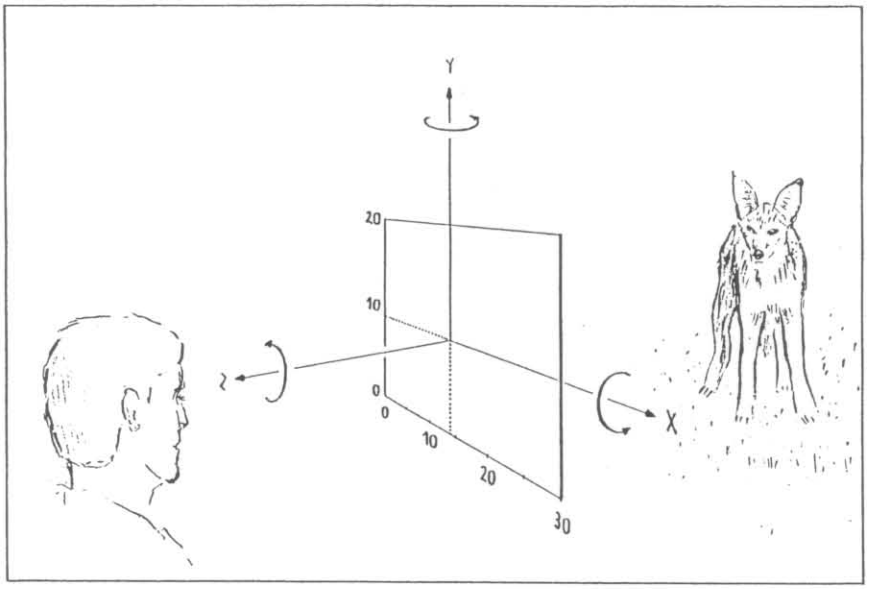
SCHAKAL 99

A computer program for the graphic representation
of molecular and crystallographic models

Tutorial manual and figures



SCHAKAL 99

A computer program for the graphic representation
of molecular and crystallographic models.

by *Egbert Keller*
Kristallographisches Institut der Universität
Hebelstr. 25 / D-79104 Freiburg i.Br. / Germany

Tel.: +49-761-203-6438
Fax: +49-761-203-6434
e-mail: Egbert.Keller@krist.uni-freiburg.de
WWW: http://www.krist.uni-freiburg.de/~kell

SCHAKAL [read 'shakahl'] is the German word for "jackal";
it is also an abbreviation for the two words SCHAttierte KALotten
[= shaded cups; shaded space-filling models].

Table of Contents

Introduction

SCHAKAL 99 is a computer program for the graphic representation of molecular and crystallographic models. A first version of this program has been developed in 1979 on a Sperry-UNIVAC computer. The program is written in FORTRAN 77. To check for differences between SCHAKAL 99 and earlier versions, see on-line manual (cf. chapter 1.2., below), table 98.

0.1. The different SCHAKAL 99 versions (Installation Hints)

a) PC versions: there are two different SCHAKAL 99 versions for PC which both have been compiled/linked with the FTN77 compiler system of Salford software:

SCHAKAL 99 / DOS	MS-DOS version (requires Salford DBOS to be installed)
SCHAKAL 99 / W32	MS-Win 95 / 98 / NT / 2000 version

The DOS version draws about 10 times faster than the Win versions; but it will probably not run on a computer designed to run under Windows 9x/NT/2000; and if, it will not have access to the general Windows printing mechanism. It is left in the state of September 2001, i.e., it is not supported any more. Nevertheless it is still part of the distribution in case there are still some DOS computers running.

After unpacking, the SCHAKAL 99 **installation program** *instw32.exe* installs the Win version and (optionally) the DOS/DBOS version. The latter requires Salford Software's 32 bit operating system DBOS (supplied on an extra diskette) to be installed separately on the computer.

On a pure DOS computer, use *instdos.exe* instead of *instw32.exe*

DOS please check (and possibly modify) the files *editt.bat* and *cppy#.bat* .

Win If a HP LaserJet is connected to your computer, please read file *readme.pc* .
Then use the Windows file explorer to link the icon *sch99.ico* either to the file *sw32.bat* or to the file *sch99w32.exe* . At a later date you might check/modify the file *sch99win.ini* .

For more information, see the file *readme.pc* .

b) UNIX/X-Windows version:

To **install** SCHAKAL 99 / Xwin after unpacking into a directory called „s99maindir", here, first inspect the files *readme.xw* and *makefile.x* to learn for possibly necessary modifications. Then ...

```
chmod +x [./]mschak
[./]mschak   (or: sh [./]mschak)
```

mschak sets up the different directories, moves the corresponding files and tries to compile/link the program. The program can be started by typing

s

Note that the program will only react to user action, if the graphics window is completely visible on the screen.

In a multi-user environment, any user should install his/her own SCHAKAL 99 user directory. From his/her home directory he/she should give the following command and then specify (on request) a hitherto non-existing directory as his/her SCHAKAL 99 user directory):

s99maindir/uschak

Any user should check (and possibly modify) the files *editt.sh* and *cppy#.sh* in his/her SCHAKAL99 user directory. After installation, or at a later date, also default path names for the different file types might be modified in the file *sch99.ini* .

For more information, see the file *readme.xw* .

0.2. Command Language

With respect to user interaction SCHAKAL differs from most other contemporary software: SCHAKAL is mainly controlled by input of "commands" each of which causes a specific action of the program or a setting of a certain switch or parameter. There are three different ways to input a command into the program:

1. by clicking one or more corresponding control field(s) (CF(s)) of the graphical user interface (GUI) with the mouse. The GUI is optional (i.e., it may be switched off); it is **not a standard MS-Windows or Motif GUI** (see fig. 25) which makes its use comparatively difficult.

2. by loading a "command file" which contains a number of SCHAKAL commands. A command file is loaded by means of the 'Use Commandfile' command. Predefined (= "Distributed") command files (DCF) are available from the DCF-directories in the 2nd & 3rd row of the GUI.

3. by typing the command on the keyboard, terminated by a <RETURN>.

Almost all SCHAKAL commands obey the following general syntax:

LL1 [LL2 [<sep> [p_1 [p_2 [p_3 [AC_1 .. AC_n [BC_1 ... BC_n]]]]]]] with

LL1	1st label word (usually a verbum), may generally be abbreviated to 1 letter
LL2	2nd label word (usually a substantive), may generally be abbreviated to 1 letter
<sep>	a separator character which gives the command a special meaning, e.g. "!" or "*"
p	a parameter value
AC	an atom code representing one or more atoms of the model
BC	a bond code which is two atom codes connected by the character "=" : AC_1=AC_2

An atom/bond code may be preceded by the "negative selection" operator "(" if it is not the first atom/bond code in a command.

All items except the first label word LL1 are optional. In general, a certain LL1 is specific for a certain class of commands. Table 1 contains most of the different 1st label words and the names of the control fields (CF) in the 1st row of the GUI by which the corresponding commands can be mainly accessed .

(Note: there is another type of "commands", which may be used to control the on-line menu. These commands consist of a number or of a special single character (see section 1.2.))

Table 1: 1st row CFs, 1st label words and thier meaning

No.	CF name	1st label word(s)	general meaning
1	USE	*USE*	use (read or write) a file
2	EDT	*EDIT* *WRITE*	edit (modify) a file write graphical text
3	WRT	*WRITE*	label atoms by names, coordinates, etc.
4	XQT	*XQT*	draw something (except text)
5	ROT	*ROT* *ALIGN* *ORIENT*	rotate the model or a part of it align the model or a part of it rotate the model on-screen
6	LST	*LIST*	give (list) some information
7	GEN	*GENERATE*	operate a main switch
8	G/S	*GENERATE / SET*	more main switches / more general drawing parameters
9	SET	*SET*	set general drawing parameters
10	MGN	*MAGNIFY* *MODIFY* *BROADEN*	set size parameters set darkness, contrast, shadow intensity set line widths
11	CHG	*CHANGE*	modify atom/bond specific parameters or flags
12	DEF	*DEFINE*	define things by atoms
13	FLT	*FILTER*	Filter atom/bond codes by certain criteria
14	ADD	*ADD* *KILL* *TRANSFORM*	add an atom or bonds remove atoms/bonds modify connection scheme
15	TRL	*TRANSLATE* *INVERT*	shift the model or a part of it invert the model or a part of it
16	MSC	*KILL* *PUSH* *ZET*	erase the screen or parts of a drawing erase text window or make it transparent operating system and GUI commands
	none	*Y* *HELP*	special command file commands get help

Examples (note that characters printed with small letters can be omitted if the commands are typed at the keyboard):

GENR Cups	(switch for space-filling [="cup"] models)
SET Light 35 45	(position the light source)
CHGE Colour 3 Cl H=C	(use colour no. 3 [=green] for Cl atoms and bonds betw. H and C)
LIST Atoms ?	(give information on atoms to be picked by mouse)
DEFN Group 7 Pb1 S3 S4	(define the numerical group 7 by the specified atoms)
DEFN $ gg7 #14	(define the special group named "$" by the atoms of the numerical group 7 and atom no. 14 in the atom list)
XQT Xqt tt ($ $=C	(draw all atoms and bonds except the atoms which have been assigned to group "$" and the bonds between these atoms and C)

0.3. Preface to tutorial sessions

Chapters 1. to 5. of this manual describe five tutorial program sessions. Performing these sessions is supposed to lead to a basic knowledge of how to work with SCHAKAL 99. Users who are interested in more details are encouraged to read SCHAKAL's hierarchical on-line manual (see 1.2.). Numbers given after an arrow (−−>) generally refer to the corresponding items within this on-line manual.

Throughout this tutorial, it is assumed, that SCHAKAL 99 has been installed correctly (see section 0.1. and files *readme.xxx*) and that the **original** system files *sch99.ini* and *sch99.mn0* and the **original** distributed command files **.scf* are accessible by the program under their **original** names. The distributed data sets *ex*.dat* will be used as demonstration examples.

Below, input which should be given by the user, is usually typed left-justified and enclosed within two blank lines. There are two cases:

a) Input generated via the mouse ...

... is printed as follows: the name of the control fields (CFs) which have to be clicked to produce a certain action are given sequentially. Each CF name is enclosed in {curly brackets}, one of two or three (differently coloured) parts of the CF may be specified as L (= left), C (center), or R (= right); two indices specify the position of the CF within the GUI. An explanation may follow in parentheses. Example:

{ other^{L} }$_{2,9}$ (light green part !!)

means to click the left (= green) part of the CF labelled "other" which is in row 2 and "main" column 9 of the graphical user interface (GUI). If a part of text output produced by the program should be clicked by the mouse (instead of a CF), the index is "Text":

{ 2 Draw Someth }$_{Text}$

b) Input typed at the keyboard ...

... is printed left-justified without curly brackets:

```
ex2
```

or within the text enclosed in 'single quotation marks'. A typed line has to be terminated by a <RETURN> [or by clicking { [OK]}$_{3,3}$. Such a <RETURN>, however, will only be accounted or in the following text, if it is the only input of a certain line (i.e., for "plain" <RETURN>s). As has already been done here, special keys of the keyboard are generally specified by their names, given in capital letters and surrounded by two brackets, e.g. <ESC>, <LEFT ARROW>, etc. If two keys appear side-by-side, they will have to be pressed concurrently (e.g. <SHIFT> <UP ARROW>).

User's input should be carefully given as specified, because there is a certain probability, that a faulty input will unpredictably influence parts of the residual session (however, a faulty command can n o t cause any "damage"). Mistakes in typed input may be corrected using the <BACK-SPACE> key or the CF { <−− }$_{4,16}$, before the line in question is sent to the program by a <RETURN>.

Session 1: First steps

1.1. Starting

To run SCHAKAL 99, click at the corresponding icon or type

```
s
```

First, the program displays a title page (*MS-Win*: you may click { Window Size} / { Help } to learn about some features which are specific for the MS-Win version). Then, it requests for a

<RETURN>

This will lead to the automatic input of the initializing command file *sch99.ini* . *sch99.ini* contains a number of regular SCHAKAL commands which are now executed one after the other. The major part of *sch99.ini* consists of 'Use... -n' commands which define prefices (= "path names") and suffices (= "extensions") to be added to file names later on. At a later time, you might edit one of the other of these commands in *sch99.ini* (probably mainly the 'U X -n' commands).

The last lines of *sch99.ini* contain some 'Zet Menu ..' commands, the last of which (a plain 'Z M') having caused the **graphical user interface (GUI)** to be displayed (the other ones are responsible for some "command-line control fields" [see below]).

The GUI (fig. 25) is of the "home-made" kind, i.e. it conforms neither to the MS-Windows nor to the OSF-Motif standards (which means that getting familiar with it will take its time). It can be seen as a matrix of control fields (CF) with 4 rows and 16 "main" columns plus one column of CFs plus the dark green vertical bar at the lower left, all CFs bearing more or less incomprehensible text. Actually, the GUI is nothing else but a "machine" to produce SCHAKAL commands via mouse clicks.

The main parts of the GUI are: the **1st row** to select groups of commands, the **command column** in the lower left part displaying such a group, then, in the 2nd and 3rd rows: the **ten differently coloured CFs** for frequently used actions (left), the **Distributed Command File (DCF) directories** (center), the **atom code array** (right; empty per default), and the two large yellow **OK buttons**. Finally, the **4th row** for access of the on-line manual (among others) and the dark green vertical bar near the command column, bearing **(user-defined) command-line CFs**.

In the upper left corner of the free area not covered by the GUI (the "**drawing area**"), SCHAKAL's command input prompt ">>>" appears (as part of the frameless **text** "**window**"), indicating, that SCHAKAL is ready for input of a command. If you once more type

<RETURN>

another prompt will appear. The same is valid if you click (left mouse button) the large yellow CF

{ [OK] }$_{3,3}$

which simply emulates the typing of <RETURN> (note that the index 3,3 specifies the position of the button in the 3rd row, 3rd (of 16) "main" column of the GUI). After two or more additional

{ [OK] }$_{3,3}$

text will be erased and the next prompt will be displayed in "line 1", again.

1.2. Help Functions and the Interactive Manual

You can get a brief explanation on most of the control fields (CFs) of the GUI by clicking (an individually coloured part of) **the CF with the right mouse button.** Try this by clicking

{ [OK] }$_{3,3}$ [**right** mouse button]

More detailed information is available from SCHAKAL's hierarchical on-line manual which consists of "Tables" and "Infos". If you type '0' or if you click

{ 0 }$_{4,4}$

"Table 0" of the manual is displayed on the screen; the 4th row of the GUI now contains some additional CFs with single characters which may be used to control the manual (see below).

DOS If you are working with SCHAKAL 99 / DOS, text will be displayed in a window which may be considerably smaller than the screen. If you think that characters are too small you may terminate with 'q' and restart SCHAKAL with a lower resolution, e.g. by using 's 2' instead of a plain 's' (see file *readme.pc*).
The text window would have a larger size, then, but drawings would be of less quality, of course.

"Table 0" is something like a "table of contents" of this manual. The figures and terms displayed in the first two columns of the table refer to the 9 chapters of the manual. Now, type '2' or click any character of the following Text:

{ 2 Draw Someth }$_{Text}$

to get a display of "Table 2" which is a "table of contents" of chapter 2. The number 5 in the first column refers to the section "Draw Lines". Type '5' or click any character of

{ 5 Draw Lines }$_{Text}$

to get a display of Table 25, which lists a group of commands controlling the drawings of lines, arrows, etc. One of the commands is 'X F' (no. 6). Type '6' or click any character of

{ 6 Frame I }$_{Text}$

to receive information on this command (which you don't have to read by now). Note, that the command is assigned to the number 256 (which means "6th item of table 25"). Type '+' or click

{ + }$_{4,6}$

to move to the "next-upper-level" table; i.e., "Table 25" is re-displayed. Now, type '-' or click

{ - }$_{4,5}$

["table/info toggle switch"] to have "Info -25" (information associated to "Table 25") displayed. The last line contains the string "(–-> 545)" which is a hint to another item of the manual. Click

{ --> 545)$_{Text}$

to have the corresponding information displayed. The last line of the page contains the string "(–-> 471)". Type '471' or click

{ --> 471)$_{Text}$

to have the corresponding information displayed. The last line says " continue ? >>>", which means that the information consists of more than one page. The next page is accessed by typing <RETURN> or by clicking

{ [OK] }

There is again a " continue ? >>>" at the bottom; the next (and last) page can be accessed as described before:

<RETURN>

To have information no. 471 re-displayed from the beginning, type '=' or click

{ = }$_{4,6}$

With "<" and ">" you could go backwards and forwards witin the series of "pages" which already have been displayed. "99" leads to the index of the manual. The item numbers given there may also
be clicked with the mouse.

Finally, let's try to get information on a certain command directly, for example, the 'Xqt Test' command. In the following, ^{C} means "central part" of the CF:

{ XQT^{C} }$_{1,4}$

then click, with the **right** mouse button the **left** part (^{L}) of the following CF in the "command column":

{ Test^{L} }$_{16,1}$ [**right** mouse button]

As expected, a brief explanation is given. Now click the CF a second time.

{ Test^{L} }$_{16,1}$ [**right** mouse button]

This generates the command 'H X T' which is an abbreviation for 'Help Xqt Test' (note that 'Help..' commands are the o n l y commands with **three** label words !).

First, the upper-level table with respect to the 'X T' command is displayed (containing a brief explanation of 'X T'); by typing '<RETURN>' or by clicking

{ [OK] }$_{3,3}$

you continue to detailed information on the command. - Once you have read the information, click the right (yellow) part (^{R}) of the CF labelled 'Test' with the **left** mouse button.

{ Test^{R} }$_{16,2}$

The screen is erased and SCHAKAL generates a display of its current [= default] screen colour map consisting of 19 "colour steps" per 13 "main colours" plus white and black (the latter presently not visible due to the black background colour). This result can be reproduced by typing

```
X T
```

1.3. Text window

Now, click another zero:

{ 0 }4,4

The text "window" will now cover a part of the test drawing. It differs from other text windows you are probably used to: First: it is frameless; second: it's size changes dynamically depending on the amount of text displayed; third: it is frequently erased by the program automatically and re-displayed with new text; fourth: displayed text cannot be scrolled.
However, the text window can to some extent be controlled by 'Push' commands. Click

{ P }2,2

This generates a plain 'Push' command which makes the text disappear. Note, that erased text can- not be regenerated except by repeating the command(s) which produced it (but see –-> 743) .

Per default, the text window is opaque. With 'Push Pellucid', it can be set to become pellucid (= transparent). This command is generated by clicking:

{ t }3,2

Now activate Table 0 for another time:

{ 0 }4,4

The text window is now transparent. Visibility of graphics is improved, but readability of text is clearly worse than before (on backgrounds other than the 'Xqt Test' drawing it's usually better). With another

{ t }3,2

you switch back to the opaque form of the text window. Check this by clicking another

{ 0 }4,4

Table 2: File types (see section 1.4.)

File type	Commands	action
Input file	'Use I', 'Add I', 'Edit I'	load or edit a structure data file
Xtaldata file	'Use X', 'Edit X'	load or edit a crystallographic data file to be translated into an Input file
Output file	'Use O'	write a structure data file
Command file	'Use C', 'Edit C'	execute or edit a file containing SCHAKAL commands
Echo file	'Use E'	open a file to record SCHAKAL commands
Printer file	'Use P'	open a file to store graphics to be printed
Writing file	'Use W', 'Edit W'	write or edit a file with text
Save file	'Use S'	save the screen image to a file
Restore file	'Use R'	restore a screen image from a file

1.4. Handling files / red and green stars

Next, you will have to load a data set describing the geometry of a molecular or crystallographic model into the program. Such data sets are usually stored in files, which are called "Input files", here. (With SCHAKAL, files are classified into different file types: "Input files", "Save files", etc.,see table 2). Loading of an Input file is accomplished with the 'Use Inputfile' command:

{ USE^{C} }$_{1,1}$

{ Inputf^{R} }$_{6,2}$

SCHAKAL requests for the name of the Input file containing the data set. **You may generally cancel input of a file name by typing "n", or by clicking**

{ n }$_{4,3}$

Now let's try it once more:

{ Inputf^{R} }$_{6,2}$

This time, type a real file name:

`ex1`

The data file *somepath//ex1.dat* (an example structure file) is now loaded. Some information on the data set is output by the program. Note, that the appropriate path (called *somepath* , here) and extension (*.dat*) have been added automatically to your file name (*ex1*) by the program. *somepath* and *.dat* are specified in a 'Use Inputf -1' command in the file *sch99.ini* .
"ex1" is called the file's **name core**.

Note: In a case where you don' like to have the program add the predefined prefix and suffix to a typed file name , just let the typed program name **start with a blank space**.

To see what we have loaded, click

{ XQT^{L} }$_{1,4}$ (red star; executes the command 'XQT Quick')

More about drawing is given in the next section. Presently, let's execute the command to load a data file once more, this time clicking the "*" (which appears in the 4th row of the GUI) as a file name:

{ Inputf^{R} }$_{6,2}$

{ * }$_{4,4}$

A list of file names is output. You will see, that *ex1* is one of the names appearing in the list. The program again asks you to specifiy a file name. To load the data file once more, click

{ `ex1` }$_{Text}$

MS-Win: Instead of a plain file name list, the usual Windows box for file name handling is generated. The file name is given as ex1.dat, here. The Windows box is handled as usual; select the file ex1.dat

Let's practice this once more:

{ Inputf^{R} }$_{6,2}$

As an answer to the file name request, type

`ex`

and add a '*' or click

{ * }$_{4,4}$

This time the list of file names is restricted to those, the name cores of which start with "ex" (it may be identical to the previous list, if all file names in the Input file directory start with "ex"). Load the file once more by selecting it from the displayed list or click { n }$_{4,4}$ to cancel.

Finally, let's try the 'Use Inputfile' command once more, using an "**empty**" filename:

{ P }$_{2,2}$ (remove the text window)

{ Inputf^{R} }$_{6,2}$

{ [OK] }$_{3,3}$ (this generates an empty filename)

The identical text output shows that the same file (*ex1*) has been loaded again. An "empty" file name generally means: re-use the file of the desired type which has been specified last by the user.

Note: **The rules described above apply to all file types used by SCHAKAL.**

Actually, there is an even faster way to re-load the same Input file: Remove the text window and click the un-labelled yellow CF on top of the command column:

{ P }$_{2,2}$

{ }$_{5,2}$

This generates a plain 'Use' command (i.e., a command which consist only of its first label word). **A plain 'Use' command causes the last file of type Input file, Command file, or Writing file (whichever has been specified last explicitely by the user) to be re-used.**

In this case it's the Input file *ex1.dat* which has been specified last and which is therefore now re-used, i.e., re-loaded. Now, erase the text window by clicking

{ P }$_{2,2}$

You will have noticed that this un-labelled CF contains a red star. This red star appears also in the 1st row's CF labelled 'Use' (i.e., the CF which gives access to the current command column). Which means that the command generated by the un-labelled CF is also generated by the red star in the 1st row (regardless of whether the corresponding command column is displayed or not):

{ USE^{L} }$_{1,1}$ (red star; executes the command 'Use')

The file *ex1.dat* is reloaded once more.
Note: The use of "=" as a file name is described in sections 1.7, 3.4, and 5.1.

1.5. Some basic commands

Now, to obtain another coarse, but fast, drawing of the EX1 molecule, we use the 'Xqt Quick' command, explicitely (note: "Xqt" is an abbreviation for "Execute"):

{ XQT^{C} }$_{1,4}$

{ Quick^{R} }$_{7,2}$

Another fast drawing of the ball-and-stick model (the default model type) appears. The CPU time needed to calculate the drawing is printed; such information will often be output when an action has been completed . As the CF labelled "Quick" contains the red star, it was this 'Xqt Quick' command which was performed when you clicked { XQT^{L} }$_{1,4}$ (red star) in section 1.4.

You will have noted, that both times before the new picture was drawn, the screen had been erased. This is because "automatic screen erasing" is switched on (per default). You may use the 'Kill Screen' command anytime to erase the screen additionally "by hand":

{ K }$_{2,2}$

Now, we want to make two drawings of some higher quality. With a **plain** 'Xqt', you may generate an outline drawing of the molecule ("outlines" is the default display style for plain 'X' drawings):

{ XQT^{R} }$_{1,4}$ (green star; executes the command 'XQT')

To generate a shaded drawing, you have to use the 'Gen Shading' command (which operates a so-called "main switch", namely a switch from "outline drawing" to "shaded drawing").

{ GEN^{L} }$_{1,7}$ (red star; executes the command 'Genr Shading')

{ XQT^{R} }$_{1,4}$ (green star; executes the command 'XQT')

Before drawing starts, a red control field labelled "<ESC>" appears which should not be confused with the control field labelled "ESC" (see below).
Whenever "<ESC>" appears, you may cancel the action which is performed presently by pressing the <ESC> key of the keyboard (but not by clicking this CF !!).

{ XQT^{R} }$_{1,4}$ (green star)

<ESC>

Drawing stops immediately or after a little while. Now let's have the drawing complete:

{ XQT^{R} }$_{1,4}$ (green star)

Per default, the imaginary light source, which affects the shading of balls and sticks is positioned at the positive Z axis (pointing towards you). You may change this with the help of the 'SET Light a1 a2.' command which rotates the light source a1 degreees about the Yaxis (pointing upwards) and then a2 degrees about the Z axis.

{ SET^{C} }$_{1,9}$

{ Light^{L} }$_{11,1}$ (light blue part !!!)

Within the GUI, red and yellow colour (of parts of CFs or of letters) usually mean that control is immediately passed from the GUI to the program (i.e, some action is performed, e.g. setting of a switch)**. Clicking light green and light blue parts of CFs leaves control within the GUI.**

With respect to the left command column, a large (small) yellow part means that there is a high (low) probability that the command is used for "immediate action".

Two parameter boxes appeared in the 2nd and 3rd row of the GUI. The two desired parameters (say, 55 degrees and 25 degrees) may be input to the program now in two different ways:

a) either by typing '55 25', followed by '<RETURN>' or { [OK] } or ...
b) by modifying the parameters in the parameter box: Click on the red and grey parts of the increase/decrease control fields within the parameter boxes to learn how to adjust the parameter values properly. Once you have set the right values, click

{ [OK] }

{ XQT^{R} }$_{1,4}$ (green star; executes the command 'XQT')

The two parameter boxes are still open. You may now modify the 2nd parameter by clicking the red part of its "increase" button until it displays a value of 95 and execute the corresponding 'Set Light ..' command by clicking { [OK] }. A

{ XQT^{R} }$_{1,4}$ (green star)

will show the effect. Note that the 'S L' command remains "latently activated" on the GUI as long as its parameter boxes are displayed in the 2nd and 3rd row. You may desactivate it by clicking

{ }$_{2,2}$ or { }$_{3,2}$

These two unlabelled CFs will generally clear the GUI (i.e. reset its default state) **and desactivate a latently active command** (if any).

Note: Within session 2 you will learn a better way to make shaded drawings.

The current settings of most "switches" and general drawing parameters may anytime be checked by means of the 'List Params' command:

{ LST^{L} }$_{1,6}$ (red star; executes the command 'List Params')

{ [OK] } (continue to next page)

Switches and parameters are given in the context of the commands which control them. You will see, that the parameter values 55 and 95 appear within the 'SET Light' command. A more compressed (1-page) output is obtained with the 'List Quick' command:

{ LST^{C} }$_{1,6}$

{ $Quick^{R}$ }$_{6,2}$

The EX1 molecule is an organo-iron cluster molecule. There are several ways to verify, that the molecule actually contains iron atoms as well as C and H atoms:

First, you might have a look at the data set *ex1.dat* itself. This is possible by means of an 'Edit...' command which makes a local or built-in editor open a file. A plain 'Edit' opens of all Input-, Command-, or Writing files the one which has been specified last by the user (cf. plain 'Use' command, page 11). In this case, the Input file *ex1.dat* is the one and only. Click

{ EDT^{L} }$_{1,2}$ (red star; executes the command 'Edit')

to generate a plain 'Edit' command which lets the editor open *ex1.dat* (note: you also might have used an 'Edit Inputfile' command and specify an "empty" file name, cf. page 11).

You will see, that *ex1.dat* contains a number of labelled lines. These are called by the old-fashioned term "cards" (to help distinguish this type of program input from the other type ("commands")). Each card starts with one label word which generally could have been abbreviated to the first two letters.

The file contains a title (TItl card), information on the cell parameters to which atom coordinates refer (CEll card; no parameter means: cubic cell with a = 1 A), the names and coordinates of the molecule's atoms (ATom cards), and an ENd card. If you have inspected the file, cancel the editor session with the appropriate button, command, or key(s).

Second, you may use the 'List Model' command to get information on the number of atoms of the different chemical types (lower half of the text window):

{ LST^{R} }$_{1,6}$ (green star; executes the command 'List Model')

Third, you may use the 'Write Label' command, which will make the program label the different atoms with their "names":

{ XQT^{L} }$_{1,4}$ (red star; executes the command 'XQT Quick')

{ WRT^{L} }$_{1,3}$ (red star; executes the command 'Write #')

The message "LPS was cleared" appears in the 4th row of the GUI. An optimization procedure is performed which determines more or less reasonable positions for the different labels; the relatively fast default optimization level 12 is used. The procedure is finished with the message: " 41 optim., 0 from LPS". We will come back to labelling in session 2.

Fourth: The molecule has been displayed with default colours: white (colour no. 0) for H, grey (1) for C, red (2) for O, green (3) for N, blue (4) for Fe, and brown (6) for the bond sticks. With the 'List Atoms' command you can check for most of these assignments:

{ LST^{C} }$_{1,6}$

{ Atoms^{R} }$_{12,2}$ (don't click { Atoms ? }$_{14,2}$!!; if you already did, click { ESC }$_{4,3}$)

Information on the molecule's atoms is listed. The second last column (labelled "C") contains the colour numbers. Other columns except atom names are not of interest, presently.

1.6. Atom codes and bond codes

Many SCHAKAL commands allow so-called atom codes (AC) or bond codes (BC) to be specified. To get an idea about the different possibilities to specify atoms/bonds, you should have a look at chapter –—> 92 of the interactive manual, first (you don't have to read every detail, by now). Type '92' (into the line displaying ">>>" or "continue ? >>>"):

```
92
```

The 'List Atoms' command may be used to practice the use of ACs/BCs a little. Click

{ Atoms^{L} }$_{12,1}$ (light blue part)

The CFs in the right part of rows 2 and 3 of the GUI turn light blue and display some (parts of) atom codes now. You may inspect the meaning of the different fields by clicking them with the right mouse button. Then click (with the left mouse button)

{ Fe }$_{2,14}$

{ [OK] }$_{3,16}$

Information on the four Fe atoms of the structure is given. The message "4A" appears in the 4th row of the GUI (at the right) to state that 4 atoms of the model matched the atom code "Fe". Now, try some more:

{ Fe } { 4 }$_{4,12}$ { [OK] } (list atom Fe4 only)

{ # }$_{2,12}$ { (} { H } { C } { [OK] } (list all atoms except H and C atoms)

Again the 'L A' command is "latently activated" as long as the atom codes are displayed in the 2nd and 3rd rows' CFs. Now let's do a similar thing with the 'List Bonds' command. Groups of atom/bond codes to be input are from now on given here within only a single pair of curly brackets. They can be generated via the mouse (as above) or typed character by character.

{ Bonds^{L} }$_{13,1}$

{ Fe=Fe H= # } { [OK] } (list all Fe-Fe bonds and all bonds to H)

{ Fe1= # (Fe=Fe } { [OK] } (list all bonds to Fe1 except any Fe-Fe bonds)

The number of bonds which matched the specified bond codes is given each time as "nB" in the 4th row of the GUI. Now let's redraw the molecule and go back to the 'List Atoms' command:

{ XQT^{L} }$_{1,4}$

{ Atoms^{L} }$_{12,1}$

{ ? }$_{2,11}$ { [OK] }

"?" used as an atom code means that atoms are to be picked by means of the mouse. While the mouse cursor was serving the GUI up to now, it is now temporarily serving the program's "working routines". This mode is called "graphics cursor mode", here.

Whenever "graphics cursor mode" is on, a CF labelled "ESC" (without <brackets> !!) appears in the 4th row of the GUI [*UNIX: additionally, the mouse cursor turns into a crosshair cursor*]. **Clicking this CF** (or typing <ESC>) **will terminate this mode**.

Now, click any atom with the left mouse button. The selected atom's name and a number indicating its position in the program's atom list ("atom list number") will be printed. Go to some other atoms and repeat the procedure. If you click an atom for the second time, the selection of this atom will be cancelled (indicated by a negative atom list number).

Whenever "graphics cursor mode" is on (and only then)**, the cursor may also be moved by the keyboard's arrow keys. Atoms can in this case also be selected/deselected with <SPACE BAR>. You may use <SHIFT> <UP ARROW> etc. to move the cursor for only one pixel.** [*UNIX:* This may not work correctly under certain UNIX systems, see file *readme.xw*]

To terminate "graphics cursor mode" now, type <ESC> or click

{ ESC }$_{4,3}$

The atoms you have picked will be listed. Note that generally, when atoms or bonds are picked via the atom code "?" (or the bond code "?=?"), the desired action is performed to all the picked atoms (bonds) only at the end of the picking procedure (i.e., when graphics cursor mode is left via "ESC". An exception to this is the 'List Geometry ?' command:

{ Geom ? }$_{18,2}$

When you now pick a number of atoms with the cursor, additionally distances (to the last atom), angles (to the last two atoms), and dihedral angles (to the last three atoms) are printed instantly. Click an atom with the right mouse button (or press <m> instead of <SPACE BAR>) to start a new series. When you leave cursor mode, no additional action will take place, afterwards. Click:

{ ESC }

To make the structure of the iron complex more comprehensible, we will now change the display colour for some of the bonds by means of some 'Change Colour' commands ('Change..' commands are generally used to modify atom/bond specific parameters). SCHAKAL's 13 "main colours" are addressed by their "colour numbers" (for more information, see session 2). With

{ CHG^{C} }$_{1,11}$

{ Colour^{L} }$_{5,1}$

a colour box (instead of a parameter box) and again the field of atom codes appear in the 2nd and 3rd row of the GUI. In the case of 'Change...' commands, the order with which things are clicked is of significance: Let's first click "Fe" *first*:

{ Fe }

you will see that the number 4 is displayed in the colour box to indicate that colour no. 4 [blue] is presently assigned to the Fe atoms (note that the colour no. of white is 0). Now click

{ = Fe }

the number is changed into 6, as colour no. 6 [brown] is assigned to Fe-Fe bonds presently.

By clicking one of the colour buttons we could change the colour of the Fe-Fe bonds. We don't do this right now, but try the other order of clicking things. To do this, we first should re-generate the 'C C' command:

{ Colour^{L} }$_{5,1}$

Now we *first* click a coloured button within the colour box. In this case the colour no. doesn't change dynamically with the clicking of atom/bond codes:

{ *yellow* } { Fe=Fe } { [OK]} (colour no. 5 [yellow] for Fe-Fe bonds)
{ *violet* } { C=C (C1=C) { [OK] } (colour no. 7 [violet] for C-C bonds except C1-C)
{ *cyan* } { C=O } { [OK] } (colour no. 9 [cyan] for C-O bonds)

Let's now check the results:

{ XQT^{L} }$_{1,4}$ (red star; executes the command 'XQT Quick'))

Note, that it is always recommendable to enhance ring systems by giving the corresponding bonds a different colour (see also fig.s 11 and 12).

Finally, we should try another mode of addressing atoms/bonds. First we select a group of atoms/bonds by drawing a polygon around them. This is done with the 'Xqt Line' command:

{ XQT^{C} }$_{1,4}$
(note: these two could [probably] be replaced by { XL }$_{10,3}$)
{ Line^{R} }$_{18,2}$

"Graphics cursor mode" is switched on (see above). By clicking the left mouse button at different positions you may now draw a grey polygon around, say, the phenyl ring or some other group of atoms. The polygon doesn't have to be a closed one. Then click

{ ESC }$_{4,3}$

Atoms within the polygon may now be addressed via the atom code "rg" (for "region"); if the polygon is not closed, the program will close it virtually by connecting its first and last corners. Now, to produce an 'X X..' command, click

{ XQT }$_{6,1}$

As an 'X L' polygon is on the screen, the atom/bond codes CF array (2nd ad 3rd row of GUI) now contains the atom code 'rg'. We now input an 'Xqt Xqt rg rg=rg' command:

{ rg rg=rg } { [OK] }

The atoms/bonds within the polygon are now shaded.

Note: the atom code 'rg' should not be mixed with other atom/bond codes within one command.

Note: the atom code 'rg' is only valid as long as the corresponding polygon is drawn on the screen. It is therefore recommended to use it preferably within the 'Define $' command (see section 2.1.).

Note: atom/bond codes may be filtered by several filters (–> 68).

1.7. Rotations and on-screen rotations

The default orientation of the EX1 molecule is not a very attractive one. You may change it by means of 'Rotate..', 'Align..' or 'Orient' commands. For example, you may align the molecule in a way, that the best plane defined by the four iron atoms is oriented parallel to the drawing plane. This is done by means of the 'Define Plane' and 'Align Plane' commands:

{ DEF^{C} }$_{1,12}$

{ Plane^{L} }$_{21,1}$ { Fe } { [OK] }

{ ROT^{R} }$_{1,5}$ (green star; executes the command 'Align Plane'))

{ XQT^{L} }$_{1,4}$ (red star; executes the command 'XQT Quick')

To make better use of the drawing area, you finally may rotate the molecule 90 degrees about the Z axis (see fig.1) and then 90 degrees about the X axis (the fastest way is to type the two follow- ing commands at the keyboard instead of clicking around on the GUI):

```
R Z  90
R X -90
```

{ XQT^{L} }$_{1,4}$ (red star)

As you will have already noticed, rotations caused by 'Rotate' and 'Align' commands are not performed on-screen. However, with the 'Orient' command, you may switch to "on-screen rotations mode" (OCR mode):

{ ROT^{L} }$_{1,5}$ (red star; executes the command 'Orient')

The 'Orient box' (fig. 25) with a number of CFs appears on the GUI. In the following, position indices of these CFs refer to the position within this box. The screen is erased and a wire model of the current structure model is displayed using the different default and modified **bond colours**. The wire model is drawn with a fixed depth-dependent darkness. Click

{ > }$_{2,4}$

for some time. As long as you press the mouse button, the wire model rotates about the Y axis (pointing upwards); it stops on releasing the button. The CFs labelled with other "arrow tips" work correspondingly; { + }$_{1,4}$ and { - }$_{3,2}$ rotate about the Z axis (pointing towards you). { s }$_{3,4}$ and { f }$_{1,2}$ make the rotation slower/faster (Note: the default speed of the rotation can be set with the 'Orient Defaults' command, which you may insert into the file *sch99.ini* to make it permanent).

{ R }$_{1,1}$ lets you switch between the modes R(otation), 90 degree F(lips), and T(ranslation) (the latter only, if the model scale factor is fixed !). { M }$_{2,1}$ is a toggle between modes M(ouse), A(rrowkeys) and S(ave) (note: the latter causes automatic saving of all images and requires a previously given 'O S' command; it should be tried out only by an experienced user !!). While in mode A, the CFs for rotation behave like the arrow keys of the keyboard's numerical keypad (NUMLOCK off !!), which also may be used to control on-screen rotations:

Press the <RIGHT ARROW> key once. The molecule begins to rotate about the Y-Axis) and continues to rotate once you release the key. Pressing the <LEFT ARROW> key once (or the <5> key) will stop the rotation. Try it again, but press the <RIGHT ARROW> key several times: rotation becomes increasingly faster. You can slow it down by pressing the <LEFT ARROW> key several times. The rotation stops finally; then the molecule starts to

rotate the other way around, again becoming increasingly faster. When the rotation is rather fast, stop it once more by the <5> key or by <UP ARROW>, immediatley followed by <DOWN ARROW>.
<SHIFT> < LEFT ARROW> and <SHIFT> < RIGHT ARROW> cause rotations about the Z-Axis and <SHIFT> <UP ARROW> and <SHIFT> < DOWN ARROW> increase or decrease the default rotation speed. [*UNIX:* this may not work correctly under certain UNIX systems, see file *readme.xw*]

Clicking { D }$_{4,1}$ will add some small "atom circles" to the wire model; clicking { D }$_{4,1}$ a 2nd time will remove the circles. { Q }$_{4,2}$ will switch to a rotatable "Xqt Quick" drawing, { L }$_{4,3}$ / { S }$_{4,4}$ will switch to rotatable outline / shaded drawings. { C }$_{4,5}$ will cause the last Command file expli- citely specified (or clicked) by the user to be re-executed. If it generates a drawing of the model, this drawing can be rotated as well. However, presently (end of 1999), mainly { D }, { Q } and { L } drawings will be of practical value due to limited computer speed.

For the meaning of any of the other CFs of the "Orient box" click it with the right mouse button for some information (see also fig. 25) and try to "learn by doing". To leave OCR mode now, click
{ ESC }$_{4,3}$

or press <ESC> . (Note: as long as the "Orient box" is up, none of the "external" CFs except { ESC }$_{4,3}$ will respond to clicking.) --- After leaving OSR mode, any orientation of interest (plus most current switch/parameter settings of the program) might be saved on a file by means of the 'Use Outputfile' command. This file may then anytime be re-loaded via 'Use Inputfile':

{ USE^{C} }$_{1,1}$

{ Outputf^{R} }$_{10,2}$

Specify a name , e.g. the rarely used ...

```
test
```

If the file already exists, answer the program's request with { [OK] }, to override the old file. The orientation/parameter settings are now stored on the file *somepath//test.dat*. Let us now change the orientation of the molecule and then reload the current orientation from *test.dat*:

```
R X 90
```

{ XQT^{L} }$_{1,4}$ (red star)

{ Inputf }$_{5,2}$

As a file name you may now use "=" which is displayed in the 4th row of the GUI. **"=" may be used to transfer a name core of the *last* used file of one certain type to a file of a certain other type** (–-> 0 -1.4). In the present case, the name core of the last Output file is transferred to the name core of an Input file:

{ =^{R} }$_{4,5}$ (yellow button)

{ XQT^{L} }$_{1,4}$ (red star)

This is the end of session 1. You may now continue with session 2 or terminate the program by typing 'q' or by clicking

{ - }$_{2,1}$

Session 2: Distributed Command Files

2.1. Using Distributed Command Files (DCFs)

If you terminated the program at the end of session 1, restart it now as described in section 1.1.
In **any** case, we will now (re-) load the *ex1.dat* Input file

{ USE^{C} }

{ Inputf^{R} }$_{6,2}$

`ex1`

{ XQT^{L} }$_{1,4}$ (red star; executes the command 'XQT Quick')

This will reproduce the drawing you got the first time within session 1. To make some defined ori- entation modifications additional to some parameter settings, you may use the "distributed Com- mand file" (DCF) *ex1.scf* . A Command file contains a number of regular SCHAKAL commands. It could be executed via the 'Use Commandfile' command. An easier way of access to a DCF is via the control field (CF) representing the appropriate **DCF directory**, in this case *other.scf* .

{ other^{L} }$_{2,9}$ (**light green** part)

The contents of the directory are displayed in a box within the left column of the GUI:

{ ex1 }$_{14,2}$

{ XQT^{L} }$_{1,4}$ (red star; executes the command 'XQT Quick')

This displays the modifications performed by *ex1.scf* : the molecule has been reoriented, per- spective has been switched on (via 'Set View..'), the bond stick thicknesses have been modified (via 'Chge Thick..') bond colours have been modified (the same way as you did in session 1), and one bond stick has been broken into four fragments (via 'Chge Fragmentation').

To draw the molecule another time, we use another DCF. In this case, we click the **yellow** part of the corresponding DCF directory, which means that the default DCF (*qs.scf*) of this directory is executed immediately:

{ quick^{R} }$_{3,4}$

The drawing is similar to an 'Xqt Quick' drawing, but shows "distance-dependent darkness" (depth cueing). Table 3 gives an overview of the different DCF directories (which, by the way, are DCFs themselves):

Table 3: DCF directories

lines	line drawings	**varyZ**	depth-dependent effects
quick	coarse but fast drawings; some demos	**cupBS**	space-filling + ball-and-stick model
pattn	different shadings for diff. chem. elements	**fourr**	"Fourier" backgrounds
shade	shaded drawings	**cryst**	crystallographic procedures
imptt	"important" part of model ($) enhanced	**resxn**	resolution enhancement
other	miscellaneous	**user1**	user's command files

DCFs which make shaded drawings are available from the DCF directory *shade.scf* .
Before you use one of these DCFs, you generally should position the program's imaginary light source via 'Set Light ..' (not necessary right now, as *ex1.scf* contains a 'S L 45 45' command). Furthermore, the background colour might be changed via a 'Set Backgr' command (see below):

{ SET^{C} }$_{1,9}$

{ Backgr^{L} }$_{8,1}$

Click one of the colours on the right side of the colour box in the 2nd row of the GUI, e.g. the colour with the number 12 (try until you got it); ignore the additional colours which appear in row 3. Once you have found colour no. 12, click { [OK] }:

{ *greyish green* } { [OK] }.

Now let's execute the default DCF *s.scf* of the DCF directory *shade.scf* to get the shaded drawing. First we clear the GUI; then we generate the command 'USE Commandfile "s"':

{ }$_{2,2}$

{ shade^{R} }$_{3,5}$ (**yellow** part)

Now click

{ shade^{L} }$_{3,5}$ (**light green** part)

You will get the list of DCFs belonging to the DCF directory *shade.scf*. Note that the names given in the box are, in most cases, not the names of the DCFs directly but their "pseudonyms". Before, when you clicked the yellow part of { shade }, the "default DCF" of this directory (with pseudonym "norml", flagged by "*") had been executed immediately.

If you click onto a pseudonym with the **right** mouse button you get a brief explanation of what the DCF does and its "real" name in [square brackets]. Click with the **right** mouse button

{ dots }$_{15,1}$ (**right** mouse button)

Generally you don't have to care for the "real" names of DCFs; if you ever want to load *sd.scf* explicitely via a 'Use Commfile' command, you may either use 'U C "sd" ' or 'U C "shade dots" ').

On top of the box with the DCF pseudonyms there are two CFs labelled "w" and "$". If you click one of these before you click the desired DCF, it will become light green, and some or all atoms of the model will be labelled. In case of { w }, all non-Hydrogen atoms will be labelled:

{ w }$_{7,1}$

{ metal }$_{10,2}$

The "$" (besides "w") referred to the "group $". This special group of atoms is defined by the user with the command 'Define $'

{ DEF^{C} }$_{1,12}$

{ $^L }$_{7,2}$
{ Fe N C1 } { [OK] }

The specified atoms may from now on be addressed by the atom codes '$' and 'dd' (the latter being a substitute for '$ $=$'). Try this with the following

{ K }$_{2,2}$

{ SET^L }$_{1,9}$ (red star; executes the command 'Set Background')

```
X X  dd
W L $
```

The DCFs in the DCF directory *imptt.scf* (and a few others) assume that you have defined the "$ group" prior to calling them. They make drawings with a selected "important" group of atoms (and the bonds between them) enhanced.

{ imptt^L }$_{2,6}$

{ size }$_{11,2}$

Now let's have a look at the DCF *ia.scf* which just has been executed. Click

{ EDT^L }$_{1,2}$ (red star)

This generates a plain 'EDIT' command which makes the editor open the most recently specified Input-, Command-, or Writing file, in this case *ia.scf* .

The DCF consists of a relatively large number of regular SCHAKAL commands and some comments (starting with #). Furthermore, It contains two "labelled partitions" at the end (look at the end of the file).
Labelled partitions are headed by a 'Y Label "strg"' command (where strg stands for a string of length 1 to 6 characters). The commands in a labelled partition are executed only, if strg has been added to the Command file name.
The labelled partition in Command file *ia.scf* are headed by a 'Y Label "w"' command and a 'Y Label "$" command. One of these labelled partitions is executed, when you click "w" or "$" in the DCF pseudo names box. (i.e., the program adds "w" or "$" to the command file name, separated by a blank space.

Let's leave the editor and try this once, explicitely, this time using the DCF *i.scf*:

{ USE^C }$_{1,1}$

{ Commandf }$_{13,2}$

```
i  $
```

It should be emphasized, that $, in this case, is not an atom code but a label of a DCF's labelled partition.

Besides the commands mentioned already, there are a number of other commands by which the appearance of (most) Command file drawings may be controlled. Some of them are briefly discussed in the following sections:

2.2. Distance-dependent effects / Model type

With SCHAKAL, three different graphical effects can be made "distance-dependent" (i.e., dependent on the depth resp. on the Cartesian Z coordinate of an atom/bond): line width, darkness and pellucidity (= transparency). DCFs which make use of these effects are collected in the DCF directory *varyZ.scf* . Its default partition (*v.scf*) uses variable pellucidity to the background:

{ varyZ^{R} }$_{3,6}$

Note: on a *white* background, generally use { varyZ^{L} } and { light }$_{10,2}$, instead.

You can add constant or distance-dependent darkness and/or pellucidity also to the DCFs collected in *shade.scf* and to some others . This is done with the commands 'Modify Darkness' and 'Modify Pellucidity' (which belong to the { MGN } group):

{ MGN^{C} }$_{1,10}$

{ DarkVar^{L} }$_{13,1}$

Set the second parameter to 6.0, and click { [OK] }, then click

{ SET^{L} }$_{1,9}$

{ }$_{2,2}$

{ shade^{R} }$_{3,5}$

to get a drawing with increasing darkness when going from the foreground to the black background. With the 3rd parameter of all commands for distance-dependent effects ('BROADEN All' is the one for variable line widths) you may select one of five functions which determine how the effect varies with Z. We don't try this by now but reset the darkness to a constant value of 1 (by a plain 'Modify Darkness' command) :

{ Darkn^{R} }$_{12,2}$

To switch to the space-filling model of the iron complex, use the 'Gen Cups' command:

{ Gen^{C} }$_{1,7}$

{ Cups }$_{7,2}$

Note: While the default partition of *varyZ.scf (*Command file *v.scf* , generating depth-dependent pellucidity) is appropriate for ball-and-stick models, *vc.scf* should be used for cup models. Try it:

{ varyZ^{L} }$_{3,6}$

{ Cups }$_{15,1}$

Note: Some distributed Command files in the directories { cupBS } and { shade } will act independently of the model type you selected previously.

Note: Distance-dependent darkness („depth”) as well as the model type [{ B }$_{1,10}$] can also be controlled from within the on-screen rotations box (see section 1.7. and fig. 25).

2.3. More atom/bond specific parameters

Commands to modify atom/bond specific parameters are described (besides others) in chapter 6 of the interactive manual. You already tried the 'Change Colour' command, above. Now, let's switch back to the ball-and-stick model and then modify some radii via 'Change Size ..':

{ BallStck }$_{6,2}$

{ CHG^{C} }$_{1,11}$

{ Size * ^{L} }$_{13,1}$

Set the parameter to 0.5, then click

{ Fe } { [OK] }

{ XQT^{L} }$_{1,4}$ (red star)

The radii of the Fe atoms have been multiplied by 0.5. A plain 'Chge Radii' without a parameter specified resets all radii to their default values:

{ Radii^{R} }$_{12,2}$

{ XQT^{L} }$_{1,4}$ (red star)

In a similar way, the 'Change Thickness' command may be used to modify the thickness of selected bond sticks.

Note: Some distributed Command files (mainly some in *cupBS.scf*) will use small radii for H (or other) atoms independently of the radii you seleceted prior to executing the Command file.

Another important atom/bond specific parameter is the "darkening function number" (DFN). This controls the maximal and minimal darkness of an atom/bond's surface in shaded drawings (additionally, darkness is controlled by the commands of table −−> 52 of the interactive manual).

The 30 basic DFNs are illustrated in fig. 5. Click

{ Darkf ^{L} }$_{6,1}$

Set the first parameter to 6 and click

{ Fe1 Fe2 } { [OK] }

{ }$_{2,2}$

{ shade^{R} }$_{3,5}$

You will see, that the appearance of two of the four iron atoms has changed significantly. With

{ Darkf ^{R} }$_{6,2}$

you re-assign default DFNs to all iron atoms.

2.4. Colour

Some general aspects of colour are best explained in front of the 'Xqt Test' colour matrix:

```
X  T
```

With SCHAKAL, colour is usually specified by the index of the desired "main colour" (i.e. the index of the corresponding row of the colour matrix). In session 1, you already learned how to change individual colours of selected bonds by specifiying the main colour index in 'Chge Colour' commands and to set the background colour to main colour no. 12 .
Additionally, some colour commands allow specification of a "colour step" index (i.e., the index of the corresponding column of the colour matrix; note that the first and the last columns do not cor- respond to colour steps but to pure white and pure black). You may try this with the 'Set Back- ground' command (another suitable command would be 'Set Colour ..', here):

{ SET^{C} }$_{1,9}$

{ Backgr^{L} }$_{8,1}$

{ *blue* } { [OK] }

This sets the background colour to a medium colour step of main colour no. 4 (blue).

Note: While for the DOS and the X-Windows version (running under an 8 bit colours X server) the background can be changed "behind" the drawing, for the MS-Win version and the X-Win version (running under a 16 or 24 bit X server) any change in background colour will erase the current drawing.

When you clicked { *blue* }, a second row of colours appeared in the colour box which displays the different colour steps of blue. Click the 5th colour step (from the left)

{ *5th blue colour step* } { [OK] }

The command 'S B 4 5' is executed and the background displays the selected colour step, now.

Another way to specify colours is by HSL colour coordinates. HSL means Hue, Saturation, Lightness. The HSL coordinates system has been used by the Tektronix company for their colour graphics terminals. The meaning of the three coordinates is explained in the following:

Hue — is defined in a cyclic range with values between 0 and 360; some characteristic hue values are

0/360 = blue	60 = magenta (violet)	120 = red
300 = cyan	240 = green	180 = yellow

Saturation — is defined in the range 0 to 100 with 0 = pure grey, 100 = maximal saturation (brightness) of the colour

Lightness — is defined in the range 0 to 100 with 0 = black, 100 = white.

The second form of the 'Set Background' command uses HSL coordinates. Click

{ Backhsl }$_{9,1}$

to get a HSL box in the right side of the first three rows of the GUI. First, click a hue value in the first row, then a saturation value in the 2nd row, then a lightness value in the 3rd row, and finally

{ [OK] }

If the colour matrix of the test drawing is still on the screen, you will see, that some colours of the colour palette are modified in order to generate the colours of the HSL box. Original colours will be reset at the time when the HSL box is erased (see below).

You may now modify the HSL values by clicking into any of the three rows and realize the result with { [OK] }, etc. Finally let's have a black background, again, and clear the GUI:

{ SET^{L} }$_{1,9}$

{ }$_{2,2}$

HSL coordinates are also used to change the characteristics of a certain main colour. This is done via the 'Set Hue' command.

{ Hue^{L} }$_{10,1}$

Within the colour box you may now select the main colour which you want to modify, e.g.

{ *red* }

A HS box is opened and you may specify new hue and saturation values for main colour no. 2. Execute the command with

{ [OK] }

{ XQT^{L} }$_{1,4}$

Oxygen atoms (to which - by default - main colour no 2 is assigned) should now display the new HS values, as well as some parts of the GUI (which uses colours 1 to 5). A list of the 13 main colours with their default and current HS values is obtained with

{ LST^{C} }$_{1,6}$

{ Hues }$_{7,2}$

Current HS values differing from default values are flagged by "*". With the following, you re-assign default HS values to all main colours and reset the background to black:

{ SET^{C} }$_{1,9}$

{ Hue^{R} }$_{10,2}$

{ $Backgr^{R}$ }$_{8,2}$

As you already have experienced, the latter command is also available from { SET^{L} }$_{1,9}$. Finally we erase the HSL box with

{ }$_{2,2}$

2.5. Perspective and stereo

The *ex1.scf* Command file contains a 'Set View 27' command which sets a view distance of 27 Angstrom to generate a corresponding perspective distortion in all drawings. Click

{ SET^{C} }$_{1,9}$

{ View^{R} }$_{14,2}$

{ XQT^{L} }$_{1,4}$

To get a drawing with perspective switched off (i.e. with parallel projection mode). You will note that some bond sticks still show a perspective tapering. This tapering is "artificial". You can get rid of it with a plain 'Set Tapering' command.

{ Taper^{R} }$_{18,2}$

{ XQT^{L} }$_{1,4}$ (red star)

To get now a rather severe perspective distortion, click/type

{ View^{L} }$_{14,1}$

Set the parameter to 10 and click { [OK] }. Then click

{ XQT^{L} }$_{1,4}$ (red star)

To regenerate the old conditions, set the parameter in the parameter box (2nd row of GUI) to 27 and click { [OK] }. Then execute a 'Set Tapering -1' command

{ Taper^{L} }$_{18,1}$

Set the parameter to -1 (good for stereo drawings, see below) and click { [OK] }.
Note, that for **space-filling** models, perspective may lead to erroneous shadows in some cases.

The 'Generate View' command switches between normal and stereo drawings.

{ GEN^{C} }$_{1,7}$

{ View 1 }$_{21,2}$

{ XQT^{L} }$_{1,4}$

A stereo pair has been generated. It may be viewed with a "mirror stereoscope".

{ }$_{2,2}$

{ shade^{R} }$_{3,5}$

Command file *s.scf* now also produced a stereo pair. Most distributed Command files would have acted this way (*ls.scf* [{ lines^{L} } / { lstreo }] and *las.scf* [{ shade^{L} } / { lstreo } ; —> fig. 14] have stereo mode "built in").

2.6. Scale factor and position of drawing

Up to now, all drawings habe been generated using a "variable" or "optimized" scale factor. I.e., the program calculated scale factor and position of the drawing (= position of the origin of the internal Cartesian coordinate system within the drawing area) to make optimal use of the drawing area. The scale factor can be fixed by a 'Mgn Model p' command where p is in cm/A. In this case, the position of the drawing within the drawing area is controlled by 2 parameters which may be modified by the 'Set Origin' command (default position = center of drawing area).

{ MGN^{C} }$_{1,10}$

{ Model^{L} }$_{5,1}$

Set the parameter to 2 and click { [OK] }.

{ XQT^{L} }$_{1,4}$

Now, although stereo mode is still active, only one drawing (namely, the left-eye part of the stereo pair) has been generated. Both parts of a stereo pair are only drawn when a "variable" scale factor is switched on! To reduce confusion, let's switch stereo mode off, now, (by a plain 'Genr View') before you use the 'Set Origin' command:

{ GEN^{C} }$_{1,7}$

{ View^{R} }$_{20,2}$

{ SET^{C} }$_{1,9}$

{ Origin^{R} }$_{21,2}$

"Graphics cursor mode" is switched on. Select any position by clicking and ...

{ XQT^{L} }$_{1,4}$

The center of the drawing (i.e., the origin of the internal Cartesian coordinate system, –—> 673) is now at the selected position. With another

{ Origin^{R} }$_{21,2}$

you may position - by means of the graphics cursor - the drawing at another location. Now click

{ Origin^{L} }$_{21,1}$

Ignore the parameter boxes and type

```
1000
```

to reset the origin position to the center of the drawing area (for drawings [with fixed scale factor] to come). The variable, calculated scale factor is re-established with a plain 'Mgn Model' by

{ MGN^{C} }$_{1,10}$

{ ModCalc^{R} }$_{6,2}$

2.7. Labelling

With most distributed command files, you may add the partition labels "w" or "$" to the command file name to invoke labelling of all non-Hydrogen or all group $ atoms (see 2.1.):

{ }$_{2,2}$

{ shade^{L} }$_{3,5}$

{ $ }$_{7,2}$ (CF turns light green)

{ simple }$_{13,2}$

The results of the labelling procedure may be influenced by some commands which have been added to the { WRT } group:

{ WRT^{C} }$_{1,3}$

First we increase the character size to 1.3 times the default size (this requires a negative size parameter):

{ MGN Insc^{L}}$_{23,1}$

Set the first parameter to -1.3 and click { [OK] }. Then, we make the characters slanted:

{ SET Insc^{L} }$_{25,1}$

Set the **2nd** parameter to 4 and click { [OK] }. Finally we restrict the arrangement of letters and numbers in atoms names to horizontal and enclose numbers in parantheses:

{ SET Wrt^{L} }$_{26,1}$

Set the **2nd** as well as the **3rd** parameter to 1 and click { [OK] }. Then click

{ USE^{L} }$_{1,1}$

The latter generates a plain 'Use' command which means that the Input-, Command-, or Writing file which has been specified last by the user is now re-used. In this case it is the file *si.scf* (including the option to label group $ atoms) which is re-used. However, labelling results are different from before due to the three commands given above.

Normally, with the 'Broaden Inscription' or 'Broaden All' command, you may set the boldness of labelling characters. Most distributed Command files, however, set the boldness of characters themselves by calling the Command file *broad.scf* . Thus, your only chance to adjust boldness in this case is to edit *broad.scf* or to use the Command file without "w" / "$" and label the drawing explicitely via 'Wrte Label ..' (see above).

If the line width of characters (and, therefore, line width of outlines or hatching lines) has been set by a distributed Command file, it will not be reset to the old value at the end of the Command file (in contrast to all other parameters). This measure allows correction of label positions by means of the 'Translate Inscription' command (which turns "graphics cursor mode" on):

{ TRL Insc }$_{22,1}$
{ Fe3 Fe1 Fe4 } { [OK] }

The graphics cursor moves to the position of the "Fe1" label. Move the cursor a little and click the left mouse button or type <SPACE BAR> . The "Fe1" label is moved to the new position. Repeat this procedure until you obtain a position you like. Then click the **right** mouse button or type <m>. The cursor appears at the "Fe3" label. You can move this label in the same way as before (don't mind if the drawing is ruined a little by these operations). With

{ ESC }

you leave cursor mode, i.e., the "Fe4" label remains untouched.
You may wonder why the Fe1 label has been processed before the Fe3 label despite the fact that, in the command, the order of the two corresponding atom codes was opposite. Now, this actually corresponds to the general behaviour of SCHAKAL in processing atom codes (ACs) in commands:

When ACs/BCs are specified, the program first flags all atoms/bonds matching the ACs/BCs, internally. Then the atoms/bonds are processed in the order given by the program's internal atoms/bonds list (an exception are the commands 'Define Plane and Define Line' where the first three (two) atoms which are specified have a special meaning).

Note: correction of labelling via 'Transl Insc' is not possible with DCFs given in the following list. With those ones flagged with "(-)", even a following explicite labelling (via 'Wrt ..') will not work (i.e., lead to strange results).

control fields	DCF name	control fields	DCF name
{ shade^{L} / lablCp }	*la.scf*	{ lines^{L} / relief }	*r.scf* (-)
{ shade^{L} / lstreo }	*las.scf*	{ cupBS^{L} / ..... }	*c*.scf* (-)

Note: Some DCFs label using a variable darkness; correcting label positions will in this case require to set the corresponding 'Modify Darkness' command again "by hand".

Note: read also text describing fig.s 11 and 12 .

Note: correction of label positions is slightly different for printer devices (see section 4.3.).

Finally, let's reset some parameter values, now:

{MGN Inscr^{R} }$_{23,2}$

{ SET Inscr^{R} }$_{25,2}$

{ SET Wrt^{L} }$_{26,1}$

Set all parameters to 0 and click { [OK] }.

Again, you may continue with session 3, now, or terminate the program either with

{ - }

or by closing the window as usual.

Session 3: Some advanced options

3.1. Erasing a part of a screen drawing

If you re-started the program right before Session 3, perform the following action:

{ Inputf^{R} }$_{6,2}$

```
ex1
```

{ Commandf^{R} }$_{13,2}$

```
ex1
```

While the whole drawing area has been erased a lot of times during sessions 1 and 2 due to "automatic screen erasing" (see 1.5.), you will now learn how to erase only a part of a drawing. First, make another drawing of the iron complex with the help of the DCF *s.scf* . Then label the iron atoms by an extra 'Wrt Label ..' command:

{ shade^{R} }$_{3,5}$

{ WRT^{C} }$_{1,3}$

{ Label^{L} }$_{6,1}$

{ Fe } { [OK] }

The last action (invoked by 'W L..') may now be cancelled with the 'Kill Last' command:

{ l }$_{2,1}$

Labelling is repeated, this time using the background colour to write the labels, which lets the latter disappear. On the screen, 'K L' should mainly be used to cancel out line-drawing operations (not shading).

Note: 'Kill Last' will not work correctly for *screen* drawings on a non-empty background (not valid for *printer* drawings [see session 4]).

The 'Kill Region' command works differently:

{ r }$_{3,1}$

The program is now in "graphics cursor mode". Click four positions which define approximately a rectangle containing the four iron atoms. After clicking { ESC }, the contents of this rectangle will be filled (irreversibly) with background colour, i.e., erased.

Note: after a switch from the "screen device" to a "printer device" (see session 4), the command 'Kill Region' will be disabled. However, another command to erase parts of the drawing ('Kill Xqt') will be enabled in this case.

3.2. Making Command files of your own

You may set up your own Command files in two ways:

a) by writing a Command file with the editor: click

{ EDT^{C} }$_{1,2}$

{ Commandf^{R} }$_{7,2}$

```
myfile
```

The editor comes up to edit the file. If the file *myfile.scf* was already existing, erase its contents. Write the following lines into it:

```
g o
x
g s -1
x x  fe fe=fe
w l  fe
g o
```

Then exit/save from the editor and generate a plain 'Use' command by clicking

{ USE^{L} }$_{1,1}$

The commands you typed are now executed one after the other. You could now modify the file with the editor (by clicking { EDT^{L} }) and execute it again (by clicking { USE^{L} }), etc.

b) by opening an Echo file, recording a series of commands, closing the Echo file, and - possibly -
edit the Echo file with the editor. Click

{ USE^{C} }$_{1,1}$

{ Echof^{R} }$_{14,2}$

```
myecho
```

If the file *somepath//myecho.scf* aleady exists, override it. The file is opened to record user action to come. When an Echo file is opened, most parameters and switches of the program are reset to their default values. **It's always wise to (re-) load an Input file** (structure data) **just after opening an Echo file:**

{ Inputf^{R} }$_{6,2}$

```
ex1
```

{ Commandf^{R} }$_{13,2}$

```
ex1
```

Now let's do some drawing including labelling and correction of labelling:

{ varyZ^{R} }$_{3,6}$

{ WRT^{C} }$_{1,3}$

{ Label^{L} }$_{6,1}$

{ Fe } { [OK] }

{ TRL Insc }$_{22,1}$

{ Fe1 } { [OK] }

now move the Fe1 label with the mouse by 'T I Fe1' as described in 2.7. and terminate with

{ ESC }$_{4,3}$

Let's then add an arrow which points to some part of the molecule ('Xqt Arrow'):

{ XQT^{C} }$_{1,4}$

{ Arrow^{R} }$_{19,2}$

With two mouse clicks you specify a straight line to which an arrow tip is added when you click

{ ESC }$_{4,3}$

Finally we write some text into the drawing ('Wrte Text', which is part of the { EDT } group):

{ EDT^{C} }$_{1,2}$

{ Text^{R} }$_{18,2}$ (don't click { Text^{R} }$_{13,2}$!! if you already did, type any word, e.g. `mytext` then click { [OK] }, then click { Text^{R} }$_{18,2}$)

The program is in "graphics cursor mode" now; click a position about 1 cm at the right of the uppermost H atom. The chemical formula of the iron complex is written. It has been taken from the TItl card of the Input file *somepath//ex1.dat* (and could have been replaced by some other text via 'Use Text'). You may move the text by clicking other positions nearby; then click

{ ESC }$_{4,3}$

(Note that size, colour, boldness, slantedness of the characters could have been modified with the commands appearing at the bottom of the command column, cf. section 4.2.).

Now let's close the Echo file *somepath//myecho.scf* with the 'Use Echofile 1' command:

{ USE^{C} }$_{1,1}$

{ Echof 1 }$_{15,2}$

To see what we have stored on this Echo file we execute it now as a Command file:

{ Commandf^{R} }$_{13,2}$

{ =^{R} }$_{4,5}$

The actions which have been recorded are "re-played" now. You could then edit the file with { EDT^{L}} and use it again with { USE^{L} }, etc.

3.3. "Fourier" backgrounds

Up to now, we had plain black, white, or other uniformly coloured backgrounds. However, SCHAKAL offers facilities to generate less trivial backgrounds. Click

{ fourr^{R} }$_{3,7}$

The drawing area is now slowly filled with grey colours of varying darkness. Actually, the image is a "Fourier image" composed of "Fourier waves". Let's have a look at *fourr.scf* :

{ EDT^{L} }$_{1,2}$

The default partition of this file (headed by 'Y Label "deflt*") generated the image. The FOurr cards in this partition contain the parameters h, k, F_{hk}, and ϕ_{hk}; each card defines one of **max. 100** Fourier components. If you are crystallographically trained, you may consider the drawing area as a 2D unit cell; h and k then have their usual crystallographic meaning, F_{hk} is the analogon to the structure amplitude (except for F_{00} it should have values < 0.1) , and ϕ_{hk} is the Fourier wave's phase (in degrees). (Note: Some partitions of *fourr.scf* contain only one or two FOurr cards with some code letters. In these cases, the background is not generated by Fourier syntheses but by some simple functions; i.e., the label 'FOurr' is somewhat misleading in these cases).

Of course, you may add more partitions to the file *fourr.scf* . Furthermore, you may modify the existing Fourier images to some degree: You may change the main colour with 'Set ColOth', and modify darkness and contrast with 'Modify Fourier' (in the { MGN^{C} }$_{1,14}$ group):

{ SET^{C} }$_{1,9}$

{ ColOth }$_{6,1}$

{ *blue* } { [OK] }

{ MGN^{C} }$_{1,14}$

{ Fourier^{L} }$_{19,1}$

Set the two parameters to 1 and -2 and click { [OK] }; then re-use *fourr.scf* by clicking

{ USE^{L} }$_{1,1}$

The bottom of *fourr.scf* contains the command 'Kill Screen 3' which switches automatic screen erasing off (as a consequence, the red CF { K }$_{3,1.5}$ has appeared on the GUI). This is to take care that the background ist **not** erased by the next drawing:

{ }$_{2,2}$

{ varyZ^{R} }$_{3,6}$

{ K }$_{3,1.5}$

The latter re-activates autom. screen erasing: the red { K }$_{3,1.5}$ is replaced by the green { K }$_{2,1.5}$.

Note: On a Fourier background (as on any non-empty background), shifting of atom labels ('Translate Insc') or of graphical text ('Write Text') will lead to unreasonable results (valid only for *screen* drawings, not for *printer* drawings, see section 4.).

3.4. Saving & Restoring screen images

Images drawn up to now could not have been sent to any printer directly (see session 4 for printer output). However, they could have been stored ("saved") onto a raster image file.There are three possible file formats: TIFF, .PCX, and .SIF (the latter being the, non-standard, SCHAKAL 92 format). If possible you should use the TIFF format to save SCHAKAL 99 screen images. Click

{ shade^{R} }$_{3,5}$

to generate a shaded drawing of the iron complex. Now, click

{ USE^{C} }$_{1,1}$

{ Savef^{R} }$_{18,2}$

`test`

The screen image is now stored onto file *somepath//test.tif* . (note: you could also have clicked { save }$_{4,10}$). You can restore it as follows:

{ K }$_{2,1.5}$ (only necessary here to see more clearly what happens)

{ Restoref^{R} }$_{19,2}$

{ =^{R} }$_{4,5}$

By specifying "=" as the file name, the name core ("test") of the Savefile is used as the name core of the Restorefile. The image is restored [*MS-Win:* storing/restoring of raster data is up to about 15 to 20 times slower than for the DOS version, presently].

[*MS-Win:* skip the rest of this section; *UNIX:* skip this if you are using an X server with >8 bit colour depth]
An image may be restored either solid or "transparently". In the latter case, the current contents of the drawing area is overlayed with the image to be restored. Let's demonstrate this by means of another "Fourier" background

{ fourr^{L} }$_{3,7}$ / { ellipt }$_{13,2}$

Another "Fourier" background is generated. Instead of adding some drawing to it (as before) you may now re-activate automatic screen-erasing and overlay the background with a stored image:

{ K }$_{3,1.5}$

{ USE^{C} }$_{1,1}$

{ Restoref^{L} }$_{19,1}$ (light blue part !!)

Select a parameter of 10 and click { [OK] }. As an answer to the file name request click another

{ [OK] }

The image is now *added* to the background. Clear the GUI now, by clicking

{ }$_{2,2}$

3.5. Generating TIFF images with enhanced resolution

SCHAKAL 99 offers a procedure which allows the generation of TIFF (or .PCX) images with a resolution of up to 5 times the (linear) resolution of the drawing area (use 'List Size' to check for the resolution of your drawing area). This procedure requires that a Command file exists which draws the image. This command file may be a DCF or a user-defined Command file. Click

{ resxn^{L} }$_{3,8}$

{ resx2* }$_{8,2}$

This starts a procedure to generate an image with 2 times the linear resolution of the drawing area. E.g., if the drawing area has a resolution of 1000 * 800, the image will have a resolution of 2000 * 1600. resx3 would generate an image of 3000 * 2400, etc.

The program asks to specify the Command file which draws the image. Click

{ shade^{R} }$_{3,5}$

The image is now generated in four parts. Each part is stored (automatically) as a TIFF file. After generation and storing of all four parts you are asked to specify the name of a Save file onto which the "sum" of all four files can be stored:

`test2`

If you restore the image with 'Use Restorefile', it will be compressed into the drawing area and look as if there was no resolution enhancement:

{ Restoref^{R} }$_{19,2}$

{ =^{R} }$_{4,5}$

However, if you print the files *test.tif* and *test2.tif* by means of an appropriate software, the latter will probably produce better results.Let's repeat the procedure now using *myecho.scf* (see 3.2.) as the Command file which draws the image:

{ resxn^{R} }$_{3,8}$

`myecho`

Store the complete image on *test3* and restore it as shown above. You will see that some atoms/ bonds drawn pellucid (by { varyZ }, when *myecho.scf* was recorded, see 3.2.) appear either more pellucid or less pellucid as compared to the original image. This is due to compression of the large TIFF file into the drawing area. This effect should not occur if the file was printed. On the screen, it could have been avoided by using { resx3 } instead of { resx2 }, above.

It is therefore recommended to rather use { resx3 } [or { resx5 }] for the generation of large TIFF files. However, the larger the resolution ....

a) ... the longer the time to generate the whole drawing and the larger the file and therefore ...
b) ... the lower the probability that your (printing or text processing) software can handle the file.

3.6. Generating non-coloured screen drawings

For some purposes you might want to have rather a black-and-white or a grey image instead of a coloured one. SCHAKAL offers a facility to generate drawings in black-and-white or in grey.

This leads us to the term "device". With SCHAKAL, drawings can be "sent" alternatively to a number of different devices (see session 4). Per default, the "screen device" (device no. 1) is active. Other devices can be accessed with the 'Genr Device' command (GUI group G/S).

Most devices can be set up in a "colour mode" or in a "non-colour mode". This is also valid for the screen device which, per default, has been set up in "colour mode". To set it up in "non-colour mode", we use the 'Genr Device * 1' command. A "*" in a 'Genr Device' command generally sets the device up in "non-colour" mode.

{ G/S^{C} }$_{1,8}$

{ Device^{L} }$_{5,1}$

The "*" appears in the upper right corner of the { [OK] } button.

{ * }$_{2,3}$ (The control field turns light blue)

{ [OK] }$_{3,3}$

The screen device is now in non-colour mode. Click

{ shade^{R} }$_{3,5}$

In shaded drawings, while in non-colour mode, the program tries to translate differences in colour into differences in darkness, to make atoms of different types easier distinguishable. E.g, the green N atom is shaded lighter than the red O atoms, although the same darkening function is assigned to
N and O per default).

Let's return now to the screen's colour mode by a plain 'Genr Device' command:

{ Device^{R} }$_{5,2}$

You may now terminate the program or continue with session 4.

Session 4: Printer drawings

4.1. Printer Devices

In case you have re-started the program right before, you should again do the following

{ Inputf^{R} }$_{6,2}$

`ex1`

{ Commandf^{R} }$_{13,2}$

`ex1`

Drawings which have been made for the screen (device no. 1; see 3.6.) cannot be printed directly on any printer. **If you want to print a drawing, you first have to switch to the appropriate "device". Then you can make a drawing and print it.** Table 4 gives an overview on SCHAKAL 99's different graphics devices:

Table 4: SCHAKAL 99's graphics devices

#	Name	type	colours	direct printer output	compatible to other software
1	Screen (default)	screen	256	**no**	as TIFF or .PCX file (via 'U S')
3	HP-GL	printer/file	6 or 8	HP plotter [1)]	yes
4	LaserJet 3-6	printer/file	b/w	HP LaserJet 3-6 [1)]	no
5	HP-GL/2	printer/file	256	HP DeskJet 1200 [1)]	probably no
6	EPS/Postscript	printer/file	256	Postscript printer	yes (EPS)
7	WMF	(printer)/file	256	**no**	yes (files may be very large!)
8	Windows Printer [2)]	wprinter/file	256	Windows printer [3)]	maybe, if directed to file

[1)] or a fully compatible printer [2)] MS-Win version only [3)] not suitable for shaded HP LaserJet drawings

For the following, you may select one of printer devices 3 to 6, or 8, the one which suits you most.
This is done by means of the 'Genr Device' command which is in the { G/S } group. Click

{ G/S^{C} }$_{1,8}$

{ Device^{L} }$_{5,1}$

set the first parameter to p_1 = 3,4,5,6, or 8 [*US user*: you probably should set the 2nd parameter to 3 to activate US A format]. Later on, you may try higher resolutions, too (2nd parameter box).Then click

{ [OK] }

MS-Win: If you chose **device no. 8**, the usual printer selection box appears; just select the printer and (possibly) adjust some of its properties (see —> 968 !), then click OK. No printing is performed right now !

The screen is erased, background colour changes to white, and a grey stripe appears at the top or at the right side of the screen. The residual white area simulates that part of the A4 (or A) paper sheet onto which something can be drawn.
Actually, the new device is a file, onto which graphics commands (to drive the printer) will be stored. Per default, the file's name is *schakal.xxx* [Difference to SCHAKAL 92 !!] with *xxx* differing for the different printer devices . Now let's draw something:

{ shade^{R} }$_{3,5}$

A shaded drawing of (possibly) poor quality is generated on the screen (invoked by Command file *s.scf*). However, the screen drawing is only an imperfect visualization of what is written onto the Printer file *schakal.xxx* , here. If you ever send this file to the printer, the final drawing should be of better quality. About printing, see chapter 4.2. .

If you want to make another drawing, you have two choices: erase the old drawing by clicking { K } (generating a 'Kill Screen' command), which also erases the contents of the Printer file, or open a new Printer file via the 'Use Printerfile' command:

{ Printerf^{R} }$_{9,2}$

```
plot2
```

The old Printer file, *schakal.xxx* , has been closed and resides still within its directory. **Only after closing** (e.g. by 'U P' using a dummy file name) **a Printer file can be used by another software**. The file may be sent to a printer after the program has been terminated (however, it will be overriden by the next SCHAKAL run which experiences a 'Genr Device p' [p = p_1] command).

The newly opened Printer file, *plot2.xxx* , now waits for printer commands to be sent to it.

Note that, in contrast to the screen device, no automatic erasing of a drawing will occur for a printer device (except when using distributed command files which contain a 'Kill Screen' at the beginning of the corresponding device partition (–—> 766)).

Let's store another drawing, now, on the new file *plot2.xxx* :

{ lines^{L} }$_{2,4}$

{ pluto }$_{10,2}$

You can generally switch temporarily to the screen device, do something (but you shouldn't do things like change the orientation of the model, of course), and then switch back to the same printer device without losing the current Printer file:

{ Device^{R} }$_{5,2}$ (plain 'Genr Device' switches to the screen device)

{ XQT^{L} }$_{1,4}$

{ WRT^{L} }$_{1,3}$

{ LST^{R} }$_{1,6}$

{ Device^{L} }$_{5,1}$

{ [OK] } [Windows printer: click also the OK button of the printer selection box]

The Printer file is re-visualized on the screen and you can continue to work on it.
Note that such an intermediate switch (from a printer device to another device and back) **with keeping the Printer file open, is only possible, if the other device is the screen device (no. 1) and if the switch back is to the same printer device (same format, same resolution) as you started from.**

4.2. Printing

If a corresponding printer (compatible with the device you selected in 4.1.) is connected to your computer and if ...

DOS ... the batch file *cppy#.bat* [*# = device number*] has been set up accordingly
UNIX ... the script file *cppy#.sh* [*# = device number*] has been set up accordingly
MS-Win ... you have selected the Windows printer (device 8) or - if not - the printer is connected to 'lpt1' or - if not - an appropriate 'U P -10# ..' [# = device number] command has been given previously (—> 155.2)

you may send the Printer file directly to the printer by a 'Use Printerfile 2' command, i.e. by:

{ Print }$_{4,10}$

The Printer file is now copied to the printer (which will probably require some time). After this, the drawing will be erased and then re-visualized on the screen [*MS-Win:* the latter is not true for the Windows printer (device 8)] . **You may stop re-visualization by pressing '<ESC>'. In this case, you should erase the incomplete drawing with { K }$_{2,1.5}$!**

When the copy or print operation has been performed, the command prompt (>>>) appears and the file is ready to store additional printer commands (you could also erase a part of the Printer file with 'K X', see 4.3.). Let's now add some other piece of text (cf. section 3.2.), this time in our own words. We could use 'Use Text' (in the { EDT } group) for a single line of text or 'Use Writingfile' for more than one line. In the latter case we first have to set up a Writing file:

{ EDT^{C} }$_{1,2}$

{ Writingf }$_{8,2}$ (don't click { Writingf }$_{15,2}$; if you already did, click { n }$_{4,3}$!)

```
mytext
```

When the editor has opened the new Writing file *mytext.txt* , write the following two lines into it:

```
to be or
not to be
```

Then save/exit from the editor. Before we write the text, we may alter size, boldness, slantedness, and colour of the characters using the four commands on bottom of the { EDT } group or typing

```
M T  1.5
B T  4 2
S I  0 3
S C  7 0 1
```

Now let's write the text with 'Use Writingfile' or with a plain 'Use' into a free space:

{ USE^{L} }$_{1,1}$ (red star)

click at some positions (only the first line of the Writing file is displayed) until you have found a good one and then leave "graphics cursor mode" with { ESC } to see the whole text. Then you can print the current drawing with another

{ Print }$_{4,10}$

4.3. Erasing a part of a printer drawing

Now, let's construct another drawing from several parts. Click

{ K }$_{2,1.5}$ (erase the current drawing)

{ XQT^{R} }$_{1,4}$ (draw the outlines; "outlines" is the default drawing mode)

{ GEN^{L} }$_{1,7}$ (generates a 'Genr Shading' command)

{ XQT^{C} }$_{1,4}$

{ XQT }$_{6,1}$

{ O } { [OK] } (shade the O atoms ['X X O'])

{ C } { [OK] } (shade the C atoms ['X X C'])

{ Fe } { [OK] } (shade the Fe atoms ['X X Fe'])

{ WRT^{C} }$_{1,3}$

{ Label^{L} }$_{6,1}$

{ Fe } { [OK] } (label the Fe atoms ['W L Fe'])

{ l }$_{2,1}$

The last click generated a 'Kill Last' command which has the same effect as previously (3.1.): The four labels disappear (on the screen as well as on the printer file). If a part of an atom/bond looks slightly damaged afterwards: this atom/bond will be o.k. on the printed drawing !

In contrast to the screen device, a 'Kill Region' will not work here. However, any action invoked previously by 'X..' or 'W..' can be cancelled (more savely than via 'Kill Last') with 'Kill Xqt n'. Type

```
K X 2
```

The screen is erased and the Printer file is re-visualized. However, the parts of the drawing generated by 'X X Fe' and 'X X C' are omitted. As the results of the 'W L..' command have already been removed from the Printer file by 'Kill Last', the 'Kill Xqt 2' command refers to these two commands.

A plain 'KILL Xqt' (n = 1) can be generated by clicking

{ x }$_{3,1}$

The shading of the O atoms will now also be removed from the drawing. If you like, you can now add more graphics to the file:

{ Fe N C1 } { [OK] } (as the 'Write Label' command is still latently active)

{ K }$_{2,1.5}$ (erase the drawing)

4.4. Portrait format

Now let's make a drawing using a "predefined pattern" :

{ GEN^{C} }$_{1,7}$

{ Outlines }$_{13,2}$

{ XQT^{R} }$_{1,4}$

The following command may be typed or generated with the GUI as usual (when you type them, you may omit the "0" after "P"):

X P 0 11 (for pattern indices, see fig.s 9 and 10)

{ GEN^{L} }$_{1,7}$ ('Genr Shading': this is to have labels exclusively **outside** the atom circles)

W L Fe

The printer paper, as simulated by the white area on the screen is in "landscape" format. However, you might prefer to have the printer drawing in "portrait" format.

Portrait format is selected by a 'Set Inscr ! 1 0' or 'Set Inscr ! 3 0' command (this command was originally only used to rotate the writing direction for labels and text)

{ G/S^{C} }$_{1,9}$

{ Inscr^{L} }$_{17,1}$

{ ! }$_{2,3}$

Set the first parameter to 1 and click { [OK] }. Then let's repeat the last drawing:

{ K }$_{2,1.5}$

G O

{ XQT^{R} }$_{1,4}$

X P 11

{ GEN^{L} }$_{1,7}$

W L Fe

Model, writing direction, light source, and shading pattern have been rotated 90 degrees about the Z axis. You can now switch back to landscape format with a plain 'S I !' :

{ Inscr^{L} }$_{17,1}$

{ ! }$_{2,3}$

Set all parameters to 0 and click { [OK] }.

4.5. Correcting label positions for printer drawings

Let's now make a drawing with some labels added:

{ DEF^{L} }$_{1,12}$ { $ }$_{5,1}$

{ Fe N O } { [OK]]

{ pattn^{L} }$_{2,5}$

{ $ }$_{7,2}$ { 3Dim* }$_{9,2}$

Command file *p3d.scf* which has been invoked uses some predefined "patterns" stored in file *patt.scf* to make a drawing of the molecule (see below).

Correction of labelling is different for "screen drawings" and printer drawings. While for a screen drawing, several 'T I' commands could be used in arbitrary order for several labels to be corrected, **all** labels of interest will have to be specifed within **one** 'T I' command for a printer drawing:

{ WRT^{C} }$_{1,3}$

{ TRL Insc }$_{22,1}$

{ N Fe1 O32 } { [OK] }

All labels disappear, then all labels except those ones of N, Fe1 and O32 are re-written (if an atom/bond looks damaged afterwards: this will not affect the Printer file). The cursor appears at the center of Fe1. Move it and click a new position; use the **right** mouse button to jump to the next atom's center, etc. Or cancel with { ESC }$_{4,3}$. Then reset the GUI:

{ }$_{2,2}$

4.6. Going back to the screen device

You're going to leave the currently active printer device now and return to the screen device (by a plain 'Genr Device'):

{ G/S^{C} }$_{1,8}$

{ Device^{R} }$_{5,2}$

The screen is erased and the background colour changes to black. Note that you could return to the last active printer device and find the currently active Printer file re-visualized on the screen and ready for additional printer commands sent to it (cf. 4.1.).

Let's reset once more most drawing parameters to their default values, now:

{ other^{L} }$_{2,9}$

{ reset }$_{13,2}$

Session 5: Crystallo-Graphics

This session (especially sections 5.1. , 5.5., and 5.7.) may also be of interest for users who only want to draw single molecules as it teaches how to obtain complete molecules from crystallographic data in general cases.

5.1. Crystallographic data sets

If the source of structural data are *crystallographic data* , the corresponding Input file (cf. 1.4.) for SCHAKAL should contain the following information:

a) the crystallographic cell parameters (CEll card)
b) names and coordinates of an asymmetric set of atoms (ATom cards); **all atom names preferably different from each other.**
c) space group information (either a SPgr card with the spacegroup symbol or a couple of SYmm/ DUpl cards with the single symmetry operations of the space group plus a 'SPgr symops' card)
d) **no** ENd card

Such data sets can be written by a text editor using published crystal structure data.

A more convenient way to obtain such data sets is to use one of the sources given in table 5 . The corresponding files can be translated with the help of the 'USE Xtaldatafile n' command into an appropriate SCHAKAL Input file.

Table 5: Files which can be translated via 'Use Xtaldata n'

n	*source*	*file type / format*	*n*	*source*	*format / remarks*
1	SHELX	.ins or .res	4	ICSD via STN	display format
2	CSD	BCCAB format	5	[PDB]	(symmetry is ignored !)
3	ICSD	"print" format	6	CIF	standard

As an example we translate the distributed file *ex2a.ics*:

{ USE^{C} }$_{1,1}$

{ Xtaldat^{L} }$_{7,1}$

Set the parameter to 4 and click { [OK] }. Then specify the file name

`ex2a`

The program now asks for the name of the Input file onto which the ICSD file is to be translated You may answer by clicking

{ =^{R} }$_{4,5}$

which means that the file *somepath//ex2a.ics* is translated into the file *someotherpath//ex2a.dat* (if the file exists already, override it by clicking { [OK] }). The Input file is loaded immediately after translation. From now on you can use this Input file directly (i.e., the translation procedure has to be performed only once per structure). --- Let's clear the GUI now with

{ }$_{2,2}$

5.2. Box sections and spherical sections

The structure, now stored in *ex2a.dat* ($Pb_3(PS_4)_2$;space group $P2_13$) will be displayed with

{ quick^R }$_{3,4}$

The program generated just one unit cell with all atoms outside the unit cell walls cut off. The picture is somewhat confusing because of the many bonds. We will get rid of some of them in section 5.4 . Let's now have a look at the data set itself:

{ EDT^C }$_{1,2}$

{ Inputf^R }$_{6,2}$

{ [OK] }

The data set (containing a number of #comment lines at the beginning) is in accordance with the rules given in 5.1. - Leave the editor. We will use *ex2a.dat* now to generate a number of sections of the crystal structure of $Pb_3(PS_4)_2$. First we load the Input file once more by clicking

{ USE^L }$_{1,1}$

The data set is loaded and the program asks you to enter more data cards from the keyboard. At the same time the GUI's "command column" is replaced by a "card label column". This is the general behaviour if the program doesn't find an ENd card at the end of a data set.

Note: Size and shape of a crystal structure section generated by SCHAKAL are controlled mainly by data cards and not by commands. These data cards may be appended to the Input file (containing the structural data) or may be input "by hand" in the process of loading the Input file (the latter if the data set doesn't contain an ENd card).
Data input given "by hand" is generally terminated by clicking { END }$_{6,2}$ or, simply,

{ [OK] }

(i.e., no additional data are given, this time) . To display the structure, we first set a view distance of 100 A by typing the corresponding 'Set View' command or by clicking (see group { SET }):

S V 100

{ quick^R }$_{3,4}$

You will again see the unit cell of the structure (as the data set contains a SPgr card, but no BOx [or PAck] card, the program has added automatically a plain 'BOx' card (without parameters), see below). Unit cell edges (which are usually "hidden" after data processing of a non-crystallographic data set) have been switched on by the program because of the SPgr card in the data set.

The unit cell's a "axis" is displayed with grey colour (main colour no. 1), the b axis with red (no. 2), and the c axis with green (no. 3) colour.

If you want to modify size and/or shape of the model you will have to re-load the Input file:

{ USE^L }$_{1,1}$

The same Input file as before is re-loaded (as the command file *qs.scf* , invoked by { quick^R }$_{3,4}$,

which has been executed just before is generally excluded from the list of Input, Command, and Writing files; cf. section 1.4.). Again you are asked to specify additional data cards. Click

{ BOX^{L} }$_{19,1}$

Set the parameter to 0.25 and click { [OK] }. Then terminate the input of data cards with another

{ [OK] }

{ quick^{R} }$_{3,4}$

The model generated this way is enlarged by 0.25 (fractional coordinates) in all directions as compared to the model generated by the plain BOx card. Besides the 1-parameter syntax of the BOx card used here, there is also a 3-parameters and a 6-parameters syntax (available in the GUI from { BOXa } and { BOXb }). With these, any desired section of the structure with boundary walls parallel to the unit cell walls may be generated, provided that the program's atom/bond capacity is not exceeded.

Now let's re-load the Input file once more:

{ USE^{L} }$_{1,1}$

This time we use a RAdius card to generate a spherical section of the crystal structure:

{ RADIUSa }$_{17,1}$

Set the parameters to 15, 0.5, 0.5, 0.5 and click { [OK] }. Then terminate with another

{ [OK] }.

{ quick^{R} }$_{3,4}$

This time we have generated a spherical section of radius 15 A, center at the center of the unit cell. Let's now try a variant of the RAdius syntax. First re-load the Input file:

{ USE^{L} }$_{1,1}$

To input additional data, click

{ RADIUS }$_{16,1}$

{ 4.0 } { Pb1 }$_{2,13}$ { [OK] }

and terminate input of data cards with

{ [OK] }

{ quick^{R} }$_{3,4}$

We get a Pb1 atom of the structure plus all atoms bonded to it. In fact the program generated a spherical section of radius 4 A, center at the first Pb1 atom found in the Input file. But all atoms in this section are "hidden" initially except Pb and the atoms bonded to it. With a plain 'Def Fragment' command, i.e., by clicking { DEF^{L} }$_{1,12}$, you could make all atoms of the spherical section visible.

5.3. The EXpand region

A BOx section or a spherical section (or a polyhedral section) may be expanded by p Angstroms by means of an EXpand card. The additional space obtained is called the EXpand region; atoms which are within the EXpand region get the "EXpand region status". The parameter p should be slightly larger than the length of the longest bond in the structure. To check for the longest bonds in $Pb_3(PS_4)_2$ we first re-load the Input file *ex2a.dat* :

{ USE^{L} }$_{1,1}$

This time, we generate another model which is a a bit more than one unit cell by typing:

```
BO .3
```

{ [OK] }

{ quick^{R} }$_{3,4}$

```
L B
```

We can now use the output produced by the 'List Bonds' command to check that the longest bonds have a length of 3.16 A. Therefore we use an EXpand parameter p of 3.2 A

{ USE^{L} }$_{1,1}$

The file *ex2a.dat* is re-loaded.

{ EXPA }$_{9,1}$

Set the parameter to 3.2 and click { [OK] }.

{ BOX^{L} }$_{19,1}$

Set the parameter to 0.25 and click { [OK] }. Terminate data input with

{ END }$_{6,2}$

{ quick^{R} }$_{3,4}$

The 'BOx 0.25' section has been expanded by an "enveloping" layer of 3.2 A thickness, the "EXpand region". We can now use this EXpand region to re-shape the model in a way that it is reduced to the 'BOx 0.25' section but with all bonds pointing from this section "outwards" represented by broken-off bond sticks. This is done with the 'Transform EXpandregion 1' command which is in the { ADD } group:

{ ADD^{C} }$_{1,13}$

{ Expand 1^{R} }$_{20,2}$

{ quick^{R} }$_{3,4}$

This representation visualizes that the model doesn't end at its borderlines but is only a section of an infinite crystal structure.

5.4. Customizing bond pattern / Surface models

For the purposes of the following we will consider the structure to consist of "isolated" tetrahedral PS_4^{3-} anions held together by Pb^{2+} cations. Therefore we will get rid of the many Pb-S bonds which make the models look rather confusingly. This aim could be achieved by three different measures:

a) using a 'Kill Bonds PB=S' command **anytime** the loading of the data set has been completed.
b) Inserting **once** a 'DIst 0 Pb S' card into the data set which would prevent any PB-S bond from being generated by the automatic bond finding routines.
c) In this special case: Telling the program to treat the Pb atom as "ionic" atoms but leave the P and S atoms "covalent". SCHAKAL will not generate bonds between "ionic" and "covalent" atoms, except by an 'Add Bonds BC' command (BC = bond code).

Measure c) has already been performed with the modified data set *ex2.dat* . Let's have a look at it:

{ EDT^{C} }$_{1,2}$

{ Inputf^{R} }$_{6,2}$

`ex2`

ex2.dat differs from *ex2a.dat* mainly by the lack of comment cards and by the two ASsm cards, which make the following atoms be treated as **I**onic or **M**olecular (= covalent) atoms. A by-product of this measure is that Pb will now be assigned its small (average) ionic radius instead of its large Van-der-Waals radius. Let's now leave the editor and load the file:

{ USE^{L} }$_{1,1}$

This time we will generate a "surface model", i.e., a model of how atoms are arranged on a surface (hkl) of a crystal of $Pb_3(PS_4)_2$. Click

{ FACE }$_{12,1}$

and set the three parameters h, k, and l to 1, 2, and 0; then click { [OK] }. Then we use the BOx card to specify the size of the surface model:

{ BOXa^{L} }$_{20,1}$

Set the three parameters to 0.5, 0.5, and 0; then click { [OK] }; terminate input with another

{ [OK] }

{ quick^{R} }$_{3,4}$

First you may check that the model doesn't contain any bonds to and from Pb (see above).

The first two parameters on the BOx card refer to the "2D unit cell" of the (120) surface (each crystallographic plane has its own 2D unit cell). This unit cell is displayed by cyan, yellow, and violet lines. It has been expanded by 0.5 fractional units along the two unit vectors. The third parameter tells the program to use the default thickness of the surface model which is the characteristic inter-lattice-plane distance d_{120} .

For more information on surface models, see figs. 20 to 24.

5.5. Complete building blocks (molecules)

A shortcoming of the models generated so far lies in the fact that generally some of the peripheral PS_4 tetrahedra are incomplete. The reason is, that atoms outside the BOx or RAdius ranges have been generally cut off strictly. However, as we consider the crystal to be composed of the building blocks Pb^{2+} and PS_4^{3-} we would like to see all the PS_4 groups intact.

In the simplest case we want to see just the *minimum number of complete building blocks,* i.e. those complete "molecules" of which the crystal can be formed by applying symmetry operations and lattice translations. It would be difficult or even impossible to figure out a BOX or RAdius range which just generates this kind of model without any additional atoms.

SCHAKAL offers a procedure which automatically generates this minimum number of complete building blocks by means of the default partition of *build.scf* (available from { cryst }).

This procedure should generally be invoked when you are interested in the structure of a molecule from crystallographic data [*]:

{ cryst^{L} }$_{2,8}$ / { build }$_{11,2}$

You are asked to specify the Input file which contains the crystallographic data set. Click

{ [OK]}

to have the file *ex2.dat* re-loaded. The procedure which follows, ends up in displaying the desired building blocks: One Pb atom and two complete PS_4 groups.

Note that this procedure and the others from *build.scf* (see below) require two conditions to be fulfilled:

a) The automatic bond finding routines employed during the loading of a data set must not generate bonds between any two atoms belonging to two different building blocks (note: this does not pertain to intermolecular H bridges generated by a DCF of the { cryst } group !**)**. I.e., for *ex2.dat* , the procedure would have failed, if any bonds between Pb and PS_4 would be generated by the bond finding routines. This has been excluded by modifying the data set (see section 5.4.). Usually, if "forbidden" bonds are generated, you would have to exclude them with one or more DIst cards to be inserted into the Input file.

b) No two atoms of two different building blocks should have the same name (e.g., C12 and C12). Generally, you simply should take care that all atoms in the data set have different names.

Note: if your structure contains any H bridges, you may use { bld+h } instead of { build } . In this case all inter- and intramolecular H bridges will be added to the complete molecule(s). *Inter*molecular H bridges will end in atoms of a small radius which belong to a molecule in the neighbourhood of the molecule in question. Use {cuthb } (also available from { cryst^{L} }) to transform such intermolecular bonds into broken-off bond sticks. As an example, you may use the file *ex5.dat* (structure of a tripeptide, spacegroup P-1, with two molecules per asymmetric unit and four molecules per unit cell).

---

[*] You also could try to load the crystallographic data set with the 'Use Inputfile 1' command, which lets the program ignore all symmetry information on the file. This strategy will work in some cases; in other cases it won't. Try it with the structure in *ex2.dat* (where it won't work).

5.6. Packing diagrams

Now we want to generate larger sections of the structure composed only of complete building blocks. Drawings of such models are usually called "packing diagrams".
The necessary procedure to be used is also a partition of *build.scf* . It first generates the minimum number of building blocks (as in section 5.5.) and stores the structure obtained on a file which is named *name_p.dat* [*DOS: p_name.dat*], where *name* is the name core of the Input file containing the crystallographic data. *name_p.dat* is then immediately loaded:

{ cryst^{L} }$_{2,8}$

{ bld+p }$_{12,2}$

Again you can answer the request for the file with crystallographic data with

{ [OK] }

to load once more the file *ex2.dat* . The procedure invoked generates the file *ex2_p.dat* [*DOS: p_ex2.dat*] and loads it right after. As *ex2_p.dat* doesn't contain an ENd card, you are asked to input more data cards. We want to have the contents of one unit cell:

{ [OK] } (termination of data input; the program adds a plain 'BOx' card)

The model displays now one unit cell; all building blocks, the gravity centers of which are inside the BOx range (i.e., inside the unit cell, here) have been included and *are complete*.

SCH 92: The data file *name_p.dat* contains an 'ASSM P(ack)' card which changes any BOx card into a PAck card; the latter card is still valid and explained in the manual but doesn't have to be given explicitely.

Next loadings of *ex2_p.dat* , once it has been generated by { bld+p } , can be performed either with 'Use Inputfile' or with { b+p2 } (instead of repeating the whole { bld+p } procedure):

{ cryst^{L} }$_{2,8}$

{ b+p2 }$_{13,2}$

`ex2_p`

The file is loaded. To generate a spherical packing diagram, click

{ RADIUSa }$_{17,1}$

Set the first parameter to 12 and click { [OK] }. Then click another { [OK] } or

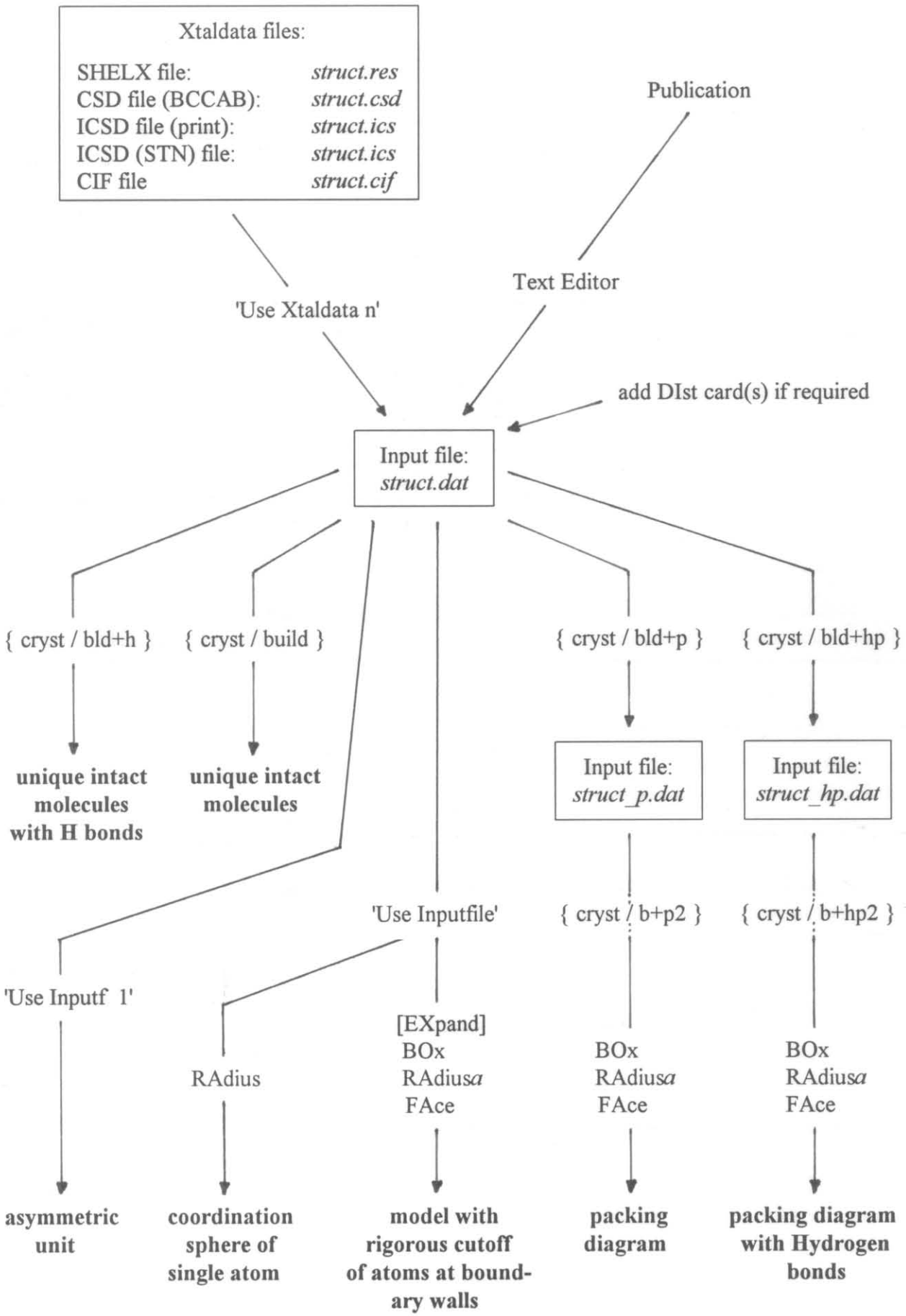
{ END }$_{6,2}$

The spherical model consists of Pb atoms and complete PS_4 tetrahedra.

Note: if your structure contains H bridges you should use the partition { bld+hp } to generate a packing diagram. This generates a file *name_hp.dat* [*DOS: hp_name.dat*] which is immediately loaded to construct the first packing diagram. If you later want to generate more packing diagrams and **if you want to start with *name_hp.dat* , you have to use { b+hp2 } to load this file** (you cannot use 'Use Inputf' in this case !!). Again, *ex5.dat* may be used to practice.

5.7. Summary

The following scheme gives an overview on the different types of crystallographic data files, the different possible results and the commands/actions which connect them:



Epilogue

To learn more about SCHAKAL, you might read the interactive hierarchical manual. To the features which have not (or hardly) been mentioned within the three sessions, belong ...

Adding a model to an existing one ('Add Inputfile' , ––> 113; 954)
The Label Position Storage (LPS; –––> -22.3)
Use of "fonts" ('Use Font' , –––> 245; see also distributed Command file *showf.scf*)
Geometry modifications (–––> 32; 35; 36)
Aligning direct or reciprocal crystallographic lattice vectors (–––> 334; 335)
Modification of the colour palette ('Gen Palette ..' , –––> 42; 957)
Switching shadows on and off (–––> 44; 583)
Intersection traces ('Gen Intersection ..' , –––> 447)
Shading modes ('Set Mode ..' ; –––> 45; fig. 6)
Contrast, intensity of shadows and highlights (–––> 52)
Density of hatching lines; dashed and dotted lines (–––> 54; fig.3)
Drawing area X,Y limits; Z limits for atoms/bonds (–––> 57)
Adding Hydrogen atoms (–––> 61)
Changing atom names ('Chge Name ..', –––> 631)
Assigning numerical group numbers to atoms ('Def Group ..', –––> 668)
Redefining the origin of the internal Cartesian coordinates system ('Def Origin ..', –––> 673)
Filter atom/bond codes ('Defn Target' , –––> 678; 'Filter ..' –––> 68)
Getting a list of (too) close Van-der-Waals contacts (–––> 697; 698)
Assigning bond sticks to edges of coord. polyhedra ('Transf Coordination ..' , –––> 715; fig. 17)
Echoing of command lines, suppressing written output (–––> 74)
Using symmetry operations explicitely (–––> 83)
Transforming crystallographic coordinates (–––> 835)
Photographing screen pictures (–––> 957)

You also might study the contents of the "distributed Command files" which give examples on how to do things. Unfortunately, "device partitions" and "labelled partitions" (–––> 76) make most distributed Command files look rather confusing. Furthermore, some procedures have been written in a comparatively complicated way (to have them work in all cases). Of course, if you write your own Command files, these don't have to contain any partitions at all and you may design them to work for none but your special problems !

Figures

Note: If viewed with „Acrobat Reader“ , use *double-page* display for this part.

If not stated otherwise, the figures within this manual (except fig. 1) have been generated on an HP LaserJet III. If not stated otherwise, a resolution of 150 dpi (which gives good results on Xeroxing) was used.

Fig. 1: Internal Cartesian Coordinate System and Physical Drawing Area

SCHAKAL's internal atom coordinates refer to the internal Cartesian coordinate system (CCS, –—> 673). The physical drawing area (PDA) is that part of the screen or printer paper onto which something can be drawn.

Dimensions of the PDA are 30 x 20 cm, here. The current origin of the CCS is positioned at X = 14 cm / Y = 9 cm. This has been done

a) automatically by the program (if the model scale factor is not fixed –—> 513)
or
b) by a 'Set Origin 14 9' , a 'Set Origin .47 .45' , or a plain 'Set Origin' command (–—> 57), if the scale factor is fixed presently.

Arrows indicate sense of rotations invoked by 'Rot..' commands (–—> 32), if the rotation angle is given as a positive number.

Fig. 2: Default orientation of the unit cell

Unit cell corners/edges are added automatically to the program's atom/bond lists when a data set is loaded. With respect to drawing they are mostly in a "switched-off" state per default. Switch them on by 'Gen Unit..' (–—> 48).

This drawing demonstrates the usual default orientation of a triclinic unit cell relative to the internal Cartesian coordinates system (CCS, –—> 673). The names of the 8 corners (automatically assigned by the program) are given.

On a colour device, per default, the a edge will be in grey, the b edge will be in red, and the c edge will be in green. All other edges will be blue.

Note: If a "surface model" (see fig. 20) has been generated, the default orientation will be modified by the program after data processing such, that the (hkl) plane in question comes to lie parallel to the drawing plane.

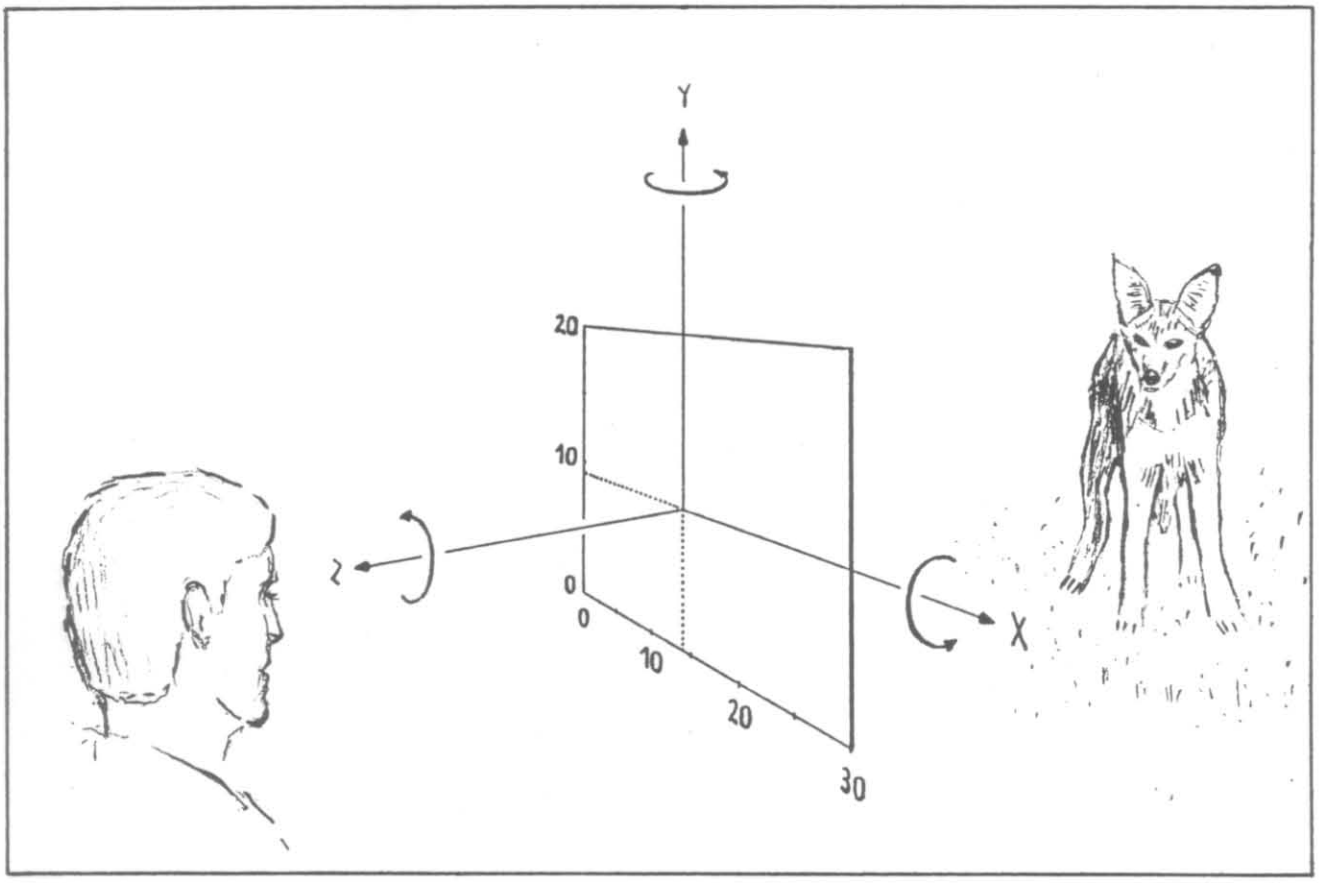


Fig. 1: Internal Cartesian Coordinates System and Physical Drawing Area

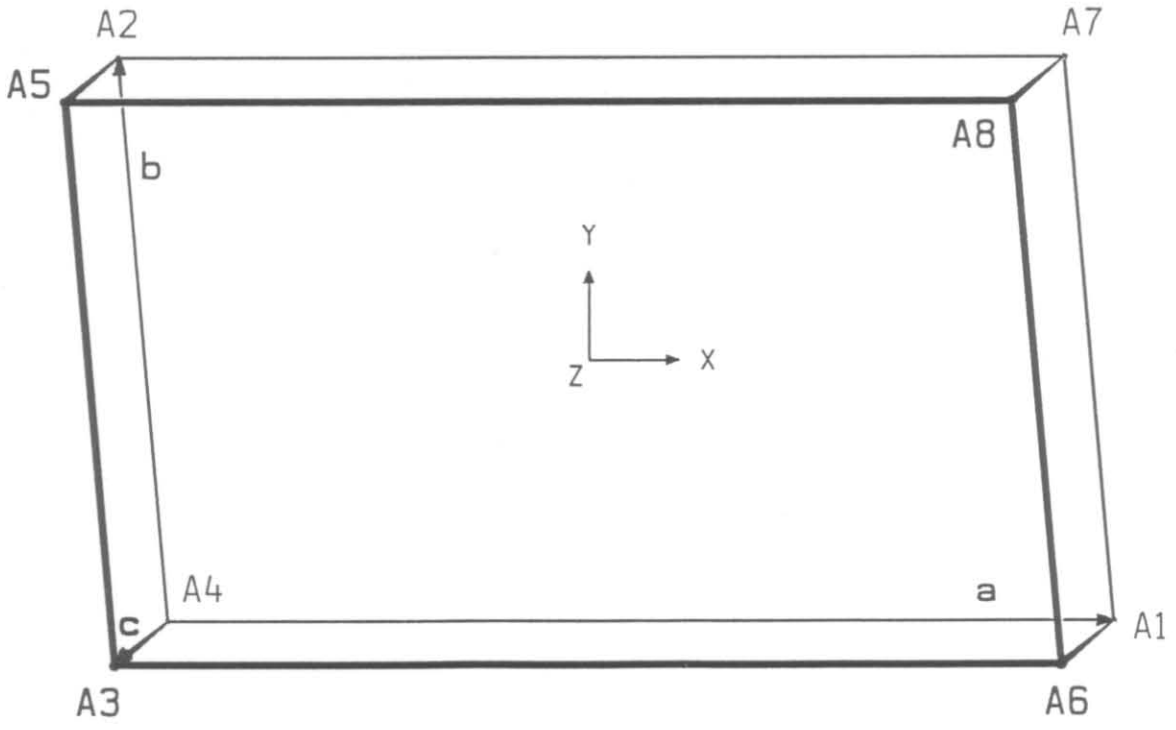


Fig. 2: Default orientation of the unit cell

Fig. 3: Illustration of some terms used in the manuals

The terms illustrated in fig.3 are related to the following commands:

Dashed lines
('Set Dash..', –––> 545)

increased line width
'Brd..', –––> 55)
reduced griddensity
('Set Griddens..', –––> 54)

hatching lines (parallels)

('Gen Hatch..', –––> 444)

north pole

('Set Pole..', –––> 541)

meridians
('Gen Merid..', –––> 445)

foreign shadow
self-made shadow
'Set Shadowl..', –––> 583, 44)

grids
('Gen Grids..', –––> 446)

highlight
('Chge Darkf..', –––> 646,
'Modf Highl..', –––> 527)

fragmentated bond
('Chge Fragm..', –––> 657)

regular dithering

('Set Mode 3n',–––> 453)

broken-off bond

('Trnsf Expnd..', –––> 712)

marginal gap
('Mgn Gaps..', –––> 515)

random dithering
('Set Mode 3n', –––> 453)

group designator
('Wrt All..', –––> 222)

plain colour fill

('Gen Shad -1', –––> 443)

outside labelling

('Set Writopt..', –––> 461)

alternative bond display mode

('Chge Bond..', –––> 658)

inside labelling

('Set Writopt..', –––> 461)

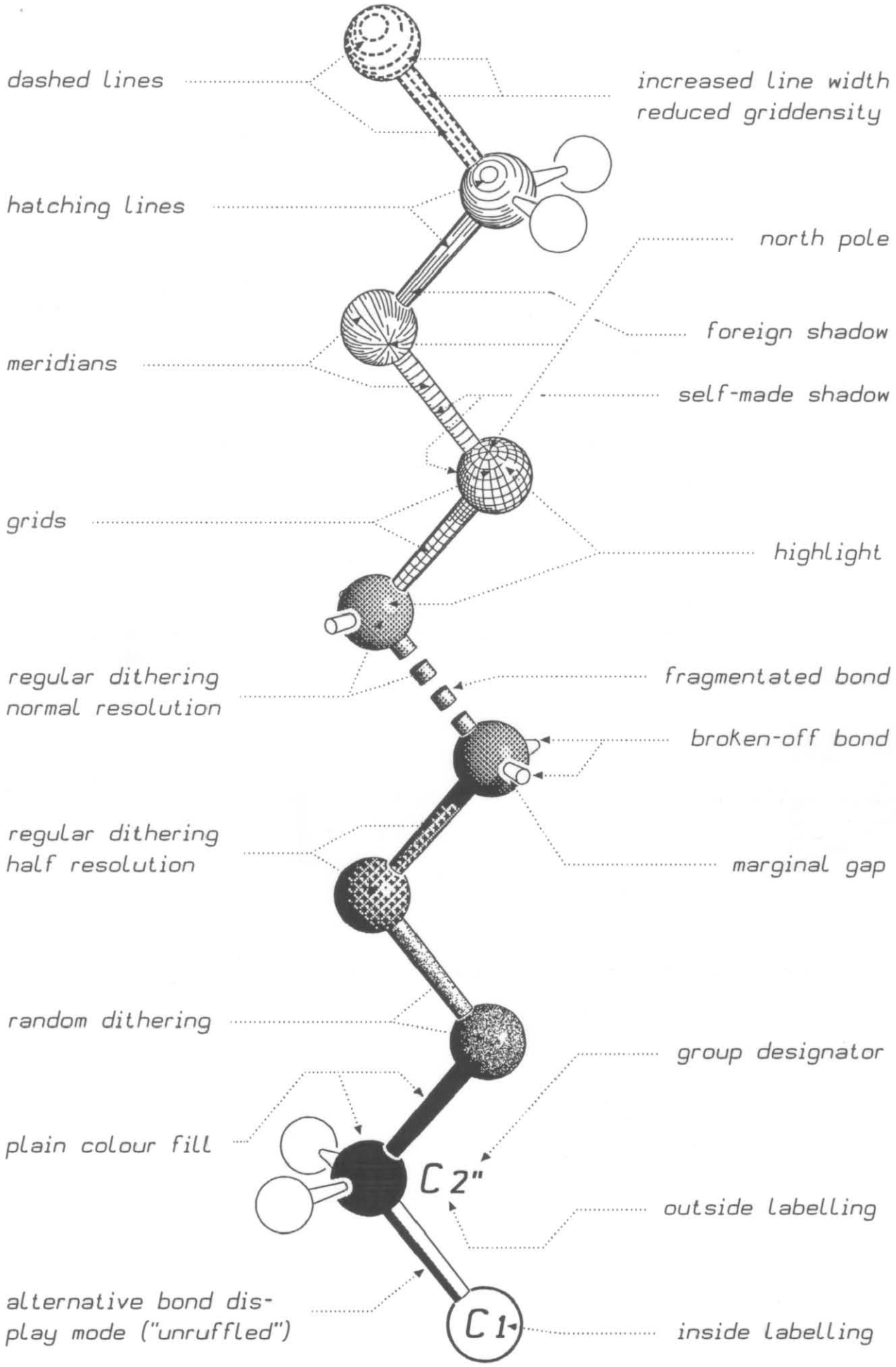


Fig. 3: Illustration of some terms used in the manuals

Fig. 4: Some examples for hatching by parallels, meridians, and grids

This drawing tries to illustrate, how the appearance of hatching patterns invoked by 'Gen Hatch p', 'Gen Merid p', and 'Gen Grids p' (––> 44) is influenced by ...

a) the parameter p
b) 'Set Pole ..' settings ("north pole" position, –––> 541)
c) 'Set Light ..' settings (light source position, –––> 528)

Via 'Set Griddens 4' (–––> 543), a griddensity of 4 lines per cm has been set for this drawing. To draw the sphere in the lower right corner, this griddensity has been reduced by 'Set Griddens *.5' (–––> 544)

The set of commands given under each sphere would not be sufficient to generate the drawing above it: To achieve this, you would have to draw the sphere's outline via 'G O : X' first, then use the specified commands, and finally give another 'X' to invoke drawing of the hatching pattern.

The white spots shown by most of the spheres ("highlights") could have been avoided if a 'Modf Highl 0 0' (–––> 527) would have been given before.

A number of predefined hatching patterns is accessible by the 'Xqt Pattern ..' command (see fig.s 9 and 10).

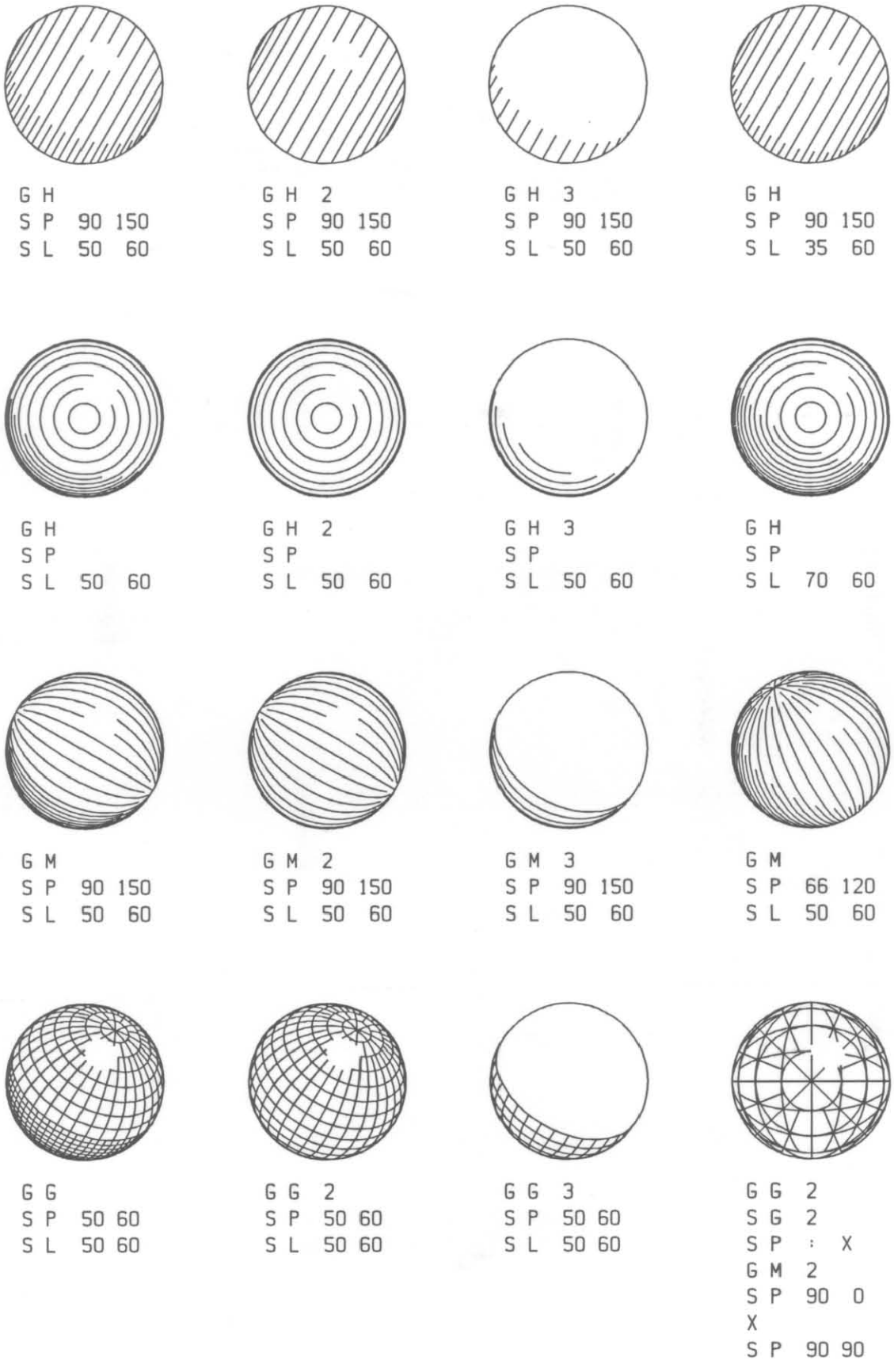


Fig. 4: Some examples for hatching by parallels, meridians, and grids

Fig. 5: Darkening functions

This drawing illustrates the 30 basic darkening functions (to be selected with 'Chge Darkf n ir', –––> 646) and the corresponding numbers n ("darkening function number", DFN).The drawing has been generated using shading mode 33 ('Set Mode 33', –––> 453).

The function with n = 11 (reserved for edges and 3D lines) is not shown.

Instead of the n = 130 function (which hardly differs from that one with n = 70), the special n = 12 function is included.

Set i = 1 or 2 to weaken or eliminate the highlight (this adds 20 or 40 to the DFN)

Set r = 1 or 2 to make the surface rougher (this adds 240 or 480 to the DFN). This will not affect shading by dots (–––> 451).

Per default, the following DFNs are assigned to the following selected types of atoms:

Metals	generally DFN = 1 (with some exceptions). DFN 1 should be used in connection with low-saturated colours (–––> 538.2) on a multi-colour device (screen).
C	DFN 26 (= 6 + 1 * 20; i.e., 6 with weakened highlight)
H	DFN 122
O, N	DFN 5
bond sticks	DFN 4

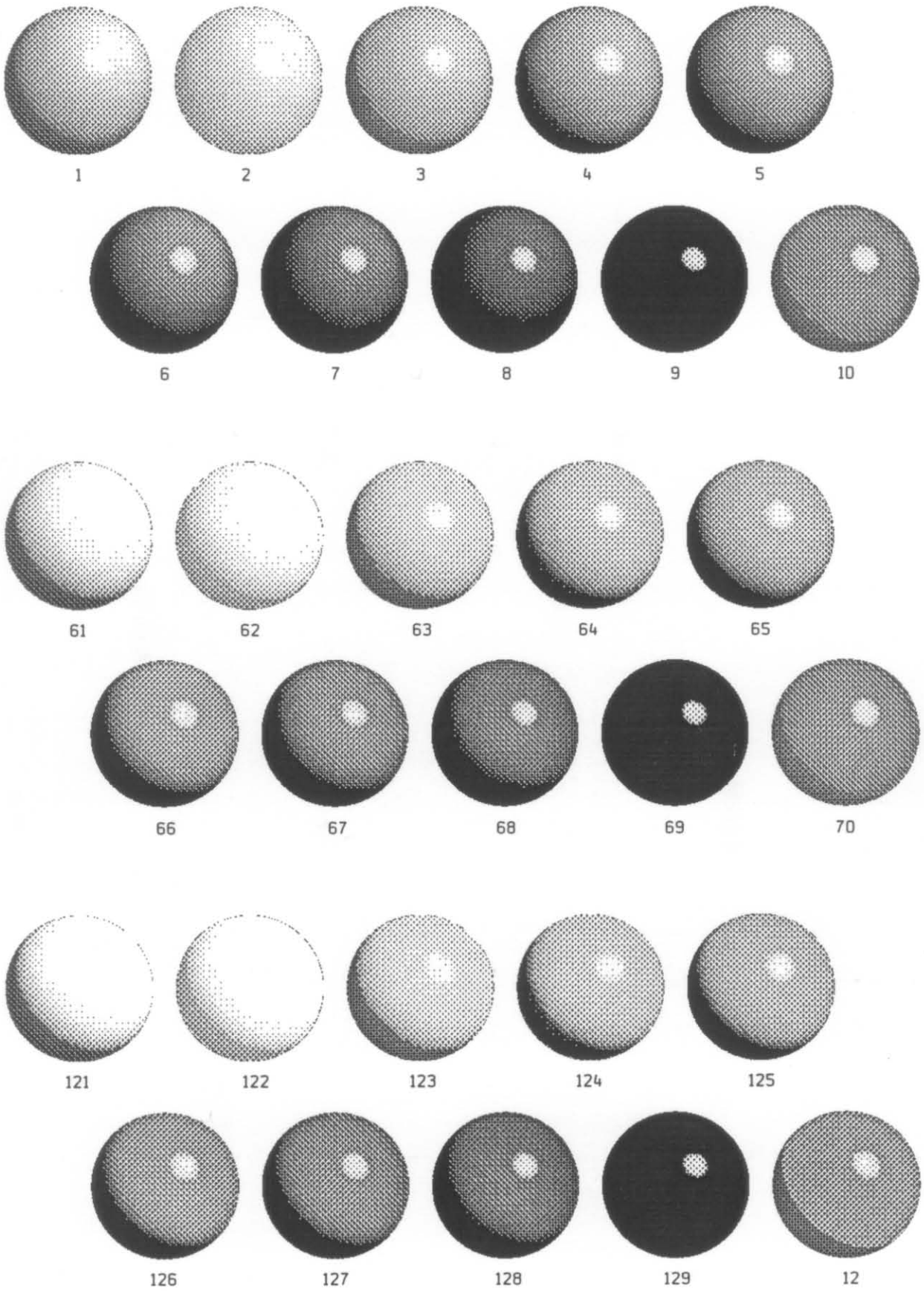


Fig. 5: Darkening functions

Fig. 6: Some shading modes

This drawing illustrates some of the shading modes which may be controlled via 'Set Mode mn' or 'Set Mode m n' (–—> 45). The 'Set Mode 1n' modes (dots/colour steps) which affect drawings on multi-colour devices only, are not illustrated, here. The 'Set Mode 2n' modes (bond shading) which affect the geometrical way in which bonds are shaded, are also not illustrated, here.

The 'Set Mode 3n' modes affect the way in which darkness variations are visualized by dots or colour steps ("dithering"). Modes 30 and 31 are called "random dithering", modes 32 - 34 are called "regular dithering (normal resolution)", and modes 35 - 37 are called "regular dithering (half resolution)". Mode 32 is not very appropriate for use on a laser printer (but see below). The corresponding sphere would look better, though, if a decreased darkness would have been selected before (e.g., by 'Modf Darkn .7').

If a multi-colour device is switched on, the meanings of the 3n modes (n > 0) differ partially from the meanings illustrated here, if shading by colour steps is switched on (which is the default for a multi-colour device).

The 'Set Mode 4n' modes control calculation of darkness. Mode 40 tries to calculate "realistic" darkness variations; modes 41 and 42 calculate purely depth-dependent darkness (the latter dependent on the 'Set Zlim..' setting (–—> 578)).

The 'Set Mode 5n' modes control shading of the peripheral region of an object, i.e, n is a measure for the width of outlines which are added to the darkening function.

The three spheres in the last row have not been drawn by means of a laser printer but by means of an HP7475A plotter (device no. 3), using shading mode 32. On a pen plotter, mode 32 is the only one which should be used for regular dithering. A DFN of 5 (see fig. 5) was assigned to the atom, here (instead of 1, above).

For the first sphere, default (= "theoretical") settings of resolution were used. The pen had already been used several times, i.e., its tip was broadened slightly. Therefore, single dots are not or hardly resolved.

For the other two spheres, the default value of 13 plotter coordinate units per pen width was increased to 15 or 17 by the command written underneath. Single dots are better resolved, in these cases. The corresponding modifications could also have been achieved by the command 'Gen Device ! 303 .033 n' (n = 15 or 17). Given in the initializing Command file *sch97.ini* , this command would make the modification permanent.

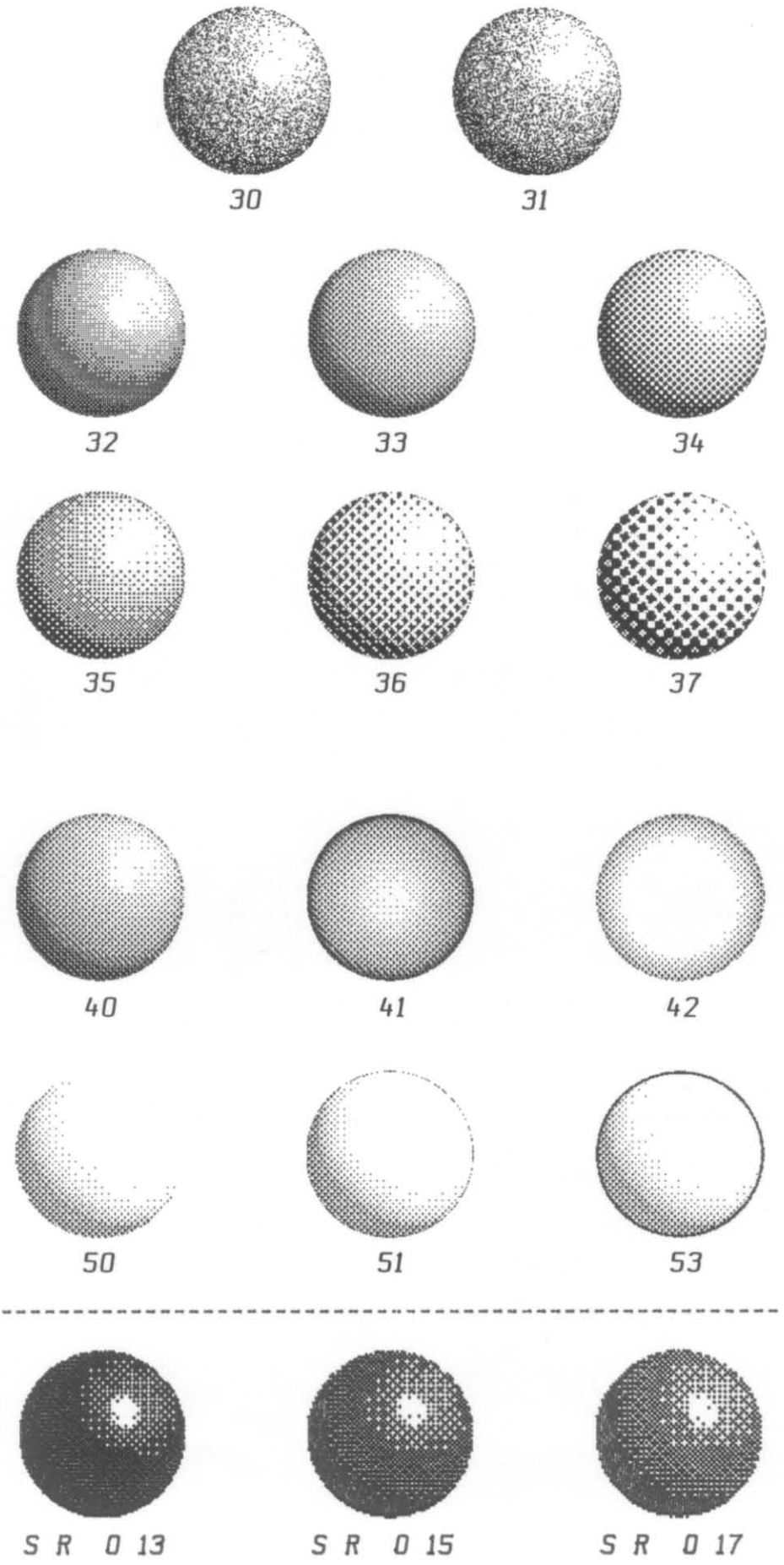


Fig. 6: Some shading modes

Fig. 7: Lines, arrows, and polygons

This drawing illustrates most of SCHAKAL's (poor) extra graphic facilities, where "extra" means: "besides the facilities to draw structure models".

The commands which have been used to generate the different parts of the drawing are given on the left side. Some commands appear in square brackets either because they reset default conditions, or because they appear already in a previous line.

The following table gives the commands in un-abbreviated form as well as the manual index and the group to wich the commands belong in the graphical user interface (GUI):

Command	Manual	GUI
Broaden Lines	(–> 556)	{ MGN }
Set Dashing	(–> 545)	{ SET }
Xqt Line	(–> 251)	{ XQT }
Xqt Arrow	(–> 254)	{ XQT }
Set Mode	(–> 45)	{ G/S }
Modify Pellucidity	(–> 523)	{ MGN }
Xqt Region	(–> 26)	{ XQT }

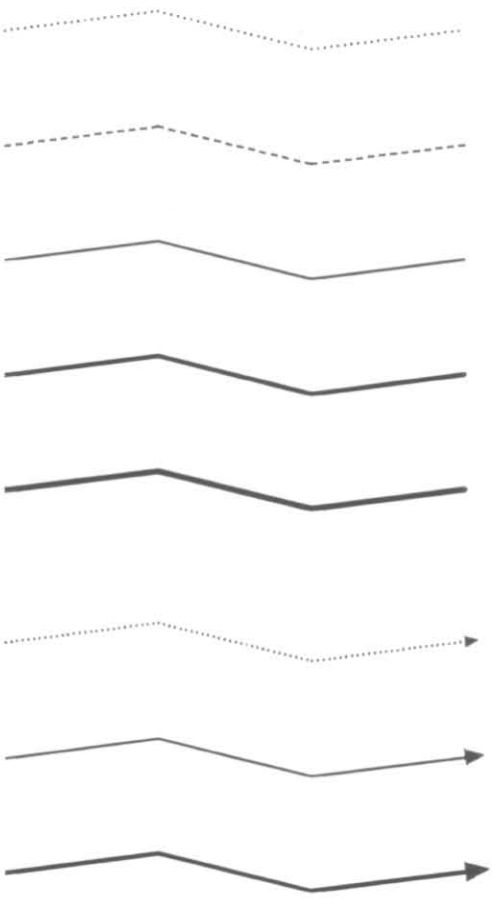
Note: In some cases, arrow tips are a d d e d to a line, such that they prolongate the line. For more info, see –> 254.

Note: When trying to generate a slightly shaded polygon for a high-resolution printer, it may happen, that no shading is to be seen on the monitoring screen becaue of a disadvantageous interference of dot drawing with the pixel pattern of the screen.

Note: Lines, arrows and regions may be defined by coordinates, by mouse clicks, or by atom positions.

```
B L 1 : S D 6 1 : X L
B L 1 : S D 10 5 : X L
B L 1 :  [ S D : ]  X L
B L 3 : X L
B L 5 : X L

B L 1 6 : S D 5 1 : X A
B L 1 10 :  [ S D : ]  X A
B L 3 10 : X A
```



```
B L 1 : S M 34 : M P 14 : X R
[ S M 34 : ]  M P 12 : X R
[ S M 34 : ]  M P  8 : X R
[ S M 34 : ]  M P  4 : X R
[ M P : ]  X R
```

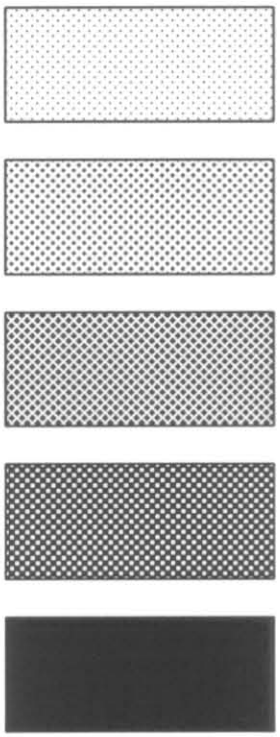


Fig. 7: Lines, arrows, and polygons

Fig. 8: Graphic text

The text in the upper part has been produced by executing an older version of the distributed Writing file *ex.txt* via the command 'Use Writingfile' (––> 16):

```
Use Writingfile
ex
```

The first text line was positioned by means of the mouse. Then, writing of the residual text was invoked by pressing <ESC>.

Note: Graphical text may be positioned by coordinates, by mouse clicks, or relative to atom positions.

Text in the lower part has been generated via 'Wrt Text..' (––> 241). It demonstrates the effect of the commands 'Mgn Text..' (––> 564), 'Brd Text..' (––> 557), and 'Set Inscr..' (–––> 466). These commands belong to the { EDT } group in the GUI.

Note, that the 'S I mn' commands (which could have also been written as 'S I m n') refer to the case where you would generate the corresponding text on the screen (which displays a "landscape" format, usually). To generate the drawing on the laser printer (in "portrait" format), the value of 1 had to be added to m , actually (cf. section 4.4.).

0123456789 -+*/=.,;:-'" !? %#$ <>()[]{}
ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ
abcdefghijklmnopqrstuvwxyz äöüß
αβχδεφγηιψκλμνωπθρσστξυζ ΔΓΛΠΣΩΞ
ÅÄåäáàâãëéèěêîíìñšÿýÓóò ¥Ø≠≡Ççı

$Fe_4(CO)_{12}[\eta^6\text{-}NC\text{-}C_6H_5]$ $(\bar{2}\ 0\ \bar{3})$

MT 1 : BT 1 : SI

MT2 MT21 MT12

BT4 **BT41** **BT14**

SI3 *SI6* *SI9*

SI10 OZIS SI30

MT 1.25 : BT 64 : SI 4

Fig. 8: Graphic text

Fig. 9: Predefined 3D patterns

This drawing illustrates the 30 predefined "3D patterns" which are contained in the distributed command file *patt.scf* .

The number underneath a sphere is the corresponding pattern number (i.e., the partition label of the corresponding labelled partition of file *patt.scf*).

Patterns are activated by means of the 'Xqt Pattern' command (–—> 217). For example,

```
Xqt Pattern  21  N          (alternative syntax: 'Xqt Pattern 0 21 N')
```

applies pattern 21 to all Nitrogen atoms.
Bonds can also be shaded or hatched with this command. An illustration is not given for this case, however.

Predefined patterns generally don't show "foreign" shadows (see fig. 3). This is, because the commands in file *patt.scf* have been designed accordingly.

Patterns with hatching lines may be modified by the following commands:

```
Set Griddens  *p          (controls density of hatching lines, –—> 544)
Brdn All  p [q [n]]       (controls the line width used for outlines and
                           hatching lines, –—> 554;  fig.15)
Modf Highl  p q           (q controls the size of highlights, –—> 527)
```

Patterns with dot shading use shading modes 33 or 34 (–—> 453). They may be modified by the following commands (–—> 52):

```
Modf Darkn  ..            (modifies the overall darkness)
Modf Shadows  s           (modifies the darkness of the shadow area)
Modf Highl  p q           (modifies intensity and size of highlights)
```

See also last section of text discussing fig. 10 .

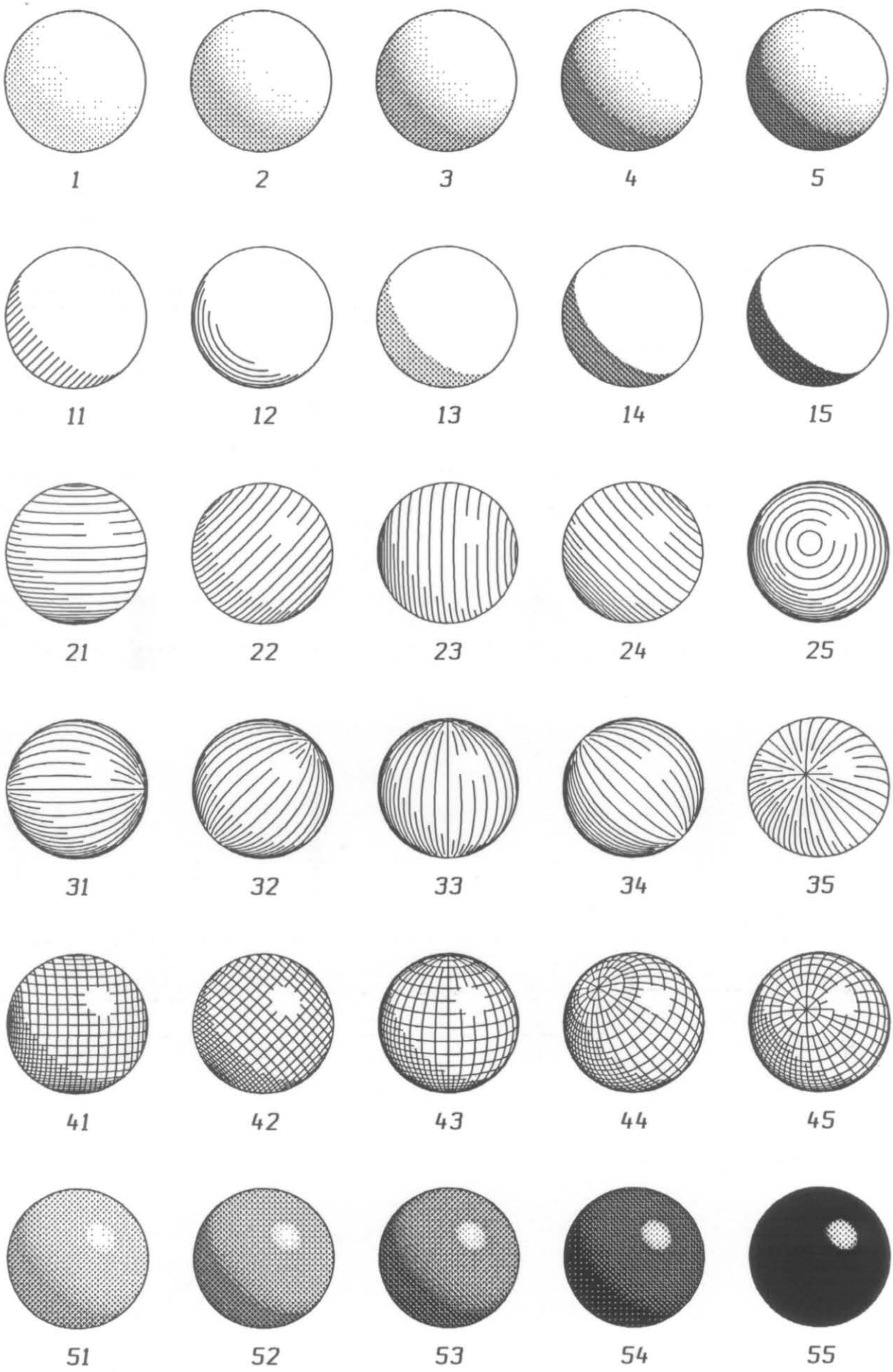


Fig. 9: Predefined 3D patterns

Fig. 10: Predefined 2D patterns

This drawing illustrates the 30 "2D patterns" contained in the distributed command file *patt.scf* . Generally, 2D pattern no. 1nm resembles closely the corresponding 3D pattern no. nm (see fig. 9). However, 2D patterns show neither a highlight nor a "self-made" shadow. A 2D pattern can be selected either with 'Xqt Pattern 1mn' or with 'Xqt Pattern 1 mn' .

Patterns with hatching lines may be modified by the following commands:

`Set Griddens *p`	(controls the density of hatching lines, –—> 544)
`Brdn All p [q [n]]`	(controls the line width used for outlines and hatching lines (–—> 554; fig.15)

Patterns with dot shading use shading modes 33 or 34 (–—> 453). They may be modified by the following command (–—> 521):

`Modf Darkn ..`	(modifies the overall darkness)

Note: Patterns with dot shading are not appropriate for a pen plotter (because of the large relative size of a single dot). Here, you should use pattern 2mn instead of pattern 1mn , as patterns 2mn base on shading mode 32 instead of 33 or 34 (see text describing fig. 6).

Note: If the atom circles are rather small (for example, because the structure contains a large number of atoms), patterns may look rather different as compared to fig. 10 (or 9), simply, because only a small number of lines or dots fit into the circle (or stick) area. On a laser printer, you may use a higher resolution (300 dpi instead of 150 dpi) in this case.

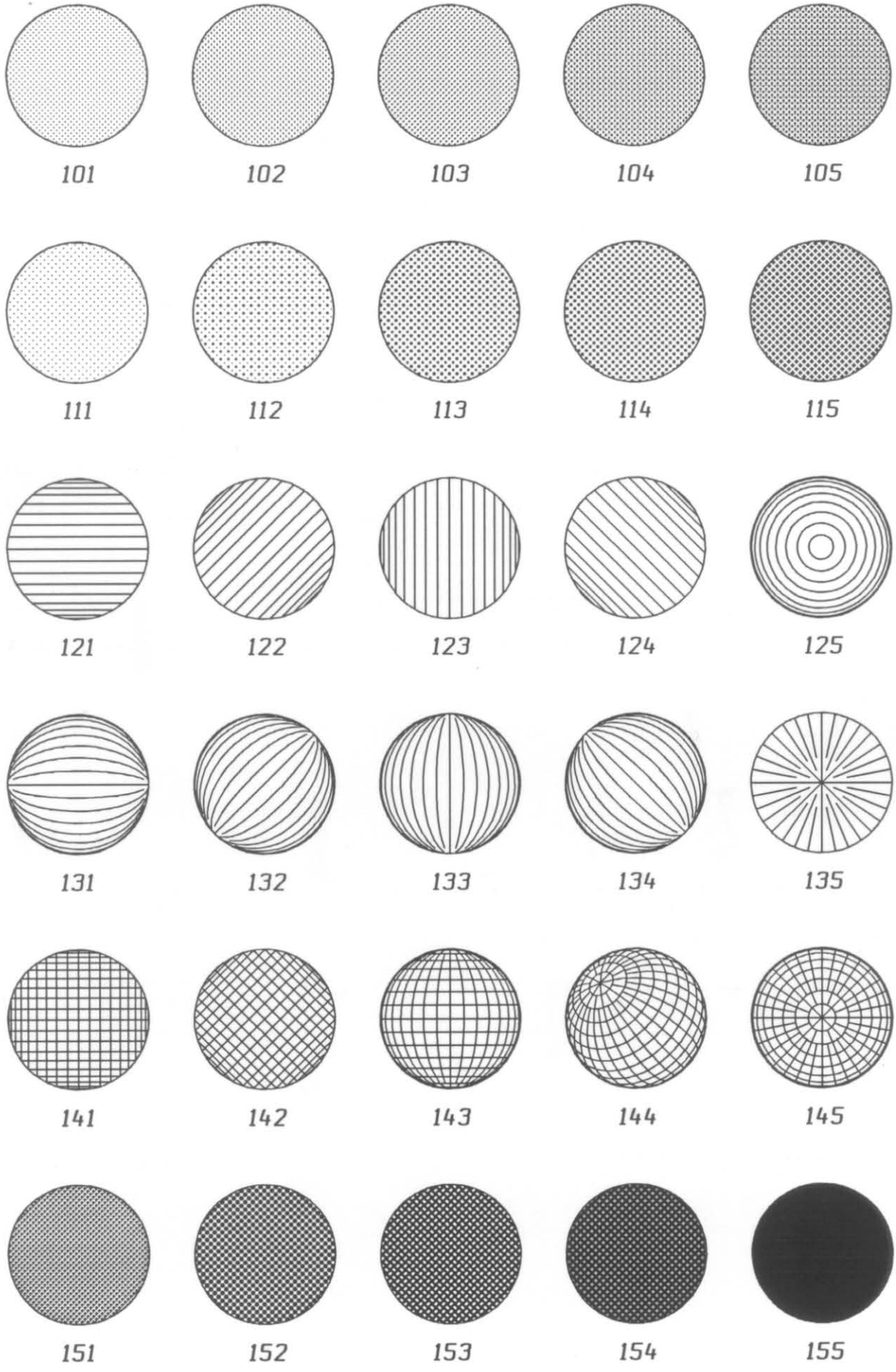


Fig. 10: Predefined 2D patterns

Fig. 11: The Penicillin molecule

This drawing of the ball-and-stick model of the penicillin molecule may serve as a demonstration for the use of 3D and 2D patterns (see fig.s 9, 10). After adjusting the orientation of the model, the drawing has been generated by the following commands:

```
Chge Thickn  *.5 H=nn          (multiply thickness of all bonds to or
                                from H atoms by 0.5, –-> 652)
Chge Thickn  *1.5 nn=nn        (multiply all stick thicknesses by 1.5)
Defn $  N1 C2 C3 S C6 C7 C8    (define group $ by atoms of bicyclic
                                ring system –-> 664)
Defn $  $  C11-C16 (H          (add phenyl C atoms to group $ )
Genr Outl : Xqt                (draw outlines)

Xqt Pattn 11 H S               (use pattern no. 11 for H and S atoms)
Xqt Pattn 41 C                 (etc.)
Xqt Pattn 21 O
Xqt Pattn 33 N
Xqt Pattn 52 nn=nn (h=nn $=$   (use pattern no. 52 for all bonds ex-
                                cept those ones between the $ atoms
                                and those ones to or from H atoms)
Xqt Pattn 155  $=$             (etc.)
Xqt Pattn  21  h=nn
```

To label the atoms, the molecule has been divided into four regions (separated by the dotted lines). For each region, an individual set of 'Set Writeopt..' / 'Set Inscription..' commands (–-> 46) has been used. The 4 different sets of commands are shown within the four different regions:

The second parameter of the 'Set Writeopt' command controls the arrangement of element symbols and numbers (horizontal/vertical) and the use of parentheses/ brackets.

'Set Inscr' controls the slantedness of the characters. As was explained for fig. 8, a value of 1 was added to the 'S I' parameter m , actually, to generate the drawing in "portrait" format.

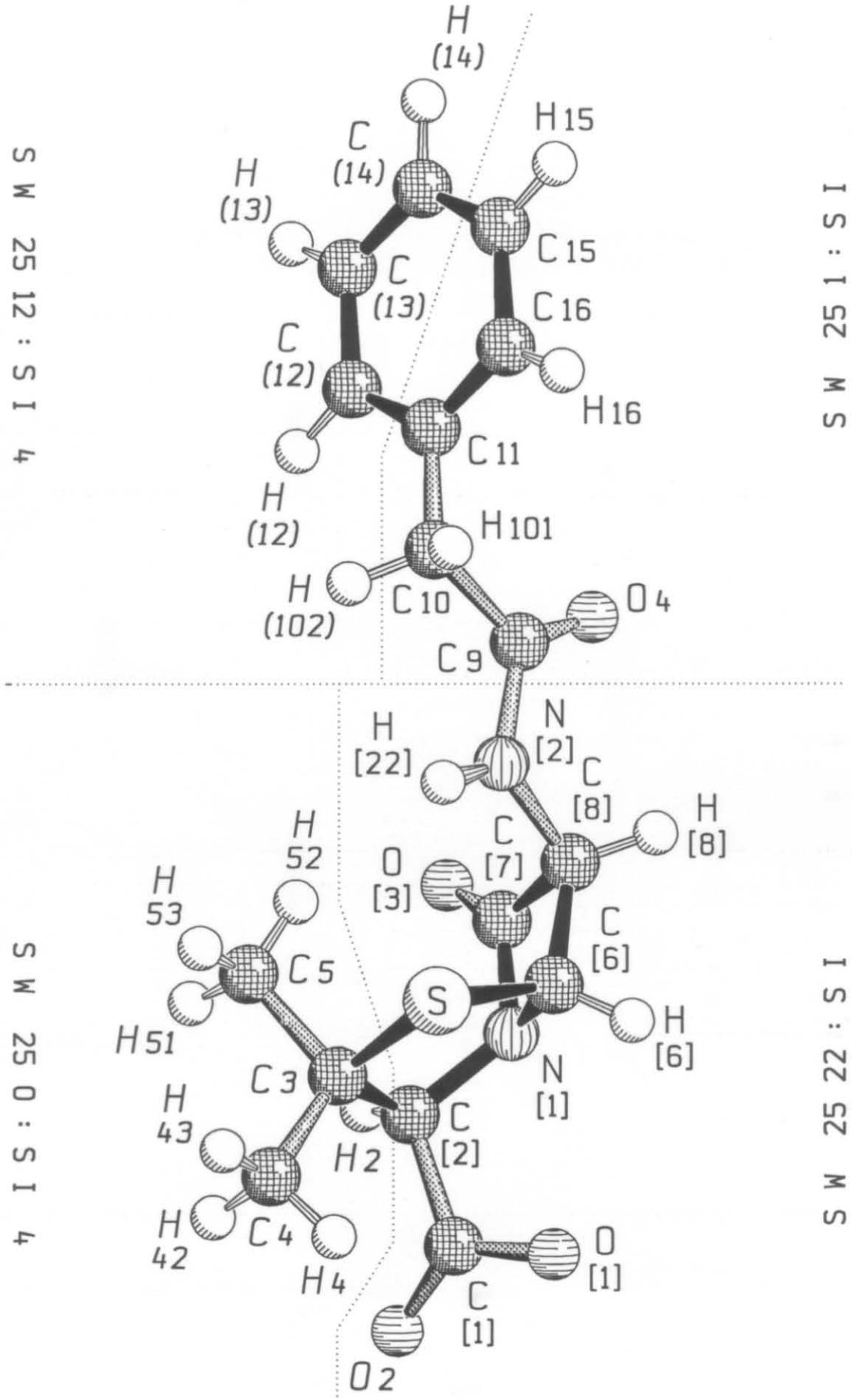


Fig. 11: The Penicillin molecule

Fig. 12: The ($C_{20}H_{24}$) Zr Cl_2 complex (I)

The drawing of this organometallic catalyst may serve as an example for several measures to make the drawing of a molecule more comprehensible (furthermore, it demonstrates labelling with names including "group designators" (––> 222)):

a) Reducing the radius of H atoms and the thickness of the bond sticks connecting them with other atoms (e.g., by 'Chge Size *.7 H H=nn') generally leads to more transparency and clarity.

b) It is recommended to generally enhance the bonds of ring systems either by colour or by thickness or by other graphic means (see also fig. 11) . Here, the 10 "aromatic" bonds within the two 5-membered rings have been made thicker via the following procedure:

```
Def $     C1 C2 C3 C7a          (group $ is defined by all 10 ring atoms, –––> 664)
Chg Thickn  *1.5  $=$           (multiply stick thicknesses by 1.5, –––> 652)
```

c) The large number of bonds generated per default between metal atoms and the atoms of organic π-bonded rings should be replaced by single "pseudo-bonds".This can be done with the DCF *pilig.scf* , wich is available on the GUI from { other^{L} } / { pi-lig } . This DCF asks you first to click a "central atom" and then to click all the π-bonded atoms of one π-ligand bonded to it.

A labelling procedure was started by 'Wrt All..' (–––> 222) and then stopped via <ESC>. This let the program know that any 'W #..' command contained in the Command file *p.scf* (below) was to be replaced by a 'W A..' command ('W A..' labels atoms with names plus group indicators).

Originally, all atoms belonged (per default) to the numerical group 1. However, atoms in the upper half of the drawing have been assigned to the numerical group 2 (to have an apostrophe added to the name as a group designator) by the following commands:

```
Filt Y   0.2 100          (Set a filter for internal Y coordinates, –––> 683)
Def Group   2 nn          (assign all atoms in the upper half to group no. 2, –––> 668)
Filt                      (switch filter off)
```

The drawing of fig. 12a was then generated with the help of the following commands:

```
Set View  30              (perspective distortion, –––> 476)
Set Inscr ! 1 4           (rotate model and establish a vertical writing direction to
                           have drawing in portrait format)
Use Commandf              (execute Command file p.scf incl. ´Wrt All..' labelling for
pattn  s92+w               non-Hydrogen atoms)
Trl Inscr  C1 C7a (gg1    (move two unsatisfyingly positioned labels, –––> 229)
```

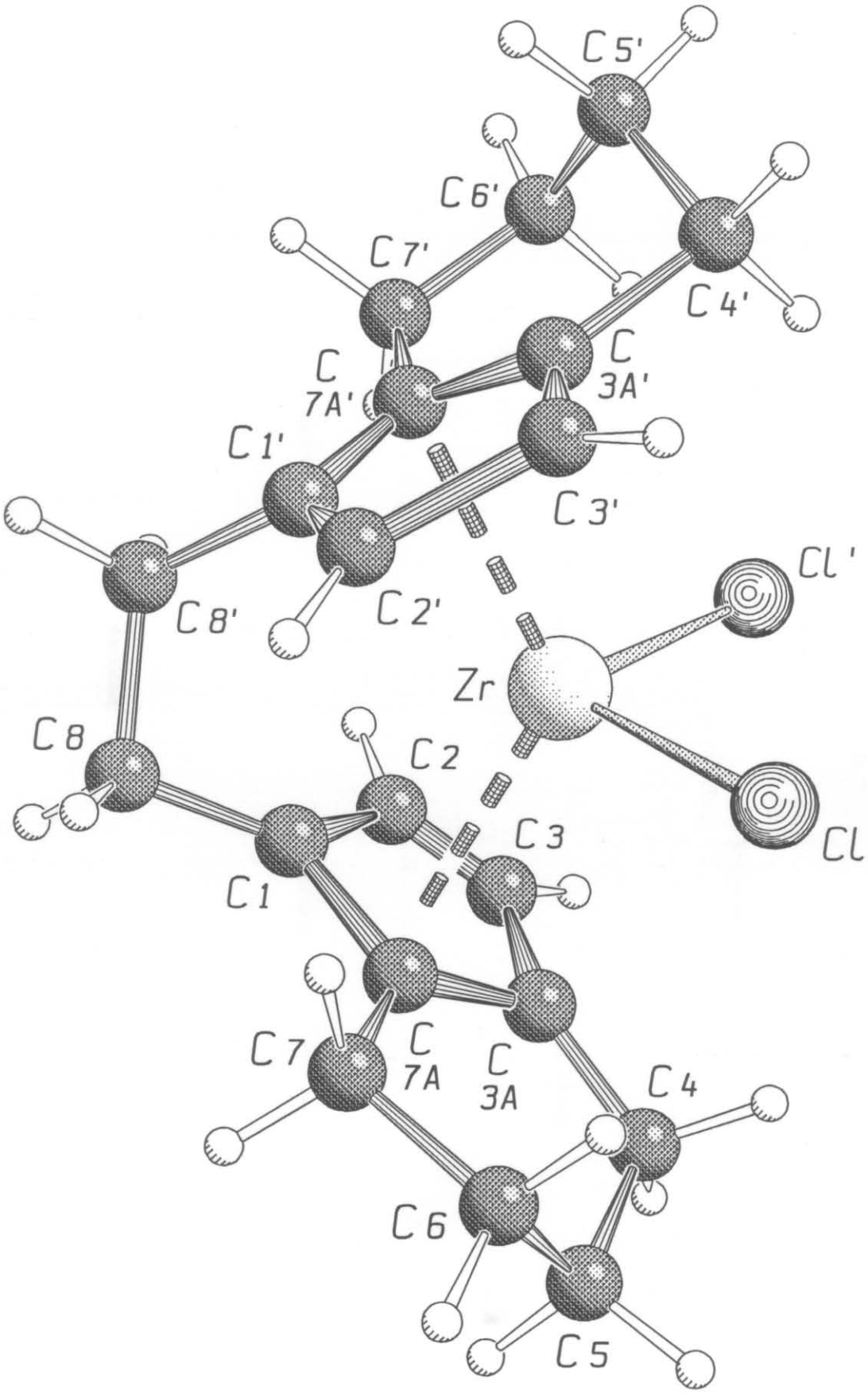


Fig. 12: The $(C_{20}H_{24})$ Zr Cl_2 complex (I)

Fig. 13: The $(C_{20}H_{24})$ Zr Cl_2 complex (II)

Fig. 13 was generated right after fig. 12 (same program run). First, the darkening function numbers assigned to several atoms/bonds (–—> 646) were modified by the following commands (for meaning of $, see fig.12, section b):

```
Chge Darkn  3 Cl,  5 C,  9 $
Chge Darkn  7 nn=nn (H=nn Zr=nn
Chge Darkn  1 $=$
```

Then the drawing was generated by the following commands

`Set Tapering  .5`	(less exaggerated tapering of bond sticks, –—> 581)
`Use Commandfile shade`	(execute command file *s.scf* without labelling)
`Mgnf Inscr  -1.5`	(increased size of characters, –—> 561)
`Brdn Inscr  3`	(increased boldness, –—> 551)
`Wrte Symbols Zr Cl`	(label selected atoms with chemical element symbols, –—> 223)

Note: Command file *s.scf* (as well as some other distributed Command files) increases the contrast between different darkening functions when used for the laser printer.

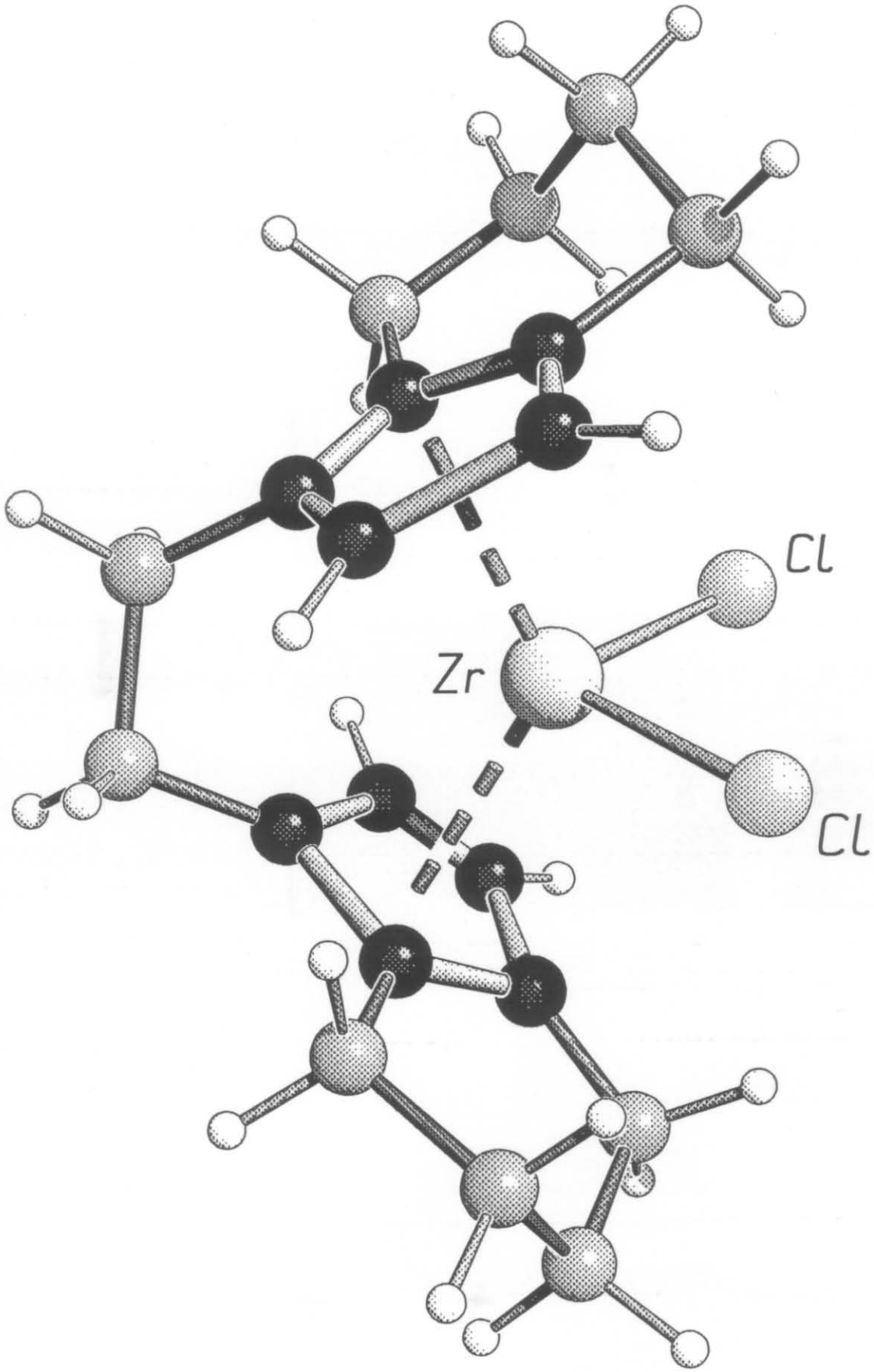


Fig. 13: The $(C_{20}H_{24})$ Zr Cl_2 complex (II)

Fig. 14: The ($C_{20}H_{24}$) Zr Cl_2 complex (III): stereo labelling

This drawing has been generated by means of the DCFs { lines / lstreo } [*ls.scf ;* upper part] and { shade / lstreo } [*las.scf* ; lower part]. Both parts are stereo drawings.

The drawings were generated by the following commands:

{ }$_{3,1.5}$	(switch a n y erasing of drawings off, –––> 724. This is to have two Command file drawings generated on the same sheet)
`Set Inscr 0 4`	(use slanted characters [actually, to have the drawing in portrait format, 'S I ! 1 4' was used; see fig. 8)
`Mgnf Inscr 1.5`	(use positive inscription size parameter !, –––> 561)
`Set Griddens *.8`	(slightly decreased line density for hatching –––> 544)
`[Mgnf Model]`	(switch to variable model scale factor [default], –––> 512)
`Set Xlim 0 .45`	(set X drawing limits to first half of drawing area, –––> 577; note that at the time, when the drawing was generated, it was actually rotated by 90 degrees on the screen. Therefore the X axis was parallel to a line which is now vertical when you look at this drawing)
`Defn $`	(group $ is defined by all atoms (–––> 664), i.e., all atoms will be labelled by *ls.scf* and *las.scf* , below *)*
{ lines^{L} }$_{2,4}$ { lstreo }$_{14,2}$	(execute Command file *ls.scf*)
`Set Xlim .45 .9`	(set X drawing limits to the second half of the drawing area)
{ shade^{L} }$_{3,5}$ { lstreo }$_{21,2}$	(execute Command file *las.scf*)
{ K }$_{3,1.5}$	(switch back to automatic screen erasing, –––> 722)

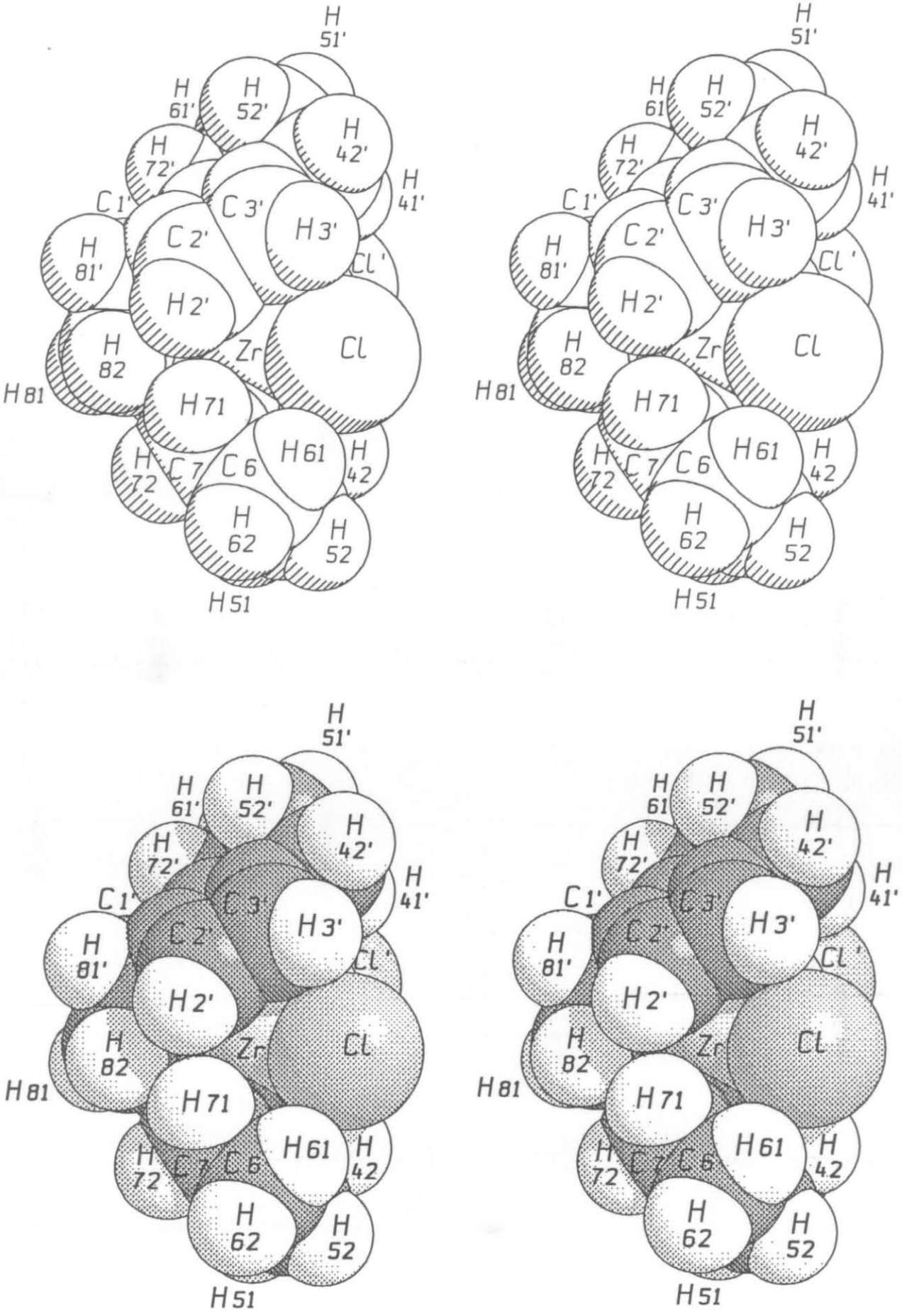


Fig. 14: The $(C_{20}H_{24})$ Zr Cl_2 complex (III): stereo labelling

Fig. 15: The $AgGaS_2$ structure

This drawing demonstrates how to generate an arbitrary section of a crystal structure with boundary walls parallel to the unit cell walls (BOx, –—> 841), and how to add the corners of more than one unit cell to the atom list (ADd, –—> 863).

Furthermore, this drawing illustrates different modes of interpolation for depth-dependent features like darkness ('M D..', –—> 522), pellucidity ('M P..',–—> 523) and line width ('B A.',–—> 554). The following data set was used to generate the model:

```
TITL AgGaS_2_  space group I4&-2d
CELL  5.750    5.750   10.290
ASSM  Ionic
ATOM   Ag      0         0         0
ATOM   Ga      0         0        1/2
ATOM   S      .2908     1/4       1/8
SPGR   I -4 2 d
```

When loading the data set, the following additional data cards were added "by hand":

```
ADD
BOX  0 2  0 1  0 1
END
```

After adjusting radii and stick thicknesses and performing the necessary rotations, the following commands were given:

`Mgnf Model 1`	(scale factor 1 cm / A)
`Mgnf Gaps -2`	(fixed size of marginal gaps, –—> 515)
`Set Tapering .5`	(reduced bond stick tapering, –—> 581)
`Set View 150`	(perspective distortion, –—> 476)
`Set Inscr ! 1 0`	(generate "portrait" format, –—> 466)

The following 'Broaden All' commands were then used to establish different depth-dependence of the line width (–—> 554) for the four different drawings (differences between the two upper drawings are only small):

upper left: `Brdn All  4 1` upper right: `Brdn All  4 1 3`

lower left: `Brdn All  4 1 4` lower right: `Brdn All  4 1 5`

Each of the fhe four drawings was then generated by the following commands:

`S O ...`	(select position for the center of the drawing, –—> 57)
`G O : X`	(draw the outlines)
`X P  41 Ag`	(use pattern 41 for Ag)
`X P  33 Ga`	(use pattern 33 for Ga)

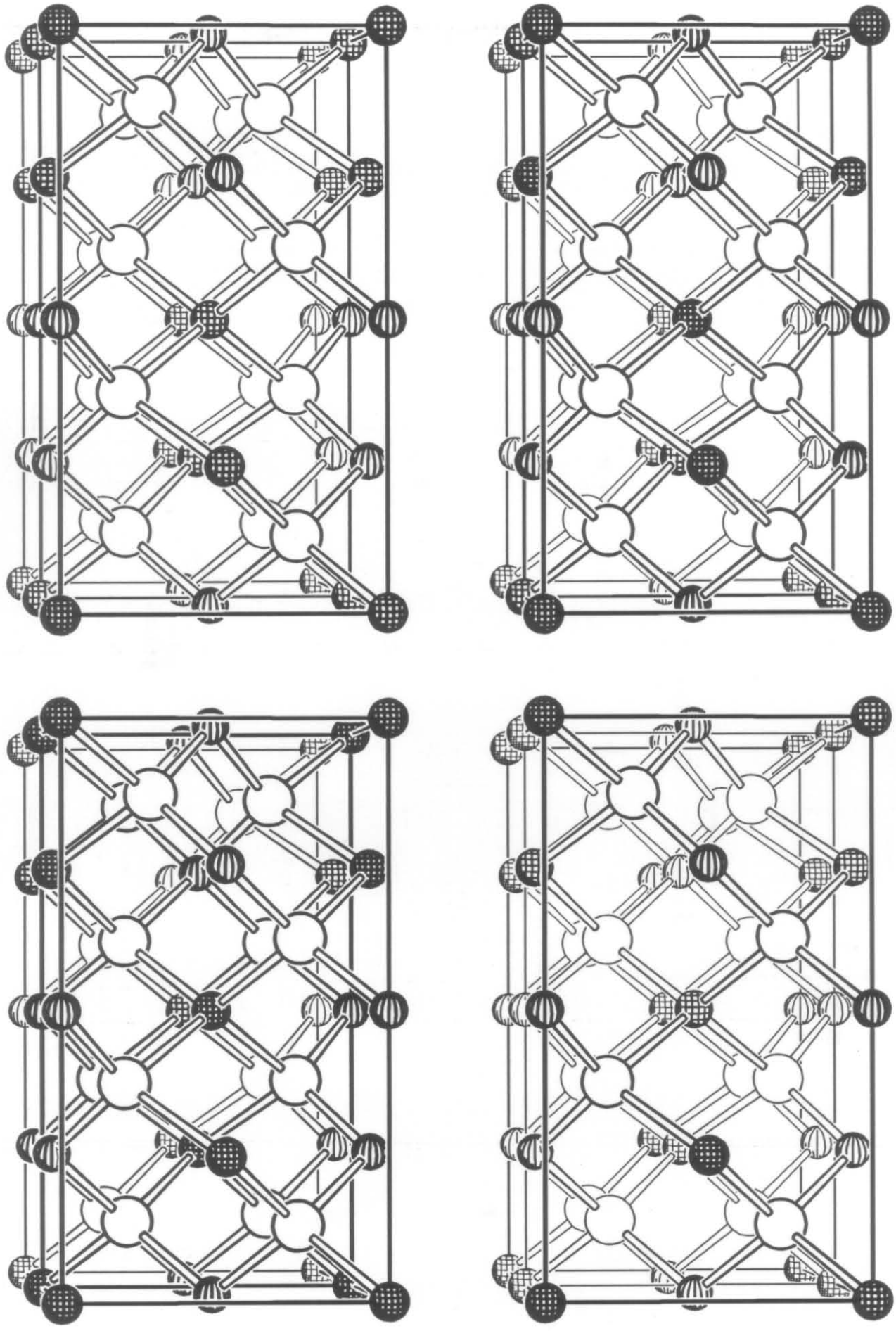


Fig. 15: The $AgGaS_2$ structure

Fig. 16: The (fluoranthene)$_2$ PF_6 structure

This drawing of the crystal structure of (fluoranthene)$_2$ PF_6 illustrates different facilities to generate drawings of crystal structures containing individual building blocks ("molecules"). First, the data set *fluor.dat* (see box, below) was transferred via { cryst^L } / { bld+p } into the data set *fluor_p.dat* (cf. section 5.6. of this manual):

```
TITL Fa_2_PF_6_
CELL   6.610  12.570  14.770  104
ATOM   P     .0000   .0000    .0000
.
.
ATOM   H5    .3830  -.0890    .8340
SPGR A 2/m
```

To both data sets (*fluor.dat* and *fluor_p.dat*) , the following lines were added (for convenience). The commands in the last four lines are executed whenever the data set is loaded.

```
ASSM Comm
STOP
A D 1 0 0                 (align the direction [100] perpend. to drawing plane)
S L 45 45                 (position the light source)
S V 80                    (set perspective)
S M 33                    (set regular dithering mode for shading)
```

To generate the different drawings, the following files were loaded and the following additional input was given "by hand":

upper drawing: *fluor.dat*

add.l data for left part:

```
END
```

add.l data for right part:

```
EXPAND  2
END

Transf Expandr 1
```

lower drawing: *fluor_p.dat*

additional data:

```
END
```

Then, the following additional commands (plus a few others) were given:

upper drawing:

```
Gen Outl : Xqt Xqt  tt

Gen Shading 1
Set Zlim  0 7.5
Mdf Darkn  .9 .2 -2
Xqt Xqt  tt
Set Zlim
Brd All 3 1
Xqt Unitcell
```

lower drawing:

```
Set Zlim 0 7.5
Gen Outl : Xqt Xqt tt
Set Zlim
Gen Shading 1

Mdf Darkn .9 .1 3
Xqt Xqt  tt

Brd All 3 1
Xqt Unitcell
```

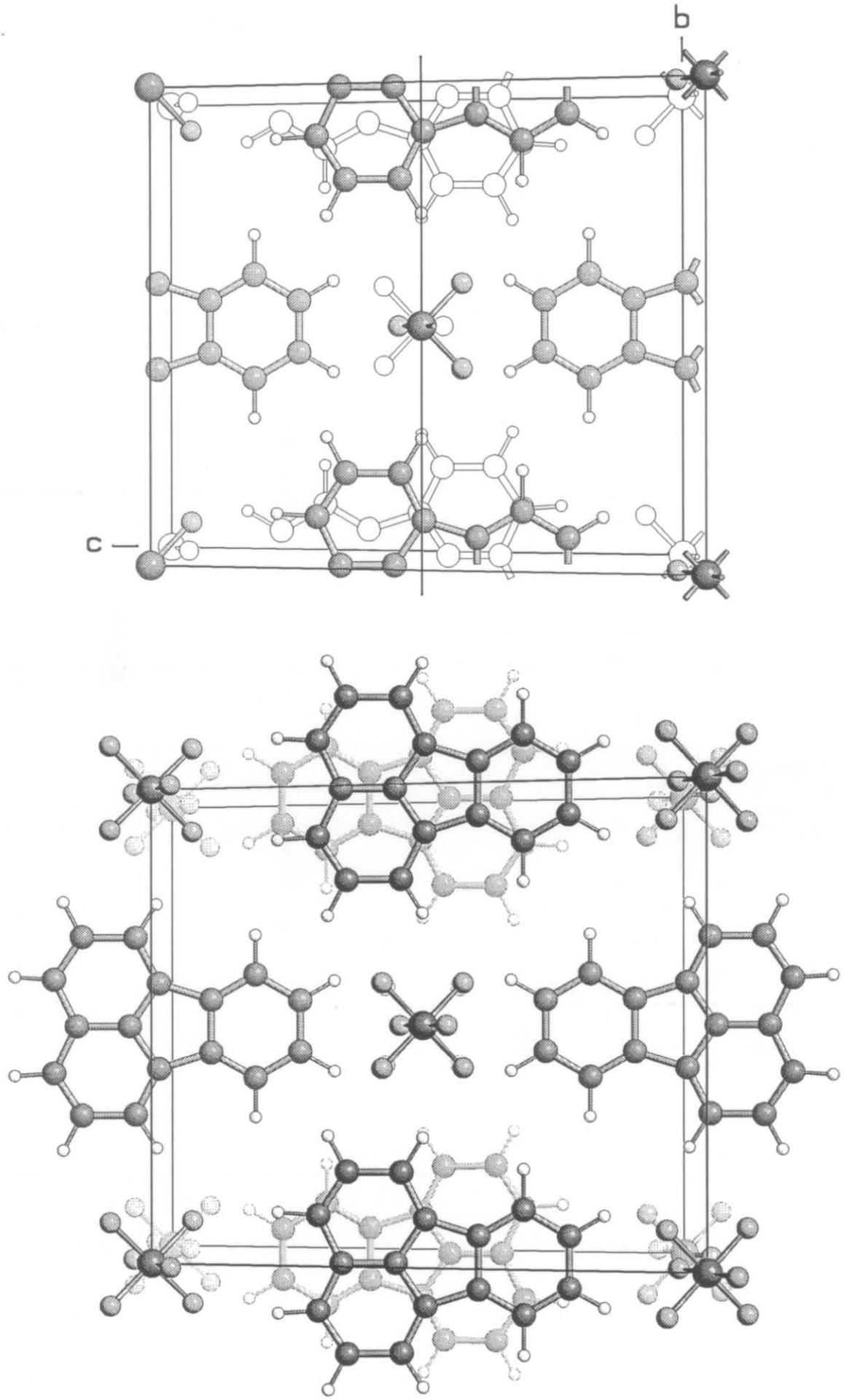


Fig. 16: The $(fluoranthene)_2\ PF_6$ structure

Fig. 17: The $YBa_2Cu_3O_7$ structure

These drawings demonstrate how addition of an EXpand region (–––> 845) to a BOx section (–––> 841) can be used to shape the boundary regions of a solid-state model. All four drawings have been generated by using the following data set, which adds an EXpand region of thickness 3.5 Angstrom to the BOx section (which is simply the unit cell, here):

```
TI  YBa 2 Cu 3 O 7
CE 3.821 3.886 11.668 AS I
AT Cu1 0 0 0
.
.
AT O4  0 .5 0
SGrp P m m m
EXpa 3.5
ASsm C
END                          (a plain BOx card is added by the program)
C D  9 Y, 122 O              (use non-default darkening functions)
C T / 3 Y=nn Ba=nn           (modify thicknesses of some bond sticks)
C R * 1.25 Y Ba Cu           (increase some radii)
R X -90; Y 10; X 10          (rotate the model)
S V 100; L 45 45; M 33       (set perspective, light position, and shading mode)
```

The data set has been made "ready-to-use" by inserting an ENd card (i.e., there is no request for input of more data cards when the data set is loaded). Furthermore, some commands have been added to the data set (cf. fig. 16).

After processing of the data set, the following 'Trnsf Expandregion..' commands (–––> 712) have been used to modify the model (note: the spheres with the smallest radii represent Cu atoms; note: "normal" atoms are those within the section defined by BOx):

upper left: `T E  Cu`

(all EXpand region atoms are deleted except those ones bonded to normal Cu atoms)

upper right: `T E Cu : T C  Cu`

(as upper left; additionally, the coordination polyhedra of the Cu atoms have been modelled by bonds)

lower left: `T E  1`

(all EXpand region atoms are deleted except those ones bonded to any normal atom. The remaining EXpand region atoms are converted into pseudo atoms at half bond distance, thus forming broken-off bond sticks)

lower right: `T E  1 Cu`

(all EXpand region atoms are deleted except those ones bonded to any normal Cu atom. Remaining EXpand region atoms are converted into pseudo atoms at half bond distance, thus forming broken-off bond sticks)

Then the drawings were generated using the commands

```
Gen Outlines :  Xqt
Gen Shadg : Xqt Xqt tt
```

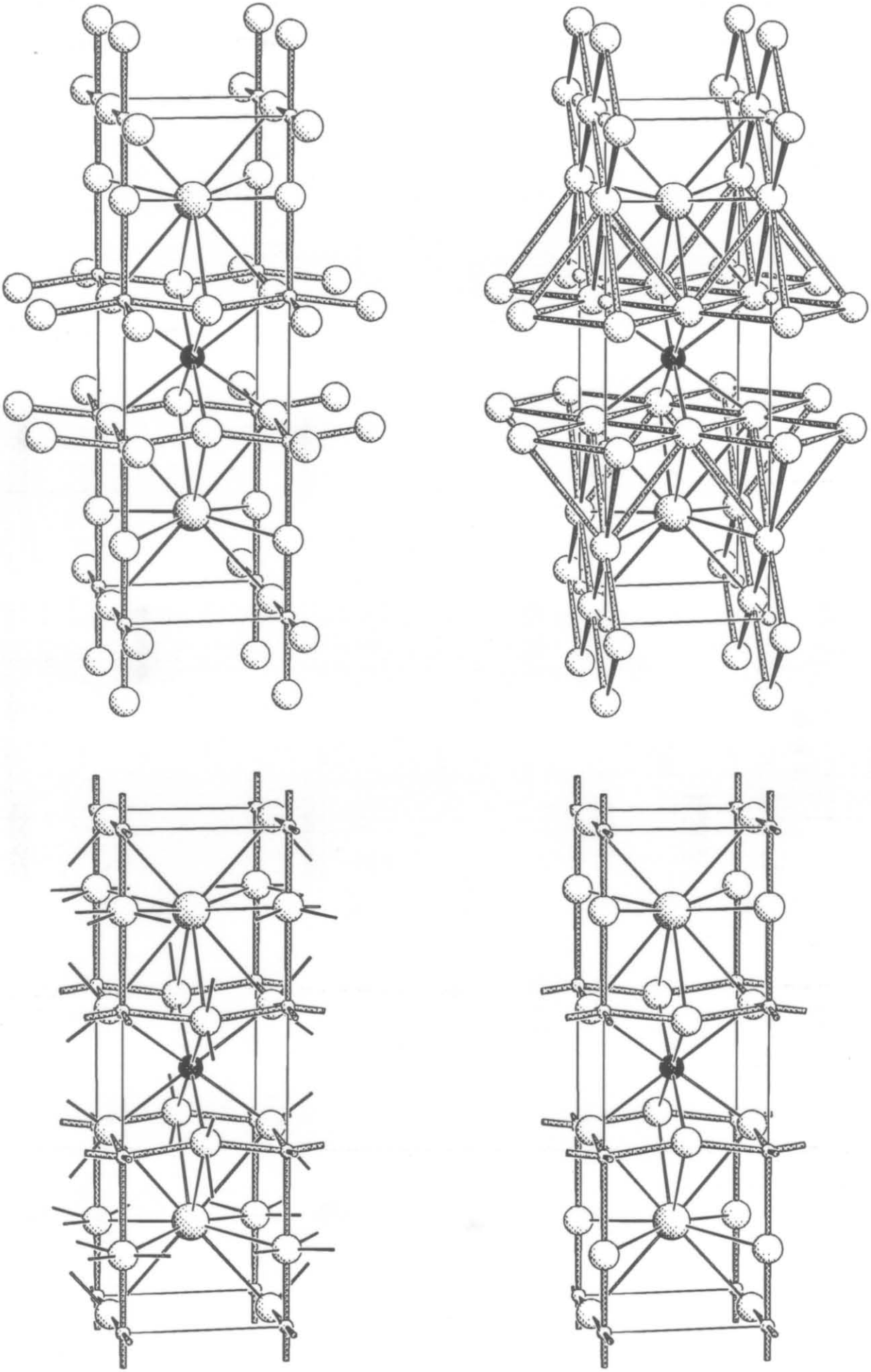


Fig. 17: The $YBa_2Cu_3O_7$ structure

Fig. 18: The ß-Bi_5O_7I structure

This drawing demonstrates the use of 'FAce h k l' cards (–> 853) to define a non-parallelepipedal section of a crystal structure. The following data set was used to generate this model:

```
CELL           18.319   4.245  13.221     108.30
ASSM  Ionic
ATOM   BI1      .2055   .0000   .3656
.
.
ATOM   O7       .5968   .0000   .5210
SPGR C 2 / m
DIST Bi I 4.2
FACE  1 0 0  2.5 2.5
FACE  0 1 0
FACE  0 0 1  2.5 2.5
FACE -2 0 1  2 2
ASSM Comm
END
Aln Dir 0 1 0
```

Instead of defining the desired crystal section by a suitable BOx card, a non-parallelepipedal section was generated here, using FAce cards (–> 853). This allowed (via the last FAce card) to cut off the two corners on top and on bottom which would have been included otherwise [*].

The syntax used for the FAce cards, here, refers to the "default position" of the corresponding planes. This is the position a plane (hkl) would have if it would be shifted from outside towards the unit cell until it touches the cell at a corner, edge, or face. The default distance of the plane to the **center** of the unit cell is multiplied by the parameter on the FAce card to achieve the final position. If two parameters are specified, the second parameter refers to the plane (-h -k -l).

The drawing was generated using the following commands (plus a few others, omitted here):

`Chge Darkn 122 I`	(assign non-default darkening functions, –> 646)
`Chge Darkn  61 Bi, 5 O`	
`Chge Radii *5`	(use non-reduced radii for ball-and-stick model, –> 642)
`Set Light 45 45`	(set the light source, –> 528)
`Set Inscr ! 1 0`	(drawing in portrait format, –> 466)
`Filt U -2  .5`	(use a filter for crystallographic x coordinates, –> 684; here, the x axis is running vertically from top to bottom)
`Defn $ nn`	(define group $ by all atoms [with x < 0.5], –> 664)
`Filt`	(switch filter off)
`Defn $  $ Bi I`	(add the remaining Bi and I atoms to group $)
{ imptt^R }$_{2,6}$	(use Command file *i.scf* to have "unimportant" atoms [i.e., those ones not belonging to group $, i.e., in this case, the O atoms of the upper half] drawn pseudo-transparently)

[*] Another method would have been to generate a normal parallelepipedal box section and cut off the corners with 'Xqt Lines' and 'Kill Atoms rg', cf. end of section 1.6.

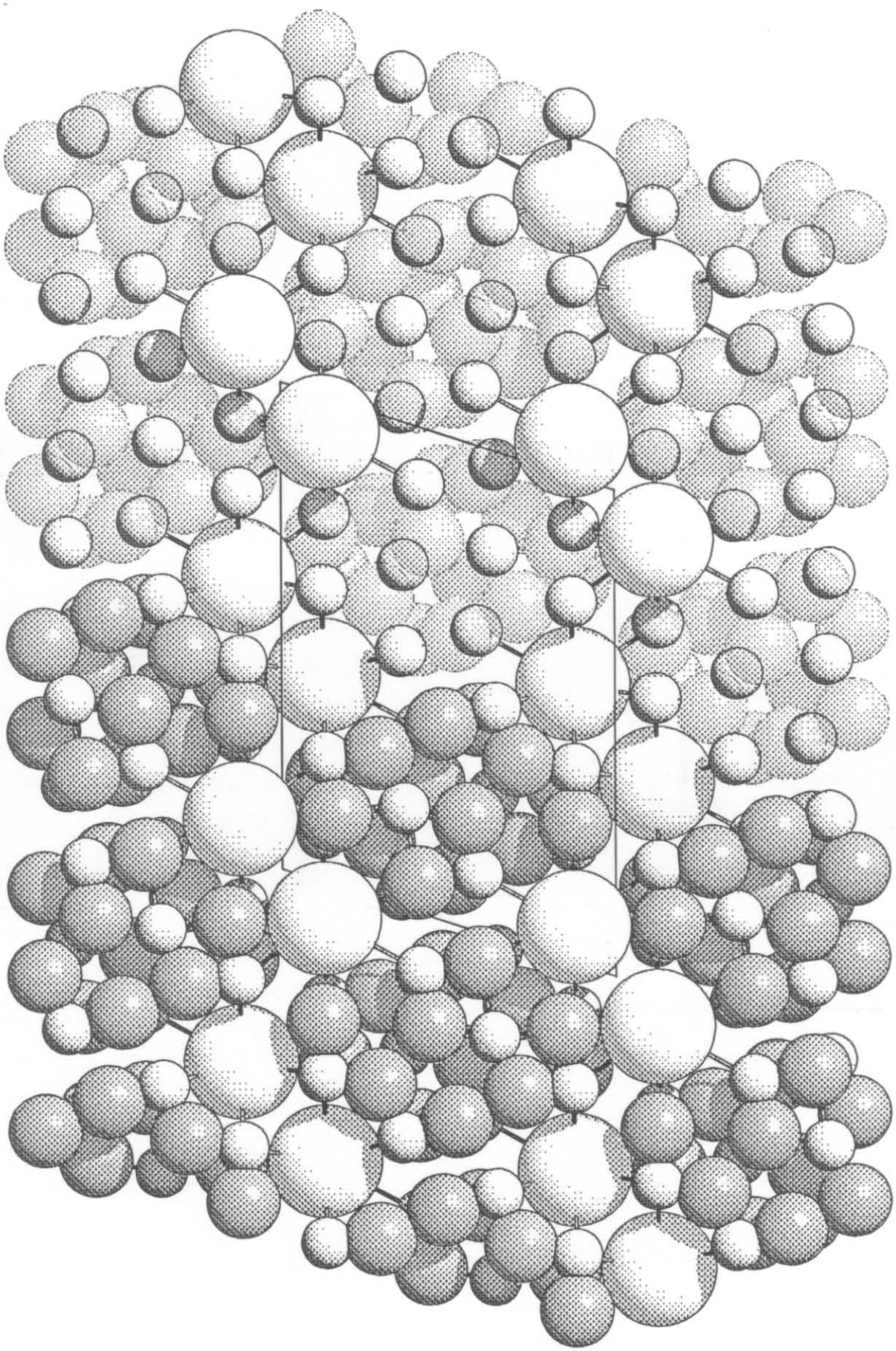


Fig. 18: The ß-Bi_5O_7I structure

Fig. 19: The $KCd_4Ga_5S_{12}$ structure

This drawing demonstrates the use of FAce cards (–—> 583; different syntax as compared to fig. 18) together with EXpand (–—> 845). The following data set has been used to define this model:

```
TITL  KCd 4 Ga 5 S 12
CELL  13.782  13.782   9.330   120
ATOM  M1      .3589   .1014   .0799
.
.
ATOM  S4      .0757   .2994  -.0021
SPGR R 3
FACE H 1.25
FACE 0 0 1 *.85 *.15
EXPA 2.5
BOX
ATOM A101  1.25  0    0
ATOM A102  1.25  1.25 0
.
.
ATOM A107  1.25  0    0
ASSM C
END
F W  -1 -.15 : K A  nn        (delete atoms with cryst. z < -.15)
F W  0.85 2  : K A  nn        (delete atoms with cryst. z > .85)
T E  1                        (generate broken bonds to the EXpand region)
F U  -.25  2 : D $  nn        (define group $ by all atoms with  -.25 < x < 2,
F V  -.25  2 : D $  $          -.25 < y < 2,   and
F W   .15  1 : D $  $ : F      +.15 < z < 1  )
D F  nn ($                    (define a fragment with all atoms except $ atoms)
C S  *1.5 tt (M               (change some radii and stick thicknesses)
```

The first FAce card specifies a hexagonal prism, default distance of planes to unit cell **origin** multiplied by 1.25 (note: with respect to this, 'FAce H..' behaves different from other 'FAce N..' cards, –—> 856).

Within the second FAce card, "*.85" selects, as a boundary plane, a version of the (001) plane which has a distance of $d_{(001)}$ * 0.85 to the unit cell **origin** (different syntax as compared to the FAce cards of fig. 18 !). "*.15" selects, as a second boundary plane, a version of the (00-1) plane which has a distance of $d_{(00\text{-}1)}$ * 0.15 to the unit cell **origin** (note: the lattice plane distance $d_{(001)}$ is equal to the height h of the unit cell in **c** direction which is equal to c itself, here, because the vector **c** is perpendicular to (001)).

The A10n atoms are used to define the dashed hexagon. It visualizes the region addressed by the 'FAce H 1.25' card.

The structure can be seen as consisting of layers parallel to (001). All EXpand atoms with z < -0.15 or z > 0.85 are deleted (to have no broken-off bond sticks above the surface, when the EXpand region is shaped with 'Trnsf EXpnd 1', –—> 713). In a certain region (see upper right part of upper drawing), two layers are removed reversibly to reveal the structure of a single layer (see commands between 'F U ..' and 'D F..').

The upper drawing displays a plan view, the lower one a side-view of the model.

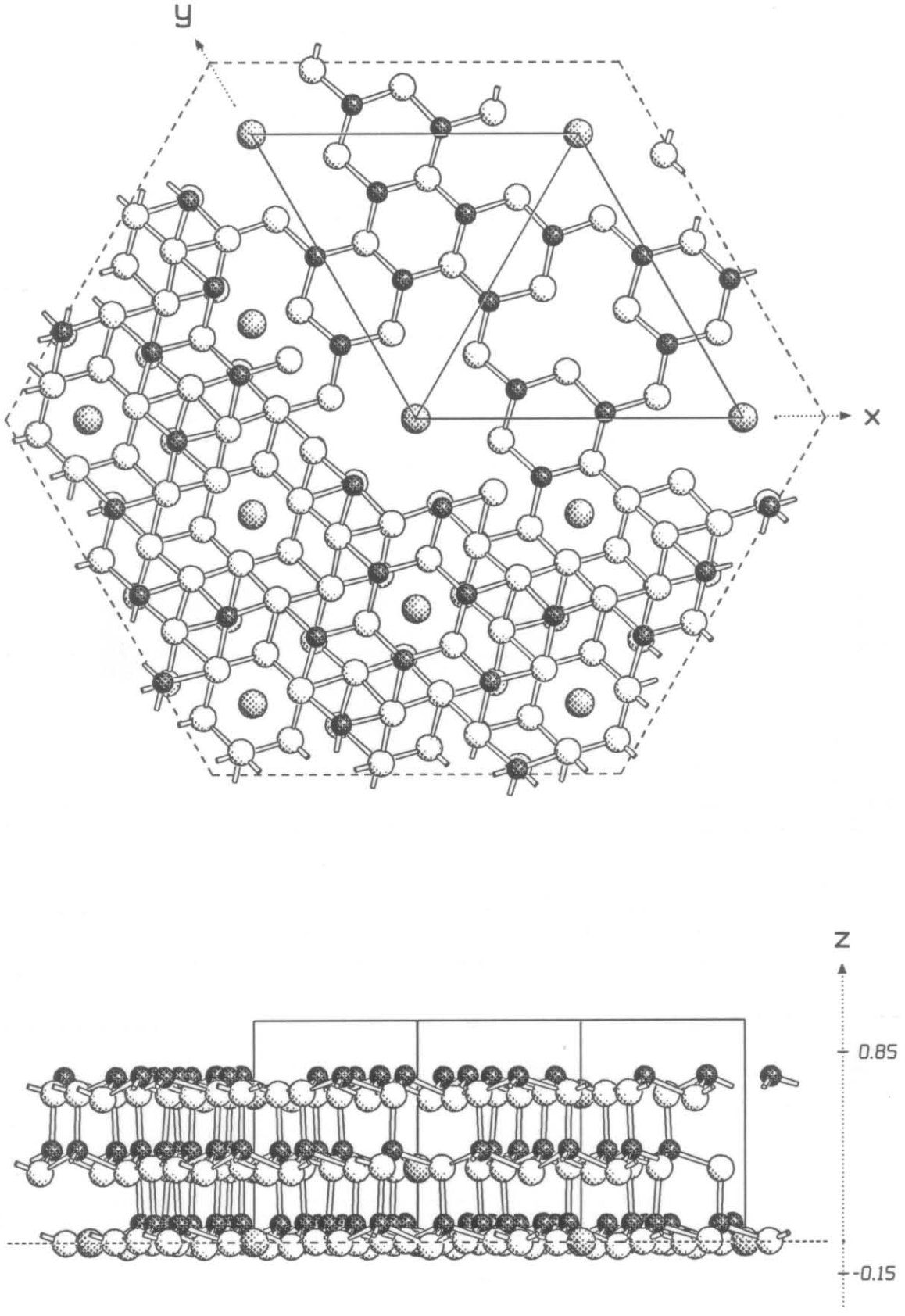


Fig. 19: The $KCd_4Ga_5S_{12}$ structure

Fig. 20: A (111) surface model of the $Pb_3(PS_4)_2$ structure (I)

While the models of fig.s 18 and 19 could be seen to be "surface models" of the (0 1 0) or (0 0 1) planes, they actually aren't surface models (but "polyhedral models") in the sense of the SCHAKAL manuals (–-> 851). Here, a model is called a "surface model", if the corresponding data set contains only **one** normal 'FAce h k l' card (note that 'FAce H 1.25' of fig. 19 replaces three normal 'FAce h k l' cards !).

The different models of fig. 20 a)-c) have been generated by the data set *ex2_p.dat* [DOS/Win16:

```
TITL  Pb 3 (PS 4 ) 2   Cubic SpGr P2 1  3
CELL   10.9394   10.9394   10.9394  90.0000  90.0000   90.0000
ASSM   P
ATOM   P1         .5743     .5743     .5743
.
.
MOL
ATOM   P2         .8498     .8498     .8498
.
.
MOL
ASSM   I
ATOM   Pb1        .4758     .2570     .6376
SPGR  P213
STOP
```

p_ex2.dat] which had been obtained from *ex2.dat* by means of { crystL } / { bld+p } (see 5.6.). For the model of fig. 20a, the following data cards have been added "by hand":

```
FACE  1 1 1
END
```

This demonstrates the "first-choice" way to generate a surface model. The FAce card contains the h,k,l values only. Size information may be given on a BOx card. As no BOx card is given, the program adds one with no parameters, i.e., just the contents of one "alternative cell" are generated (note that an additional 'BOx 0 1 0 1 0 1' would have led to the same results).

The alternative cell is composed of the plane's 2D unit cell [defined by the two unit vectors "vector 1" ($\mathbf{v_1}$) and "vector 2" ($\mathbf{v_2}$)] and the vector $\mathbf{d}_{(111)}$, –-> 851.2 . Because of the latter fact, the thickness of the generated layer is $d_{(111)} * 1$.

For the model of fig. 20b, the following data cards were added "by hand":

```
FACE  1 1 1
BOX  -1 2  -.5 1.5  0 1
END
```

Again a layer of thickness $d_{(111)}$ is generated. The lateral boundary walls, however, are now located at $-1 * v_1$, $2 * v_1$, and $-0.5 * v_2$, $1.5 * v_2$.

Note that a side-view of fig. 20b would look very similar to the lower part of fig. 21.

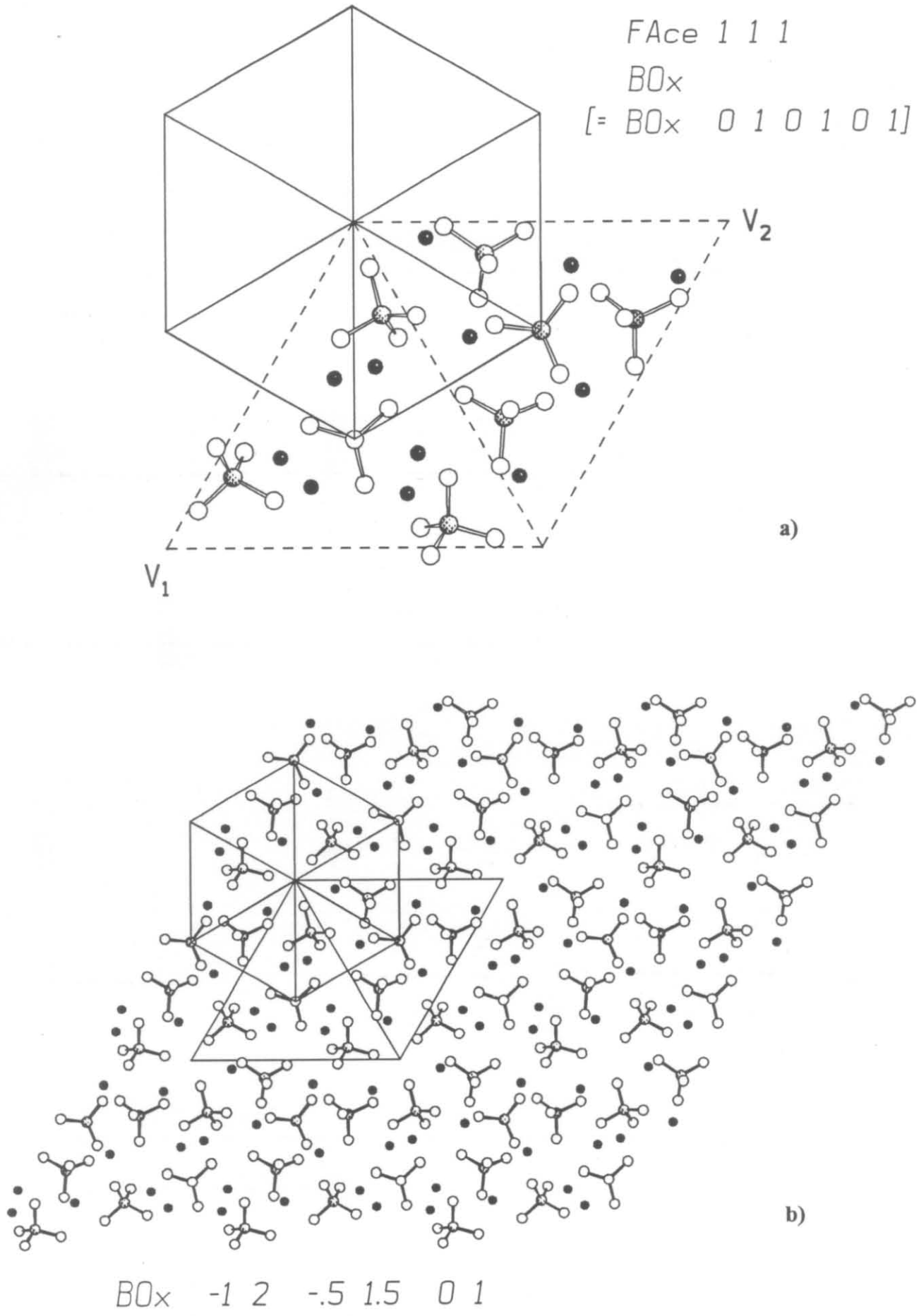


Fig. 20: A (111) surface model of the $Pb_3(PS_4)_2$ structure (I)

Fig. 21: A (111) surface model of the $Pb_3(PS_4)_2$ structure (II)

The same data set, *ex2_p.dat* as described for fig. 20 was used. For the model of fig. 21, the following data cards were added "by hand":

```
FACE  1 1 1 *1 *0
RADI  24
END
```

This is the so-called "alternative" way, to generate a surface model:

The FAce syntax is one of those which have been designed for polyhedral models, originally (see fig. 19). Correspondingly, a (111) plane with distance $d_{(111)}$ * 1 and a (-1-1-1) plane with distance $d_{(111)}$ * 0 to the unit cell origin are established.

The "RAdius 24" card excludes all molecules for which the center of gravity has a distance of more than 24 Angstrom to the unit cell origin.

While the upper drawing displays a plan view of the model, the lower drawing gives a side-view of the same model. Dotted lines are the traces of the (111) lattice planes.

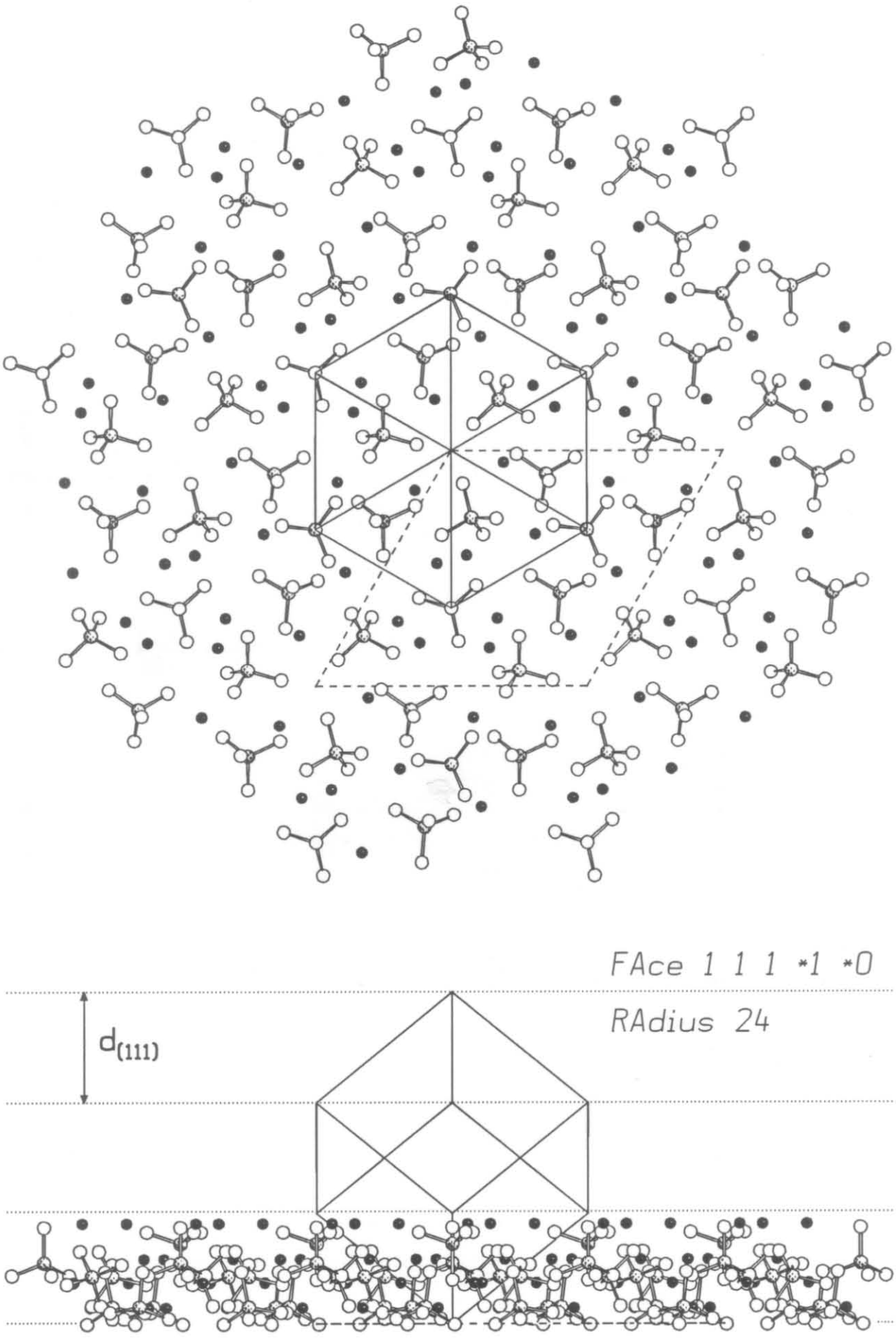


Fig. 21: A (111) surface model of the $Pb_3(PS_4)_2$ structure (II)

Fig. 22: A (230) surface model of the $Pb_3(PS_4)_2$ structure (I)

The data set described with fig. 20 has also been used for the drawings of fig. 22. Here, the following data cards were added "by hand":

```
FAce  2 3 0
BOx   0 1 0 1 z1 z2
```

with different values for z1,z2 in the three different cases. Note that the **range** defined by BOx always **must contain the unit cell origin (**only in connection with FAce !). A comparison of fig.s 22a and 22b demonstrates, how a layer of constant thickness can be "moved" parallel to the plane normal.

If you compare fig. 22a with the side view given in fig. 21 (which would not differ significantly from a side-view of fig. 20, lower part) you will notice that $d_{(230)}$ is much smaller than $d_{(111)}$ [which is due to the general rule, that $d_{(hkl)}$ is the smaller, the larger the absolute values of h, k, and l]. As a compensation, the 2D unit cell of the (230) plane is correspondingly larger than the 2D unit cell of the (111) plane (compare fig. 23 to fig. 20).

Dashed lines are the traces of the (230) lattice planes. If you are not very familiar with Miller indices, you may read the following section:

The Miller index of a set of lattice planes can be determined by the following simple method:

Look at the plane which is the first neighbour to the plane which contains the coordinates system origin (i.e., the unit cell origin). In drawing a), the trace of this first-neighbour-plane is the one which is the upper borderline of the shaded bar. This plane intersects the **a** axis at 1/*2* **a**, the **b** axis at 1/*3* **b**, and the **c** axis (which is parallel to the plane) at infinite = 1/*0* **c**. This corresponds to a Miller index of *2 3 0* .

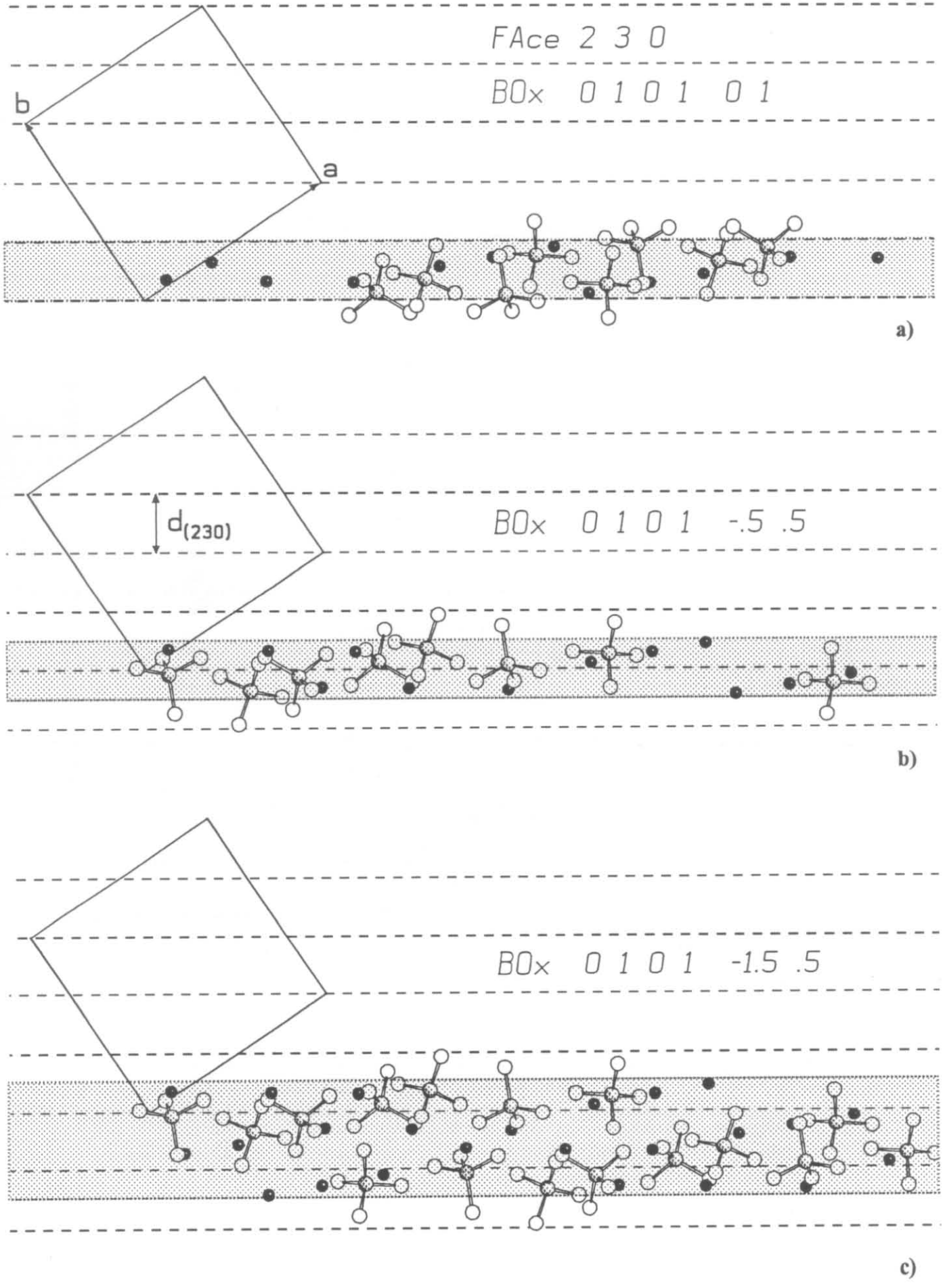


Fig. 22: A (230) surface model of the $Pb_3(PS_4)_2$ structure (I)

Fig. 23: A (230) surface model of the $Pb_3(PS_4)_2$ structure (II)

Fig. 23a gives a side-view of the model*) displayed in fig. 22a;
Fig. 23b gives a side-view of the model*) displayed in fig. 22c.
(*) a slightly modified BOx card [first two parameters] was used for the drawings of fig. 23).

Edges of the structure's unit cell have been omitted, here. However, the edges of the 2D unit cell of the (230) plane (dashed lines, –> 545) have been switched on exclusively by a 'Gen Unitcl 5 -1' command (–> 486).

If you look at fig. 23a, you will see, that this surface model looks rather strange: there are zones where no PS_4 ions are present at all. This is because the thickness of this layer (= $d_{(230)}$) is too small (cf. fig. 22a).

You will have to generate a thicker layer in this case to obtain a reasonable model of the surface. This has been done with the model shown in fig.s 22c/23b which correspond to a layer thickness of $d_{(230)} * 2$.

Generally, you will have to generate thicker layers (in terms of $d_{(hkl)}$) for larger h,k,l values to get a reasonable surface model.

There is a tiny problem, however: a surface model's "alternative cell" (with $v_3 = d_{(hkl)}$) contains just as many atoms as the structure's unit cell itself. Therefore, a model with a layer thickness of $d_{(hkl)} * 2$ will contain at least 2 times the atoms contained in the normal unit cell (provided that at least one complete 2D cell is enclosed between the lateral faces). Thus, you may exceed the program's atom capacity if you continue to increase the layer thickness of your model (but see fig. 24, 7th parameter on BOx card).

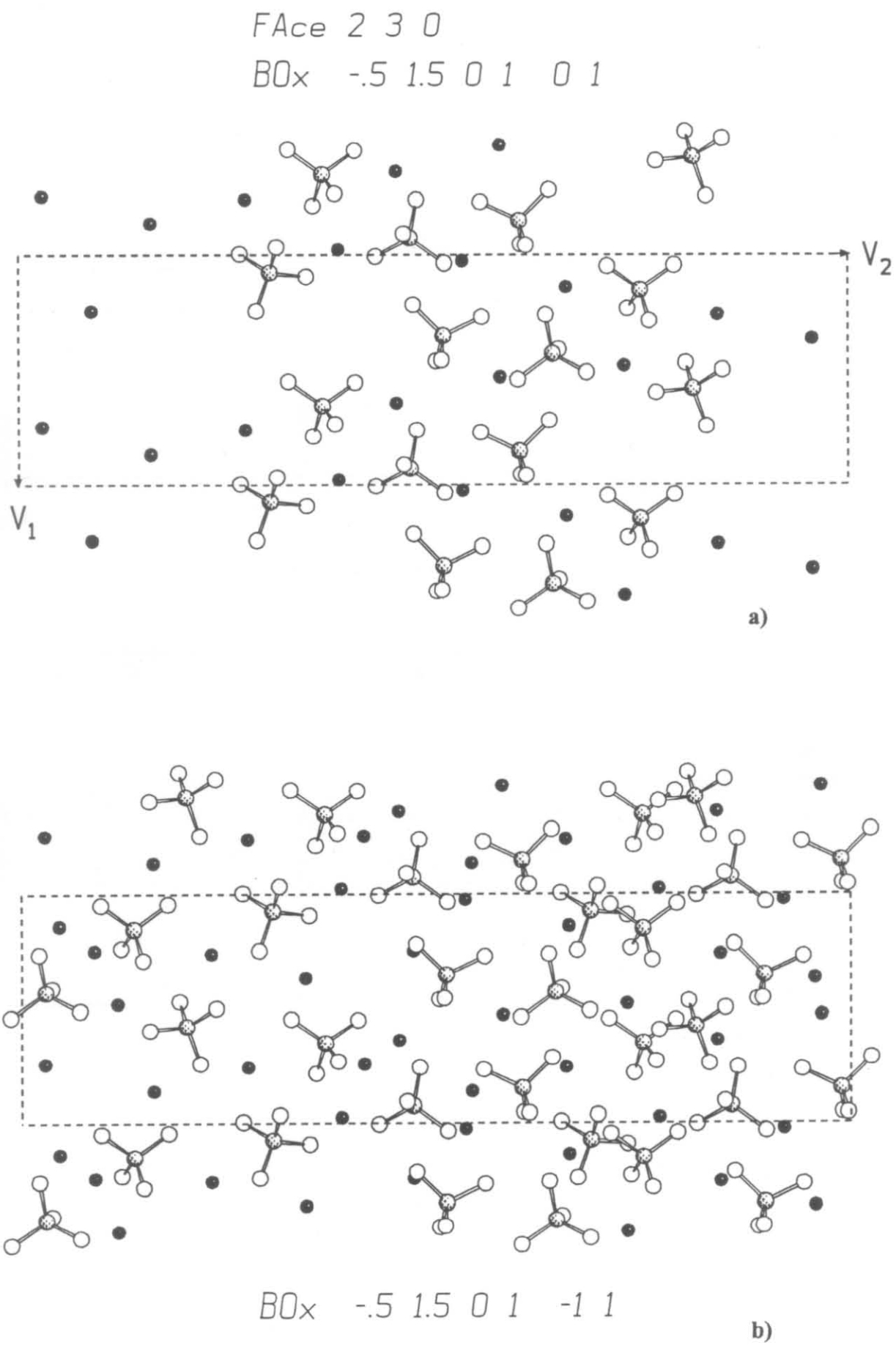


Fig. 23: A (230) surface model of the $Pb_3(PS_4)_2$ structure (II)

Fig 24: A (011) surface model of the (fluoranthene)$_2$ PF_6 structure

The data set to generate the (011) surface model of the (fluoranthene)$_2$ PF_6 was identical to the one for the lower part of of fig. 16 (*fluor_p.dat*):

```
TITL Fa 2 PF 6
CELL   6.610  12.570  14.770   90.00  104.00   90.00
ASSM   P
ATOM   P     .0000   .0000    .0000
.
.
ATOM   F2   -.0977   .0881   -.0731
MOL
ATOM   C1    .2252   .0000    .4489
.
.
ATOM   H5    .3830  -.0890    .8340
SPGR A 2/m
```

The following data cards were added "by hand":

```
FAce 0 1 1
BOx -2 3  -1.5 2.5  0 1  .5
END
```

With the last parameter on the PAck card, it was achieved that the more distant half of the atoms (when seen from the surface itself) was not included in the model (thus saving memory space to increase the lateral width of the model but retaining an intact surface).

To have the uppermost layer contain fluoranthene molecules instead of PF_6 ions, the following BOx card would have to be given:

```
BOx -2 3  -1.5 2.5  -.5 .5  0.01
```

The drawing was then generated with the following commands:

`Genr Unitc  5 -1`	(edges of the plane's 2D unit cell are switched on, exclusively, –> 486)
`Mgnf Model  1`	(fix the model scale factor, –> 511)
{ varyZ }$_{3,6}$ { uni }$_{19,2}$	(use Command file *f.scf*)
`Unlk Model`	(unlock the model for geometry modifications, –> 36)
`Trsl Z  20 A91-A94`	(shift the corners of the 2D unit cell "upwards", –> 851.3)
`Xqt Unitc`	(draw 2D unit cell edges, –> 218)

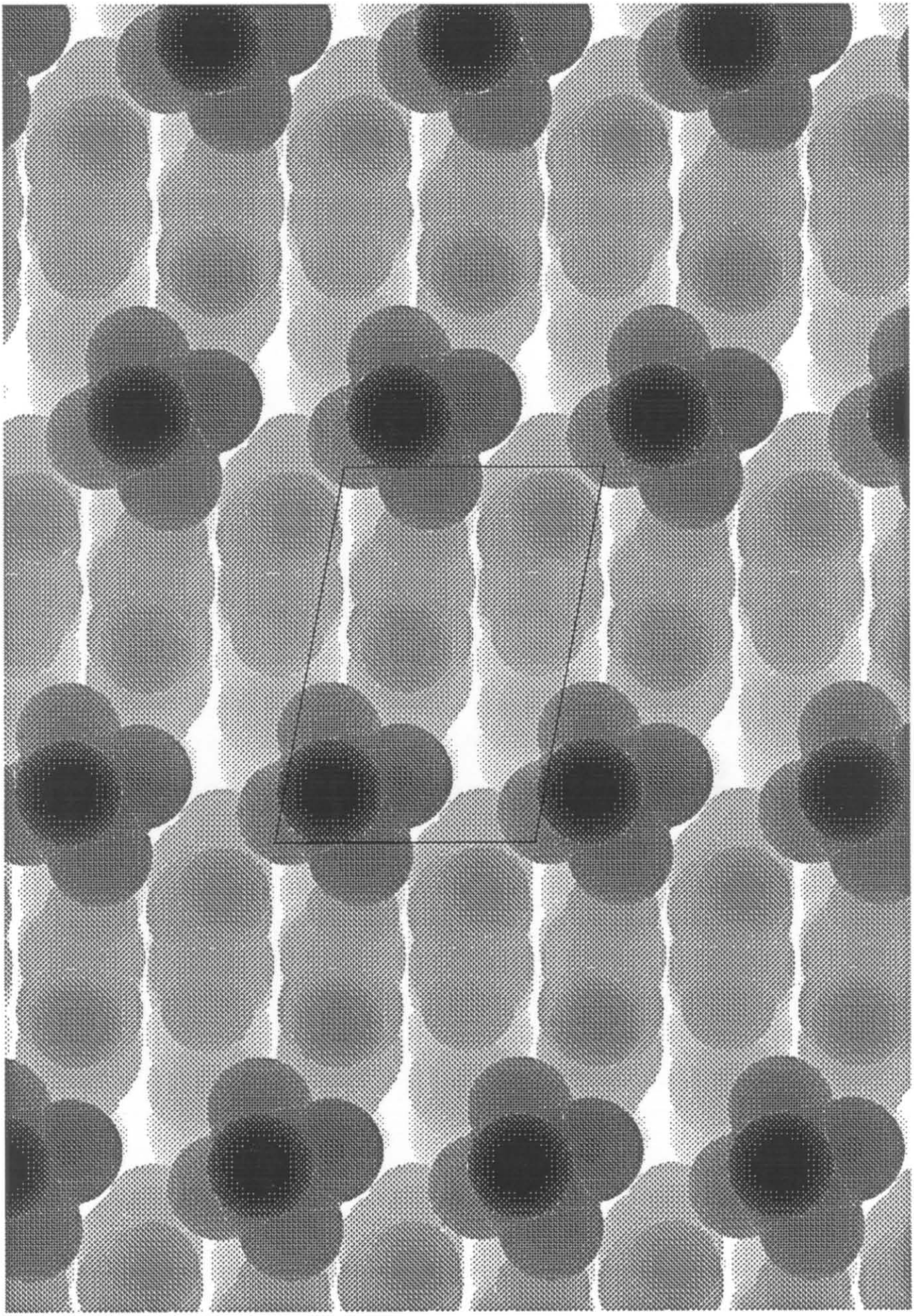


Fig. 24: A (011) surface model of the $(fluoranthene)_2PF_6$ structure

Fig 25: The Graphical User Interface (GUI)

#	***Explanation***	–––>
A	the "1st row" allows to select a group of commands or execute certain commands	931
B	the 10 "coloured control fields" launch some certain, frequently used commands.	933
C	header of the "command column"; [almost] all commands beneath start with 'Use...'.	932
D	control field (CF) for the 'Use Xtaldat' command (light blue part)	932
E	CF for the 'Use Xtaldat' command (yellow part); immediate execution when clicked	932
F	a "command line control field"; may be removed or replaced by another one	774
G	a "green star", indicating that this command is accessible from the 1st row directly.	931
H	"green star" referring to the one labelled "G".	931
I	clears the GUI; use it to remove "parameter boxes", "colour boxes", etc.	933
J	works like <RETURN>, i.e., may be used to send a command to the program	932.2
K	these two CFs may be used to access the hierarchical on-line manual	936
L	opens a directory of "distributed command files" (DCF)	934
M	executes a DCF (namely the "default DCF" of this directory)	934
N	the text "window" which is frameless and of variable size	73
O	the "graphics area" (or "drawing area"), where something can be drawn	2
P	may be used to store a screen drawing as a TIFF file or to print a printer drawing	171, 153
Q	these CFs may contain atom codes (e.g. chemical element symbols)	935
R	may contain numbers or a short message	935
S	may be used to emulate <BACKSPACE>	936
T	same as "J"	932.2
a	toggles between M(ouse), A(rrow_keys) and - possibly - S(ave) mode	311.3
b	toggles between R(otation), F(lip), and - possibly - T(ranslation) mode	311.2
c	makes motion faster (i.e., increases step width)	311.2
d	one of the six CFs to control motion (e.g. rotation, while in R mode)	311
e	may be used to stop motion	
f	these two CFs control the labelling of atoms	311.5
g	fixes the scale factor [sc. f.] and reduces it stepwise	311.6
h	visualizes the current value of the sc. f.; click somewhere to select a new (fixed) one	311.6
i	fixes the sc. f. and increases it stepwise	311.6
j	displays the current value of the sc. f.	311.6
k	sets a special value of the sc. f. : sc.f. = zero (i.e., sc. f. = non-fixed, variable)	311.6
l	switches the unit cell edges on and off	311.7
m	toggles between ball-and-stick and space-filling model	311.7
n	may be used to leave on-screen rotations mode	311
o	undoes rotations and translations	311.5
p	switches to wire model with or without small atom spheres	311.4
q	switches to 'Xqt Quick' drawing	311.4
r	switches to outline drawing	311.4
s	switches to standard shaded drawing	311.4
t	makes motion slower (i.e., reduces step width)	311.2
u	switches to drawing produced by the last active command file	311.4
v	toggles between current line width and line width 1 (fastest)	311.5
w	switches stereo mode on and off	311.7
x	may be used to leave on-screen rotations mode	

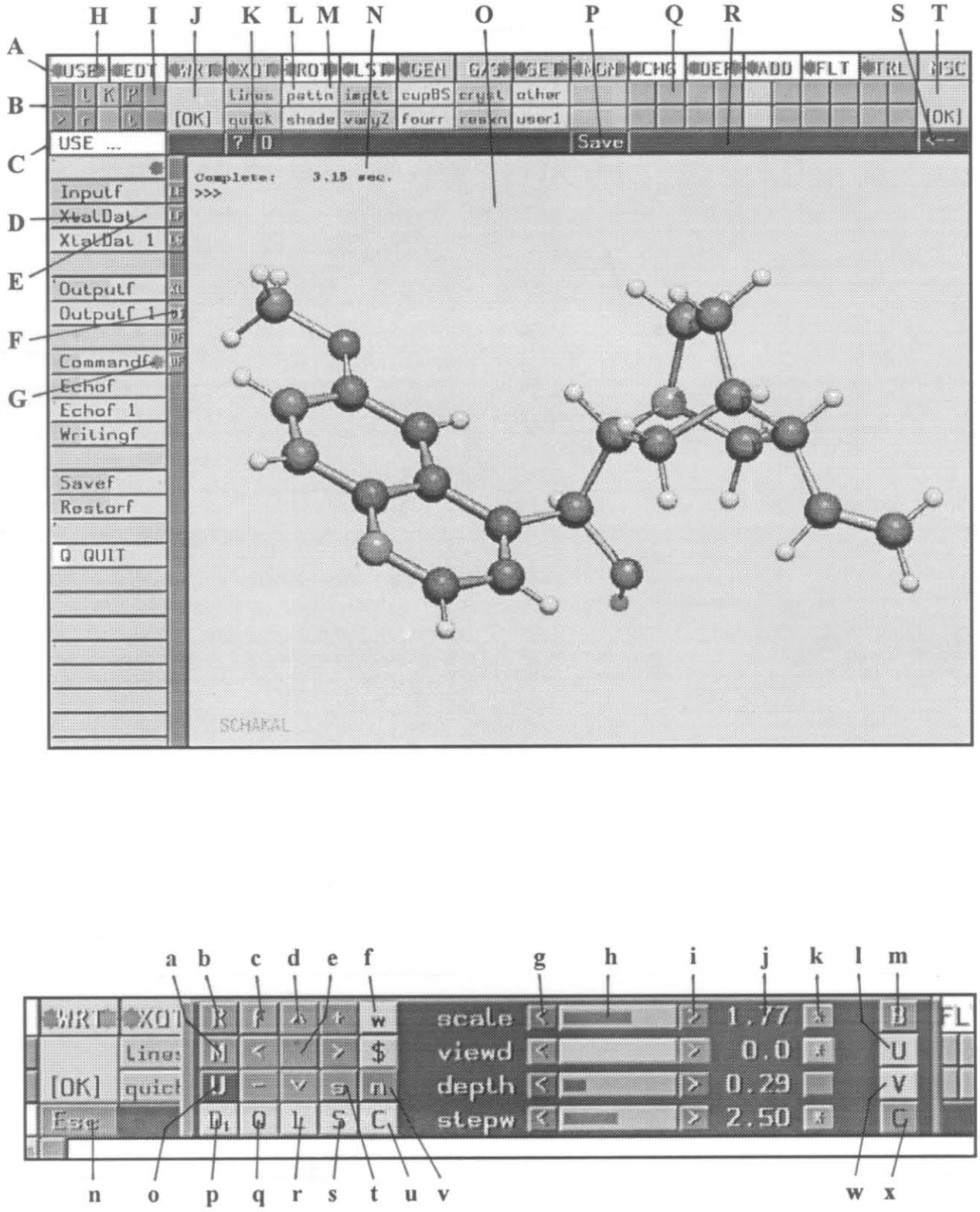


Fig. 25: The Graphical User Interface (GUI)
top: default state; bottom: on-screen rotations box

References

The structures stored in the files *ex*.dat* and those ones mentioned within this manual were taken from the following sources:

$Fe_4(CO)_{12}(NC\text{-}C_6H_5)$ (file *ex1.dat*)
E. Keller & D. Wolters
Chem Ber. **117** (1984), 157
Remark: some atom names and molecular geometry in *ex1.dat* differ from those ones in the publication

$Pb_3(PS_4)_2$ (file *ex2.dat* and figs. 20-23):
E. Post & V. Krämer
Mat. Res. Bulletin **19** (1984), 160

Quinine (file *ex3.dat* and fig. 25)
S. Kashino & M. Haisa
Acta Cryst. **C39** (1983), 310

Penicillin (file *ex4.dat* and fig. 11):
D. Crowfoot, C.W. Bunn, B.W. Rogers-Law & A. Turner-Jones
in "The Chemistry of Penicillin", edited by *H.T. Clarke, J.R. Johnson & R. Robinson,*
1949, pp. 310, Princeton University Press
Remark: original conformation of the molecule has been modified to generate fig. 11

Glycyl-glycyl-glycine (β form) (file *ex5.dat*)
T. Srikrishnan, N. Winiewicz & R. Parthasarathy
Intern. J. of Peptide and Protein Res. **19** (1982), 103

$(C_{20}H_{24})\ Zr\ Cl_2$ (fig.s 12 - 14):
F.R.W.P. Wild, M. Wasiucionek, G. Huttner & H.H. Brintzinger
J. Organomet. Chem. **288** (1985), 63
Remark: some atom names are different from those ones in the publication

$AgGaS_2$ (fig. 15):
S.C. Abrahams & J.L. Bernstein
J.Chem. Phys. **59** (1973), 1625

$(fluoranthen)_2\ PF_6$ (fig.s 16 and 24):
V. Enkelmann, B.S. Morra, Ch. Kröhnke, G. Wegner & J. Heinze
Chem. Phys. **66** (1982), 303

$YBa_2Cu_3O_7$ (fig. 17):
K. Brodt, H. Fuess, E.F. Paulus, W. Assmus & J. Kowalewski
Acta Cryst. **C46** (1990), 354

ß-Bi_5O_7I (fig. 18):
J. Ketterer, E. Keller & V. Krämer
Z. Kristallogr. **172** (1985), 63

$KCd_4Ga_5S_{12}$ (fig. 19):
H. Schwer, E. Keller & V. Krämer
Z. Kristallogr. **204** (1993), 203

Index