Tutorial 1: Play with thiocyanate

James Foadi
email j.foadi@bath.ac.uk

June, 2019

Main aim of this tutorial is to get you rapidly acquainted with some of the most common and useful functions of the package `crone`.

1 Load data related to the structure

There are two types of data related to each 1D structure, the 'direct-space' data and the 'reciprocal-space' data. The first type of data include the position of each atom in the structure, its B factor (a measure of the thermal vibration of the atom) and the occupancy (which essentially describes what fraction of the specific atom is contained in each unit cell). Part of these data are also the unit cell's length, a, and the type of symmetry (only $P1$ or $P\bar{1}$ for 1D structures). The second type of data include experimental data related to the structure, i.e. data due to the x-ray diffraction of the structure itself. These data are not always the same, but should at least consist of the Miller index h and of the observed amplitudes of the structure factors and their experimental errors. The first type of data are known as *sdata*, while the second type are known as *fdata*. Both types are R named lists.

The function responsible to load an `sdata` object is `load_structure`. If called without argument, this function returns all the structures included in `crone`'s internal library of structures:

```
library(crone)
load_structure()

##
## 1D structures available for loading:
##    beryllium_fluoride
##     carbon_dioxide
##     cyanate
##     nitronium
##     thiocyanate
##     xenon_difluoride
##     pinkerton2015
## Please use an appropriate name - e.g.  load_structure("cyanate")
```

Let's try and load data corresponding to thiocyanate:

```
# Make sure to type the underscore!
sdata <- load_structure("thiocyanate")

# The object returned by load_structure is a named list
class(sdata)

## [1] "list"

names(sdata)

## [1] "a"   "SG"  "x0"  "Z"   "B"   "occ"
```

`a` and `SG`, as said before, are the unit cell's length (in angstroms) and space group symbol, respectively:

```
sdata$a

## [1] 4.969

sdata$SG

## [1] "P1"
```

The next four elements of the list are vectors of equal length, each one containing the rest position of the atom, the corresponding atomic number, the B factor and the occupancy:

```
sdata$x0

## [1] 1.000 2.813 3.969

sdata$Z
```

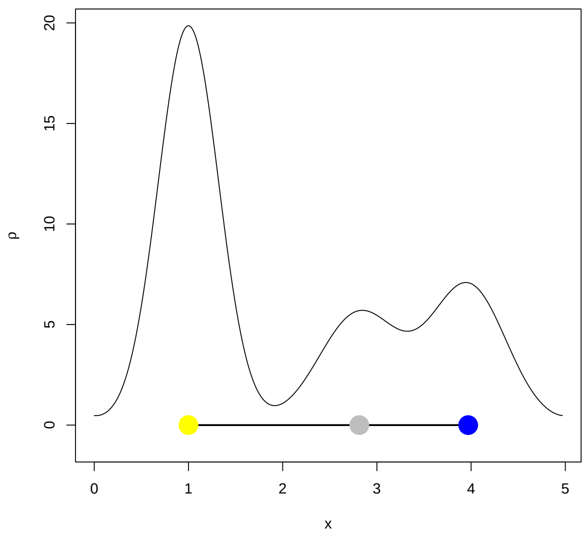
```
## [1] 16  6  7

sdata$B

## [1]  5.000 13.333 11.429

sdata$occ

## [1] 1 1 1
```

Thus, we have a sulphur (Z=16) at position $x = 1$ in the cell, a carbon (Z=6) at $x = 2.813$ and a nitrogen (Z=7) at $x = 3.969$. The sulphur, being the heaviest element, vibrates less than the nitrogen (smaller B factor), which vibrates less than the carbon. Also, each one of these atoms is fully present in all unit cells of the 1D crystal.

2 Calculate and plot the structure

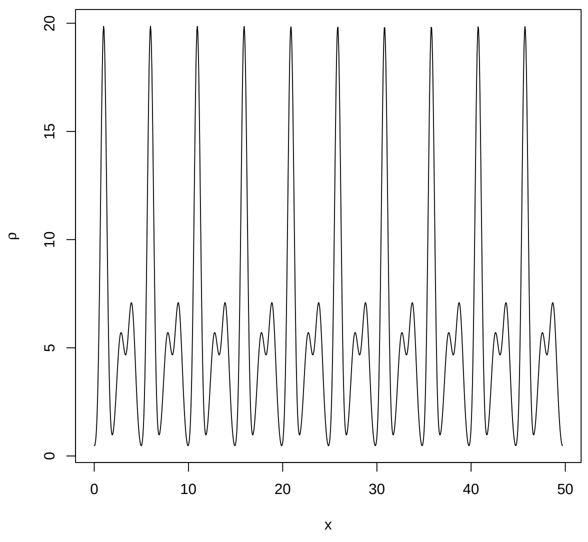
The actual electron density $\rho(x)$ of this structure can be calculated using the function `structure_gauss`. This can be then plotted to show the extent of the structure. Coloured discs corresponding to the accepted colour coding for chemical elements can be added in the bottom part of the plot, at the exact locations of the atomic centres:

```
rtmp <- structure_gauss(sdata)
names(rtmp)

## [1] "x"  "rr"

plot(rtmp$x,rtmp$rr,type="l",xlab="x",ylab=expression(rho),
     ylim=c(-1,max(rtmp$rr)))
segments(sdata$x[1],0,sdata$x[3],0,lwd=2)
points(sdata$x[1],0,pch=16,cex=3,col="yellow")
points(sdata$x[2],0,pch=16,cex=3,col="grey")
points(sdata$x[3],0,pch=16,cex=3,col="blue")
```



In fact, a grid containing several adjacent unit cells can be provided to the function `structure_gauss` so that the extent of the crystal periodicity is displayed. For 10 contiguous cells we have, for instance:

```r
x <- seq(0,10*sdata$a,length=1000)
rtmp <- structure_gauss(sdata,x)
plot(rtmp$x,rtmp$rr,type="l",xlab="x",ylab=expression(rho))
```



3 The calculation of structure factors

The product of a diffraction process are intensities whose square roots are, essentially, the amplitudes of the structure factors, $|F_h|$. Amplitudes and phases related to the given structure can be calculated using the function `strufac`. Structure factors will be computed for every value of the Miller indices provided:

```
hidx <- c(0,1)
ftmp <- strufac(hidx,sdata)
names(ftmp)

## [1] "Fmod" "Fpha"

ftmp$Fmod  # Structure factors' amplitudes for h=0,1

## [1] 29.000000  6.289007

ftmp$Fpha  # Structure factors' phases for h=0,1

## [1]  0.00000 75.66567

hidx = -2:2
ftmp <- strufac(hidx,sdata)
ftmp$Fmod  # Friedel's law |F(-h)|=|F(h)|

## [1] 12.500798  6.289007 29.000000  6.289007 12.500798

ftmp$Fpha  # Friedel's law phi(-h)=-phi(h)

## [1] -147.71590  -75.66567    0.00000   75.66567  147.71590
```

4 Electron density as Fourier synthesis

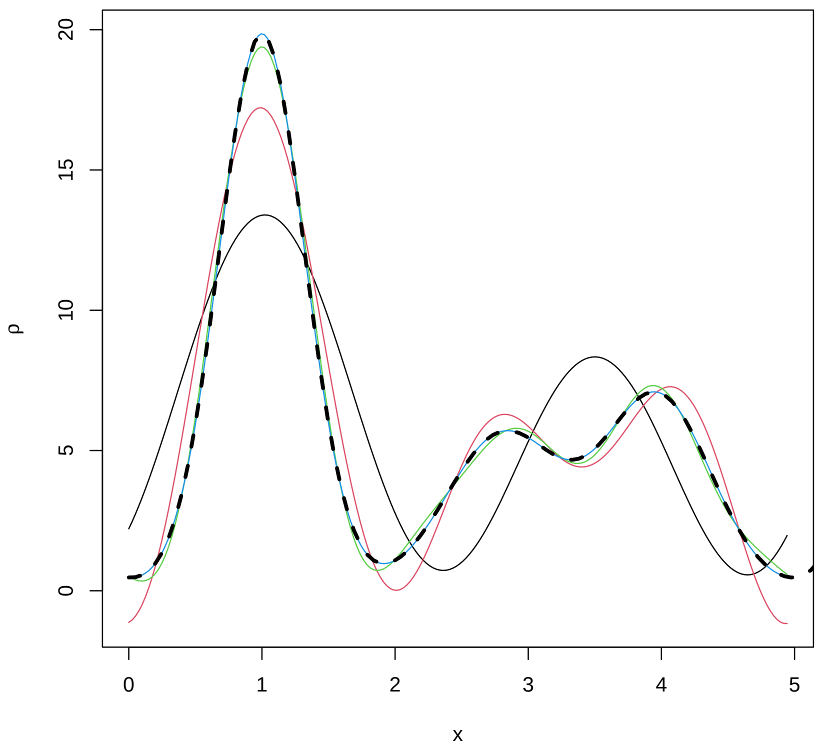
The electron density can be recovered starting from the experimental data (the structure factors' amplitudes) and the structure factors' phases, if these are known. The calculation is known as Fourier synthesis

$$\rho(x) = \frac{1}{a}|F_0| + \frac{2}{a}\sum_{h=1}^{h_{\max}} |F_h| \cos\left(2\pi h \frac{x}{a}\right),$$

and it is carried out by the function `fousynth`. The input for this function are the available indices, the corresponding amplitudes and phases of the structure factors, the unit cell length and the number of points for the regular grid used to calculate the electron density. In the following example let's calculate the approximated electron density using 2,3,5, 10 Fourier components, and compare it with the exact analytic density.

```
hidx <- 0:10
ftmp <- strufac(hidx=hidx,sdata=sdata)

# Grid
N <- 200

# Approximation with hmax=2
Fmod <- ftmp$Fmod[1:3]
Fpha <- ftmp$Fpha[1:3]
hidx <- 0:2
rtmp1 <- fousynth(sdata$a,Fmod,Fpha,hidx,N)

# Approximation with hmax=3
Fmod <- ftmp$Fmod[1:4]
Fpha <- ftmp$Fpha[1:4]
hidx <- 0:3
rtmp2 <- fousynth(sdata$a,Fmod,Fpha,hidx,N)

# Approximation with hmax=5
Fmod <- ftmp$Fmod[1:6]
Fpha <- ftmp$Fpha[1:6]
hidx <- 0:5
rtmp3 <- fousynth(sdata$a,Fmod,Fpha,hidx,N)

# Approximation with hmax=10
Fmod <- ftmp$Fmod[1:11]
Fpha <- ftmp$Fpha[1:11]
hidx <- 0:10
rtmp4 <- fousynth(sdata$a,Fmod,Fpha,hidx,N)

# Limits for plot
lims <- range(rtmp1$rr,rtmp2$rr,rtmp3$rr,rtmp4$rr,rtmp$rr)
```

```r
# Comparisons
plot(rtmp1$x,rtmp1$rr,type="l",xlab="x",ylab=expression(rho),
     ylim=lims)
points(rtmp2$x,rtmp2$rr,type="l",col=2)
points(rtmp3$x,rtmp3$rr,type="l",col=3)
points(rtmp4$x,rtmp4$rr,type="l",col=4)
points(rtmp$x,rtmp$rr,type="l",col=1,lwd=3,lty=2)
```



The Fourier synthesis with $h_{\max} = 10$ is quite close to the exact analytic density.

5 Load experimental data

The structure factors related to a specific structure can be loaded with the function `load_data`. This function will produce the *fdata* type. Let's practice with the thiocyanate structure.

```
fdata <- load_data(sname="thiocyanate")

# Names of fdata elements
ntmp <- names(fdata)
for (a in ntmp) {
  ltmp <- sprintf("%s  ",a)
  cat(ltmp)
}

## a  SG  hidx  Fobs  sigFobs  Phicalc

# Comparison between observed and calculated amplitudes
fdata$hidx[1:4]

## [1] 1 2 3 4

fdata$Fobs[1:4]

## [1]  6.613 12.467  9.226  3.454

ftmp$Fmod[2:5]

## [1]  6.289007 12.500798  9.582935  3.117837

# Comparison between stored phases and phases calculated
# with the Fourier synthesis
fdata$Phicalc[1:4]

## [1]   75.7  147.7 -149.9  -57.2

Fpha[2:5]

## [1]   75.66567  147.71590 -149.88296  -57.15538
```

The small differences between the previously-calculated structure factors' amplitudes and the loaded ones is due to experimental errors being artificially introduced when simulating the observed structure factors (see [1]).

References

[1] E. Smith, G. Evans and J. Foadi. ”An effective introduction to structural crystallography using 1D Gaussian atoms”. In: *Eur. J. Phys.* **38** (2017) DOI: 10.1088/1361-6404/aa8188.