DISCUS RefineEllipsoid tutorial

In this tutorial you will learn to:

- refine parameter values that describe an ellipsoidally shaped nanoparticle to obtain a good fit to an “experimental” PDF

The main tool for this tutorial is the macro “refine.mac” found in the REFINE_ELLIPSOID folder. This macro takes no parameters and is started as the following example shows:

```
@refine.mac
```

Start the discus_suite, navigate to the REFINE_ELLIPSOID folder and try it out.

DISCUS provides two refinement algorithms, a classical least squares driven refinement algorithm and the more flexible but more involved optimizer based on a evolutionary algorithm. Here we focus on the least squares algorithms in the “refine” section of the DISCUS_suite.

The refine section needs to know the data against which you want to refine the model (line 13), the list of parameters whose value is to be refined (lines 26 to 35) and information on the refinement itself (lines 38 to 44).

For each of the parameters you need to provide on the newparam command line a name and a starting value. Optionally you can provide an allowed range a flag to refine or fix the parameter and information that will help the refinement (“points:” “shift”)

The parameter name is actually a variable name. Use this name in the main refinement macro here “discus_main.mac” to build your structure. Any valid variable name is fine. Check the build in help to learn about special parameter names like “P_eta”.

You need to provide a starting value (“value:”) for each of these parameters. If you know that a parameter is restricted to a limited numerical range use the “range:[lower, upper]” parameter. Check the build in help to learn about the flexible use of this optional parameter.

Any classical least squares algorithm needs to know how much the calculated function changes if a parameter is changed. Assume we want to refine a linear dependence to experimental data:

$$ycalc = P_1 * x + P_2$$

Here we need to know the derivatives of ycalc with respect to P1 and P2. Within a Rietveld program these derivative for all parameters are encoded and you do not have to worry about them. As REFINE dos not really know what kind of function you are going to refine in the slave macro “discus_main.mac”, REFINE needs to calculate these derivatives numerically. To do so it calculates the function with the slave macro at “points:” shifted around the current value. The points are +- spaced with either 3 or 5 points. The parameter value is shifted by its current value*shift. “points:5” gives a better estimate of the derivative at higher computational cost. For parameters like lattice parameters that modify a powder pattern/PDF continuously use a small shift around 0.003. Parameters like a nanoparticle diameter will not change the structure if the parameter value is changes at extremely minute values, as you will not add/remove atoms. Use a larger relative shift like 0.01 to 0.02.

The main work macro, here "**discus_main.mac**" builds a crystal structure and calculates the function like powder diffraction pattern of PDF based on the current parameter values. Most of it is equivalent to the "top" macro in ELLIPSOID" with three essential differences:

- the first line must be "branch discus" (or "branch kuplot") to step from REFINE into DISCUS (or KUPLOT)
- the last two lines must be an "exit" to leave DISCUS and to go back to REFINE and the final "finished" line. This last line simple made programming life easier.
- the slave macro must ensure that the last (or single) data set in KUPLOT is the calculated function

In the current example "**discus_main.mac**" check lines 1; 63 and 64; and 57 to 62. These correspond to the three bullet points above.

An option is to provide a second macro on the "run" command (line 49 in refine.mac") with the optional parameter "plot:kuplot_macro_name" here "plot:k_inter.mac" you can instruct REFINE to display the observed, calculated and difference as the refinement progresses. This plot macro must be a pure KUPLOT macro, do not start with a line like "kuplot" or "branch kuplot". At the end you must, however, use a line "exit" to go back from KUPLOT to REFINE.