WestCHEM University of Glasgow Department of Chemistry

IUCr Commission on Crystallographic Computing Siena 2005: Crystallographic Computing School

Louis Farrugia

Lecture-1

Using available tools

1. Types of software tools available

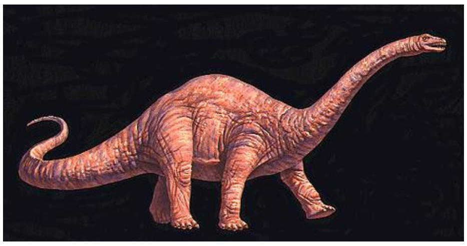
2. How to use them in your software ?

3. Where to get them from ?

From the Computing School website …

School History

First the earth cooled, then there were dinosaurs..



FORTRANOSAURUS – a real dinosaur ?

1. FORTRAN is an ancient language from 1950's – extinction long predicted, but still being maintained and modernised – FORTRAN95 in current use. FORTRAN 2003 is latest standard (but as yet no compilers !)
2. FORTRAN is an efficient language and well suited to scientific computing
3. There exist vast libraries of FORTRAN sources for scientific computing
4. It is *possible* to write modern software in FORTRAN.

Types of software tools currently available

- programming languages & compilers/interpreters
- GUI (graphical user interfaces)
- 'scientific' libraries
- pre-built 'scientific' applications
- utility applications

Programming languages used in crystallographic programs and libraries

- FORTRAN
- C
- C++
- Java, Python
- other scripting & GUI languages

Modern applications may be multi-language

GUI's – Graphical User Interfaces

GUI's are now ubiquitous and seemingly necessary

Advantages of the GUI approach

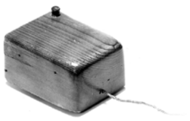
•easy to use and learn programs - commonality of interface
•scientific programs - programmer regulates numerical input
•options/possible pathways made obvious - less need for extensive reference manuals

Disadvantages of the GUI approach

• *too* easy to use and learn programs - no understanding of underlying methodologies
•scientific programs - too restrictive for complex problems
•reluctance to read extensive reference manuals

GUI's – Graphical User Interfaces

Makes full use of VDU screen with a pointing device -mouse (patented 1964)



GUI's – Graphical User Interfaces

Disadvantages of the GUI approach

• *too* easy to use and learn programs

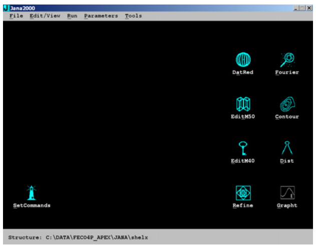
I have question about WinGX software.
when I launched DATA - - > Cad4 - - > XCad4 , the program displays :

```
  5464 reflections processed, of which 56 were standards (5408 netto)
     593 reflection(s) with zero or negative intensity
       0 reflection(s) rejected with intensity less than -9999.00
       1 reflection(s) deviated from scan centre by more than DANG
       0 reflection(s) not measured because collision predicted
       0 reflection(s) not measured because chie > 100 deg
     999 reflection(s) measured on pre-scan only
       0 reflection(s) with a count loss > 1%
       0 reflection(s) with too strong intensities
       0 reflection(s) with bad backgrounds
```

My question is : *what are standards ?*

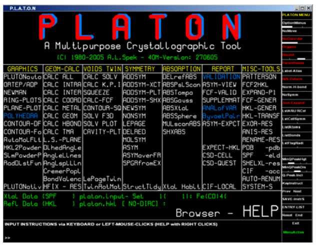
Ways of producing GUI's for applications

• Do it yourself
• Use GUI libraries



Structure: C:\DATA\FECO4P_APEX\JANA\shelx

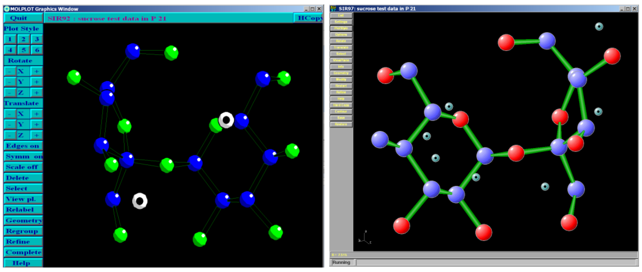
JANA2000



Xtal Data (SPF) platon.input- Set ((1): Fe(CO)4(
Refl Data (HKL) platon.hkl [NO-DIRC] :

INPUT INSTRUCTIONS via KEYBOARD or LEFT-MOUSE-CLICKS (HELP with RIGHT CLICKS)

PLATON



SIRWARE Programs SIR92 / SIR97
http://www.ic.cnr.it/

GUI libraries – many commercial but some free. Many are cross-platform. A small sample …

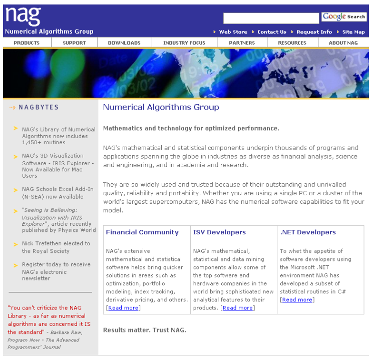
• Java
• Python - http://python.org/
• wxWidgets - http://www.wxwidgets.org/
• wxPython - http://www.wxpython.org/
• Tcl/tk - http://tcl.activestate.com/
• Qt - http://www.trolltech.com/tp
• VGUI - http://vgui.sourceforge.net/
• GTK+ (Gimp tool kit)- http://www.gtk.org/
• Fast Light Toolkit - http://www.fltk.org/
• FOX - http://www.fox-toolkit.org/

Many are written in C/C++, so interfacing to languages like FORTRAN is difficult - Steep learning curve to use.

“Scientific” libraries – mathematical and crystallographic
Some examples ...

- **NAG** – Numerical Algorithms Group
- **CCSL** - Cambridge Crystallographic Subroutine Library
- **CCTBX** – Computational Crystallographic Toolbox – Ralf Grosse-Kunstleve
- **CLIPPER** – OO crystallographic libraries – Kevin Cowtan
- **CrysFML** - Crystallographic Fortran 95 Modules Library - Juan Rodríguez- Carvajal – LLB (Fullprof suite)
- **CIFTBX** – tools for reading/writing CIF’s (also CIFLIB for mmCIF)
- **GETSPEC** – tools for space group symmetry
- **FPRIME** – tool for X-ray dispersion corrections

- crystallography source code archives & “museums”
- graphics libraries



NAGBYTES

- NAG's Library of Numerical Algorithms now includes 1,450+ routines
- NAG's 3D Visualization Software - IRIS Explorer - Now Available for Mac Users
- NAG Schools Excel Add-In (N-SEA) now Available
- *"Seeing is Believing: Visualization with IRIS Explorer"*, article recently published by Physics World
- Nick Trefethen elected to the Royal Society
- Register today to receive NAG's electronic newsletter

"You can't criticize the NAG Library - as far as numerical algorithms are concerned it IS the standard" - *Barbara Raw, Program Now - The Advanced Programmers' Journal*

Numerical Algorithms Group

Mathematics and technology for optimized performance.

NAG's mathematical and statistical components underpin thousands of programs and applications spanning the globe in industries as diverse as financial analysis, science and engineering, and in academia and research.

They are so widely used and trusted because of their outstanding and unrivalled quality, reliability and portability. Whether you are using a single PC or a cluster of the world's largest supercomputers, NAG has the numerical software capabilities to fit your model.

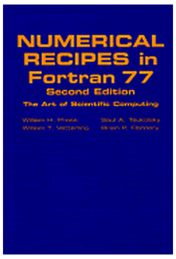
Financial Community	ISV Developers	.NET Developers
NAG's extensive mathematical and statistical software helps bring quicker solutions in areas such as optimization, portfolio modeling, index tracking, derivative pricing, and others. [Read more]	NAG's mathematical, statistical and data mining components allow some of the top software and hardware companies in the world bring sophisticated new analytical features to their products. [Read more]	To whet the appetite of software developers using the Microsoft .NET environment NAG has developed a subset of statistical routines in C# [Read more]

Results matter. Trust NAG.

Numerical algorithms group
http://www.nag.co.uk

languages
FORTRAN & C

BLAS - http://www.netlib.org/blas/
LAPACK - http://www.netlib.org/lapack/



William H. Press
Brian P. Flannery
Saul A. Teukolsky
William T. Vetterling

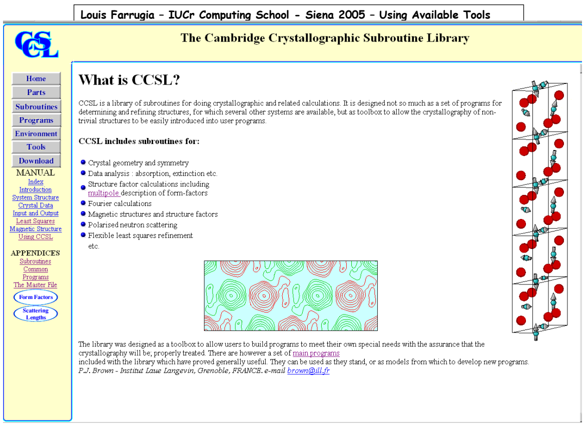
languages

FORTRAN 77/90/95

C, C++

http://www.numerical-recipes.com

Crystallographic library tools



The Cambridge Crystallographic Subroutine Library

What is CCSL?

CCSL is a library of subroutines for doing crystallographic and related calculations. It is designed not so much as a set of programs for determining and refining structures, for which several other systems are available, but as toolbox to allow the crystallography of non-trivial structures to be easily introduced into user programs.

CCSL includes subroutines for:

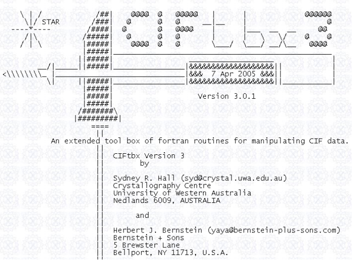
- Crystal geometry and symmetry
- Data analysis : absorption, extinction etc.
- Structure factor calculations including multipole description of form-factors
- Fourier calculations
- Magnetic structures and structure factors
- Polarised neutron scattering
- Flexible least squares refinement

etc.

The library was designed as a toolbox to allow users to build programs to meet their own special needs with the assurance that the crystallography will be; properly treated. There are however a set of main programs included with the library which have proved generally useful. They can be used as they stand, or as models from which to develop new programs.
P.J. Brown - Institut Laue Langevin, Grenoble, FRANCE. e-mail brown@ill.fr

Jane Brown - http://www.ill.fr/dif/ccsl/html/ccsldoc.html
language - FORTRAN

CIFtbx



```
&&&  7 Apr 2005 &&&
                    Version 3.0.1

An extended tool box of fortran routines for manipulating CIF data.

           CIFtbx Version 3
                by

           Sydney R. Hall (syd@crystal.uwa.edu.au)
           Crystallography Centre
           University of Western Australia
           Nedlands 6009, AUSTRALIA

               and

           Herbert J. Bernstein (yaya@bernstein-plus-sons.com)
           Bernstein + Sons
           5 Brewster Lane
           Bellport, NY 11713, U.S.A.
```

FORTRAN API - Designed to be incorporated into a program

memory containing CIF is shared between library and application

later versions offer XML output

http://www.iucr.org/iucr-top/index.html

CIFtbx

Modifications as used in WinGX – as a dynamically linked library

•memory associated with library functions & CIF is isolated
•*public functions* used to place data into library memory, to extract data from memory
• variables initialised *dynamically* using ciftbx_init() rather than *statically* through DATA statements

Used as engine to read CIF files in WinGX and Ortep for Windows

• advantages – easy to read CIF and write out structured CIF's
• disadvantages – will not read a CIF with syntax error(s)
only loads one CIF

CIFtbx

```
 program main
include 'ciftbx.cmn'
logical      f1
character*10 buf1,buf2,buf3,buf4,buf5,buf6,bufa

call getarg(1,bufa)
if(bufa == ' ') stop
f1 = init_(10,11,12,5)
f1 = ocif_('bvparm.cif')
if(.not. f1) stop 'CIF not found'
f1 = data_('bond_valence_parameters')
if(.not. f1) stop 'Block not found'
n = 0
do
  f1 = char_('_valence_param_atom_1',buf1)
  if(.not. loop_) exit
  if(buf1 /= bufa) cycle
  f1 = char_('_valence_param_atom_2',buf2)
  if(buf2 /= bufb) cycle
  f1 = char_('_valence_param_atom_1_valence',buf3)
  f1 = char_('_valence_param_atom_2_valence',buf4)
  f1 = char_('_valence_param_Ro',buf5)
  f1 = char_('_valence_param_B',buf6)
  if(buf3 == '9') buf3 = '?'
  if(buf4 == '9') buf4 = '?'
  n = n + 1
  write(*,*) buf1,buf3,buf2,buf4,buf5,buf6
enddo
write(*,*) n
end
```

462,336 bytes

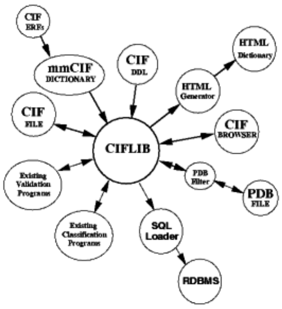
```
program main
include 'ciftbx.h'
logical      f1
character*10 uf1,buf2,buf3,buf4,buf5,buf6,bufa

call getarg(1,bufa)
if(bufa == ' ') stop
call ciftbx_init()
f1 = init_(10,11,12,5)
f1 = ocif_('bvparm.cif')
if(.not. f1) stop 'CIF not found'
f1 = data_('bond_valence_parameters')
if(.not. f1) stop 'Block not found'
n = 0
do
  f1 = char_('_valence_param_atom_1',buf1)
  if(.not. loop_() ) exit
  if(buf1 /= bufa) cycle
  f1 = char_('_valence_param_atom_2',buf2)
  if(buf2 /= bufb) cycle
  f1 = char_('_valence_param_atom_1_valence',buf3)
  f1 = char_('_valence_param_atom_2_valence',buf4)
  f1 = char_('_valence_param_Ro',buf5)
  f1 = char_('_valence_param_B',buf6)
  if(buf3 == '9') buf3 = '?'
  if(buf4 == '9') buf4 = '?'
  n = n + 1
  write(*,*) buf1,buf3,buf2,buf4,buf5,buf6
enddo
write(*,*) n
end
```

12,288 bytes

Other CIF parsing libraries

PyCifRW **James Hester – ANU implemented in Python**
http://www.ansto.gov.au/natfac/ANBF/CIF/



CIFLIB

C Class Library – Rutgers
***J. Appl. Cryst.* (1997) 30, 79-83.**

designed for mmCIF

GETSPEC

- written by I David Brown, McMaster
- incorporated into WinGX
- incorporated into LMPG software of Jean Laugier
- FORTRAN source code deposited at CCP14

This program calculates the symmetry operators (general positions) and special positions for any setting of any space group based on the Hall space group symbol

SGINFO (C code) – Ralf Grosse-Kunstleve → CCTBX

GETSPEC

incorporated into WinGX as a DLL

public functions load data (Hall symbol) and retrieve space group information

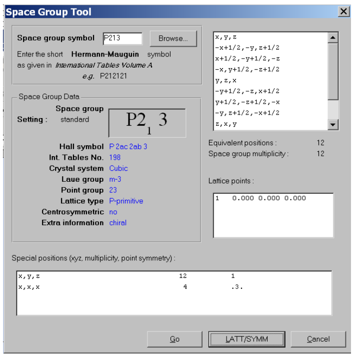
integer function gspHall(Hall_symbol)
interprets Hall symbol and loads Seitz matrices in memory

integer function gspNsym()
returns number of independent symmetry operations

integer function gspNLatt()
returns number of lattice translations

subroutine gspSymXYZ(n,string)
returns xyz notation for symmetry operations into character array *string(n)*

GETSPEC



Space group symbol P213
Enter the short **Hermann-Mauguin** symbol as given in *International Tables Volume A* *e.g.* P212121

Space Group Data
Space group
Setting : standard
$P2_1 3$
Hall symbol P 2ac 2ab 3
Int. Tables No. 198
Crystal system Cubic
Laue group m-3
Point group 23
Lattice type P-primitive
Centrosymmetric no
Extra information chiral

```
x,y,z
-x+1/2,-y,z+1/2
x+1/2,-y+1/2,-z
-x,y+1/2,-z+1/2
y,z,x
-y+1/2,-z,x+1/2
y+1/2,-z+1/2,-x
-y,z+1/2,-x+1/2
z,x,y
```

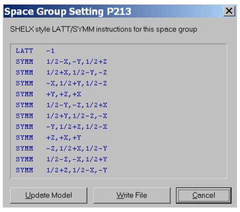
Equivalent positions : 12
Space group multiplicity : 12

Lattice points :

```
1   0.000 0.000 0.000
```

Special positions (xyz, multiplicity, point symmetry) :

```
x,y,z          12     1
x,x,x           4    .3.
```



SHELX style LATT/SYMM instructions for this space group

```
LATT   -1
SYMM   1/2-X,-Y,1/2+Z
SYMM   1/2+X,1/2-Y,-Z
SYMM   -X,1/2+Y,1/2-Z
SYMM   +Y,+Z,+X
SYMM   1/2-Y,-Z,1/2+X
SYMM   1/2+Y,1/2-Z,-X
SYMM   -Y,1/2+Z,1/2-X
SYMM   +Z,+X,+Y
SYMM   -Z,1/2+X,1/2-Y
SYMM   1/2-Z,-X,1/2+Y
SYMM   1/2+Z,1/2-X,-Y
```

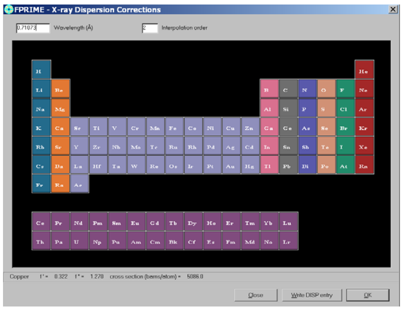
Read Hermann-Mauguin symbol

Look-up table -> Hall symbol

Load Hall symbol into DLL

Public functions query SG

FPRIME


FPRIME - X-ray Dispersion Corrections

0.71073 Wavelength (Å)

2 Interpolation order

Copper f' = 0.322 f" = 1.270 cross section (barns/atom) = 5086.0

H He
Li Be B C N O F Ne
Na Mg Al Si P S Cl Ar
K Ca Sc Ti V Cr Mn Fe Co Ni Cu Zn Ga Ge As Se Br Kr
Rb Sr Y Zr Nb Mo Tc Ru Rh Pd Ag Cd In Sn Sb Te I Xe
Cs Ba La Hf Ta W Re Os Ir Pt Au Hg Tl Pb Bi Po At Rn
Fr Ra Ac
Ce Pr Nd Pm Sm Eu Gd Tb Dy Ho Er Tm Yb Lu
Th Pa U Np Pu Am Cm Bk Cf Es Fm Md No Lr
Close
Write DISP entry
OK

Code by Bob von Dreele
Los Alamos N'tnl Lab

Also by Sean Brennan

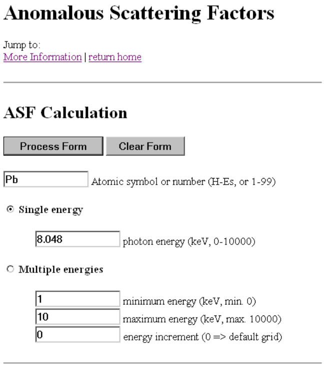
Both implemented in WinGX

Calculates f' and f'' and abs cross-section

FORTRAN sources

ftp://ftp.lanl.gov/public/gsas/windows
ftp://apollo.apsl.anl.gov/pub/cross-section_codes/

http://www-phys.llnl.gov/Research/scattering/asf.html


Anomalous Scattering Factors

Jump to:
More Information | return home

ASF Calculation

Pb Atomic symbol or number (H-Es, or 1-99)

● Single energy

8.048 photon energy (keV, 0-10000)

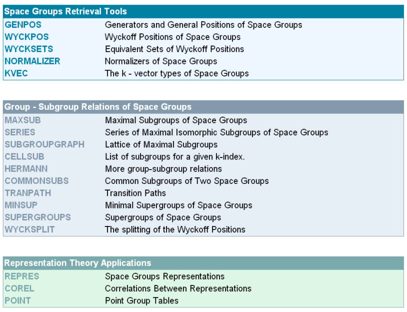
○ Multiple energies

1 minimum energy (keV, min. 0)
10 maximum energy (keV, max. 10000)
0 energy increment (0 => default grid)

Crystallographic Webservers

Bilbao Crystallographic Server
http://www.cryst.ehu.es/

[A Web Site with Crystallographic Tools Using the International Tables for Crystallography]



Space Groups Retrieval Tools	
GENPOS	Generators and General Positions of Space Groups
WYCKPOS	Wyckoff Positions of Space Groups
WYCKSETS	Equivalent Sets of Wyckoff Positions
NORMALIZER	Normalizers of Space Groups
KVEC	The k - vector types of Space Groups

Group - Subgroup Relations of Space Groups	
MAXSUB	Maximal Subgroups of Space Groups
SERIES	Series of Maximal Isomorphic Subgroups of Space Groups
SUBGROUPGRAPH	Lattice of Maximal Subgroups
CELLSUB	List of subgroups for a given k-index.
HERMANN	More group-subgroup relations
COMMONSUBS	Common Subgroups of Two Space Groups
TRANPATH	Transition Paths
MINSUP	Minimal Supergroups of Space Groups
SUPERGROUPS	Supergroups of Space Groups
WYCKSPLIT	The splitting of the Wyckoff Positions

Representation Theory Applications	
REPRES	Space Groups Representations
COREL	Correlations Between Representations
POINT	Point Group Tables

Crystallography source code archives & "museums"

•http://www.ccp14.ac.uk/ - mirror sites for much software

•http://sdpd.univ-lemans.fr/museum/ - Armel le Bail
•http://www.chem.gla.ac.uk/~paul/GX/ - GX source code
•http://www.ccp14.ac.uk/ccp/ccp14/ftp-mirror/larryfinger

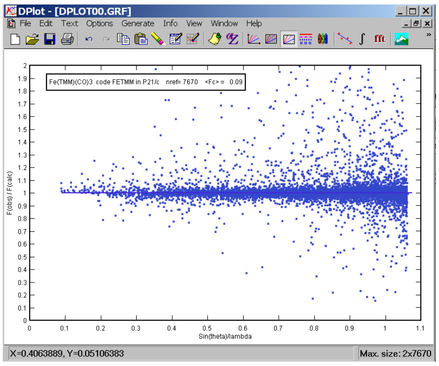
Application tools

Data plotting- DPLOT - http://www.dplot.com

Dplot is commercial software – scientific data plotting

a large number of formats

very flexible presentation


DPlot - [DPLOT00.GRF]

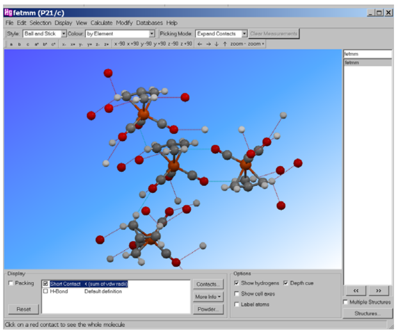
Fe(TMM)(CO)3 code FETMM in P21/c nref= 7670 <Fc> = 0.09

X=0.4063889, Y=0.05106383

Max. size: 2x7670

CCDC programs - http://www.ccdc.cam.ac.uk/

Mercury features include:
- Input of hit-lists from ConQuest, or other format files such as CIF, PDB, MOL2 and MOLfile
- Location and display of intermolecular and/or intramolecular hydrogen bonds, short nonbonded contacts, and user-specified types of contacts
- The ability to build and visualise a network of intermolecular contacts


fetmm (P21/c)
File Edit Selection Display View Calculate Modify Databases Help
Clear Measurements
a b c a* b* c* x- x+ y- y+ z- z+ x-90 x+90 y-90 y+90 z-90 z+90 ← → ↓ ↑ zoom - zoom +
fetmm
fetmm
Display
Reset
Contacts...
More Info
Powder...
Options
<< >>
Structures...

Style: Ball and Stick
Colour: by Element
Picking Mode: Expand Contacts

□ Packing
☑ Short Contact < (sum of vdw radii)
□ H-Bond Default definition

☑ Show hydrogens ☑ Depth cue
□ Show cell axes
□ Label atoms
□ Multiple Structures

Click on a red contact to see the whole molecule

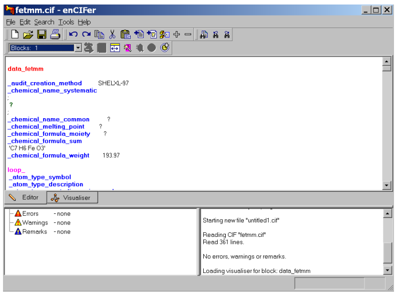
CCDC programs - http://www.ccdc.cam.ac.uk/

enCIFer - CIF checking, editing and visualisation software from the CCDC

CIF (Crystallographic Information File) is now a standard means of information exchange in crystallography.

It is also the recommended way of submitting data to the CSD.

enCIFer provides an intuitive, user-friendly interface to add information safely to the resultant CIF without corrupting the strict syntax.



```
data_fetmm

_audit_creation_method        SHELXL-97
_chemical_name_systematic
;
 ?
;
_chemical_name_common         ?
_chemical_melting_point       ?
_chemical_formula_moiety      ?
_chemical_formula_sum
 'C7 H6 Fe O3'
_chemical_formula_weight      193.97

loop_
 _atom_type_symbol
 _atom_type_description
```

Errors - none
Warnings - none
Remarks - none

Starting new file "untitled1.cif"

Reading CIF "fetmm.cif"
Read 361 lines.

No errors, warnings or remarks.

Loading visualiser for block: data_fetmm

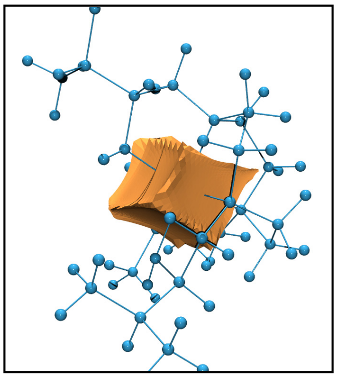
Ray traced graphics - http://www.povray.org

Ray traced graphics - http://www.povray.org

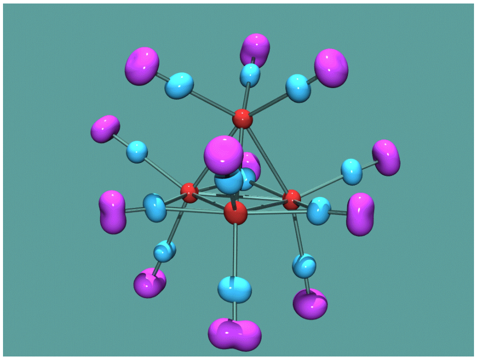
```
#version 3.6 ;
#include "colors.inc"
#include "textures.inc"
#include "metals.inc"
#declare  Surface_Texture = texture {
                    pigment {color Copper }
                    finish { Shiny } }
#declare    Atom_Texture = texture {
                    pigment {color SteelBlue }
                    finish { Shiny } }
#declare    Bond_Texture = texture {
                    pigment {color SteelBlue }
                    finish { Shiny } }
#declare    Atom_Radius  =   0.150;
#declare    Bond_Radius  =   0.020;
#declare   View_Distance  =  37.597;

global_settings { assumed_gamma 2.2 ambient_light rgb < 1, 1, 1 > }
camera {
       location < 0.0 , 0.0 , View_Distance >
       angle    20.0
       up      < 0.0 , 1.0 , 0.0 >
       right    <-1.33, 0.0 , 0.0 >
       look_at  < 0.0 , 0.0 , 0.0 >
      }
background { color White }
light_source { < 0.0,   0.0, 100.0 >
     color red 2.0 green 2.0 blue 2.0 }
light_source { < 0.0, 100.0,    0.0 >
     color red 2.0 green 2.0 blue 2.0 }

union {
    cylinder {<     0.0000,    0.0000,    0.0000 >
              <     1.2166,   -0.3406,    1.6317 > Bond_Radius
             texture { Bond_Texture } }
object {
        sphere  { < 0.0, 0.0, 0.0 >, Atom_Radius }
            texture   { Atom_Texture }
            scale     <     1.0000,    1.0000,    1.0000 >
            translate <     0.0000,    0.0000,    0.0000 >
  }
```



Ray traced graphics - http://www.povray.org



The adp's of $Rh_4(CO)_{12}$ shown as the RMSD surface (PEANUT plot)

Debugging tools

Static code analysers

•FTNCHEK http://www.dsm.fordham.edu/~ftnchek/ for FORTRAN code

•SPLINT http://www.splint.org/ for C & C++ code

ftnchek is a static analyzer for Fortran 77 programs. Its purpose is to assist the user in finding semantic errors. Semantic errors are legal in the Fortran language but are wasteful or may cause incorrect operation.

Splint is a tool for statically checking C programs for security vulnerabilities and coding mistakes.

SPAG – plusFORT – http://www.polyhedron.co.uk

Converts legacy FORTRAN-66 code into F77 or F95 compliant code – cures the “rats-nest” problem.

before

```
 148 IF(VIEW)152,150,152
 150 W(12,1)=1.
     W(12,2)=1.
     IF(W(6,1)-W(6,2))165,175,175
 152 DO 160 I=1,2
     DO 155 J=10,12
 155 W(J,I)=-W(J-6,I)
     W(12,I)=W(12,I)+VIEW
 160 W(13,I)=VV(W(10,I),W(10,I))
     IF(W(13,2)-W(13,1))165,175,175
```

after

```
100  if ( view/=0 ) then
        do i = 1,2
           do j = 10,12
              w(j,i) = -w(j-6,i)
           enddo
           w(12,i) = w(12,i) + view
            w(13,i) = vv(w(10,i),w(10,i))
        enddo
        if ( w(13,2)>=w(13,1) ) goto 200
     else
        w(12,1) = 1.
        w(12,2) = 1.
        if ( w(6,1)>=w(6,2) ) goto 200
     endif
```

Where to get software tools ?

- Web search engines
- CCP14 - http://www.ccp14.ac.uk/ a web-site with vast number of downloads and links
- SINCRIS - http://www-ext.lmcp.jussieu.fr/sincris-top/logiciel/