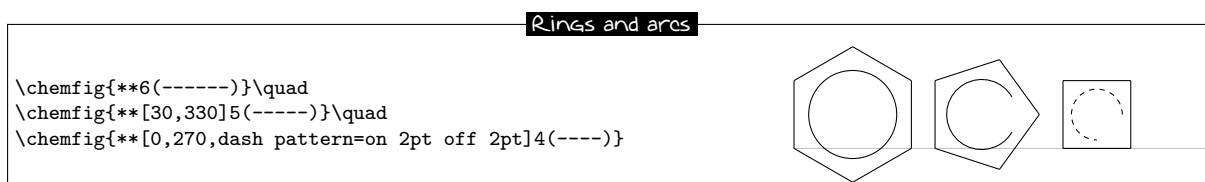


It is possible to draw a circle or an arc in the inside of a ring. To do so, the following syntax is used:

`<atom>**[<angle 1>,<angle 2>,<tikz>]<n>(<code>)`

where each field of the optional argument takes its default value if it is empty:

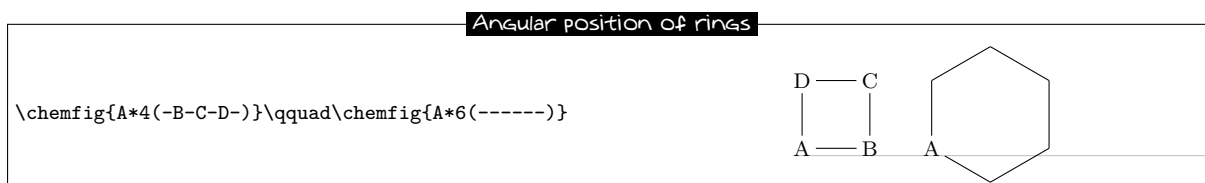
- `<angle 1>` and `<angle 2>` are the absolute angles of the start and finish of the arc. These default to 0° and 360° respectively so that a circle is drawn by default;
- `<tikz>` is the code that will be passed to `tikz` for drawing the arc.



10.2 Angular position

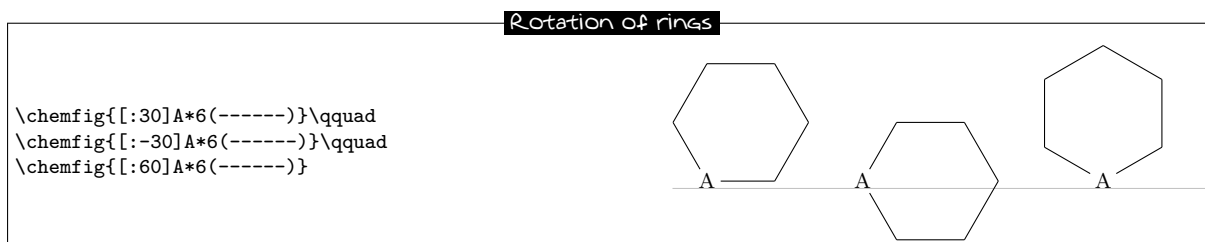
10.2.1 At the start

As can be seen in the examples above, the rule is that the attachment atom “A” is always at the south-west of the ring. Furthermore, the ring is always constructed counterclockwise, and the last bond descends vertically onto the attachment atom:



If this angular position is not convenient, it is possible to specify another angle using the optional argument at the beginning of the molecule. Here is a 6-cycle which has been rotated by $+30^\circ$, by -30° , and lastly

by $+60^\circ$:



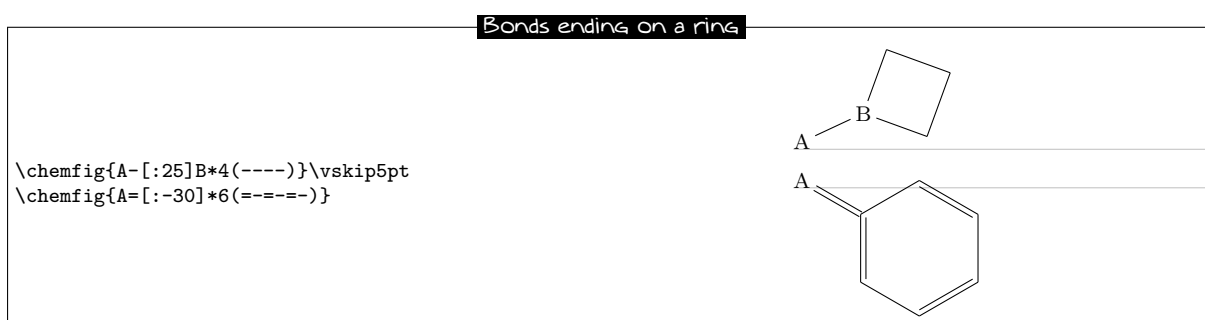
10.2.2 After a bond

When a ring does not begin a molecule and one or more bonds have already been drawn, the default angular position changes: the ring is drawn in such a way that the bond ending on the attachment atom bisects the angle formed by the first and last sides of the ring.

Here is a simple case:



The rule remains valid, whatever the angle of the preceding bond:

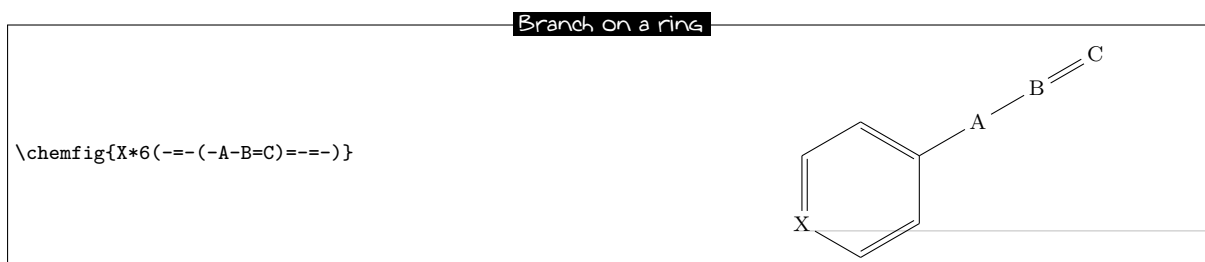


10.3 Branches on a ring

To have branches attached to the vertices of a ring, we use the syntax we have already seen:

`<atom>(<code>)`

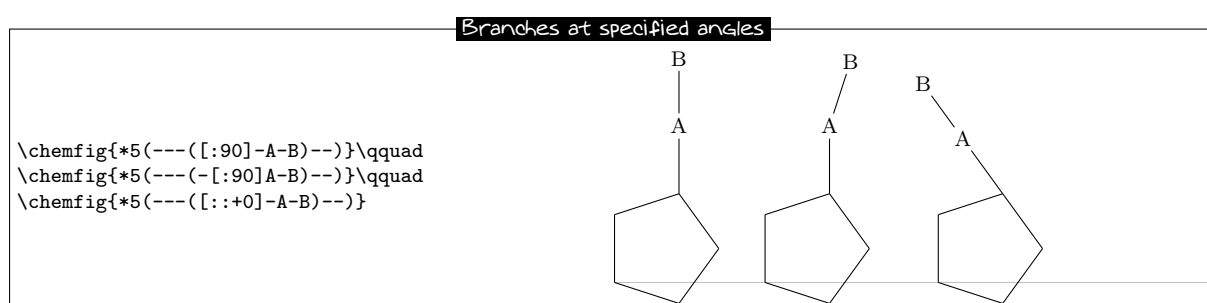
where the `<code>` is that of the sub-molecule and the `<atom>` is at the vertex. Unique to rings, the default angle of the sub-molecule is not 0° but is calculated so that it will bisect the sides leaving the vertex:



A sub-molecule can be attached to the first vertex of a ring, just like the other vertices:

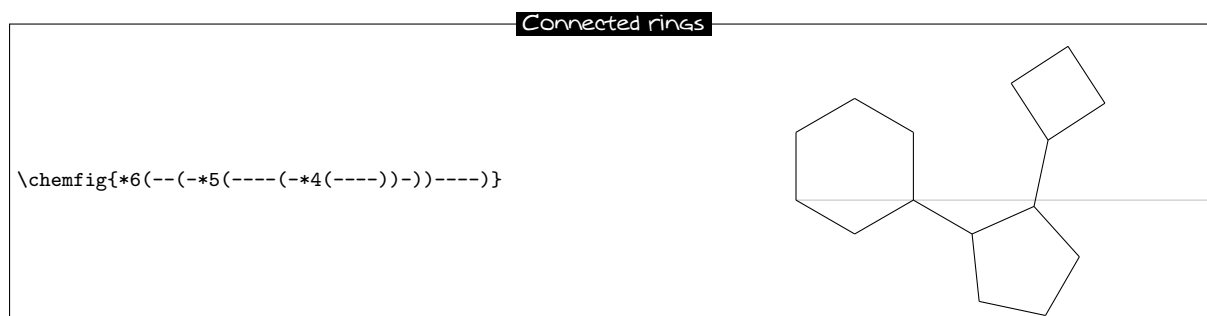


If one wants the bond leaving a vertex not to be the bisector of its sides, one can tinker with the optional global parameter or the optional bond parameter:



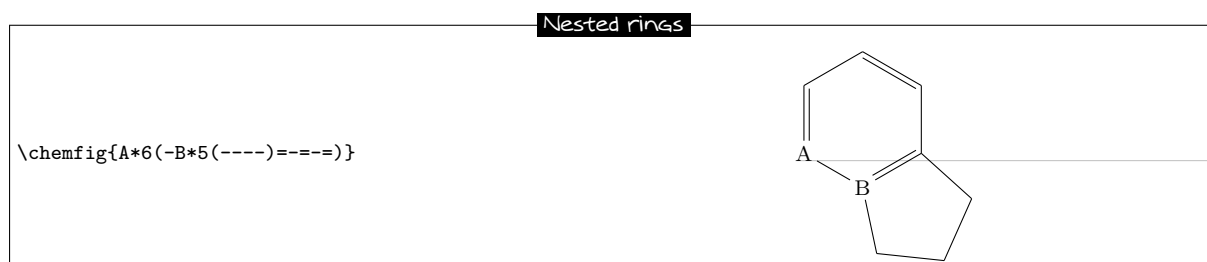
It is worth noting that in the third example, where a relative angle of 0° was given, the bonds of the branch are drawn in line with the preceding bond in the ring. This is the rule on page 12 which specified that the reference angle was that of the bond last drawn.

We can now connect together rings with bonds:



10.4 Nested rings

To “glue” two rings together, the syntax is only slightly different: the vertex is specified where the other ring is going to start. Simply follow this vertex by the usual syntax for a ring. Here for example is a 5-ring which is attached to the second vertex of a 6-ring:

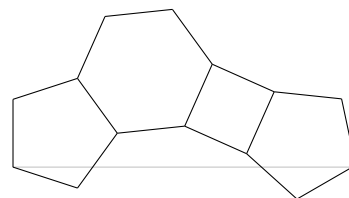


Note that the ring which is going to be attached to the main ring has an angular position such that two of the rings' sides coincide. In addition, the 5-ring has only four bonds "----". In effect, the fifth will be useless because it is the second side of the 6-ring, which has already been drawn.

It is quite possible to glue multiple rings together:

Multiple nested rings

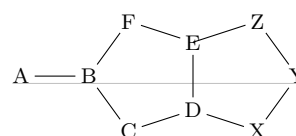
```
\chemfig{*5(--*6(-*4(-*5(----)--)----)----)}
```



There is a case where a trick must be used. It can be seen in this example that the fourth side of the second 5-ring just passes through the centre of atom "E".

Flawed drawing

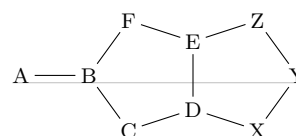
```
\chemfig{A-B*5(-C-D*5(-X-Y-Z-)-E-F-)}
```



This is normal because the second 5-ring (which is attached to atom "D") is drawn *before* **ChemFig** knows about atom "E". In this case, it is necessary to use two hooks to draw the bond Z-E:

Distant bond and ring

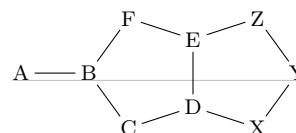
```
\chemfig{A-B*5(-C-D*5(-X-Y-Z?-E?-F-))}
```



We could also use a `` at the last vertex of the 5-ring:

Using `\phantom`

```
\chemfig{A-B*5(-C-D*5(-X-Y-Z-\phantom{E})-E-F-)}
```

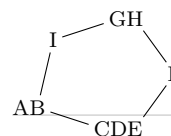


10.5 Rings and groups of atoms

Some care must be taken with rings when one or more vertices are made up of groups of atoms:

Ring and groups of atoms

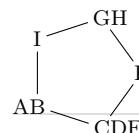
```
\chemfig{AB*5(-CDE-F-GH-I-)}
```



In order for the ring to have a regular shape, it is necessary to override the **ChemFig** mechanism which automatically calculates the departure and arrival atoms of bonds. Here, C-F and F-G must be connected by using the optional argument of these bonds:

Forced departure and arrival atoms

```
\chemfig{AB*5(-CDE-[,,,1]F-[,,,1]GH-I-)}
```



PART IV

Advanced usage

1 Displaying atoms

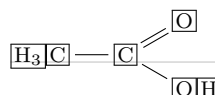
Once a molecule has been split into atoms, the macro `\printatom` is called internally by **ChemFig** in order to display each atom. Its sole argument is the code of the atom to be displayed (e.g. “`H_3`”). By default, this macro enters math mode and displays its argument with the math font family “rm”. It is defined by the following code:

```
\newcommand*\printatom[1]{\ensuremath{\mathrm{#1}}}
```

One can modify the code of this macro to customize how atoms are displayed. In the following example, we redefine `\printatom` so that each atom will be enclosed in a rectangular box:

Redefinition of `\printatom`

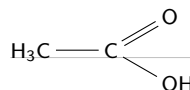
```
\fboxsep=1pt
\renewcommand*\printatom[1]{\fbox{\ensuremath{\mathrm{#1}}}}
\chemfig{H_3C-C(=[:30]O)(-[:-30]OH)}
```



Here is how to redefine it to use the “sf” font family of math mode:

Atoms displayed with “sf” font family

```
\renewcommand*\printatom[1]{\ensuremath{\mathsf{#1}}}
\chemfig{H_3C-C(=[:30]O)(-[:-30]OH)}
```



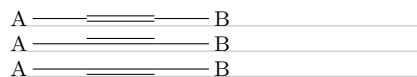
2 Shifted double bonds

All double bonds are made up of two line segments, and these segments are drawn on either side of the imaginary line along which a single bond would be drawn. It is possible to shift a double bond so that one of the line segments lies on the imaginary line. The other segment is then shifted above or below the bond. Actually, it is more correct to say “left” or “right” of the imaginary line, as the bond is traversed in the direction of drawing.

To shift the bond to the left, write “`=^`” and to shift it to the right, write “`=_`”:

Shifted double bonds

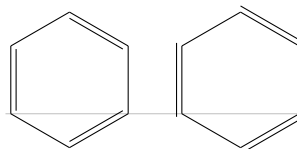
```
\chemfig{A==B}\par
\chemfig{A=^B}\par
\chemfig{A=_B}
```



In rings, double bonds are automatically shifted to the left. However, they can be shifted to the right by specifying it with “`=_`”:

Shifted double bonds and rings

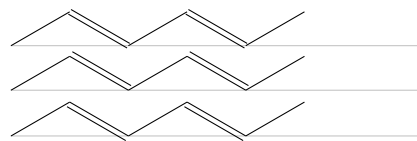
```
\chemfig{*6(==)}\quad
\chemfig{*6(=_)}
```



Shifted bonds are particularly useful in drawing skeleton diagrams of molecules consisting of carbon chains with double bonds. They give a continuous zig-zag path, whereas the path will be broken with regular double bonds:

Shifted bonds and skeleton diagrams

```
\chemfig{-[:30]=[:-30]-[:30]=[:-30]-[:30]} \par
\chemfig{-[:30]=^[:-30]-[:30]=^[:-30]-[:30]} \par
\chemfig{-[:30]=_[:-30]-[:30]=_[:-30]-[:30]}
```



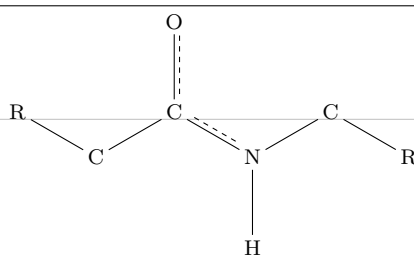
3 Delocalized double bonds

It is sometimes necessary to draw a double bond so that one line segment is dashed while the other is solid⁴. This feature is not hardcoded into **ChemFig**; instead *tikz*, its “decorations” library and its programmable styles make it possible.

First of all, after having loaded the “decorations” library by putting `\usetikzlibrary{decorations}` in the preamble, a decoration named “ddbond” (for Dashed Double Bond) is defined (lines 1 to 14) and then two *tikz* styles called “lddbond” and “rddbond” (lines 15 and 16). For the former, the dashed line segment is on the left of the solid middle segment; for the latter it is on the right. These styles can be called with the fifth optional argument of bonds:

Custom decorations

```
\pgfdeclaredecoration{ddbond}{initial}
{
  \state{initial}[width=4pt]
  {
    \pgfpathlineto{\pgfpoint{4pt}{0pt}}
    \pgfpathmoveto{\pgfpoint{2pt}{2pt}}
    \pgfpathlineto{\pgfpoint{4pt}{2pt}}
    \pgfpathmoveto{\pgfpoint{4pt}{0pt}}
  }
  \state{final}
  {
    \pgfpathlineto{\pgfpointdecoratedpathlast}
  }
}
\tikzset{lddbond/.style={decorate,decoration=ddbond}}
\tikzset{rddbond/.style={decorate,decoration={ddbond,mirror}}}
\setatomsep{4em}
\chemfig{[:-30]R-C-[:60]C(-[:60,,,rddbond]O)-[,,,,lddbond]N(-[:60]H)-[:60]C-R}
```



4 Saving a sub-molecule

ChemFig is capable of saving a <code> as an alias for reuse in a more compact form in the code of a molecule. This is particularly useful when the <code> appears several times.

To do this, one gives the command

```
\definesubmol{<name>}{<code>}
```

which saves the <code> for recall in the code of the molecule via the shortcut “!**{name}**”. All alphanumeric characters⁵ are accepted for the <name>. The <code> can itself contain other aliases.

⁴Thanks to Andreas BRÖERMANN who suggested this feature and who gave me the solution to this problem.

⁵Actually all the characters that can be put between `\csname...` and `\endcsname` are allowed.

A previous definition can be overwritten by

`\redefinesubmol{<name>}{<code>}`

Here is a code which draws the pentane molecule. An alias “xy” was defined beforehand for the code CH₂:

Pentane

```
\definesubmol{xy}{CH_2}
\chemfig{H_3C-!{xy}-!{xy}-!{xy}-CH_3}
```

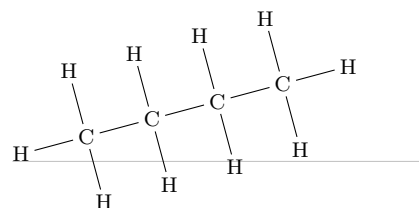


In this case the technique is not very interesting because “!{xy}” is just as long to type as the code it replaces.

But in certain cases, this feature saves a lot of space in the code of the molecule and increases readability. In the following example, we draw the complete structural diagram of butane. We will define an alias “CH₂” for the sub-molecule CH₂. If we use only relative angles, it is possible to rotate the entire molecule to any given angle by using the optional global angle parameter which specifies the default bond angle of the main molecule. It is set to 15° here:

Butane

```
\definesubmol{CH2}{C(-[:+90]H)(-[:-90]H)}
\chemfig{[:15]H-!{CH2}-!{CH2}-!{CH2}-!{CH2}-H}
```



5 Decorations

5.1 Lewis diagrams

The macro `\lewis` allows placement of pairs of electrons, of single electrons, or of empty slots. This syntax is used:

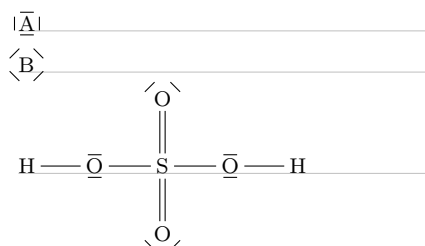
`\lewis{<n1><n2>...<ni>,<atom>}`

where the `<n1>...<ni>` represent the desired positions (in multiples of 45°) around the `<atom>`. These whole numbers must be between 0 and 7.

This command can also be used inside the argument of `\chemfig`:

The \lewis macro

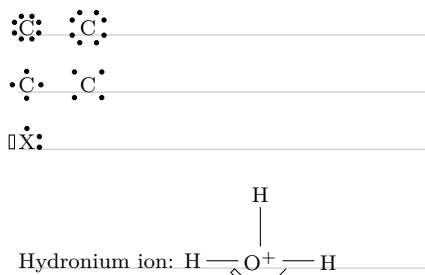
```
\lewis{0246,A}\par\medskip
\lewis{1357,B}\par\medskip
\chemfig{H-\lewis{26,0}-S(=[2]\lewis{13,0})
(=[6]\lewis{57,0})-\lewis{26,0}-H}
```



If one wishes to draw two electrons instead of a line, follow the integer with a “:”. If one wishes to draw a single electron, follow it with a “.”. To draw a lacuna, follow it with a “|”:

Lewis diagrams

```
\lewis{0:2:4:6:,C}\quad\lewis{1:3:5:7:,C}\par\bigskip
\lewis{0.2.4.6.,C}\quad\lewis{1.3.5.7.,C}\par\bigskip
\lewis{0:2.4|,X}\par\bigskip
Hydronium ion: \chemfig{H-\lewis{5|7,0^+}(-[2]H)-H}
```



All the decorations drawn by `\lewis` are not included in the bounding box of the atom; they are drawn afterwards. A consequence of this is seen in the two examples above, where the frame does not appear to be properly fitted to the drawing of the molecule, which extends downward slightly. This will be seen more often in this the “Decorations” chapter, which presents commands which do not change the bounding box.

This can be seen more clearly by drawing an `\fbox` around decorated atoms:

Bounding box and the <code>\lewis</code> macro	
<code>\fboxsep0pt</code>	
<code>\fbox{\lewis{0.2.4.6.,A}}\par\medskip</code>	
<code>\fbox{\lewis{13,B}}</code>	

Several parameters can be set with the help of the macro

`\setlewis{<dim1>}{<dim2>}{<tikz code>}`

If an argument is empty, it takes its default value.

- `<dim1>` is the distance between the bounding box and the decoration. It is 1.5pt by default;
- `<dim2>` is the length of the line segment representing a pair of electrons. It is 1.5ex by default;
- `<code tikz>` is code which is passed directly to `tikz`. This code is empty by default.

Parameters for the <code>\lewis</code> macro	
<code>\setlewis{4pt}{1.5em}{red}</code>	
<code>\chemfig{A-\lewis{26,B}-C}\par\bigskip</code>	
<code>\setlewis{}{}{line width=0.4pt}</code>	
<code>\chemfig{A-\lewis{21,B}-C}</code>	

5.2 Stacking characters

The macros

`\chemabove[<dim>]{<code>}{<stuff>}`

and

`\chembelow[<dim>]{<code>}{<stuff>}`

place the `<stuff>` above and below the `<code>` respectively at a vertical distance `<dim>`, without changing the bounding box of `<code>`. The length `<dim>` is 1.5pt by default.

These commands are independent of the macro `\chemfig` and can be used either inside or outside its argument.

They are especially useful in rings, if care is taken to put braces around the letters A, B, C and D in order to prevent **ChemFig** from starting a new atom on these letters:

Stacking in rings	
<code>\chemfig{*5(-\chembelow{A}{B}--\chemabove{C}{D}--)}</code>	

They are sometimes useful for placing pseudo-exponents which do not change the bounding box of the atoms, so that the bonds do not end up being too short:

Hydronium ion	
<code>\chemfig{H-\chemabove{\lewis{5 7,0}}{\quad\scriptstyle+}(-[2]H)-H}</code>	

5.3 Chemical reactions

To write chemical reactions, **ChemFig** provides a command for the signs and a command for the arrows.

The command `\chemsign[<dim>]<sign>` typesets the `<sign>`, which is surrounded on both sides by an unbreakable horizontal space `<dim>` defaulting to 0.5em.

The command `\chemrel[<arg1>][<arg2>]{<arrow code>}` draws an arrow where the optional arguments `<arg1>` and `<arg2>` are placed above and below the arrow respectively, without modifying its bounding box. The `<arrow code>` is passed directly to `tikz` except when it involves “<>”, which allows two arrows to be drawn one above the other.

Types of arrows

```
A\chemrel{->}B\par
A\chemrel{<->}B\par
A\chemrel{<->}B\par
A\chemrel{<>}B\par
A\chemrel{->,red,thick}B
```

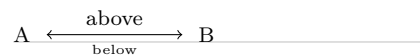


The command `\setchemrel{<dim1>}{<dim2>}{<dim3>}` allows settings of the dimensions used in drawing the arrow:

- `<dim1>` is the vertical spacing between the arrow and the optional text above and/or below it. If the argument is empty, it defaults to 2pt;
- `<dim2>` is the unbreakable horizontal space inserted before and after the arrow. If the argument is empty, it defaults to 0.7em;
- `<dim3>` is the length of the arrow. If the argument is empty, it defaults to 4em.

Text above arrows

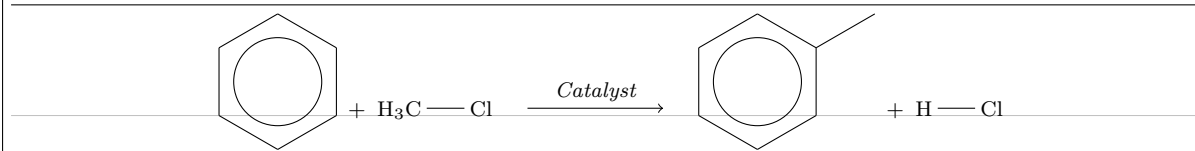
```
\setchemrel{1pt}{6em}
A\chemrel[\footnotesize above][\tiny below]{<->}B
```



Here is an example of a chemical reaction:

Chemical reaction

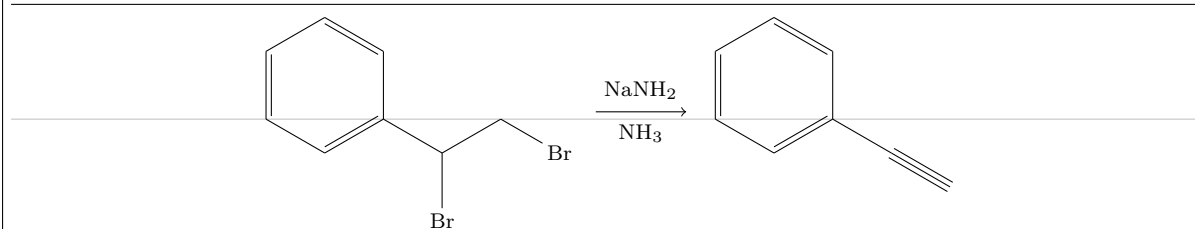
```
\setchemrel{0pt}{1.2em}{6em}
\chemfig{*6(-----)}\chemsign+\chemfig{H_3C-Cl}
\chemrel[\itshape\footnotesize Catalyst]{<->}
\chemfig{*6(---(-)---)}\chemsign+\chemfig{H-Cl}
```



And another:

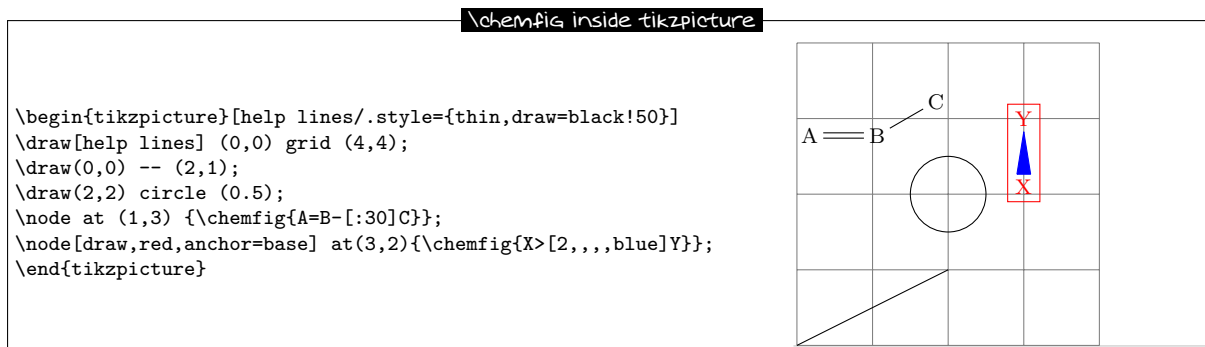
Synthesis of phenylacetylene

```
\chemfig{*6(==*6(-(-Br)-(-Br))-==)}
\chemrel[\chemfig{NaNH_2}][\chemfig{NH_3}]{<->}
\chemfig{*6(=-(--)-==)}
```



6 Using `\chemfig` in the `tikzpicture` environment

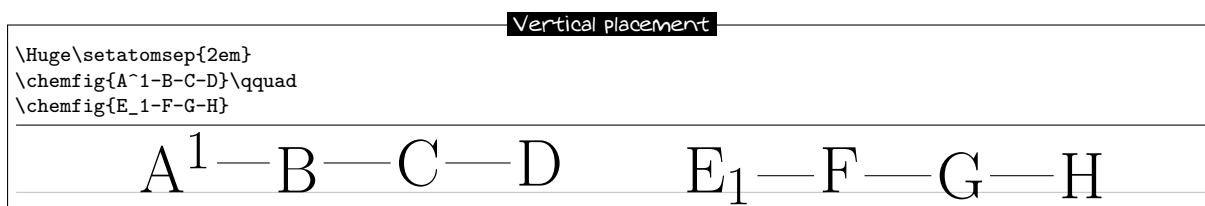
It is possible to call the `\chemfig` inside a `tikzpicture` environment:



7 Vertical alignment

In some cases with condensed structural diagram of molecules having horizontal bonds, the placement of groups of atoms is incorrect.

Careful study of the following example shows that the groups of atoms are not correctly aligned on the baseline:

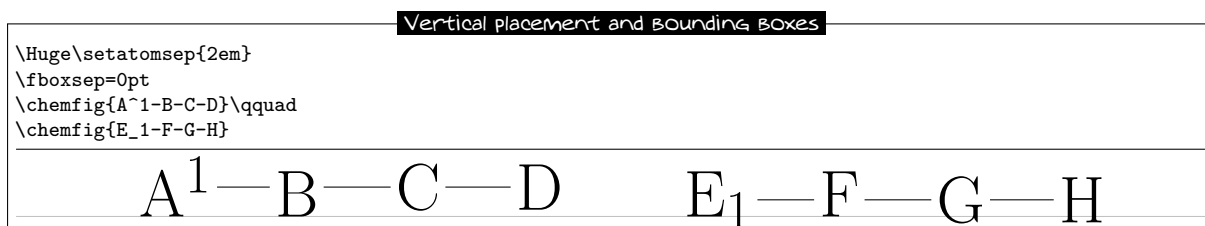


Surprisingly, the second atom is correctly aligned while the last two undergo a vertical shift which seems to be the results of the different height of the bounding box of the atoms “ A^1 ” and “ E_1 ”.

In order to understand this phenomenon, we need to consider how **ChemFig** places groups of atoms relative to each other. Let us limit ourselves to the case of horizontal bonds in order to simplify terminology, although the algorithm is the same for other bonds. A horizontal bond leaves from the middle of the right side of the bounding box of the departure atom of this bond. The arrival atom is positioned in such a way that the middle of the left side of its bounding box is at the end of the bond. It follows that the vertical placement of the arrival atom depends on the height of the departure atom. To limit this phenomenon, **ChemFig** adds to each arrival atom the `\vphantom` of the departure atom, but does not include it in the contents of the arrival atom; this `\vphantom` is not intended to affect the following atoms.

The defective alignment can thus be explained. The atoms “B” and “F” are aligned correctly as they reflect the height of the atoms before them because of their `\vphantom`. For the atoms “C” and “G”, the heights of the immediately preceding atoms are taken into account, but those of the atoms “ A^1 ” and “ E_1 ” are ignored! It follows that these atoms are a little too high or too low, depending on the height of these bonds.

We can show this by making visible the bounding boxes of the atoms; one sees clearly that the atoms “B” and “F” have bounding boxes that reflect the heights of the immediately preceding atoms:



Since there is no satisfactory manual solution, this problem can be worked around manually by putting *inside* the third atom a `\vphantom` having the same height as the first, so that the height affects the following atoms:

Vertical placement workaround

```
\Huge\setatomsep{2em}
\chemfig{A^1-B-{\vphantom{A^1}C}-D}\qqquad
\chemfig{E_1-F-{\vphantom{E_1}G}-H}
```



8 Beyond chemistry

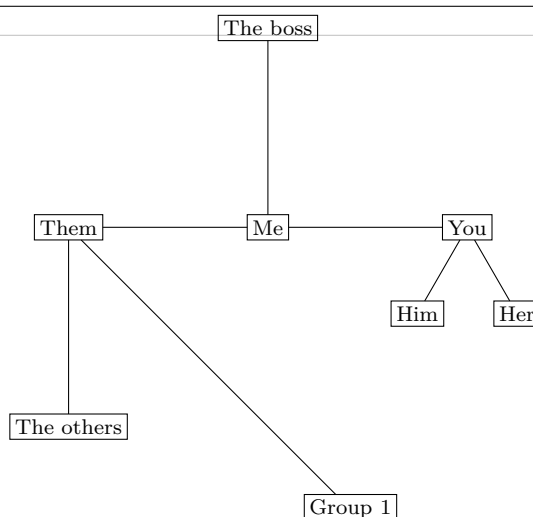
At heart **ChemFig** is a tool for drawing graphs, and this tool has been programmed to adapt it for chemistry. In some ways it is possible to return **ChemFig** to its roots to draw organization charts or other diagrams represented by graphs.

Each atom is contained in a `tikz` node. By default these nodes have an “inner sep” and an “outer sep” equal to 0pt. They are rectangular as seen on page 10. These defaults can be overwritten with the macro `\setnodestyle`, the argument of which is passed to `tikz` and specifies the style of the nodes containing the atoms.

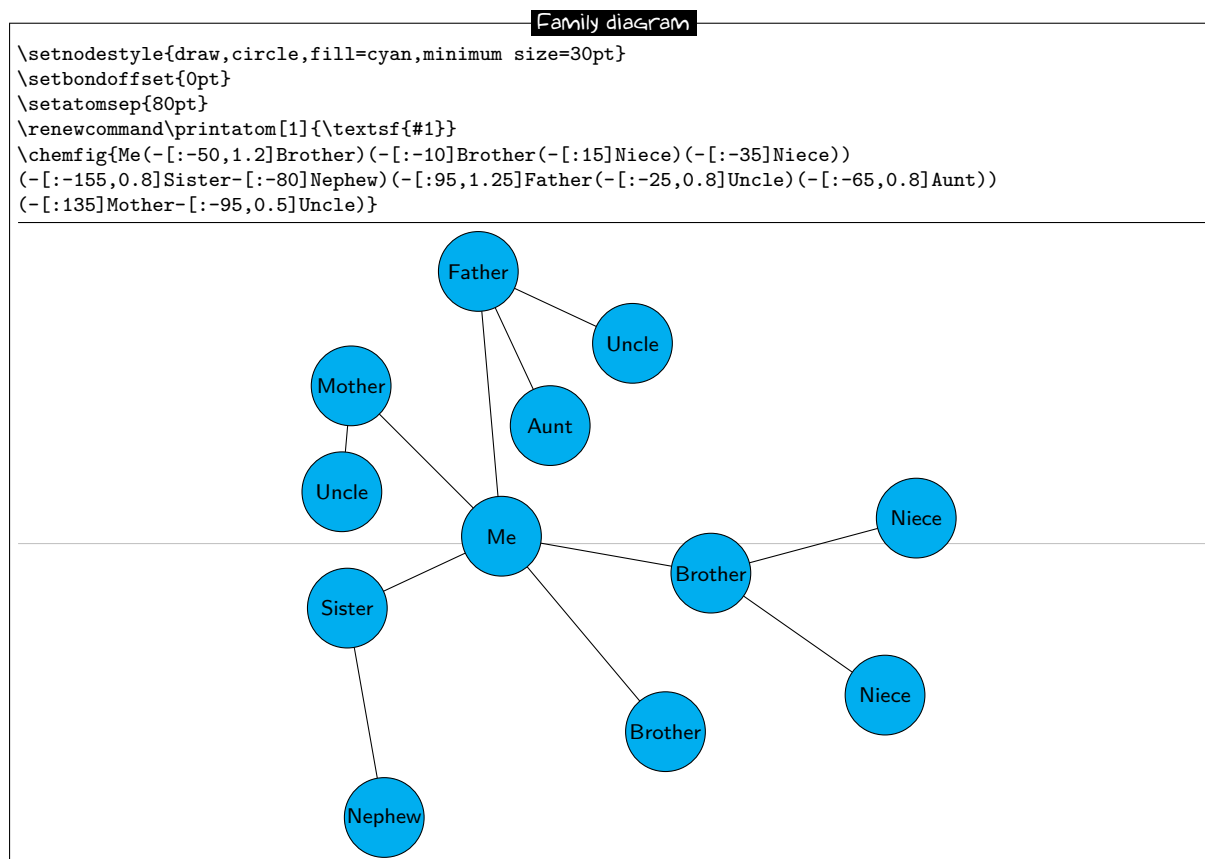
In this example we specify only “draw,inner sep=2pt”, which has the effect of drawing the outline of the nodes and separating the outline and node contents by 2pt. We also specify `\setbondoffset{0pt}` so that the bonds touch the edges of the nodes. The interatomic spacing is increased to 75pt. Finally, the command `\printatom` is made as simple as possible so that math mode is no longer used and spaces are thus preserved.

An organization chart

```
\setnodestyle{draw,inner sep=2pt}
\setbondoffset{0pt}
\setatomsep{75pt}
\renewcommand\printatom[1]{#1}
\chemfig{The boss- [6]Me(- [4]Them(- [6]The others)(- [7,2]Group 1))-You(-[: -120,0.5]Him)(-[: -60,0.5]Her)}
```



Here is another organization chart where the nodes are circular and coloured cyan:

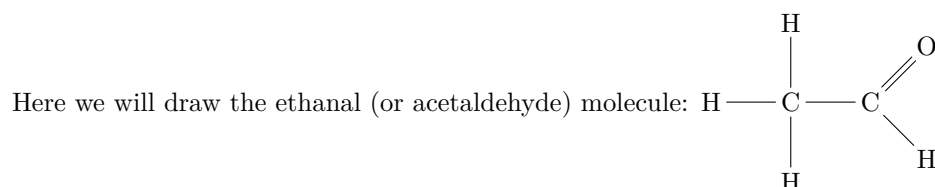


9 Annotated examples

In this chapter, several molecules will be drawn, putting into use the methods previously described. The aim here is to show a logical order for putting together a molecule so that the user unfamiliar with **ChemFig** will learn how to construct complex molecules. The construction steps will be shown to help with this learning process.

In addition, several possibilities — some intuitive and others less so — will be shown which give the same graphical results, with the objective being to show that **ChemFig** allows some flexibility in encoding molecules. One can see how each is put together and adopt the methods with which one is most comfortable.

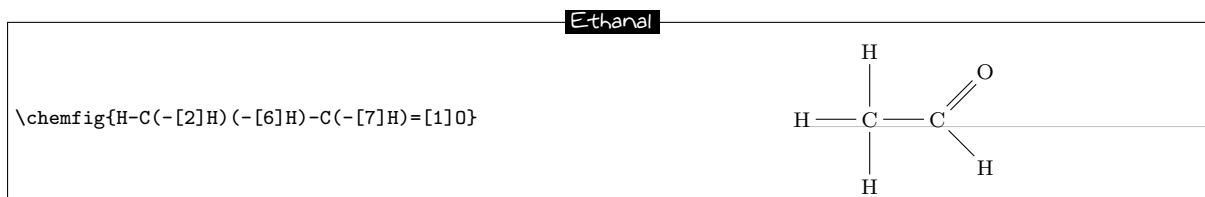
9.1 Ethanal



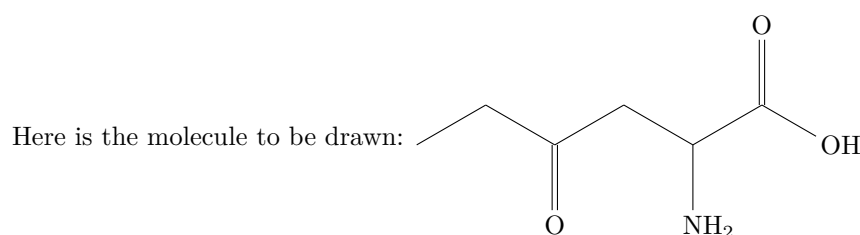
The best method for non-cyclic molecules is to select the longest chain. Here one could take “H-C-C=O” for example. The bond C=O is tilted to 45° by using the predefined angle “[1]”. This gives a “backbone” of the molecule to which the branches merely have to be added:



The three hydrogen atoms still have to be placed at the correct orientation with the help of predefined angles. The first is at 90° with the branch “(-[2]H)”, the second at 270° with “(-[6H])”, and the one on the right at 315° with “(-[7]H)”:



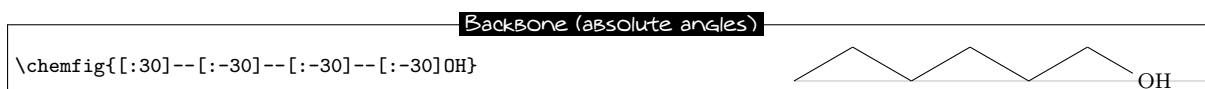
9.2 2-amino-4-oxohexanoic acid



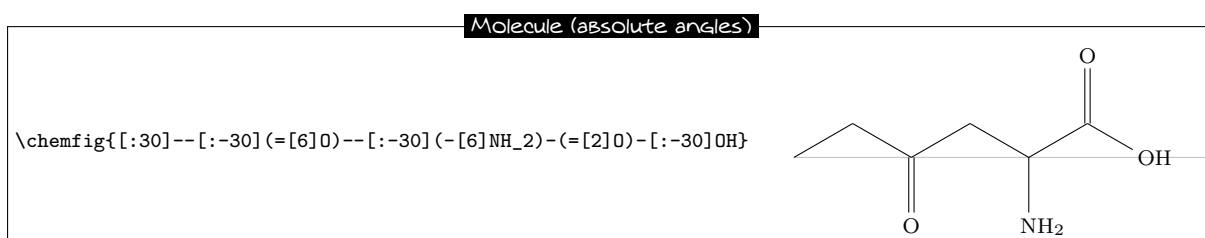
As is often the case for most molecules, there are several methods and for each several different ways of getting the result. Here we will look at four different methods.

9.2.1 Absolute angles

We will first of all draw the central chain with absolute angles. We set the default angle to $+30^\circ$ with the optional argument, and so only the descending bonds need to have their absolute angle set to -30° :

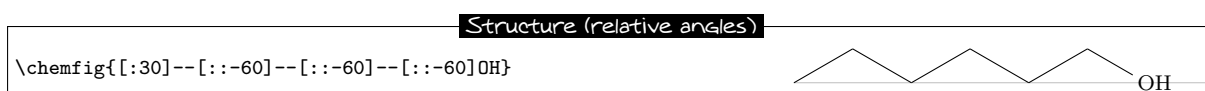


The branches “(=[6]O)”, “(-[6]NH₂)” and “(=[2]O)” still have to be added to the correct vertices:

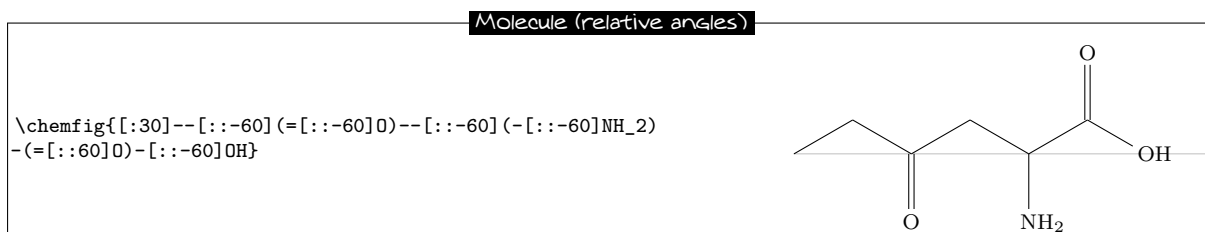


9.2.2 Relative angles

A more general approach uses only relative angles, in this way:



then

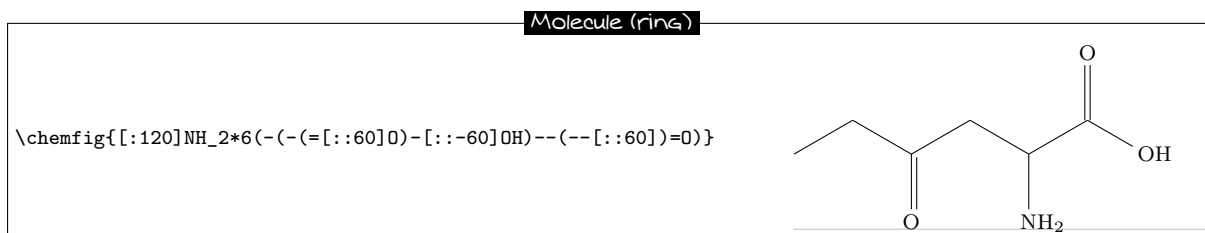


9.2.3 Ring

Since the angles between the bonds are 120° , it is possible to use a 6-ring, although this method is less natural. Here we take advantage of the fact that a ring can be left unfinished. The ring must be rotated 120° so that the first vertex is to the south-east of the ring:



Now the branches must be added to the right vertices:

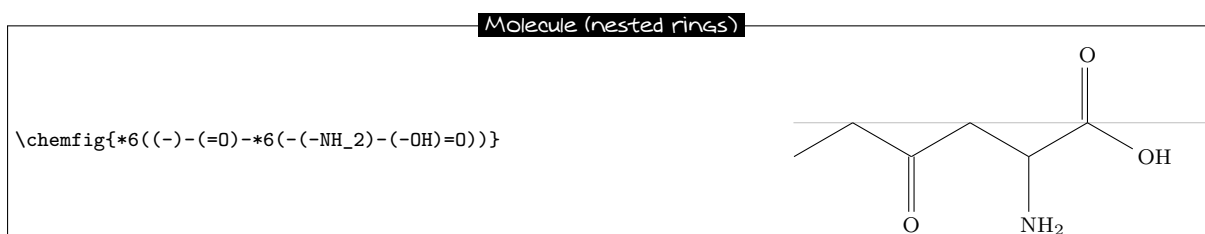


9.2.4 Nested rings

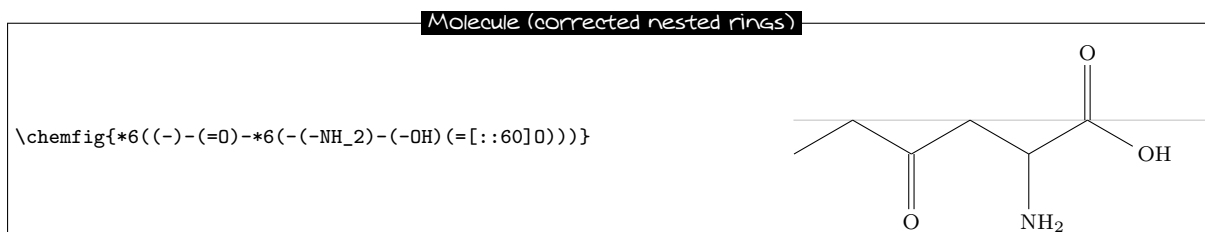
Delving deeper into the ring method, we can also consider nesting incomplete 6-rings. We could start with this backbone:



And then add the bonds which leave the vertices of these rings. There are no angles to worry about because the bonds leaving the rings are the bisectors of the sides of the ring, exactly what we want here:



A close look shows that the second line segment of the double bond to the oxygen atom is *inside* the incomplete 6-ring⁶. Despite its brevity, this code does not give a perfect drawing. This can of course be corrected by adding a little to the code:



9.3 Glucose

The goal here is to represent the glucose molecule according to several different conventions.

⁶This was also true for the preceding method with one ring.

9.3.1 Skeleton diagram

The code here looks like that of 2-amino-4-oxohexanoic acid. This gives almost the same structure with absolute angles, except here the default angle is -30° :

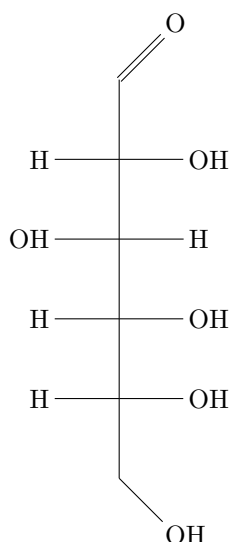


Adding the branches is no problem. We use predefined absolute angles:

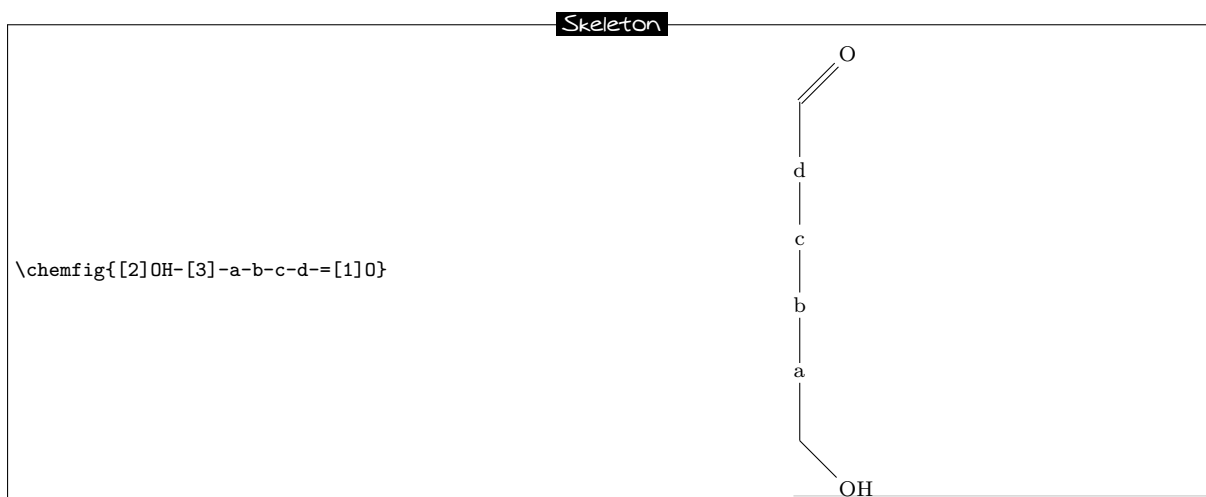


9.3.2 Fisher projection

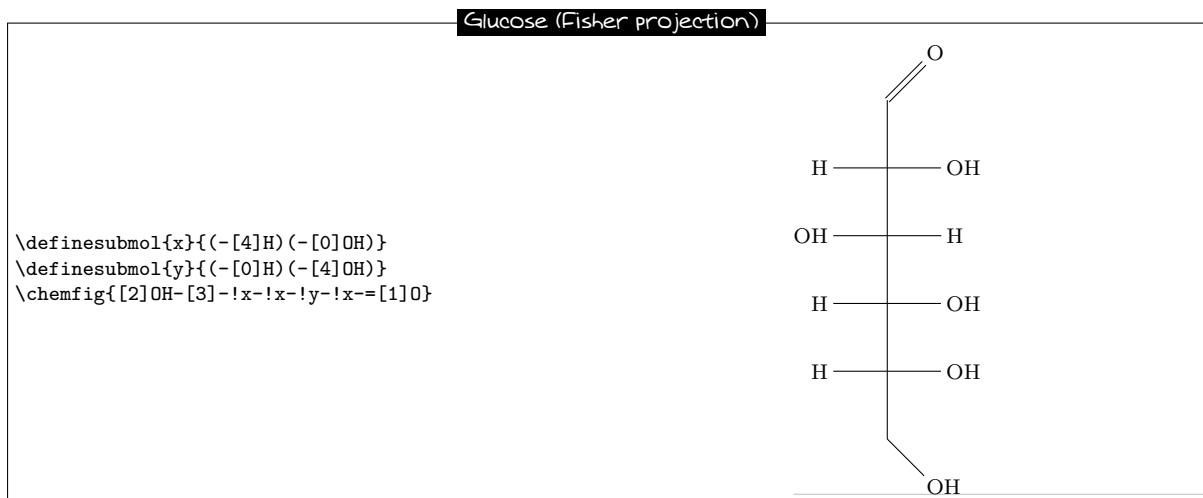
The goal is to get the molecule below:



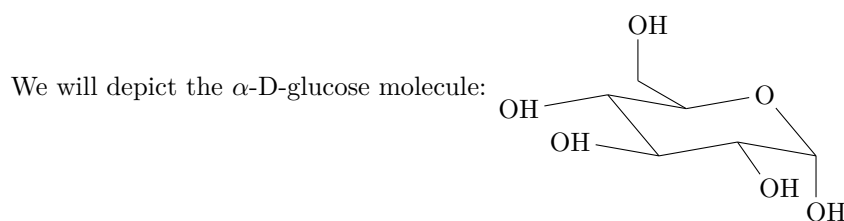
The idea is to begin to draw the longest chain vertically by giving a default angle of “[2]”. Here is the skeleton, where we have added lower case letters at the end of each vertical bond:



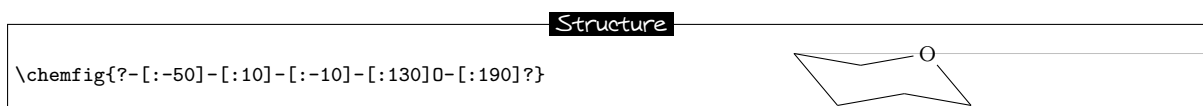
Next we define two aliases for the horizontal bonds and the atoms at their ends. Lets choose “x” which we will put in place of the lower case a, c and d, and “y” which will replace the letter c. Since these alias are just one character, we do not need braces and can write “!x” instead of “!{x}”:



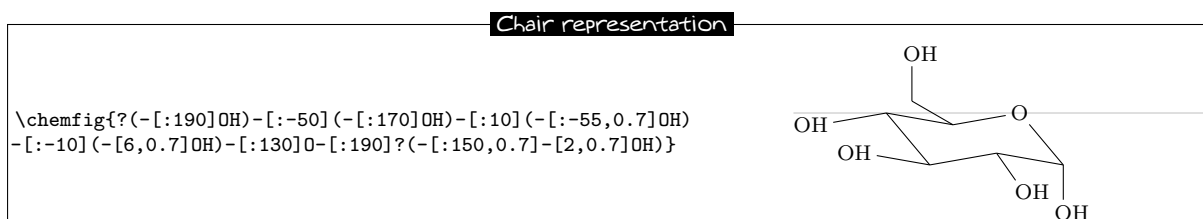
9.3.3 “Chair” representation



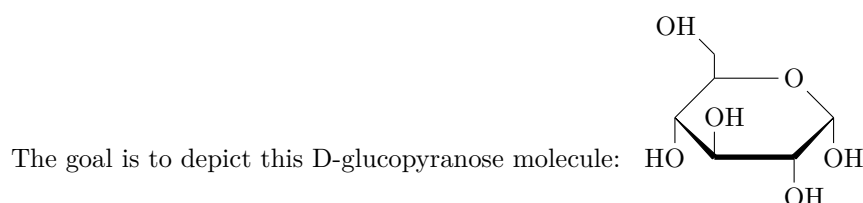
To do this, we will first of all draw five sides of the chair and link the first vertex to the last with a hook “?”. We will use the following absolute angles, running counterclockwise: -50° , 10° , -10° , 130° , 190° .



Now we simply add the branches inside parentheses. The angles are chosen to give the best impression of presentive, and some bonds are shortened by a factor of 0.7:



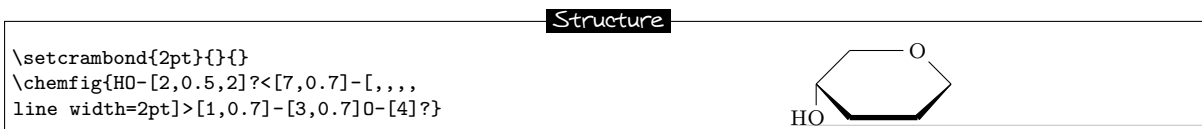
9.3.4 Haworth projection



First of all we will choose the longest chain, which starts at the “HO” group on the left and continues through fives sides of the ring. The ring will be closed with a hook. For the vertical bond which leaves from the first “HO” group, we need to specify that it will leave from the second atom using the optional

argument. Furthermore, it will be shortened with a coefficient of 0.5. Its optional argument will thus be “[2,0.5,2]”.

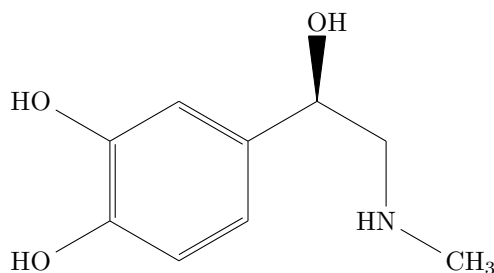
Next, to give the impression of perspective to the ring, the diagonal bonds will be shortened by a coefficient of 0.7. For the bold diagonal lines we will use Cram bonds, having redefined the base of the triangles to be 2pt. The bold horizontal bond needs to be drawn with a thickness of 2pt, and so its optional argument will be “[0,,,line width=2pt]”. Here is the skeleton of the molecule:



All that needs to be done now is to add the branches at the correct places, giving the right absolute angles and sometimes reducing the length to better give the illusion of perspective:



9.4 Adrenaline

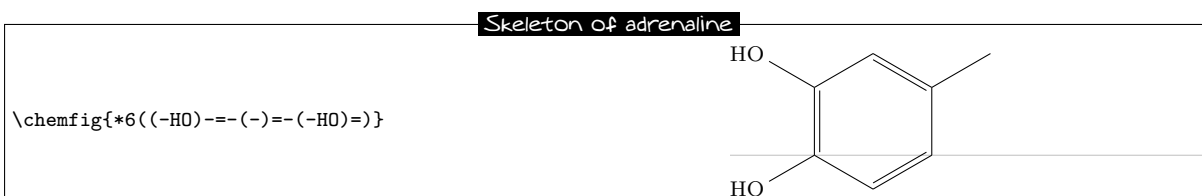


We want to draw the adrenaline molecule:

We are going to use two different methods.

9.4.1 Using one ring

First of all, we start with a 6-ring and we draw the start of the branches which leave it:

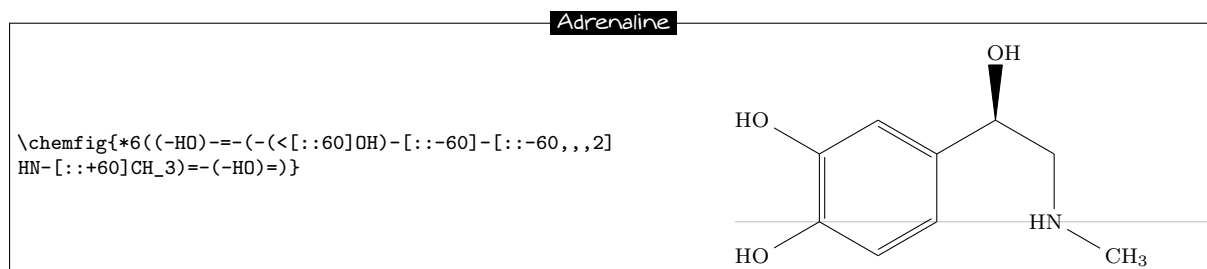


The branch on the right still needs to be completed using, for example, relative angles:



Then we need to add a Cram-bonded OH and indicate that the bond which arrives at “HN” does so on the

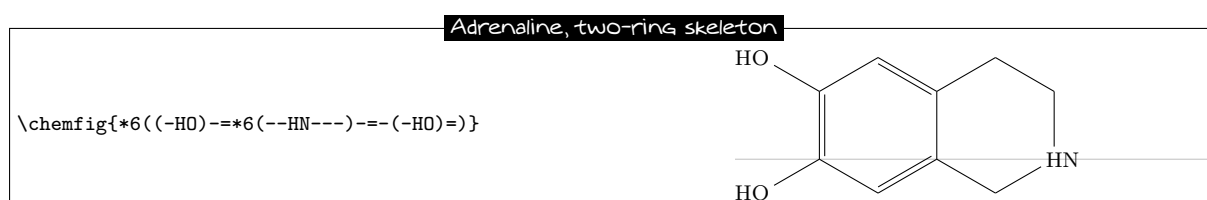
second atom, i.e., “N”. We use the fourth optional argument of the bond:



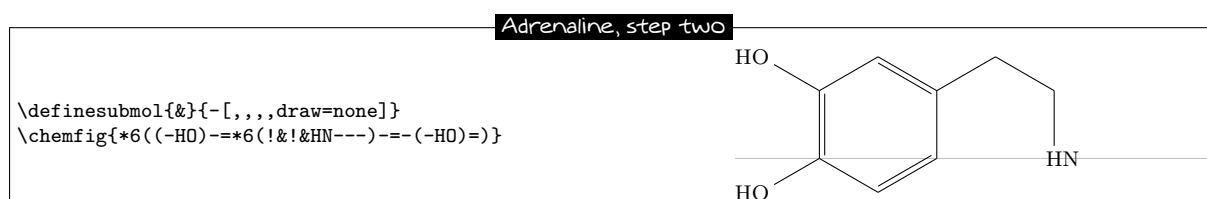
9.4.2 Using two rings

This method is less natural, but the goal is to show here how to make a bond invisible.

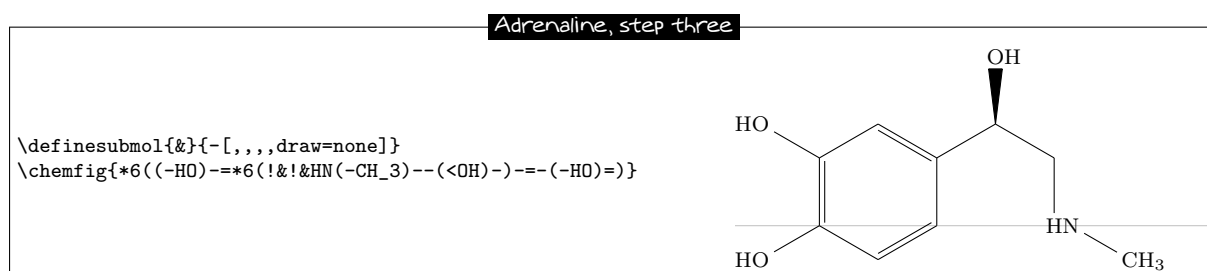
We could improve this code by considering that the drawing of the adrenaline molecule is made of two 6-rings adjacent to each other:



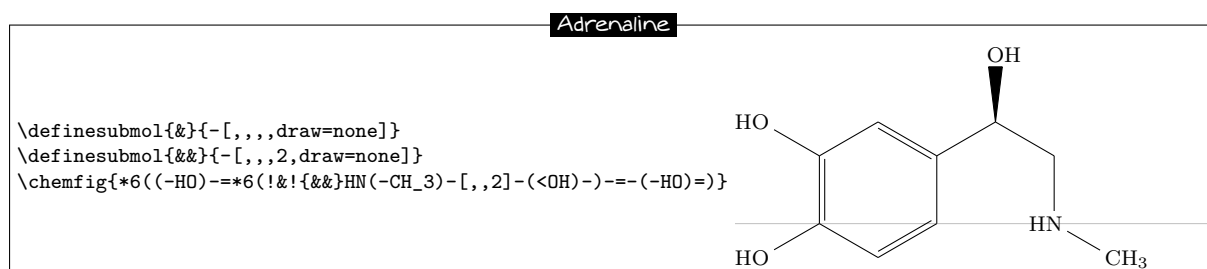
Now the first two bonds of the ring on the right need to be made invisible. To do this we use the argument that is passed to `tikz`, specifying “`draw=none`”. These bonds will thus have this code: “`-[,,,,draw=none]`”. To keep the code readable, we define an alias named “`&`” for these bonds:



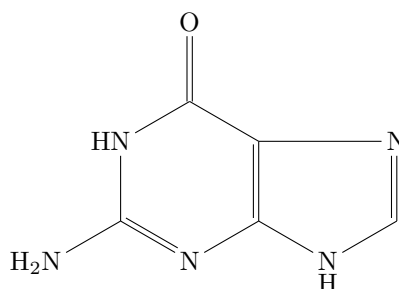
The rest becomes easy; just add the branches to the right vertices:



To finish, we specify that the bonds that *arrive at and leave from* “HN” must do so at the second atom. We therefore define another alias for the invisible bond which arrives at “HN”:



9.5 Guanine

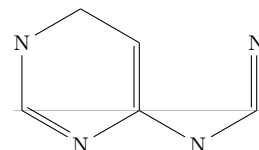


We will draw the guanine molecule:

First of all, let's begin by drawing the nested rings, putting just the nitrogen atoms at the vertices:

Guanine, skeleton

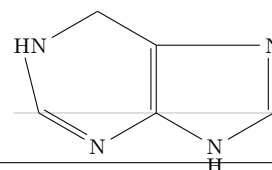
```
\chemfig{*6(=N-*6(-N=N)---N-)}
```



Then we can draw the horizontal bond in the right ring with a hook. We will also place a hydrogen atom under the nitrogen atom of the 5-ring with the command `\chembelow{N}{H}`. We also need to write “HN” instead of “N” at the vertex at the upper left of the molecule:

Guanine, step two

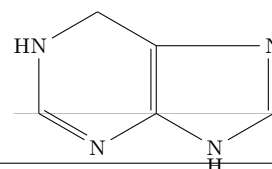
```
\chemfig{*6(=N-*6(-\chembelow{N}{H}--N?)=?--HN-)}
```



We note that one bond leaves from the wrong atom⁷! The automatic calculation mechanism must be corrected so that the bond leaves from the second atom “N” instead of the first. To do this we give an optional argument for the last bond of the first 6-ring “[, , 2]”:

Guanine, step three

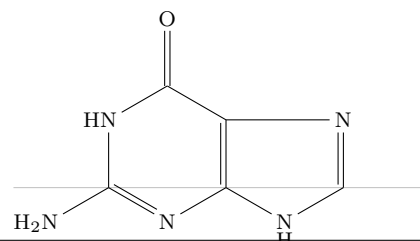
```
\chemfig{*6(=N-*6(-\chembelow{N}{H}--N?)=?--HN-[ , , 2])}
```



Simply add the branches to the right vertices. Note especially the branch leaving the first vertex of the first 6-ring “(-N₂N)”:

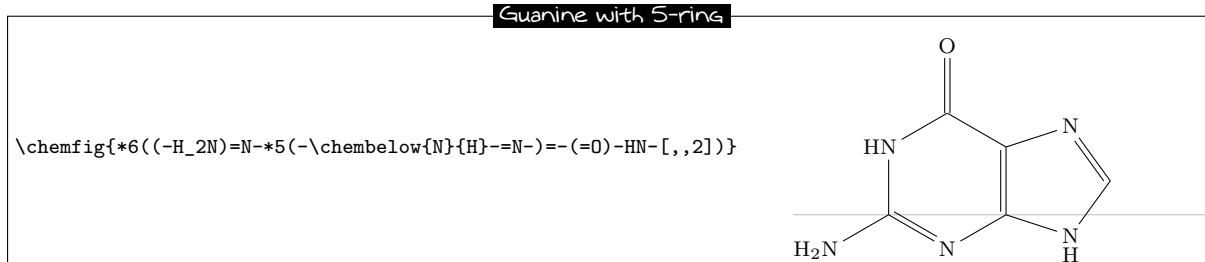
Guanine

```
\chemfig{*6((-H_2N)=N-*6(-\chembelow{N}{H}--N?)=?-(=O)-HN-[ , , 2])}
```



⁷This seems illogical because the angle of the bond from the HN group toward the first vertex lies between -90° and 90° ; ChemFig should therefore leave from the second atom. To explain this contradiction, one must know that in rings, the last bond always links the last vertex to the first, ignoring the *calculated theoretical* angle of this bond (which here is -90°). ChemFig uses this theoretical angle to determine the departure and arrival atoms, but does not use it to draw the bond because the two ends are already defined. The departure atom for the last bond is thus the first atom.

We could also draw the same molecule with a regular 5-ring, as is sometimes done:



10 List of commands

The commands created by **ChemFig** are:

Commands	Description
<code>\chemfig<code></code>	draws the molecule whose design is described by the <code><code></code>
<code>\printatom</code>	displays the atoms within the molecules. It can be redefined to customize the output. See page 23
<code>\setnodestyle{<style tikz>}</code>	using tikz syntax, this macro defines the style of nodes containing the atoms. See page 29
<code>\definesubmol{<name>}{<code>}</code>	creates an alias <code>!<name></code> which replaces <code><code></code> in the code of the molecule being drawn. See page 24
<code>\redefinesubmol{<name>}{<code>}</code>	replaces a pre-existing alias <code>!<name></code> with the new <code><code></code> . See page 25
<code>\setcrambond{<dim1>}{<dim2>}{<dim3>}</code>	sets the dimensions of the triangles representing Cram bonds: <code><dim1></code> is the size of the base, <code><dim2></code> is the spacing between the dashes and <code><dim3></code> is the side of the dashes. See page 11
<code>\setatomsep{<dim>}</code>	sets the interatomic distance. See page 11
<code>\setbondoffset{<dim>}</code>	sets the space between bonded atoms and the bond. See page 11
<code>\setdoublesep{<dim>}</code>	sets the spacing between the two lines of a double bond. See page 5
<code>\lewis{<codes>,<atom>}</code>	displays the <code><atom></code> and places Lewis dot decorations as specified in the <code><code></code> . The dots drawn do not change the bounding box. See page 25
<code>\setlewis{<dim1>}{<dim2>}{<tikz code>}</code>	sets the Lewis dot parameters; <code><dim1></code> is the distance between the atoms and the decoration, <code><dim2></code> is the length of the line representing the pair of electrons and <code><tikz code></code> is code which is passed directly to tikz. See page 26
<code>\chemsign[<dim>]<sign></code>	draws the <code><sign></code> , placing before and after it an unbreakable space of length <code><dim></code> . See page 26
<code>\chemrel[<txt1>][<txt2>]{<arrow code>}</code>	draws an arrow described by its <code><arrow code></code> and places the optional text <code><txt1></code> and <code><txt2></code> above and below the arrow respectively. See page 27
<code>\setchemrel{<dim1>}{<dim2>}{<dim3>}</code>	sets the dimensions of the arrow drawn with the command <code>\chemrel</code> : <code><dim1></code> is the vertical space between the arrow and the optional text, <code><dim2></code> is the unbreakable horizontal space inserted before and after the arrow and <code><dim3></code> is the length of the arrow. See page 27
<code>\chemabove[<dim>]{<txt1>}{<txt2>}</code>	writes <code><txt1></code> and places <code><txt2></code> above, leaving <code><dim></code> of vertical space. This command does not change the bounding box of <code><txt1></code> . See page 26
<code>\chembelow[<dim>]{<txt1>}{<txt2>}</code>	writes <code><txt1></code> and places <code><txt2></code> below, leaving <code><dim></code> of vertical space. This command does not change the bounding box of <code><txt1></code> . See page 26

PART V

Gallery

This manual concludes with drawings of molecules of varying complexity.

The curious user can look at the `<code>` of each molecule, though it does become less attractive the more complex the molecule gets. Indeed, beyond a certain level of complexity, though it is fairly easy to write `<code>`, it becomes much harder to read the `<code>` to analyse it afterwards. We quickly reached the limits of immediate readability of the code of a complex drawing.

Anyway, I hope that this package will help all L^AT_EX users wishing to draw molecules. Given the newness of this package, I hope that you will be forgiving with bugs you encounter and send me an [email](#) to let me know of any malfunctions or suggestions for improvement.

Christian TELLECHEA

TODO list :

- allow the user to specify, if he wishes for one bond, the distance δ (see page 10) between the ends of this bond and the bounding boxes of the atoms it links. One syntax idea would be to use the character “#”:

`<atom><bond characters>#(dim1,dim2)[<optional argument>]<atom>`

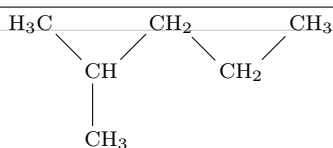
where `<dim1>` and `<dim2>` would be the dimensions δ at the start and end of the bond;

- add Newman projections to the features of ChemFig.

★
★ ★

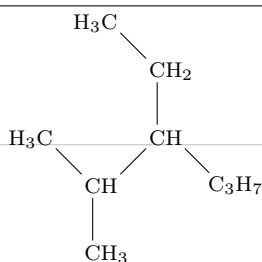
2-methylpentane

```
\chemfig{[7]H_3C-CH(-[6]CH_3)-[1]CH_2-CH_2-[1]CH_3}
```



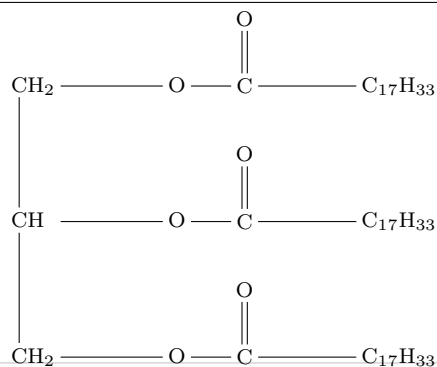
3-ethyl-2-methylhexane

```
\chemfig{H_3C-[7]CH(-[6]CH_3)-[1]CH(-[7]C_3H_7)-[2]CH_2-[3]H_3C}
```

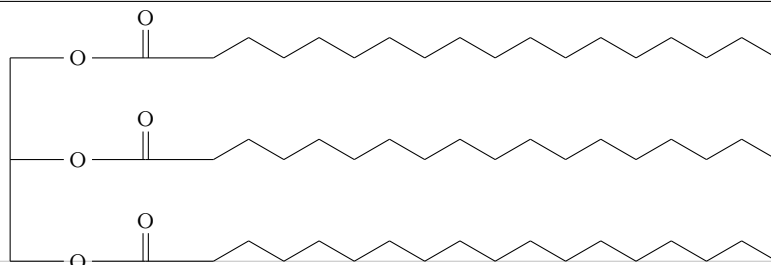


Stearine, condensed structural diagram

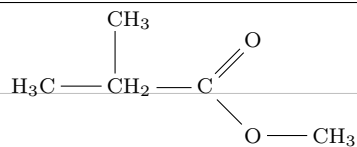
```
\definesubmol{@}{([0,2]-O-[0,1]C(=[2,1]O)-C_{17}H_{33}))}
\chemfig{[2,2]CH_2!@-CH_{\phantom 2}!@-CH_2!@}
```

**Stearine, skeleton diagram**

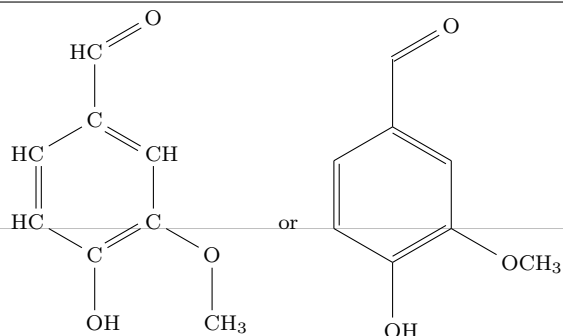
```
\definesubmol{x}{-[:+30,.6]-[: -30,.6]}
\definesubmol{y}{-O-(=[2,.6]O)-!x!x!x!x!x!x!x}
\chemfig{[2](!y)-[,1.5](!y)-[,1.5](!y)}
```

**Methyl 2-methylpropanoate**

```
\chemfig{H_3C-CH_2(-[2]CH_3)-C(=[1]O)-[7]O-CH_3}
```

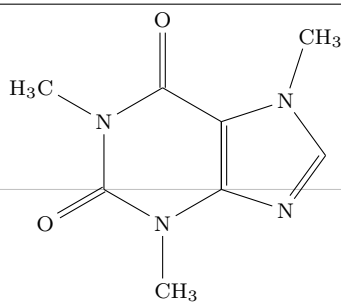
**Vanillin**

```
\chemfig{HC*6(-C(-OH)=C(-O-[: -60]CH_3)-CH=C(-[, ,2]HC=[: -60]O)-HC=[: ,2])} \quad or \quad
\chemfig{*6(-(-OH)=(-OCH_3)-(-[: -60]O)-=)}
```

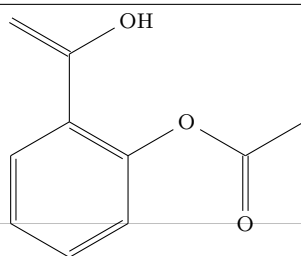


Caffeine

```
\chemfig{*6((=O)-N(-CH_3)-*5(-N=N(-CH_3)-)=--(=O)-N(-H_3C)-)}
```

**Aspirin**

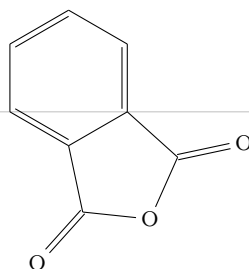
```
\chemfig{*6(=-(O-[: -60](=[: -60]O)-[: +60])=-(=[: +60])-[: -60]OH)-=)}
```



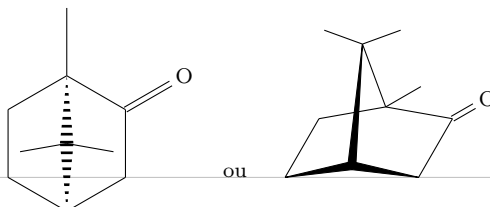
Aspirin is a registered trademark of Bayer in many countries.

Phthalic anhydride

```
\chemfig{*6(=*5(-(=O)-O-(=O)-)-===-)}
```

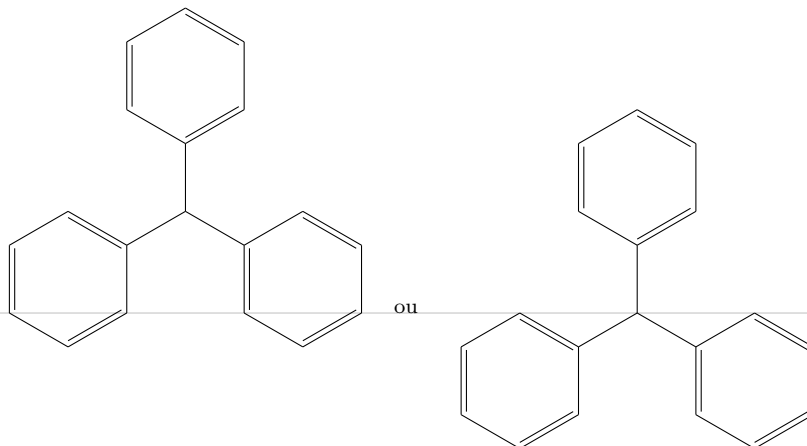
**Camphor**

```
\chemfig{*6(-(<[:120](-[: -100,0.7])(-[:100,0.7]))--(=O)-(-<[:120])--)}
\quad ou \quad
\setcrambond{3pt}{-}{-}
\chemfig{<[:10](>[:85,1.8]?(-[:160,0.6])-[:20,0.6])
>[: -10]-[:60](=[:30,0.6]O)-[:170]?(-[:30,0.6])-[:190]-[:240]}
```



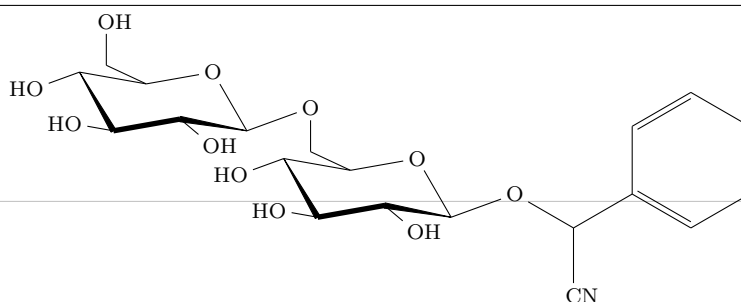
Triphenylmethane

```
\chemfig{*6(==*6(-(*6(==--)))*6(==--))==)}
\quad ou \quad
\definesubmol{@}{*6(==--)}
\chemfig{(-[:30]!@)(-[:90]!@)(-[:210]!@)}
```



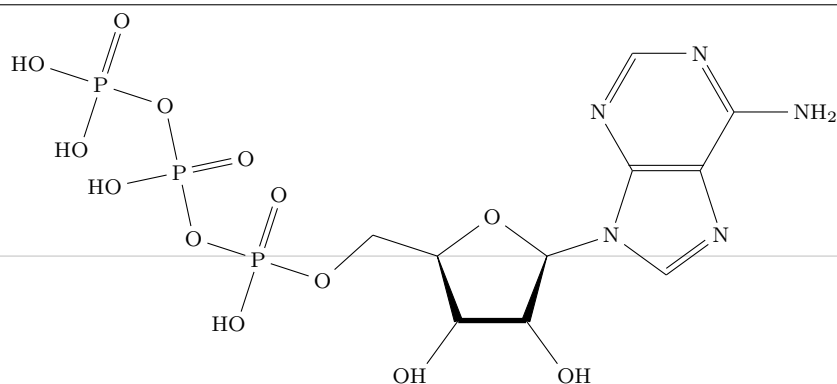
Amygdalin

```
\setcrambond{2pt}{-}{-}
\definesubmol{c1}{-[:200]-[:120]O-[:190]}
\definesubmol{c2}{-[:170](-[:200,0.7]HO)<[:300](-[:170,0.6]HO)
-[:10,,,line width=2pt](-[:40,0.6]OH)>[:10]}
\definesubmol{csub}{-[:155,0.65]-[:90,0.65]}
\chemfig{O(!{c1}(!{csub}O(!{c1}(!{csub}OH)!{c2}))!{c2})-[:30](-[:90]CN)-[:30]*6(==--))}
```



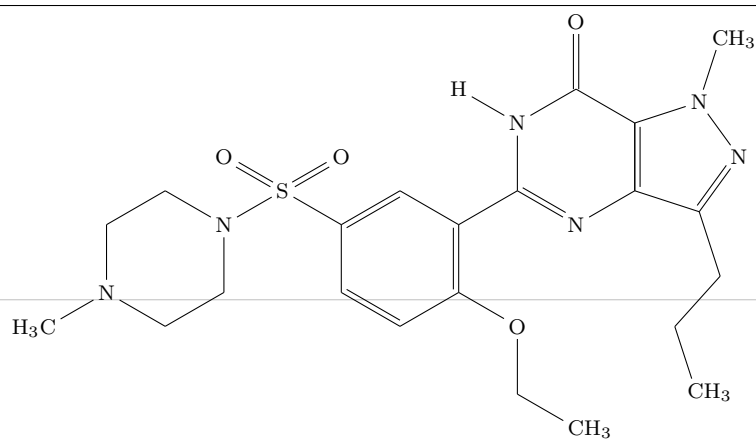
Adenosine triphosphate

```
\setcrambond{3pt}{-}{-}
\definesubmol{a}{-P(=[:90]O)(-[:90]HO)-}
\chemfig{[:54]*5((--[:60]O[:60]!aO[:60]!aO[:60]!aHO)))<(-OH)
-[,,,,line width=2pt](-OH)>(-N*5(-N*6(-(-NH_2)=N=N-)-)-O-)}
```



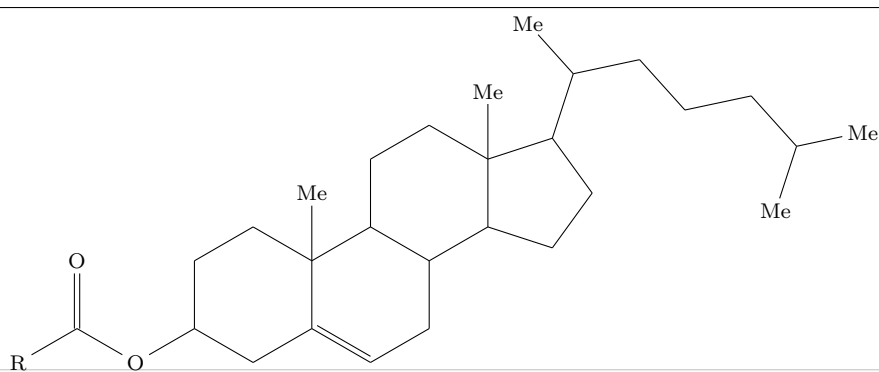
Viagra

```
\chemfig{N*6((-H_3C)---N(-S(=[::+120]O)(=[::+0]O)-[::-60]*6(=-(O-[::-60]-[::+60]CH_3)
=(-*6(=N-*5(-([::-60]-[::+60]CH_3)=N-N(-CH_3)-=)--(=O)-N(-H)-))=))---)}
```



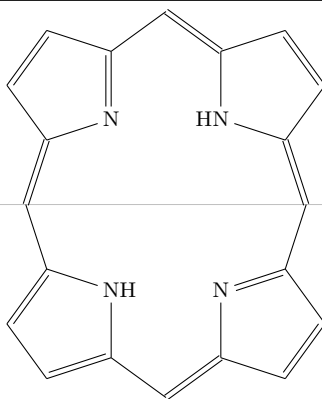
Cholesterol ester

```
\chemfig{[:30]R-(=[::+60]O)-[::-60]O-*6(--*6(==*6(-*5(---(-([::+60]Me)
-[::-60]-[::-60]-[::+60]-[::-60](-[::-60]Me)-[::+60]Me)-)-(-[::+0]Me)---))--(-([::+0]Me)---))}
```



Porphyrin

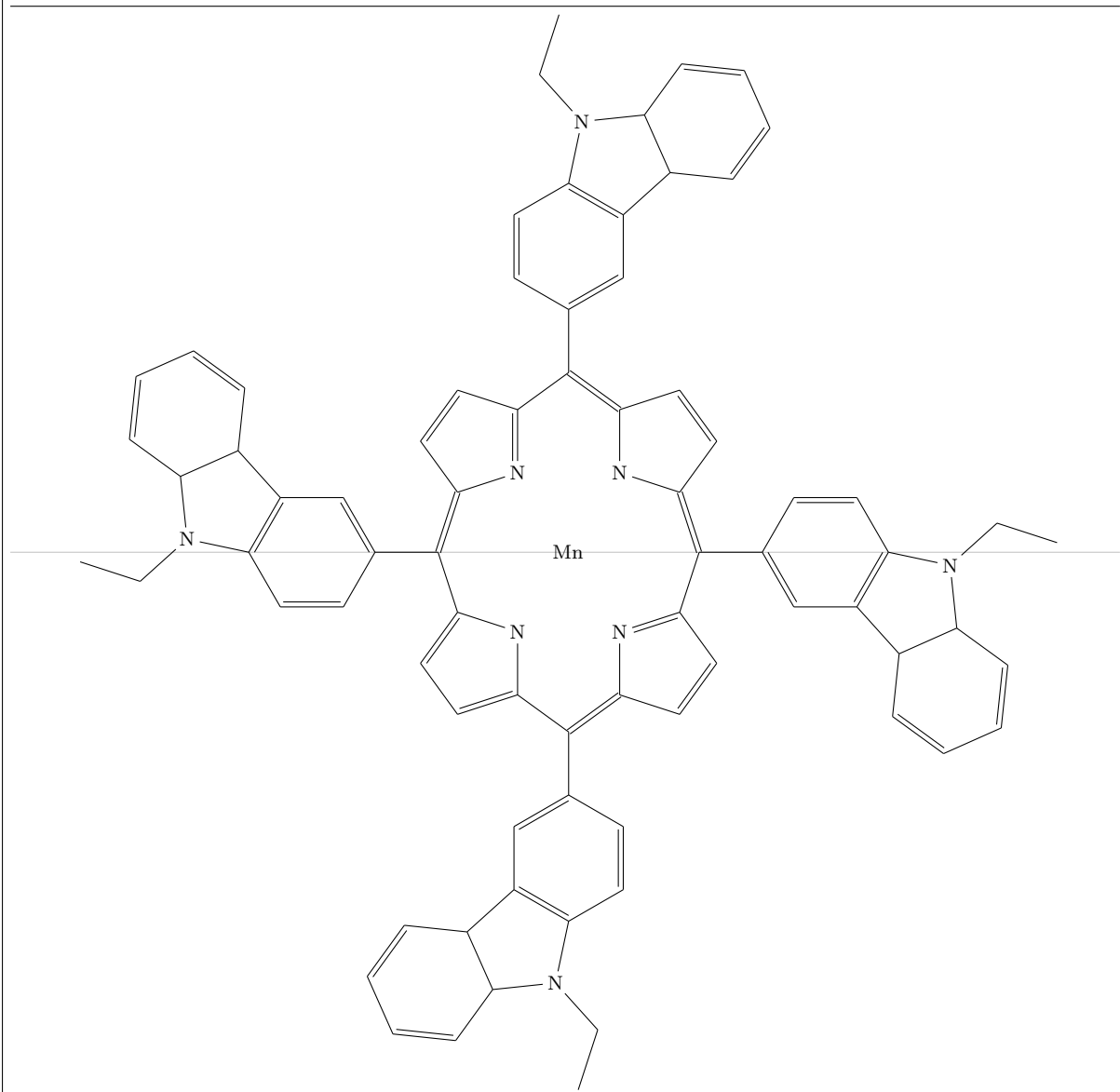
```
\chemfig{?=[::+72]*5(-N(=[::-72]*5(-[, , 2]HN-[ , 2](-=[::-36]*5(=N(-=[::-72]*5(-NH-[ , 1]?=))
--)))-))=)}\chemfig{?=[::+72]*5(-N(=[::-72]*5(-[, , 2]HN-[ , 2](-=[::-36]*5(=N(-=[::-72]*5(-NH-[ , 1]?=))
--)))-))=)}
```



Manganese 5,10,15,20-tetra(N-ethyl-3-carbazolyl) porphyrin

```
\definesubmol{A}{*6(=*5(=*6(==--)--N(--[::-60])--==--)}
\chemfig{([::+180]-!A)=[::+72]*5(-N(-(-[::+54]!A)=[::-72]*5(-N(-[::-33,1.5,,draw=none]Mn)
-(-[::+72]!A)-[::-36]*5(=N(-(-[::+54]!A)-[::-72]*5(-N(-)--))--))--))--)}

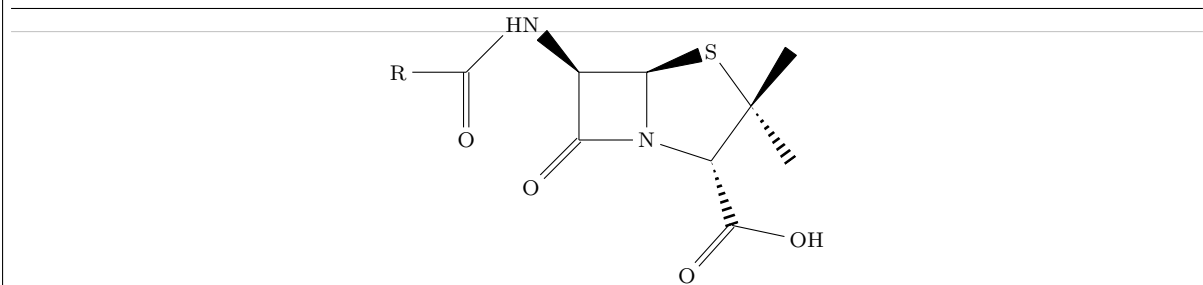
```



Penicillin

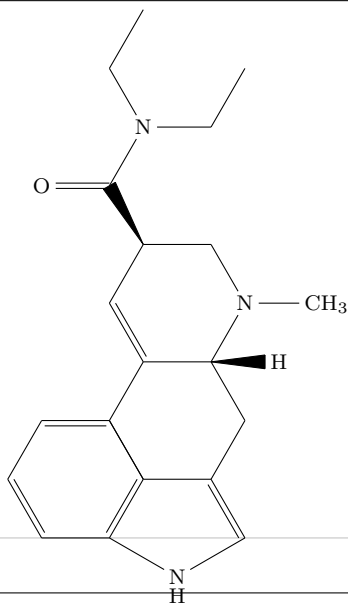
```
\chemfig{[: -90]HN(-[: -45](-[: -45]R)=[::+45]O)>[: +45]*4(- (=O)-N*5(-(<[: -60]O)
-[: +60]OH)-(<[: +0])(<[: -108]-S>--))}

```

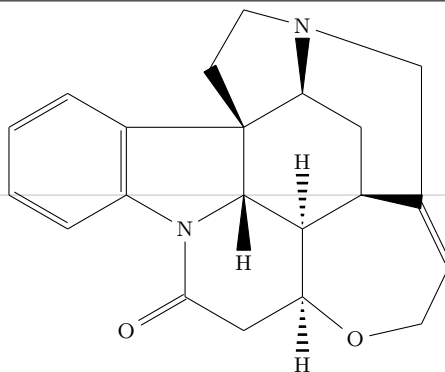


LSD

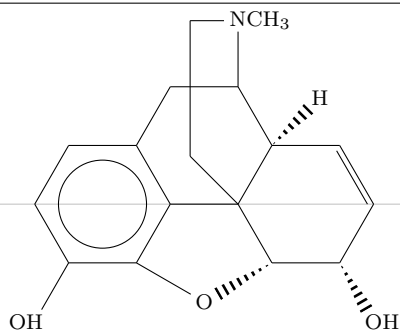
```
\chemfig{[:150]?*6(=*6(--*6(-N(-CH_3)--(<[:+60]O)-[::-60]N(-[:+60]-[::-60])
-[::-60]-[:+60])--)([::-120]<H)---)*6(==--(-[::-30,1.155]\chembelow{N}{H}{?})=))}
```

**Strychnine**

```
\chemfig{*6(=*6(-N*6(-=O)--([::-120]<:H)*7(-O--=?[O]([::-25.714]-[,2]?[1]))
-*6(-?[0,{>}--(<N?[1]?[2])-([::-90]-[::-60]?[2]))(<[:+0]H)-([::+120]<H)--?)=?==)}
```

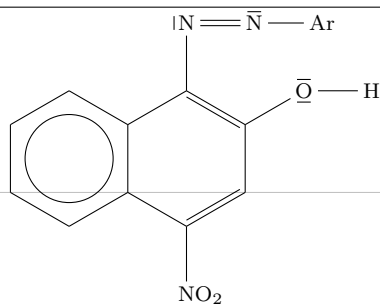
**Codeine**

```
\chemfig{[: -30]**6(-(-OH)-?*6(-(-[3]-[2,2]-[0,.5])**6(-(<[: -150,1.155]O?)
-(<:OH)--)-(<:[1]H)-(-[2]NCH_3)--)}{}
```



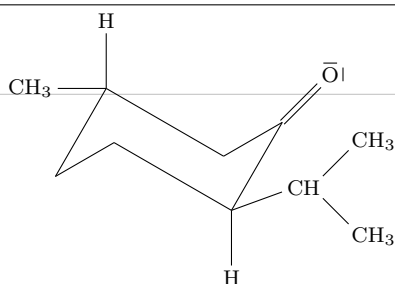
A dye (red)

```
\chemfig{*6(--*6(-(-NO_2)=(-\lewis{26,0}-[0]H)=(-\lewis{4,N}=[0]\lewis{2,N}-[0]Ar)-)----)}
```



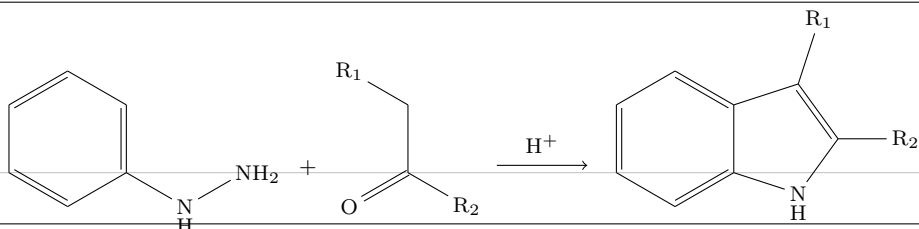
Menthone

```
\chemfig{CH_3-?(-[2]H)(-[: -30,2]-[: +60](=[1]\lewis{20,0})-[: -150,1.5](-[:20]CH(-[1]CH_3)(-[7]CH_3))(-[6]H)-[: -90,2]-[: +60]?)}
```



Fischer indole synthesis

```
\chemfig{*6(=*6(-\chembelow{N}{H}-NH_2)=--)}
\chemsign+
\chemfig{([:-150]O)(-[: -30]R_2)-[2]-[:150]R_1)\chemrel[\mathrm{H^+}]{->}
\chemfig{*6(=*5(-\chembelow{N}{H})-(-R_2)=(-R_1)-)=--=)}
```

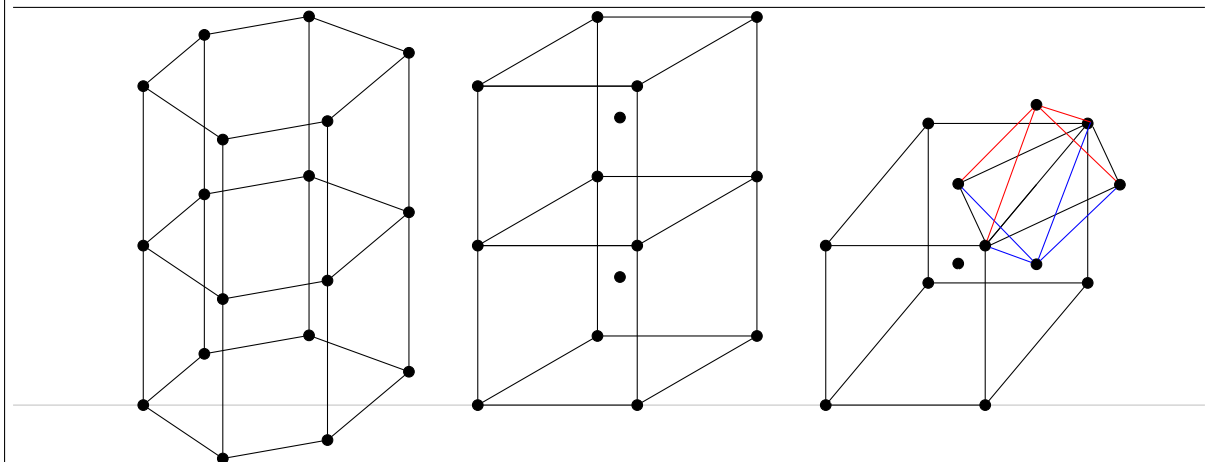


Cristallography

```

\newcommand\disk{\tikz\draw[fill=black,overlay](0,0)circle(2pt);}
\setatomsep{20pt}
\renewcommand\printatom[1]{#1}
\setbondoffset{2pt}
\definesubmol{hat}{-[:40,1.5]\disk-[:30,2]\disk-[:30,2]\disk-[:120,2]\disk-[:30,2]\disk}
\definesubmol{motif}{-[:40,1.5]\disk(-[2,3])-[:30,2]\disk
(-[2,3])-[:30,2]\disk(-[2,3])-[:120,2]\disk(-[2,3])-[:30,2]\disk}
\chemfig{\disk?{a}(-[2,3]\disk?{b}(-[2,3]\disk?{c}!\hat?{c})!\motif?{b}(-[2,3]))!\motif?{a}(-[2,3])}
\quad
\definesubmol{motif}{\disk(-[2,3])(-[:42,3.6,,draw=none]\disk)-[:30,2.6]\disk
(-[2,3])-[:0,3]\disk(-[2,3])-[:150,2.6]\disk(-[4,3])-[:2,3]-[:4,3]}
\redefinesubmol{hat}{\disk-[:30,2.6]\disk-[:0,3]\disk-[:150,2.6]\disk-[:4,3]}
\chemfig{!\motif!!\motif!!\hat!}
\quad
\redefinesubmol{motif}{(-[2,3])(-[:25,2.75,,white]-[:2,1.5,,white]\disk)-[:50,3]\disk
(-[2,3])-[:50,3]\disk(-[2,3])-[:130,3]\disk-[:2,3]-[:4,3]\disk}
\redefinesubmol{hat}{-[:50,3]\disk-[:50,3]\disk-[:130,3]\disk}
\chemfig{-[:4,3]\disk!\motif(-[:25,2.75,,draw=none]\disk?[uat]?[dat](-[:0,2.75]?[uat1]?[dat1]
-[:90,1.3]\disk?[uat2]?[dat2]-[:90,1.3]-[:25,2.75])!\hat?[uat3]?[dat3]-[:50,1.5]
(-[:6,1.5,,draw=none]\disk?[dat,,blue]?[dat1,,blue]?[dat2,,blue]?[dat3,,blue])
-[:2,1.5,,draw=none]\disk?[uat,,red]?[uat1,,red]?[uat2,,red]?[uat3,,red])}

```



Taxotere

```

\chemfig{-[:30](-[5])(-[7])-[:60]-[:60]0-[:60](=[:45]0)-[:90]HN>[:60](-[:60]**6(-----))
-[:30](<[:2]OH)-[:60](=[:6]0)-[:60]0>[:60]*7(---?(<[:120]OH)-(<[:1]CH_3)(<[:90]CH_3)
-(-[:1](<[:80]HO)-[:0](=[:60]0)-[:7](<[:130]CH_3)-[:75](<[:2]OH)-[:60]-[:60](<[:30]0-[:90])
-[:60](<[:90])(<[:30]0-[:7](-[:6]CH_3)=[:0]0)-[:60]-[:6]-[:5,1.3]?(<[:7]0-[:5](=[:60]0)
-[:6]**6(-----))=(-[:2]CH_3)-)}

```

