Structure Solution from Powder Data-I: direct methods

IUCr Commission on Crystallographic Computing

Bangalore 2017

Crystallographic Computing School

Mixed-Language Programs

Sir2014

EXPO2014

computational methods

FORTRAN

graphical user interface (GUI)

C++

QualX2

computational methods and GUI

FORTRAN

database communication

C++

Programming Enviroment

Hardware

Processor	: 12x Intel(R) Xeon(R) CPU E5-1650 0 @ 3.20GHz
Memory	: 15G
Operating System	: Ubuntu 16.04.2 LTS

Processor	: 32x Intel(R) Xeon(R) CPU E5-2690 0 @ 2.90GHz
Memory	: 62G
Operating System	: CentOS release 6.8 (Final)

MARCONI (System A1)@CINECA

Model:	: Lenovo NeXtScale
Racks	: 21
Nodes	: 1.512
Processors	: 2 x 18-cores Intel Xeon E5-2697 v4 (Broadwell) at 2.30 GHz
Cores	: 36 cores/node, 54.432 cores in total
RAM	: 128 GB/node, 3.5 GB/core
Peak Performance	: 2 PFlop/s

Editors

vi/vim, gedit, jedit, geany

Languages

Fortran, C++

Compilers

INTEL(ifort,icc), GNU (gcc,gfortran), PGI (pgf90,pgcc)

Debuggers and profilers

Totalview, IDB, gdb, Valgrind, gprof

Scientific libraries

IntelMPI, OpenMPI

FORTRAN

- Old language but still dominant for numerical computing and HPC
- Large amount of legacy code in crystallography is written in Fortran
- Fortran is easier to learn than C++
- Many revisions: 66, 77, 90, 95, 03, 08
- Dynamical allocation/deallocation and Fortran array handling features
- Object programming, operator overloading and polymorphism, inheritance

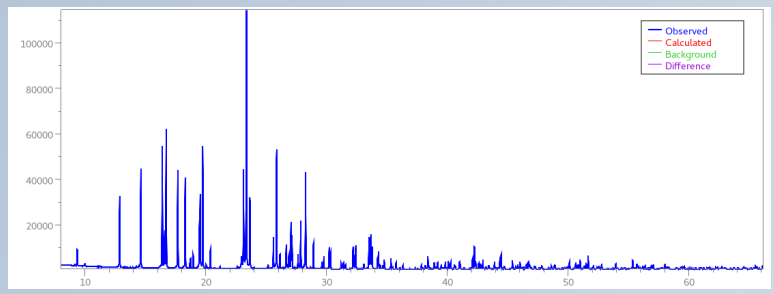
C++

- Object oriented,large standard library, boost library
- Containers
- Templates
- Database application and GUI

Cross platform GUI toolkit

- Qt
- WxWidgets
- GTK+
- Tk

Crystal Structure Determination Process



Experimental powder diffraction pattern

Indexing & space group determination

unit cell & space group

Structure solution

the biggest challenge

initial structural model

Rietveld method

final crystal structure

Methods of Structure Solution

Structure solution

Traditional approaches:

- direct methods
- Patterson methods

Direct space methods

Alternative words:real space, global optimization, global search

Other methods:

- charge flipping
- molecular replacement

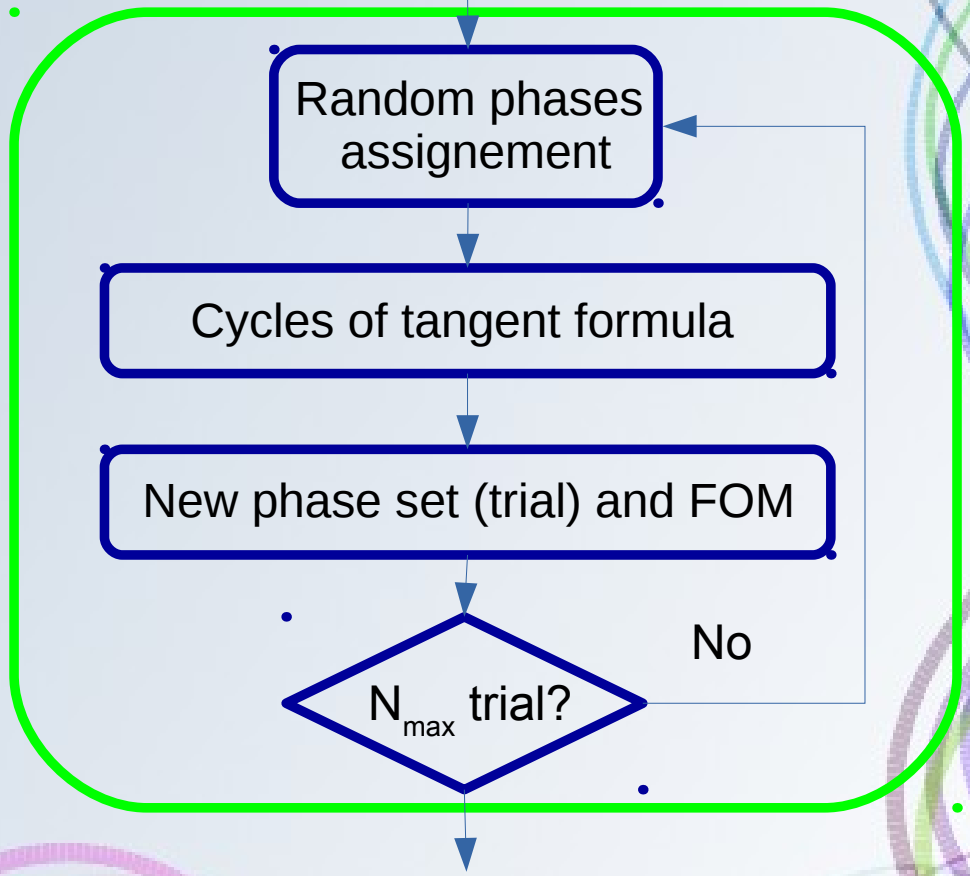
.....

W. I. F. David and K. Shankland. Structure determination from powder diffraction data. Acta Cryst. A64, 52 (2008)

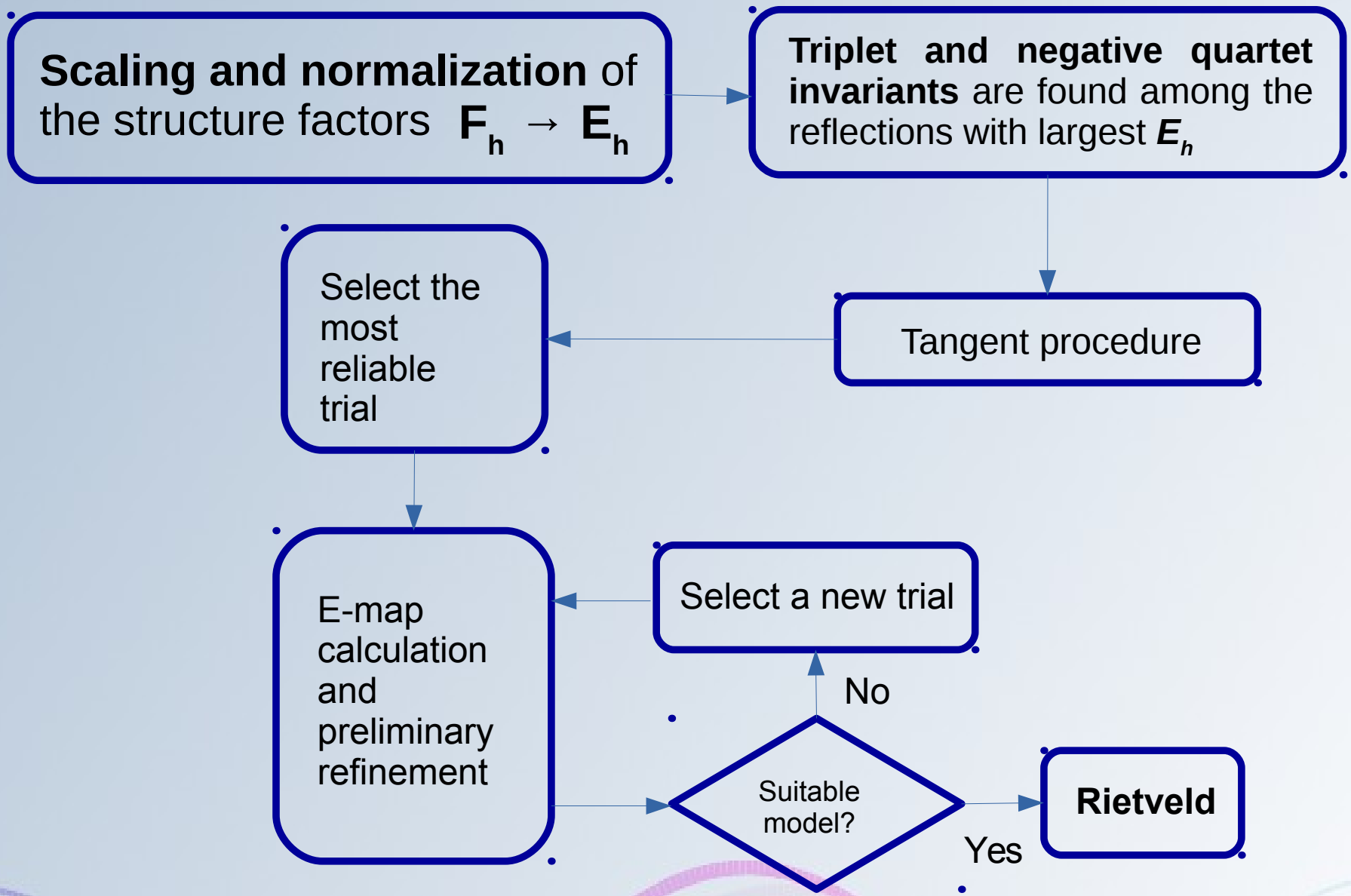
A Typical Direct Methods Procedure

Scaling and normalization of the structure factors $\mathbf{F_h} \rightarrow \mathbf{E_h}$

Triplet and negative quartet invariants are found among the reflections with largest $\mathbf{E_h}$



A Typical Direct Methods Procedure



Completion of the Crystal Structure and Preliminary Refinement

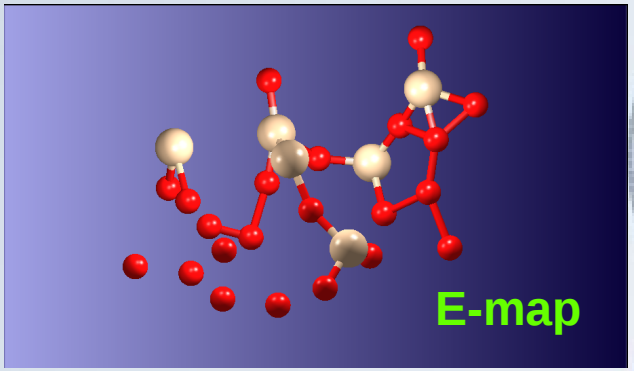
Partial structure
$\{\mathbf{X}, B\}$

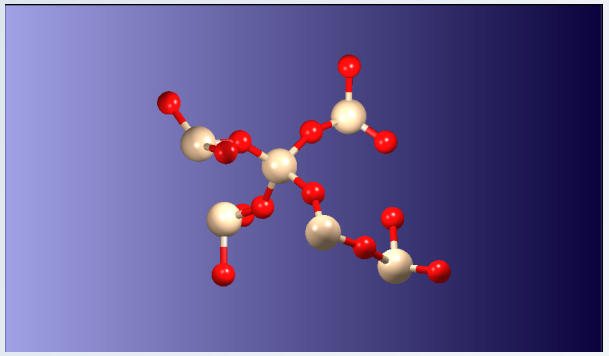
Structure factor calculation
$\{F_{calc}\}$

Least-squares refinement
$\{\mathbf{X}, B\} \rightarrow \{\mathbf{X}', B'\}$

Structure factor calculation
$\{F_{calc}\}$

Observed Fourier *
$F_{obs} e^{i\phi_{calc}}$





*F_{obs} may be replaced by $(F_{obs} - F_{calc})$ or by $(2F_{obs} - F_{calc})$

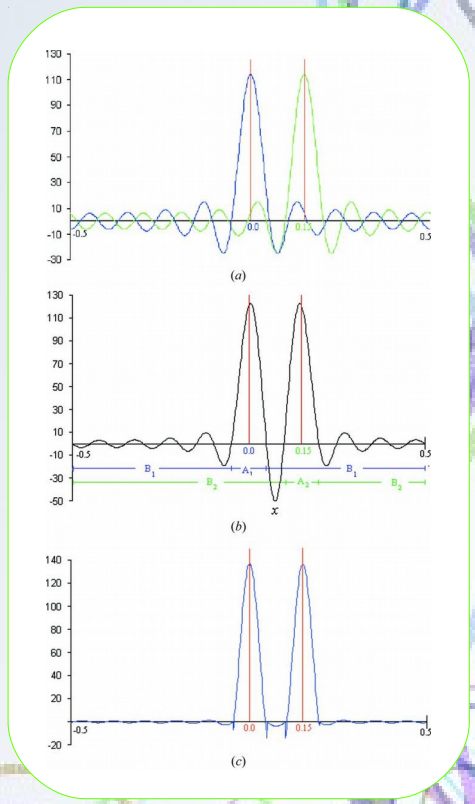
Resolution Bias Correction Algorithm (RBM)

Electron density of a crystal structure $\rho(\mathbf{r}) = \sum_{j=1}^{N} \rho_j(\mathbf{r} - \mathbf{r}_j)$

Calculated electron-density map $\rho'(\mathbf{r}) = \frac{1}{V} \sum_{\mathbf{h}} F_{\mathbf{h}} exp(-2\pi i \mathbf{h} \cdot \mathbf{r})$

Features of $\rho'(\mathbf{r})$

- Negative in more or less extended regions
- Atomic peaks show deformed profile and are surrounded by series of negative and positive ripples
- Atomic peak are shifted from the correct position $(\mathbf{r_j} \rightarrow \mathbf{r_j'})$



$\rho'(\mathbf{r})$ **RBM** $\rho'_{mod}(\mathbf{r}) \approx \rho(\mathbf{r})$

Altomare, A., Cuocci, C., Giacovazzo, C., Kamel, G.S., Moliterni, A. & Rizzi, R. (2008): Minimally resolution biased electron-density maps. Acta Cryst. A64: 326–336.

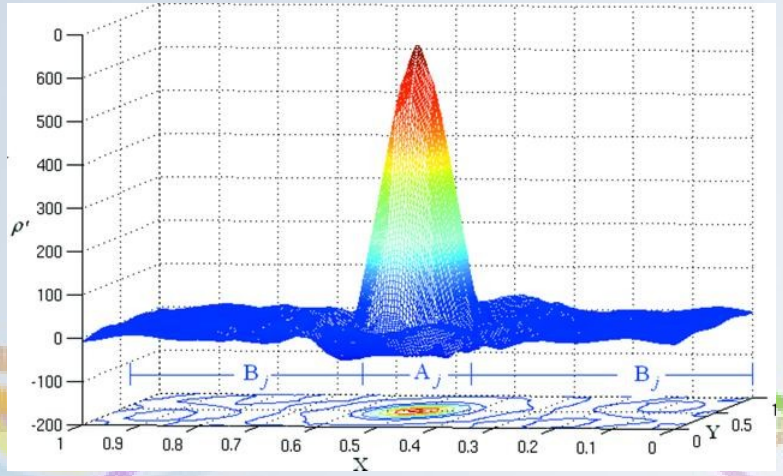
Resolution Bias Correction Algorithm (RBM)

$$\rho'(\mathbf{r}) = \rho(\mathbf{r}) \otimes T[\Phi(\mathbf{r}^*)] = \rho(\mathbf{r}) \otimes \zeta(\mathbf{r}) =$$
$$\sum_{j=1}^{N} \rho_j(\mathbf{r} - \mathbf{r}_j) \otimes \zeta(\mathbf{r}) = \sum_{j=1}^{N} \rho'_j(\mathbf{r} - \mathbf{r}'_j)$$

Each *j*-th atomic peak, in the electron density map, is replaced by a two-component function, constituted by the *main peak* and by the corresponding *ripples*

$$\rho'_j(\mathbf{r} - \mathbf{r}'_j) = \rho'_{[main]_j}(\mathbf{r} - \mathbf{r}'_j) + \rho'_{[ripples]_j}(\mathbf{r} - \mathbf{r}'_j)$$

$$\zeta(\mathbf{r} - \mathbf{r}'_j) = \zeta_{[A]_j}(\mathbf{r} - \mathbf{r}'_j) + \zeta_{[B]_j}(\mathbf{r} - \mathbf{r}'_j)$$



Resolution Bias Correction Algorithm (RBM)

The ripple contribution is subtracted

$$\rho''(\mathbf{r}) = \rho'(\mathbf{r}) - \sum_{j=1}^{N} \rho'_{[B]\,j}(\mathbf{r} - \mathbf{r}'_j) \simeq \sum_{j=1}^{N} [\rho'_j(\mathbf{r} - \mathbf{r}'_j) - c'_j \zeta_{[B]_\zeta}(\mathbf{r} - \mathbf{r}'_j)]$$

Fitting of atomic peaks with Gaussian function

$$\rho''(\mathbf{r}) \approx \sum_{j=1}^{N} c_j G((r); \sigma_j, \mathbf{r}_j)$$

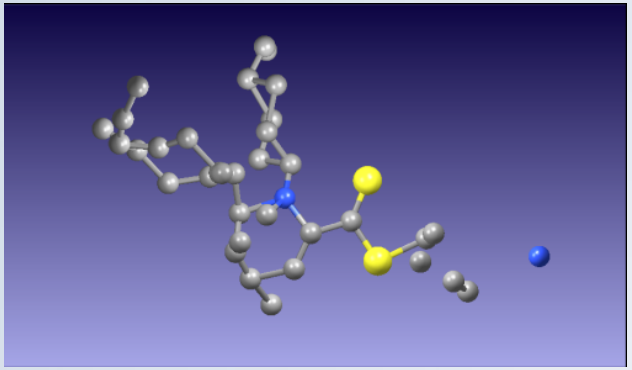
Default Strategy of Model Optimization in the EXPO Program

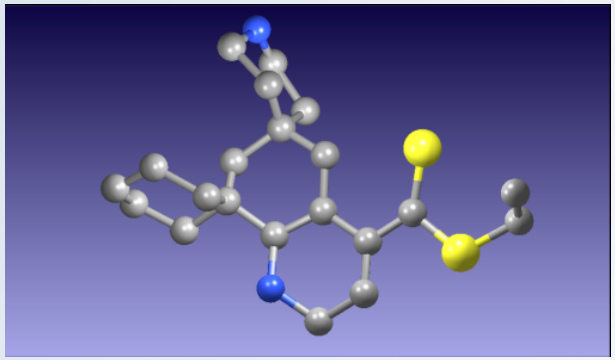
Partial structure
$\{\mathbf{X}, B\}$

Fourier map
$2F_{obs} - F_{calc}$

RBM

Least-squares refinement
$\{\mathbf{X}, B\} \rightarrow \{\mathbf{X}', B'\}$





Improvements In Electron Density-map Generation

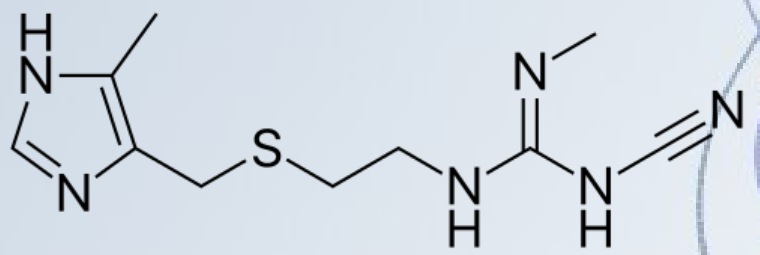
- **COVMAP** (*) exploits three kinds of information:
  - The chemical interpretation of the model peaks
  - Some basic crystal chemistry rules, essentially the bond distances expected for the pairs of pivot atoms.
  - The efficiency of the crystallographic residual $R_F = \sum_{\mathbf{h}} ||F_{obs} - F_{calc}|| / \sum_{\mathbf{h}} F_{obs}$

*Altomare, A., Cuocci, C., Giacovazzo, C., Moliterni, A. & Rizzi, R. (2012): COVMAP: a new algorithm for structure model optimization in the EXPO package. J. Appl. Cryst. 45: 789-797.

- The **Random-model-based method (RAMM)** (**) strategy skips the phasing step by Direct Methods and substitutes the Direct Methods model by a starting fully random model driven towards the correct solution by cyclic combination of **COVMAP** → **WLSQ** → **FT** → **RBM**.

**Altomare, A., Cuocci, Moliterni, A. & Rizzi, R. (2013): RAMM: a random-model-based method for solving ab initio crystal structure using EXPO package. J. Appl. Cryst. 46: 476-482.

```
   program example2
!
!  Crystal structure solution of cimetidine by direct methods from synchrotron powder diffraction
!
   use iso_fortran_env, only: stdout => OUTPUT_UNIT,ERROR_UNIT
   use gen_frm
   use commandsmod
   use errormod
   use crystal_phase
   use expo_main, only: param_options_type,spoolr
   use General, only: lo,POW_FILE,iasun
   use molcom, only: kscreen
   use variables, only: cryst
   implicit none
   type(error_type)             :: err
   type(param_options_type)     :: param_opt
   integer                      :: iend,ier,newf,active_file
   character(len=*), parameter :: expofold = '../files/'
!
!  Initialize libexpo
   call InitExpo2002(0)
   lo = stdout
   kscreen = 0
   iasun = expofold//'expo.spg'
   call load_chemical_tables(expofold,err)
   if (err%signal) then
       write(ERROR_UNIT,'(a)')' Message: '//err%msg()
       stop 1
   endif
!
!  Set up commands for structure solution of cimetidine
   call new_cmd(commands,2)
   call commands(1)%set('data',['pattern cimetidine.dat',                                &
                                'wave 1.52904','synchrotron',                            &
                                'cell 10.6986  18.8181 6.8246 90.000   111.284    90.000', &
                                'space P 21/n',                                          &
                                'cont   (C10H16N6S)4'])
   call commands(2)%set('continue')
   call spoolr(active_file,newf,param_opt,ier)
!
!  Execute commands
   call gescom(POW_FILE,iend,ier)
   if (ier /= 0) stop 2
!
   call cryst(1)%print(stdout)
   call crystal_file_export(cryst(1),'cimetidine.cif')
!
   end program example2
```



cimetidine

$C_{10}H_{16}N_6S$

Libexpo Installation

- Download the file `libexpo-1.17.08.tar.gz` from http://www.ba.ic.cnr.it/softwareic/expo/tutorials-and-lectures/

- On Ubuntu 14.04 LTS and subsequent versions all the required dependencies can be installed running the following command:

```
sudo apt install g++ automake autoconf gfortran libcurl4-gnutls-dev libgtk-3-dev libgl1-mesa-dev libopenbabel-dev
```

- Get the sources and make the libexpo library:

```
tar xvfz libexpo-1.17.08.tar.gz
mkdir libexpo
cd libexpo
../libexpo-1.17.08/configure
make
```

Programming Examples

- Download the file `libexpo_examples.tar.gz` from www.ba.ic.cnr.it/softwareic/expo/expo2014-download/

- Compile the `example1`

```
tar xvzf libexpo_examples.tar.gz
cd libeexpo_examples
cd example1
make
```

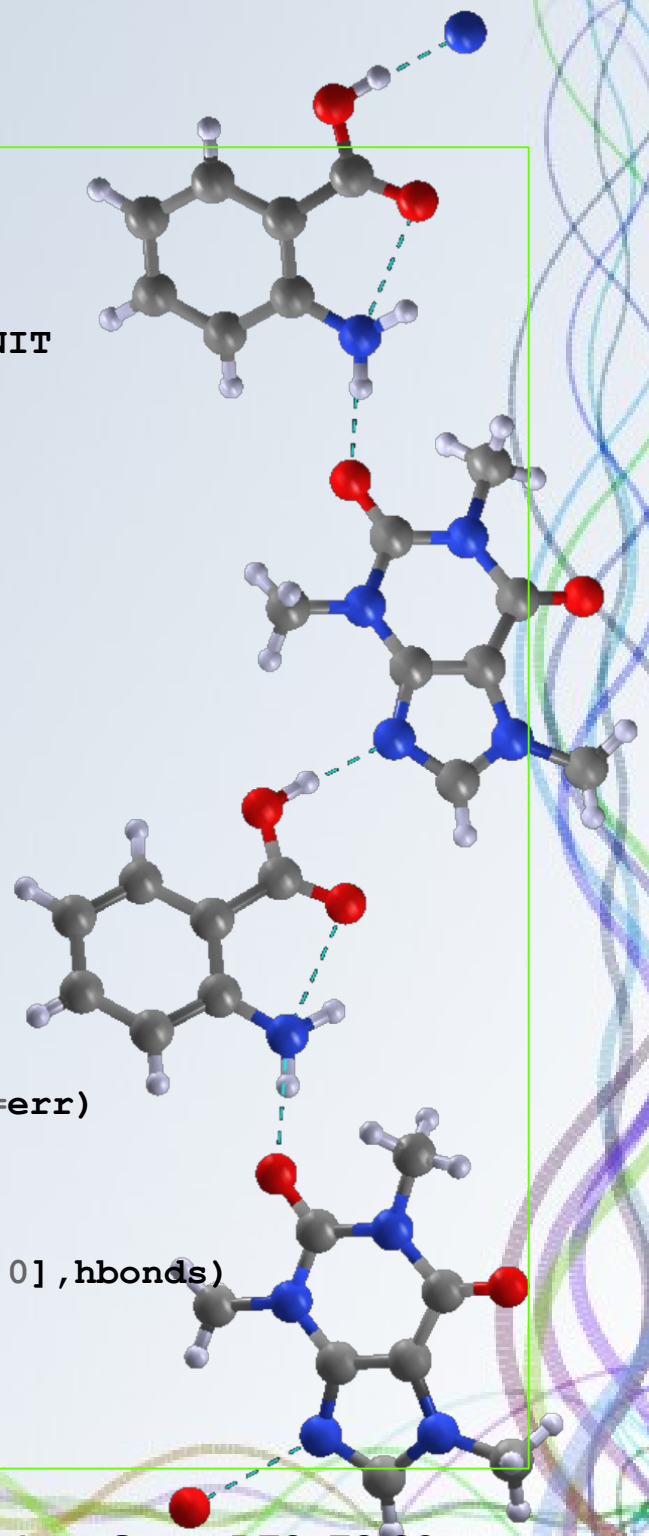
- Makefile used to compile the `example1`

```
LIBEXPODIR = $(HOME)/libexpo
FCOM = gfortran
FOPT = -O2 -I $(LIBEXPODIR) -L $(LIBEXPODIR)
LOPT = -lexpo `pkg-config --libs gtk+-3.0` -lX11 -lstdc++ -lGL -lcurl -lopenbabel

testlib:        example1.f90
        $(FCOM) -o example1 example1.f90 $(FOPT) $(LOPT)
```

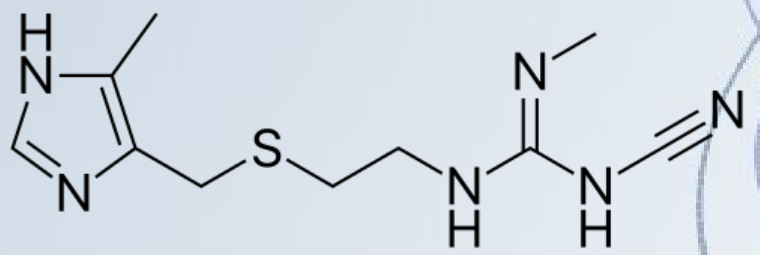
example1

```
   program example1
!
!  Find hydrogen bonds
!
   use iso_fortran_env, only: stdout => OUTPUT_UNIT, ERROR_UNIT
   use crystal_phase
   use gen_frm
   use errormod
   use connect_mod
   implicit none
   type(crystal_phase_t)                          :: crystal
   type(error_type)                               :: err
   type(atom_type), dimension(:), allocatable :: atoms
   type(bond_type), dimension(:), allocatable :: bonds
   type(bond_type), dimension(:), allocatable :: hbonds
!
   call load_chemical_tables('../files/',err)
   if (err%signal) then
       write(ERROR_UNIT,'(a)')' Message: '//err%msg()
       stop 1
   endif
!
   call crystal_file_import(crystal,'caf_ana_formII.cif',err=err)
   if (err%signal) stop 2
   call crystal%print(stdout)
!
   call crystal_find_contacts(crystal,atoms,bonds,.false.,[0,0],hbonds)
   call print_connect(atoms%lab,legm=hbonds,kpri=stdout)
!
   end program example1
```



Madusanka, N., Eddleston, M. D., Arhangelskis, M. & Jones, W. (2014). Acta Cryst. B70, 72-80.

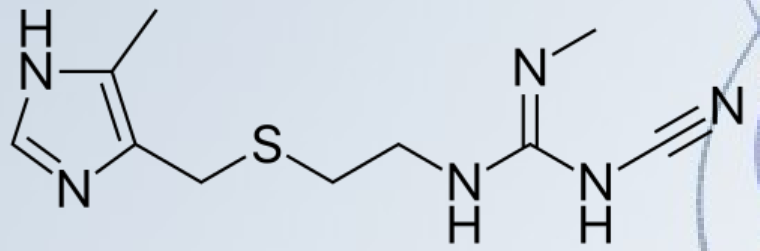
```
   program example2
!
!  Crystal structure solution of cimetidine by direct methods from synchrotron powder diffraction
!
   use iso_fortran_env, only: stdout => OUTPUT_UNIT,ERROR_UNIT
   use gen_frm
   use commandsmod
   use errormod
   use crystal_phase
   use expo_main, only: param_options_type,spoolr
   use General, only: lo,POW_FILE,iasun
   use molcom, only: kscreen
   use variables, only: cryst
   implicit none
   type(error_type)             :: err
   type(param_options_type)     :: param_opt
   integer                      :: iend,ier,newf,active_file
   character(len=*), parameter :: expofold = '../files/'
!
!  Initialize libexpo
   call InitExpo2002(0)
   lo = stdout
   kscreen = 0
   iasun = expofold//'expo.spg'
   call load_chemical_tables(expofold,err)
   if (err%signal) then
       write(ERROR_UNIT,'(a)')' Message: '//err%msg()
       stop 1
   endif
!
!  Set up commands for structure solution of cimetidine
   call new_cmd(commands,2)
   call commands(1)%set('data',['pattern cimetidine.dat',                                &
                                'wave 1.52904','synchrotron',                            &
                                'cell 10.6986  18.8181 6.8246 90.000   111.284    90.000', &
                                'space P 21/n',                                          &
                                'cont   (C10H16N6S)4'])
   call commands(2)%set('continue')
   call spoolr(active_file,newf,param_opt,ier)
!
!  Execute commands
   call gescom(POW_FILE,iend,ier)
   if (ier /= 0) stop 2
!
   call cryst(1)%print(stdout)
   call crystal_file_export(cryst(1),'cimetidine.cif')
!
   end program example2
```



cimetidine

$C_{10}H_{16}N_6S$

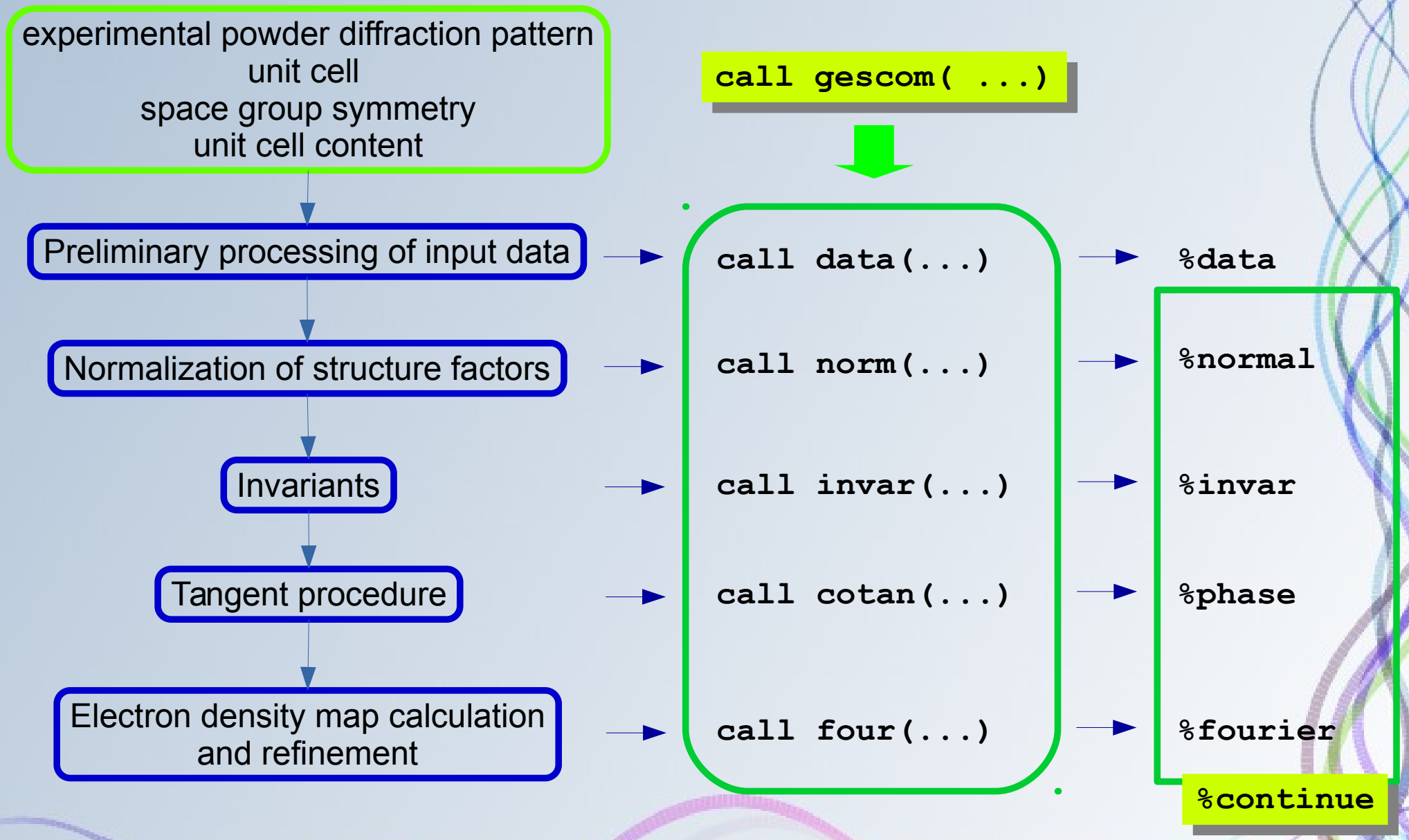
```
program example2
!
!  Crystal structure solution of cimetidine by direct methods from synchrotron powder diffraction
!
   use iso_fortran_env, only: stdout => OUTPUT_UNIT,ERROR_UNIT
   use gen_frm
   use commandsmod
   use errormod
   use crystal_phase
   use expo_main, only: param_options_type,spoolr
   use General, only: lo,POW_FILE,iasun
   use molcom, only: kscreen
   use variables, only: cryst
   implicit none
   type(error_type)              :: err
   type(param_options_type)      :: param_opt
   integer                       :: iend,ier,newf,active_file
   character(len=*), parameter   :: expofold = '../files/'
!
!  Initialize libexpo
   call InitExpo2002(0)
   lo = stdout
   kscreen = 0
   iasun = expofold//'expo.spg'
   call load_chemical_tables(expofold,err)
   if (err%signal) then
      write(ERROR_UNIT,'(a)')' Message: '//err%msg()
      stop 1
   endif
!  Set up commands for structure solution of cimetidine
   call new_cmd(commands,2)
   call commands(1)%set('data',['pattern cimetidine.dat',                                &
                                'wave 1.52904','synchrotron',                            &
                                'cell 10.6986  18.8181 6.8246 90.000   111.284    90.000', &
                                'space P 21/n',                                          &
                                'cont   (C10H16N6S)4'])
   call commands(2)%set('continue')
   call spoolr(active_file,newf,param_opt,ier)
!
!  Execute commands
   call gescom(POW_FILE,iend,ier)
   if (ier /= 0) stop 2
!
   call cryst(1)%print(stdout)
   call crystal_file_export(cryst(1),'cimetidine.cif')
!
   end program example2
```



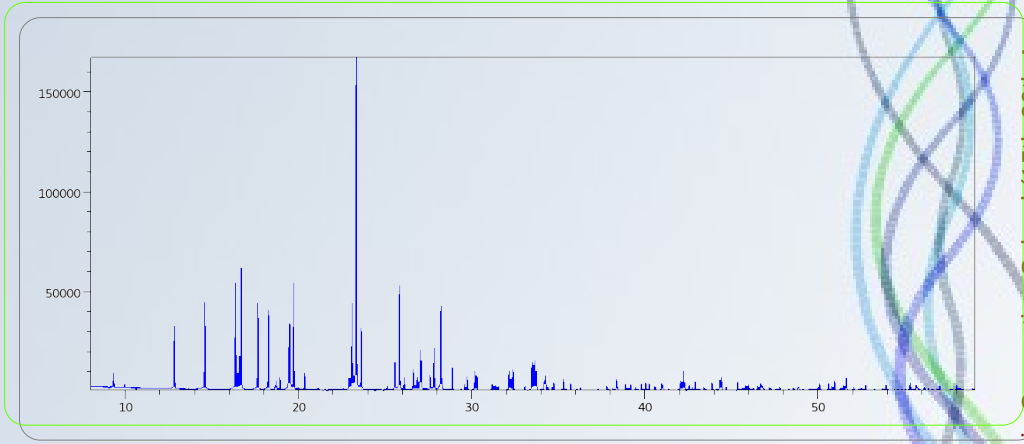
cimetidine

$C_{10}H_{16}N_6S$

Expo default strategy for crystal structure solution

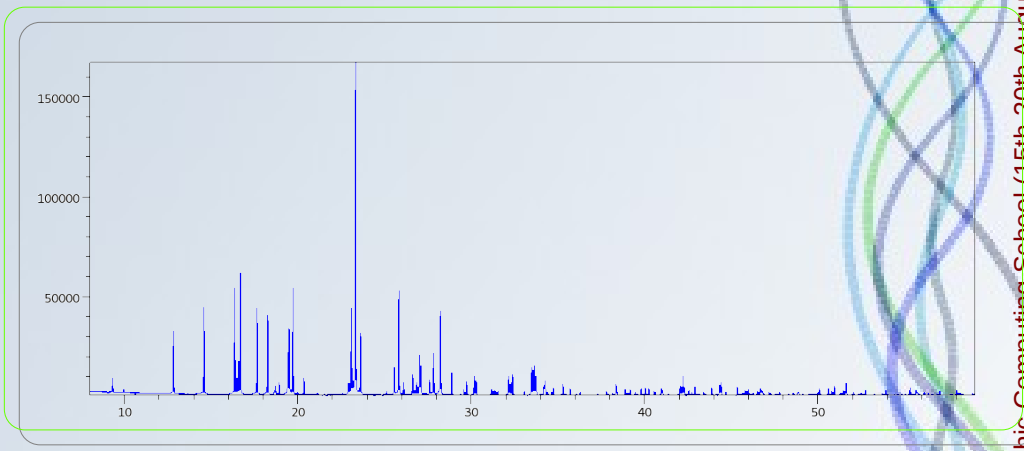


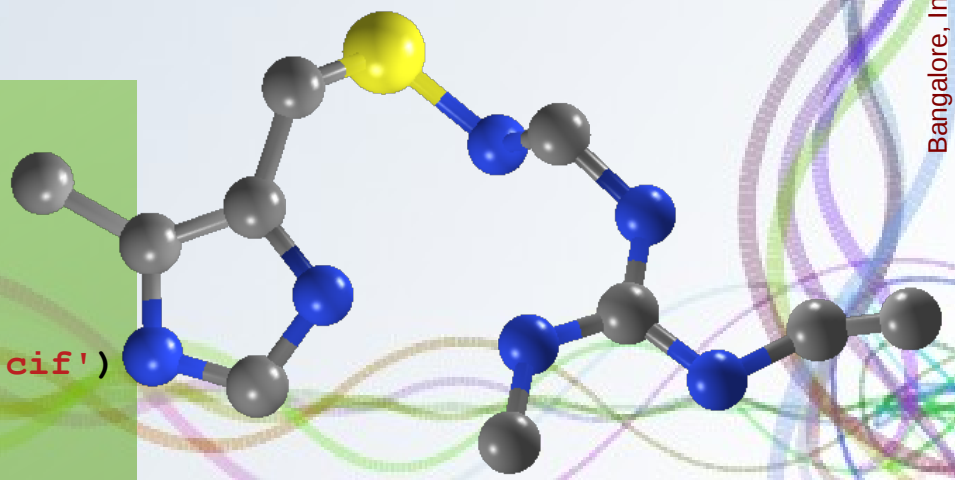
```
    program example2
!
!   Crystal structure solution of cimetidine by direct methods from synchrotron powder diffraction
!
    use iso_fortran_env, only: stdout => OUTPUT_UNIT,ERROR_UNIT
    use gen_frm
    use commandsmod
    use errormod
    use crystal_phase
    use expo_main, only: param_options_type,spoolr
    use General, only: lo,POW_FILE,iasun
    use molcom, only: kscreen
    use variables, only: cryst
    implicit none
    type(error_type)              :: err
    type(param_options_type)      :: param_opt
    integer                       :: iend,ier,newf,active_file
    character(len=*), parameter :: expofold = '../files/'
!
!   Initialize libexpo
    call InitExpo2002(0)
    lo = stdout
    kscreen = 0
    iasun = expofold//'expo.spg'
    call load_chemical_tables(expofold,err)
    if (err%signal) then
        write(ERROR_UNIT,'(a)')' Message: '//err%msg()
        stop 1
    endif
```



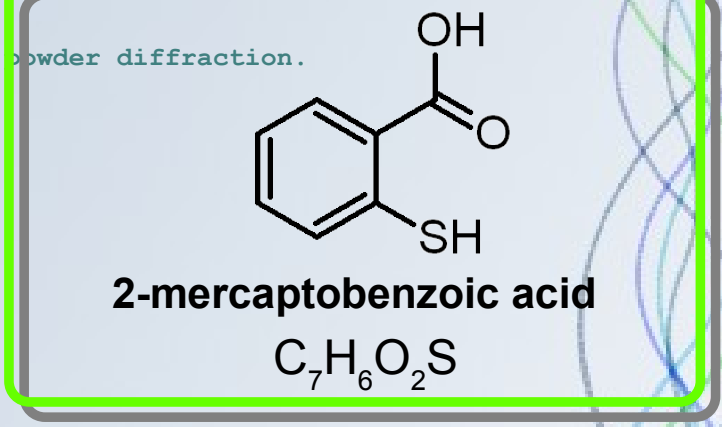

```
!
!   Set up commands for structure solution of cimetidine
    call new_cmd(commands,2)
    call commands(1)%set('data',['pattern cimetidine.dat',
                                 'wave 1.52904','synchrotron',
                                 'cell 10.6986  18.8181 6.8246 90.000   111.284    90.000',
                                 'space P 21/n',
                                 'cont   (C10H16N6S)4'])
    call commands(2)%set('continue')
    call spoolr(active_file,newf,param_opt,ier)
!
!   Execute commands
    call gescom(POW_FILE,iend,ier)
    if (ier /= 0) stop 2
!
    call cryst(1)%print(stdout)
    call crystal_file_export(cryst(1),'cimetidine.cif')
!
    end program example2
```

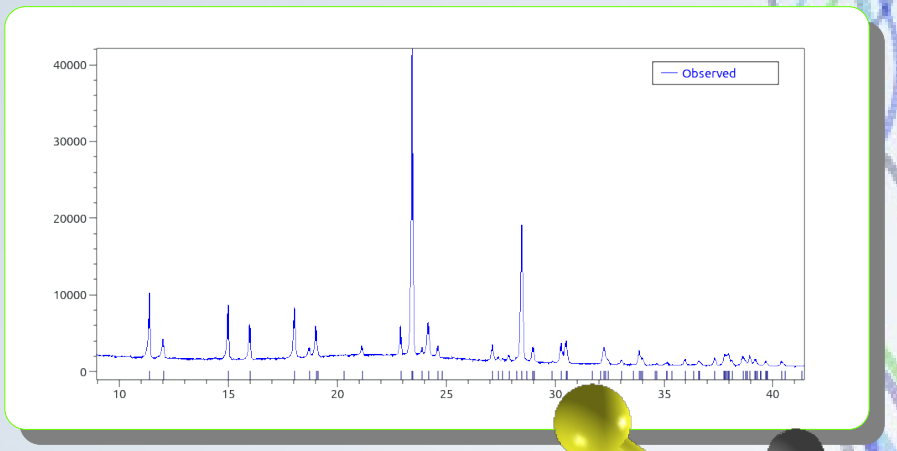
```
   program example2
!
!  Crystal structure solution of cimetidine by direct methods from synchrotron powder diffraction
!
   use iso_fortran_env, only: stdout => OUTPUT_UNIT,ERROR_UNIT
   use gen_frm
   use commandsmod
   use errormod
   use crystal_phase
   use expo_main, only: param_options_type,spoolr
   use General, only: lo,POW_FILE,iasun
   use molcom, only: kscreen
   use variables, only: cryst
   implicit none
   type(error_type)             :: err
   type(param_options_type)     :: param_opt
   integer                      :: iend,ier,newf,active_file
   character(len=*), parameter :: expofold = '../files/'
!
!  Initialize libexpo
   call InitExpo2002(0)
   lo = stdout
   kscreen = 0
   iasun = expofold//'expo.spg'
   call load_chemical_tables(expofold,err)
   if (err%signal) then
       write(ERROR_UNIT,'(a)')' Message: '//err%msg()
       stop 1
   endif
!
!  Set up commands for structure solution of cimetidine
   call new_cmd(commands,2)
   call commands(1)%set('data',['pattern cimetidine.dat',                          &
                                'wave 1.52904','synchrotron',                      &
                                'cell 10.6986  18.8181 6.8246 90.000   111.284    90.000', &
                                'space P 21/n',                                    &
                                'cont   (C10H16N6S)4'])
   call commands(2)%set('continue')
   call spoolr(active_file,newf,param_opt,ier)
!
!  Execute commands
   call gescom(POW_FILE,iend,ier)
   if (ier /= 0) stop 2
!
   call cryst(1)%print(stdout)
   call crystal_file_export(cryst(1),'cimetidine.cif')
!
   end program example2
```

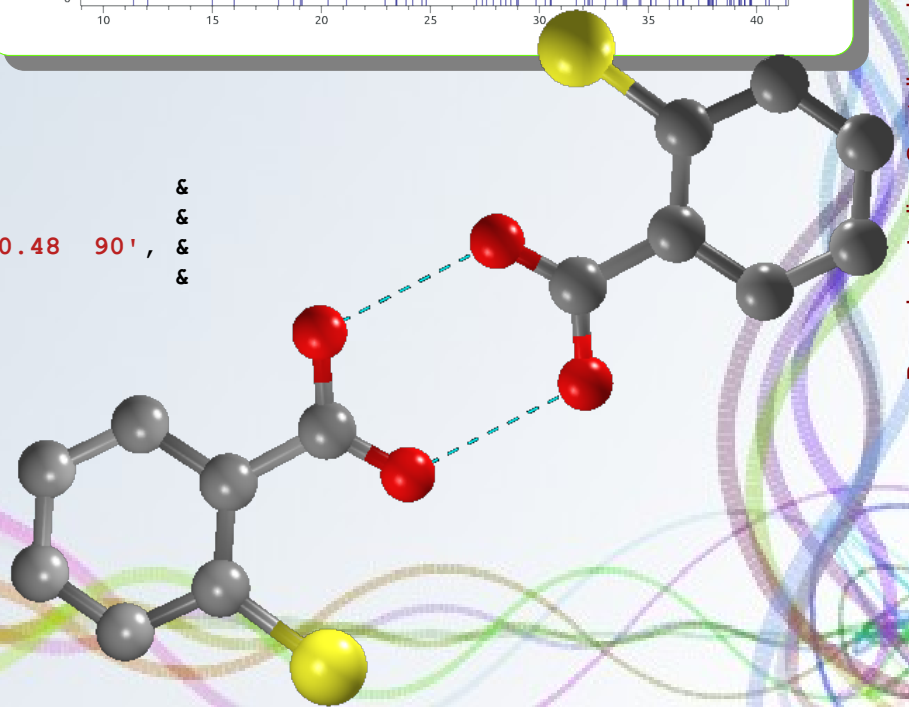




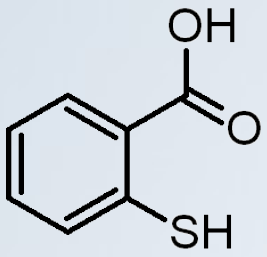
```
   program example3
!
!  Crystal structure solution of 2-mercaptobenzoic acid by direct methods from powder diffraction.
!  'Alltrials' strategy is used.
!
   use iso_fortran_env, only: stdout => OUTPUT_UNIT,ERROR_UNIT
   use gen_frm
   use commandsmod
   use errormod
   use crystal_phase
   use expo_main, only: param_options_type,spoolr
   use General, only: lo,POW_FILE,iasun
   use molcom, only: kscreen
   use variables, only: cryst
   implicit none
   type(error_type)             :: err
   type(param_options_type)     :: param_opt
   integer                      :: iend,ier,newf,active_file
   character(len=*), parameter :: expofold = '../files/'
!
!  Initialize libexpo
   call InitExpo2002(0)
   lo = stdout
   kscreen = 0
   iasun = expofold//'expo.spg'
   call load_chemical_tables(expofold,err)
   if (err%signal) then
       write(ERROR_UNIT,'(a)')' Message: '//err%msg()
       stop 1
   endif
!
!  Set up commands for structure solution of cimetidine
   call new_cmd(commands,2)
   call commands(1)%set('data',['pattern merca.xy',                          &
                                'wave 1.54056',                              &
                                'cell 7.885  5.976   14.949  90.0  100.48  90', &
                                'space P 21/c',                              &
                                'cont (C7H6O2S)4'])
   call commands(2)%set('alltrials')
   call spoolr(active_file,newf,param_opt,ier)
!
!  Execute commands
   call gescom(POW_FILE,iend,ier)
   if (ier /= 0) stop 2
!
   call cryst(1)%print(stdout)
   call crystal_file_export(cryst(1),'merca.cif')
!
   end program example3
```





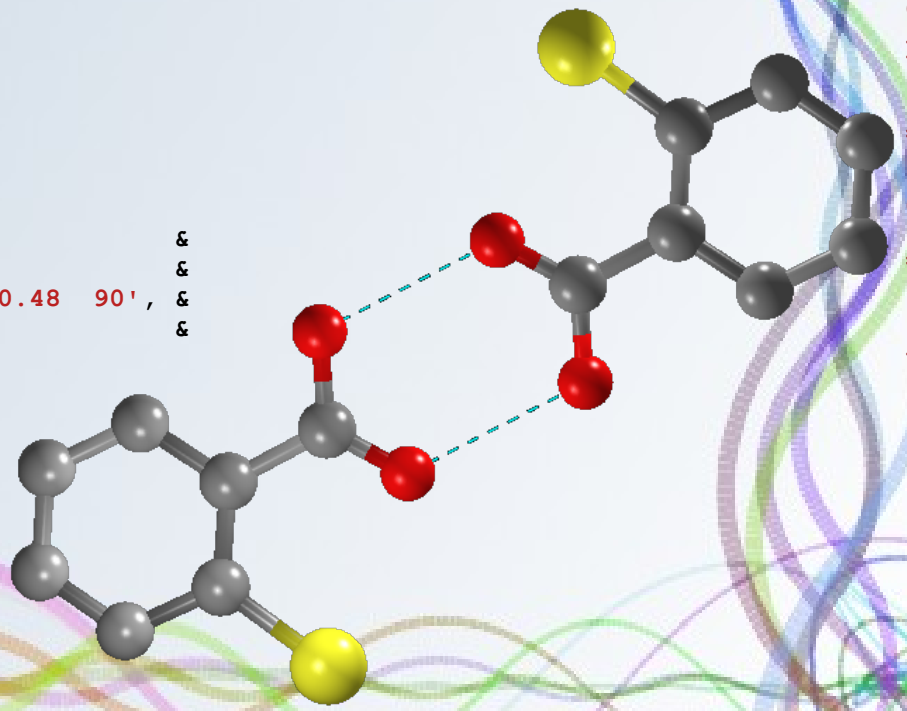


```
   program example3
!
!  Crystal structure solution of 2-mercaptobenzoic acid by direct methods from powder diffraction.
!  'Alltrials' strategy is used.
!
   use iso_fortran_env, only: stdout => OUTPUT_UNIT,ERROR_UNIT
   use gen_frm
   use commandsmod
   use errormod
   use crystal_phase
   use expo_main, only: param_options_type,spoolr
   use General, only: lo,POW_FILE,iasun
   use molcom, only: kscreen
   use variables, only: cryst
   implicit none
   type(error_type)             :: err
   type(param_options_type)     :: param_opt
   integer                      :: iend,ier,newf,active_file
   character(len=*), parameter :: expofold = '../files/'
!
!  Initialize libexpo
   call InitExpo2002(0)
   lo = stdout
   kscreen = 0
   iasun = expofold//'expo.spg'
   call load_chemical_tables(expofold,err)
   if (err%signal) then
       write(ERROR_UNIT,'(a)')' Message: '//err%msg()
       stop 1
   endif
!
!  Set up commands for structure solution of cimetidine
   call new_cmd(commands,2)
   call commands(1)%set('data',['pattern merca.xy',                    &
                                'wave 1.54056',                        &
                                'cell 7.885  5.976   14.949  90.0  100.48  90', &
                                'space P 21/c',                        &
                                'cont (C7H6O2S)4'])
   call commands(2)%set('alltrials')
   call spoolr(active_file,newf,param_opt,ier)
!
!  Execute commands
   call gescom(POW_FILE,iend,ier)
   if (ier /= 0) stop 2
!
   call cryst(1)%print(stdout)
   call crystal_file_export(cryst(1),'merca.cif')
!
   end program example3
```



2-mercaptobenzoic acid

$C_7H_6O_2S$

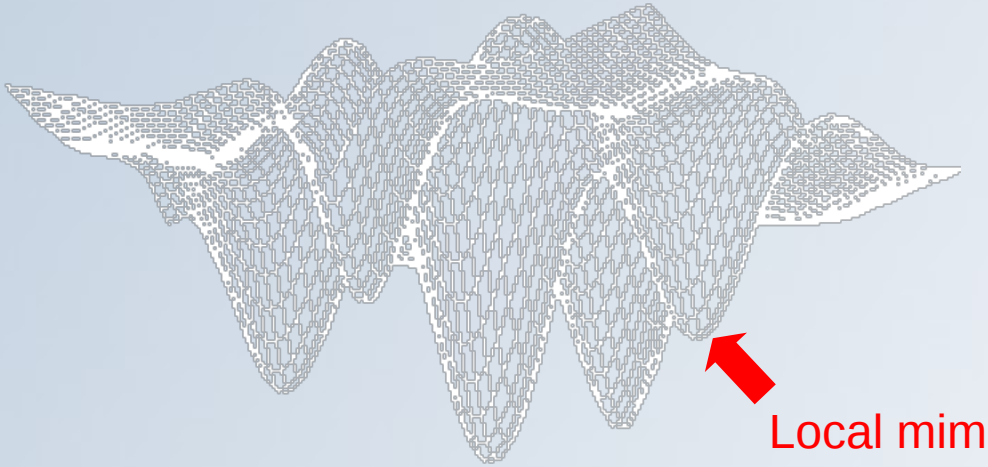


Global Optimization Methods

Find $\mathbf{x}^* = \min\{F(\mathbf{x})\}$, where $\mathbf{F}: \mathbb{R}^n \rightarrow \mathbb{R}$

X = fractional coordinates of (*x,y,z*) *or*

X = postion (x,y,z), orientation (θ, φ, ψ), torsion angles (τ_1, τ_2, … τ_n) of molecular fragments



Local optimization methods → **Structure refinement**

Global optimization methods → **Structure solution**

Global optimization methods

- Deterministic methods
  - *Branch and Bound methods*
  - *Cutting Plane methods*
  - *Interval methods*
  - *…….*

- Heuristic strategies
  - *Genetic Algorithms* (GA)
  - *Simulated Annealing* (SA)
  - *Tabu Search*
  - *Ant Colony Optimization*
  - *Particle Swarm Optimization* (PS)
  - *Bee Algorithms*
  - *Firefly Algorithms*
  - *Harmony Search*
  - *Big Bang-Big Crunch*
  - *……*

Global optimization methods

- Deterministic methods
  - *Branch and Bound methods*
  - *Cutting Plane methods*
  - *Interval methods*
  - *…….*

- Heuristic strategies
  - *Genetic Algorithms* (GA) *
  - *Simulated Annealing* (SA)*
  - *Tabu Search*
  - *Ant Colony Optimization*
  - *Particle Swarm Optimization* (PS)*
  - *Bee Algorithms*
  - *Firefly Algorithms*
  - *Harmony Search*
  - *Big Bang-Big Crunch* *
  - *……*

(*) employed in solving crystal structure

Global optimization methods

- Deterministic methods
  - *Branch and Bound methods*
  - *Cutting Plane methods*
  - *Interval methods*
  - *.....*

- Heuristic strategies
  - *Genetic Algorithms* (GA) *
  - ***Simulated Annealing (SA)****
  - *Tabu Search*
  - *Ant Colony Optimization*
  - *Particle Swarm Optimization* (PS)*
  - *Bee Algorithms*
  - *Firefly Algorithms*
  - *Harmony Search*
  - *Big Bang-Big Crunch **
  - *.....*

Widely used and with the largest impact

Various modifications:
- *parallel tempering* (PT)
- *adaptive simulated annealing*

(*) employed in solving crystal structure

Global optimization methods

- Deterministic methods
  - *Branch and Bound methods*
  - *Cutting Plane methods*
  - *Interval methods*
  - *.....*

- Heuristic strategies
  - *Genetic Algorithms* (GA) *
  - ***Simulated Annealing (SA)****
  - *Tabu Search*
  - *Ant Colony Optimization*
  - *Particle Swarm Optimization* (PS)*
  - *Bee Algorithms*
  - *Firefly Algorithms*
  - *Harmony Search*
  - *Big Bang-Big Crunch **
  - *.....*

Widely used and with the largest impact

Various modifications:
- *parallel tempering* (PT)
- *adaptive simulated annealing*

(*) employed in solving crystal structure

Global optimization methods

- Deterministic methods
  - *Branch and Bound methods*
  - *Cutting Plane methods*
  - *Interval methods*
  - *…….*

- Heuristic strategies
  - *Genetic Algorithms* (GA) *
  - ***Simulated Annealing (SA)****
  - *Tabu Search*
  - *Ant Colony Optimization*
  - *Particle Swarm Optimization* (PS)*
  - *Bee Algorithms*
  - *Firefly Algorithms*
  - *Harmony Search*
  - *Big Bang-Big Crunch **
  - *…….*

(*) employed in solving crystal structure

Widely used and with the largest impact

Various modifications:
- *parallel tempering* (PT)
- *adaptive simulated annealing*

***Software* **:** DASH (SA), EXPO (SA), ENDEAVOUR (SA), FOX (PT), GEST (GA), PeckCryst (PS), PowderSolve (SA), PSSP (SA), TOPAS (SA), …

commercial software are in red

(**) Cerny, R. & Favre-Nicolin, V. (2007). Z. Kristallogr. 222, 105–113.

Codes for Global Optimization Methods

simann	simulated annealing	F77,f90,C	http://netlib.org/opt/simann.f
Annel	implementations of SA	C/C++	http://www.taygeta.com/annealing/simanneal.html
ASA	adaptive simulated annealing	C	http://www.ingber.com/#ASA-CODE
PIKAIA	genetic algorithm	F90,MPI	http://www.hao.ucar.edu/modeling/pikaia/pikaia.php
DIRDFN	global optimization problems	F90	http://www.dis.uniroma1.it/~lucidi/DFL/
GlobSol	interval software, rigorous global search	F90	http://interval.louisiana.edu/kearfott.html
PaGMO	Parallel Global Multiobjective Optimizer	C++,MPI	http://esa.github.io/pagmo/

For a wide list:
http://plato.asu.edu/sub/global.html
http://www.mat.univie.ac.at/~neum/glopt.html

Comparison

Traditional approaches

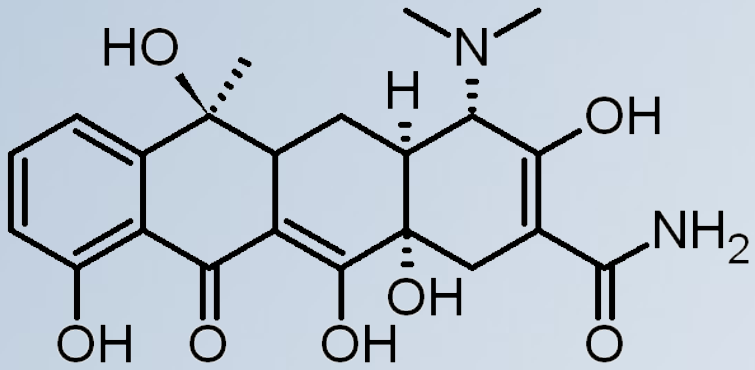
Do not use chemical knowledge

Complexity of the problem depends on the number of non H-atoms in the a.u.

Direct space methods

Can incorporate a massive amount of prior chemical information

Complexity of procedure depends on the number of degrees of freedom (DoF).



tetracycline (32 non-H atoms and 8 DoF) can be solved using global optimization

Clegg, W. & Teat, S. J., (2000). Acta Cryst. C56, 1343-1345.

Take advantage by using data of higher resolution

High resolution is not needed. Default resolution: 2-2.5 Å.

Generally require less time to run

Take time and patience. For large molecules: faster computer, run overnight, parallel program

Download this lecture

http://www.ba.ic.cnr.it/softwareic/expo/tutorials-and-lectures/

Contact, software download and info

http://www.ba.ic.cnr.it/softwareic/expo/

Acknowledgements

Research team

A. Altomare, A. Moliterni, R. Rizzi, N. Corriero and A. Falcicchio

Other collaborators

G. Cascarano, R. Mallamo, F. Ciriaco