Multiscat

Close coupled calculation code, originally by *Manalopalous* and *Ramsay* and later modified by *Buckland, Knowling, Jardine, Lee, Riley* and *Tuddenham.*

This version performs scattering calculations for a “numerically” defined potential – i.e. it does not require an analytical function for the potential, but instead reads it in as a series of numbers. The filenames used in this version are suffixed by **_fay** to distinguish them from older versions, but I've not included this suffix when describing the files below.

I've attempted to provide a brief “user guide” below. Where my knowledge is incomplete, I've indicated by ????. Details of the close coupled method may be found in *J. Ramsay's* (Univ. Nottingham), *D. Riley's* and *G. Lee's* theses.

Red refers to filenames
Blue refers to subroutine names
Green refers to program variables, some of which you must define

The subroutines in **scatsub.f** and **diagsub.f** can be thought of as a “black box”. The important sections for a typical user are

- **multiscat.f90** (main program)
- **potsub.f** (contains subroutines for reading in and interpolating potential)
- **multiscat.conf** (input configuration file)
- potential input files (filename **potxxxxx.in**)
- list of Fourier component labels (filename **FourierLabels.in**)

Potential input and Fourier labels file

The potential must be supplied as Fourier components, for a range of different z values. The range and number of z points in the input potential must be defined in **multiscat.conf,** as **stepzmin**, **stepzmax** and **nzfixed**, respectively The number of Fourier components supplied is given by **nfc**, also defined in **multiscat.conf**.

The **nfc** Fourier component labels for the input potential should be supplied in **FourierLabels.in**, as a list e.g.

```
0 0
-1 0
1 0
1 1
```

etc...

The input potential files should consist of a list of **nzfixed** complex values for each Fourier component, giving the potential (in meV) at each z value in the range **stepzmin** – **stepzmax**. The Fourier components should appear in the same order as in **FourierLabels.in**.

e.g. if we were using **nzfixed**`=3`, **stepzmin**`=0` and **stepzmax**`=2` (hopelessly unrealistic values, but ok for an example here!), the potential input file associated with the Fourier list above would read

```
(Real(V00(z=0)), Imag(V00(z=0)))
(Real(V00(z=1)), Imag(V00(z=1)))
(Real(V00(z=2)), Imag(V00(z=2)))
(Real(V-10(z=0)), Imag(V-10(z=0)))
(Real(V-10(z=1)), Imag(V-10(z=1)))
(Real(V-10(z=2)), Imag(V-10(z=2)))
(Real(V10(z=0)), Imag(V10(z=0)))
(Real(V10(z=1)), Imag(V10(z=1)))
(Real(V10(z=2)), Imag(V10(z=2)))
```

......etc.

Input configuration file

- **itest** set to zero to output just the specular intensity (e.g. if you just want to study bound state resonances) and 1 to output diffraction peak intensities as well.
- Gmres preconditioner ????
- **nsf** no. of sig figs required for convergence. Try adjusting this until the calculation results converge; the main program has hardlimits of `2<`**nsf**`<10`, which will overwrite what you put in the conf file if it falls outside this range, but of course, you can always edit the main program
- **nfc** no. of potential Fourier components to use
- **zmin**,**zmax** the z integration range for the calculation; try adjusting to achieve convergence;
- vmin ???
- **dmax** max negative energy of closed channels; determines the number of channels to be included in the calculation; adjust to achieve convergence
- **imax** max index of channels in calculation; determines the number of channels to be included in the calculation; adjust to achieve convergence
- range of kinematic conditions (**theta**, **phi**, **E**) that calculations will be performed for]
- **a1**, **a2**, **b1** unit cell parameters; there is a schematic in subroutine basis in scatsub.f that showed what **a1**, **a2** and **b1** correspond to; for a square unit cell, one of them will be zero
- **nzfixed** no. of z points supplied in the potential input file
- **stepzmin**, **stepzmax** z range over which your input potential is supplied; doesn't necessarily have to be the same as the calculation integration range **zmin** - **zmax**, but the potential interpolation is likely to fail if **stepzmax** `<` **zmax**;
- **startindex**, **endindex** if you want to perform the calculations for multiple potential input files, then they should be in a list **startindex** – **endindex**, with the filename format **potxxxxx.in**, where `xxxxx=index`; to use one potential file only, just set **startindex** and **endindex** to the same value
- **helium mass** in amu

Using the code

If you're using **gfortran**, you can compile using the **Makefile**, by typing make at the command line. You will need to edit the **Makefile** if using something other than gfortran.

To run multiscat, type `./multiscat multiscat.conf`

The program will loop over the potential files supplied. The potential is read in by subroutine

loadfixedpot. Currently set up to read in four header lines at the top of the potential file, before the actual numerical potential values, but if you want more/fewer header lines in your potential files, modify accordingly.

For each kinematic condition (i.e. each combination of **theta**, **phi**, **E**) requested in the input file, a scattering calculation is performed. For each calculation, the subroutine **find_mz** calculates the number of z values, **m**, that will be included in the calculation integration, which depends on the incident kinematic conditions and **vmin** and **nsf**. For the **m** z-points required for the calculation, the subroutine **potent** in **potsub.f** interpolates the input potential. Therefore, you can achieve a more accurate calculation by having a large value of **nzfixed**, so that the program is interpolating from a well-sampled z range.

The subroutines in **scatsub.f** and **diagsub.f** then determine the diffraction and specular intensities.

Output files

If **itest=0**, for each kinematic condition, the specular intensity calculated, **I00**, is outputted. In addition, the sum of diffraction intensities (**sum**) is outputted. It should equal 1, or close to 1. If the calculation has failed, the program sets **sum=-1**. To output the specular intensities to a file (by default, it will output it to the screen), when initiating multiscat, add >, followed by the output filename, to the run command i.e. **./multiscat multiscat.conf > outputfilename**

If **itest=1**, in addition to the above, a full set of diffraction intensities are outputted for each kinematic condition. A diffraction output file will be generated for each potential file, **diffracxxxxx.out**.

Other things to be aware of

- **Convergence** if the calculation has succeeded, **sum** should be 1 (or 0.99999...). However, this doesn't mean the calculation is converged. To check for convergence, try increasing the no. of Fourier components in the potential, the z integration range, the max closed channel energy etc. until the diffraction intensities calculated are independent of further increases in these parameters
- **rmlmda** the program runs in units of **rmlmda, which is** equal to **hbar^2/2m**, in units of **amu**, **meV** and **Angstrom**. Your potential should be supplied in meV, but subroutine **loadfixedpot** will convert it to program units
- **numbers** you may see that there are numbers in the code when opening files e.g. **open (80,file=inputfile)** . The numbers have no actual significance, but are just used as an identifier for that file, so that in later lines of code, the shorthand **80** is used to refer to the inputfile.
- **multiscat.inc** hard limits on parameter values, which is incorporated when compiling; with increases in computing power, you may find that you want to have higher limits on these values (e.g. using lots of Fourier components and channels is no longer prohibitive in terms of cpu run-time), so you can modify the contents of **multiscat.inc**, but remember to recompile afterwards!

Fay Tuddenham

January 2011
faytuddenham@gmail.com