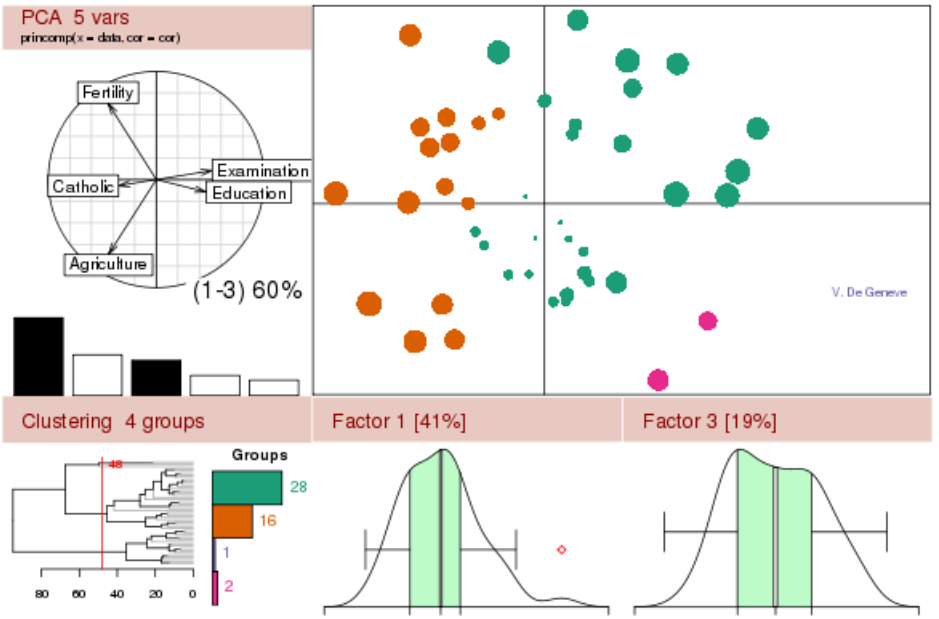
Imperial College London

diamond

A short introduction to the R statistical programming language



Diamond Light Source - April 2013 – PRACTICE WITH CRYSTALLOGRAPHY

James Foadi

MPL, Imperial College London - Diamond Light Source Ltd, Oxfordshire

A not-so-simple way to load code

After having downloaded the file “dmtz.tgz”, copy it to your working directory and unpack it in the following way:

```
tar -zxvf dmtz.tgz
```

This operation should create a subdirectory called “dmtz”. In order to use all files in that directory you will have to create the environment variable “RCOURSE_HOME”, which points to that directory. This can be done with the following command:

```
export RCOURSE_HOME=./dmtz
```

assuming you are in the directory containing subdirectory “dmtz”.

A not-so-simple way to load code

Now start an interactive R session:

```
module load R
R
```

Inside the R session you can now load all code using:

```
> source("dmtz/all_load.R")
```

What's inside your working space (use "ls()")? It looks like we have imported nothing ...

A not-so-simple way to load code

Now try:

```
> ls(all=TRUE)
 [1] ".crystal_system"            ".dcl"
 [3] ".d_hkl"                   ".extract_symmetry_info"
 [5] ".findHM"                   ".frac_to_orth"
 [7] ".full_symm_strings"          ".lword"
 [9] ".mkhome"                    ".op_xyz_list_to_matrix_list"
[11] ".op_xyz_to_matrix"           ".orth_to_frac"
[13] ".readBH"                    ".readH"
[15] ".readIR"                   ".readMTZ"
[17] ".readMTZHeader"               ".syminfo_to_matrix_list"
[19] ".syminfo_to_op_xyz_list"     ".SYMMETRY_file"
[21] ".tagRead"                   ".tranSG"
[23] ".translate_SG"               ".triclinic_to_orthogonal_01"
[25] ".triclinic_to_orthogonal_02" ".writeMTZ"
```

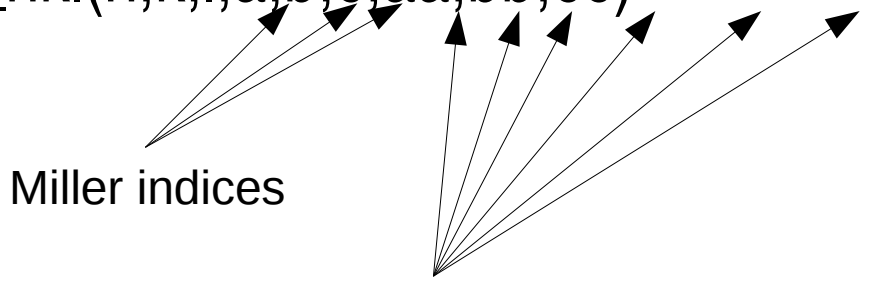
The cRy package

These are all parts of the “cRy” package for macromolecular crystallography (D. Waterman and myself). All functions start with a dot “.” because they need to be visually hidden to avoid a messy listing of the working directory. They become visible with the “all=TRUE” option in “ls”.

The R help feature is still not available for these functions because the package is not complete yet. But in this session I will show you how to use some of these functions.

Find resolution of a reflection, given Miller indices and cell parameters

Use function ".d_hkl": .d_hkl(h,k,l,a,b,c,aa,bb,cc)




```
> .d_hkl(0,0,1,10,10,10,90,90,90)
[1] 10
> .d_hkl(0,1,0,10,10,10,90,90,90)
[1] 10
> .d_hkl(0,0,0,10,10,10,90,90,90)
[1] Inf
> .d_hkl(-5,2,5,10,10,10,90,90,90)
[1] 1.360828
```

Read reflections from an MTZ file

Use function ".readMTZ": .readMTZ(filename,message=TRUE)


```
> data <- .readMTZ("a8a_calc.mtz")
[1] "COLUMN H                H        0.0000      136.0000    0"
[1] "COLUMN K                H        0.0000      136.0000    0"
[1] "COLUMN L                H      -37.0000       37.0000    0"
[1] "COLUMN Fnew             F        0.0769    23893.9355    1"
[1] "COLUMN PHInew           P     -179.9987      180.0000    1"
```

Read reflections from an MTZ file

This function returns a named "list":

```
> class(data)
[1] "list"
> names(data)
[1] "reflections"   "header"       "batch_header"
```

Let's explore each field:

```
> class(data$reflections)
[1] "data.frame"
> class(data$header)
[1] "list"
> class(data$batch_header)
[1] "NULL"
```

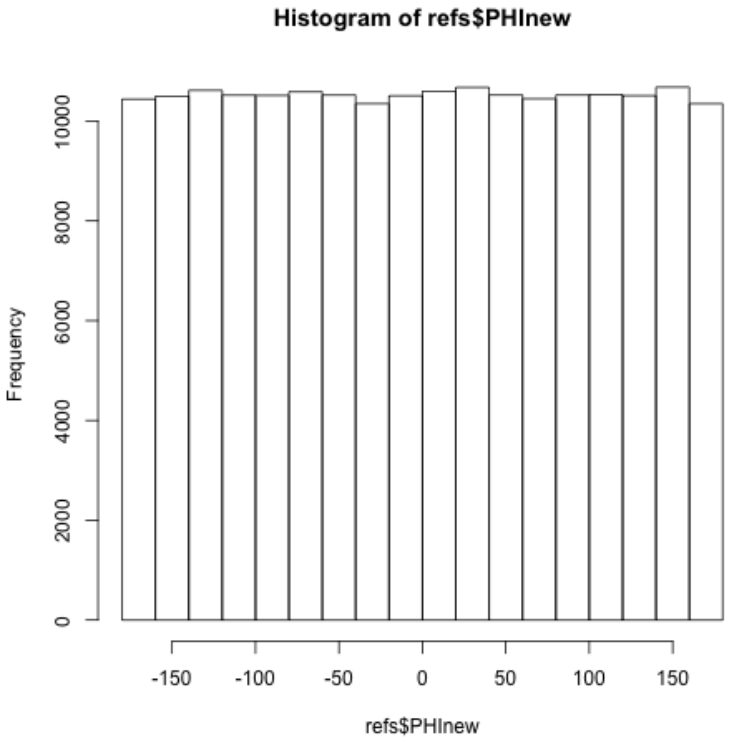
Imperial College London

diamond

Read reflections from an MTZ file

Diffraction data are stored in the data frame, “reflections”:

```
> refs <- data$reflections
> str(refs)
'data.frame':  189404 obs. of  5 variables:
 $ H     : num  0 0 0 0 0 0 0 0 0 0 ...
 $ K     : num  0 0 0 0 0 0 0 0 0 0 ...
 $ L     : num  -36 -33 -30 -27 -24 -21 -18 -15 -12 -9 ...
 $ Fnew  : num  20.8 45.5 80.7 35.4 59.7 ...
 $ PHInew: num  -109 -126 -101 103 133 …
> hist(refs$PHInew)
```



Find min and max resolution

First find out cell parameters from “header”:

```
> hdr <- data$header
> names(hdr)
 [1] "NCOL"    "CELL"    "SORT"    "SYMINF"  "RESO"    "NDIF"    "SYMM"
 [8] "PROJECT" "CRYSTAL" "DATASET" "DCELL"   "DWAVEL"  "COLUMN"  "HISTORY"
> hdr$CELL
[1] 157.27 157.27  37.45  90.00  90.00 120.00
```

Calculate a resolution vector (using R parallel features):

```
> resos <- .d_hkl(refs$H,refs$K,refs$L,157.27,157.27,37.45,90,90,120)
> length(resos)
[1] 189404
> length(refs$H)
[1] 189404
```

Calculate min and max using “range”:

```
> range(resos,na.rm=TRUE)
[1]  0.9999793 78.6350000
```

Let's play with symmetry

Use ".translate_SG" to associate SG number to HM symbol:

```
> .translate_SG(19)
$msg
[1] "P 21 21 21"

$ans
[1] TRUE

> .translate_SG(146)
$msg
[1] "R 3 :H"

$ans
[1] TRUE

> SG <- "P 1 2 1"
> .translate_SG(SG,SG_in="xHM",SG_out="number")
$msg
[1] 3

$ans
[1] TRUE
```

Let's play with symmetry

Extract symmetry operators:

```
> opl <- .syminfo_to_op_xyz_list("P 1 2 1")
> mlist <- .op_xyz_list_to_matrix_list(opl)
> class(mlist)
[1] "list"
> names(mlist)
[1] "PG" "T"  "C"
```

Explore “PG” part of symmetry list:

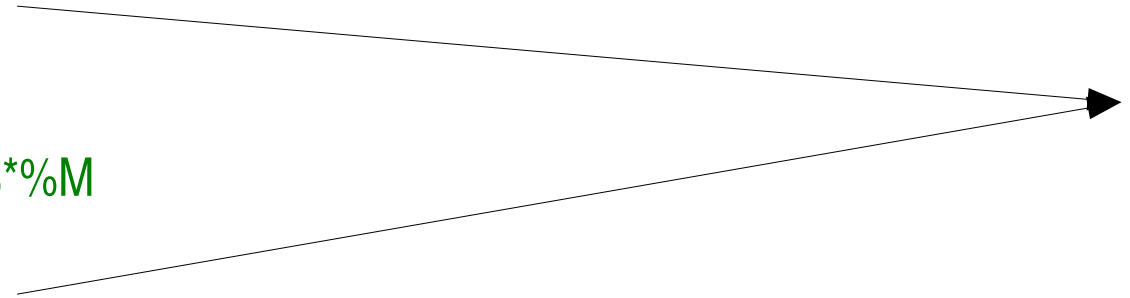
```
> class(mlist$PG)
[1] "list"
> length(mlist$PG)
[1] 2
> mlist$PG
[[1]]
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1

[[2]]
     [,1] [,2] [,3]
[1,]   -1    0    0
[2,]    0    1    0
[3,]    0    0   -1
```

Let's play with symmetry

Calculate all symmetry equivalents of reflection (-1,2,1):

```
> M <- mlist$PG[[1]]
> eq_hkl <- c(-1,2,1)%*%M
> eq_hkl
     [,1] [,2] [,3]
[1,]   -1    2    1
> M <- mlist$PG[[2]]
> eq_hkl <- c(-1,2,1)%*%M
> eq_hkl
     [,1] [,2] [,3]
[1,]    1    2   -1
```



Two equvalents

Let's play with symmetry

Calculate all symmetry equivalents of reflection (-1,2,1):

```
> M <- mlist$PG[[1]]
> eq_hkl <- c(-1,2,1)%*%M
> eq_hkl
     [,1] [,2] [,3]
[1,]   -1    2    1
> M <- mlist$PG[[2]]
> eq_hkl <- c(-1,2,1)%*%M
> eq_hkl
     [,1] [,2] [,3]
[1,]    1    2   -1
```

Two equvalents