Introduction to R-matrix calculations

A D Whiteford and N R Badnell

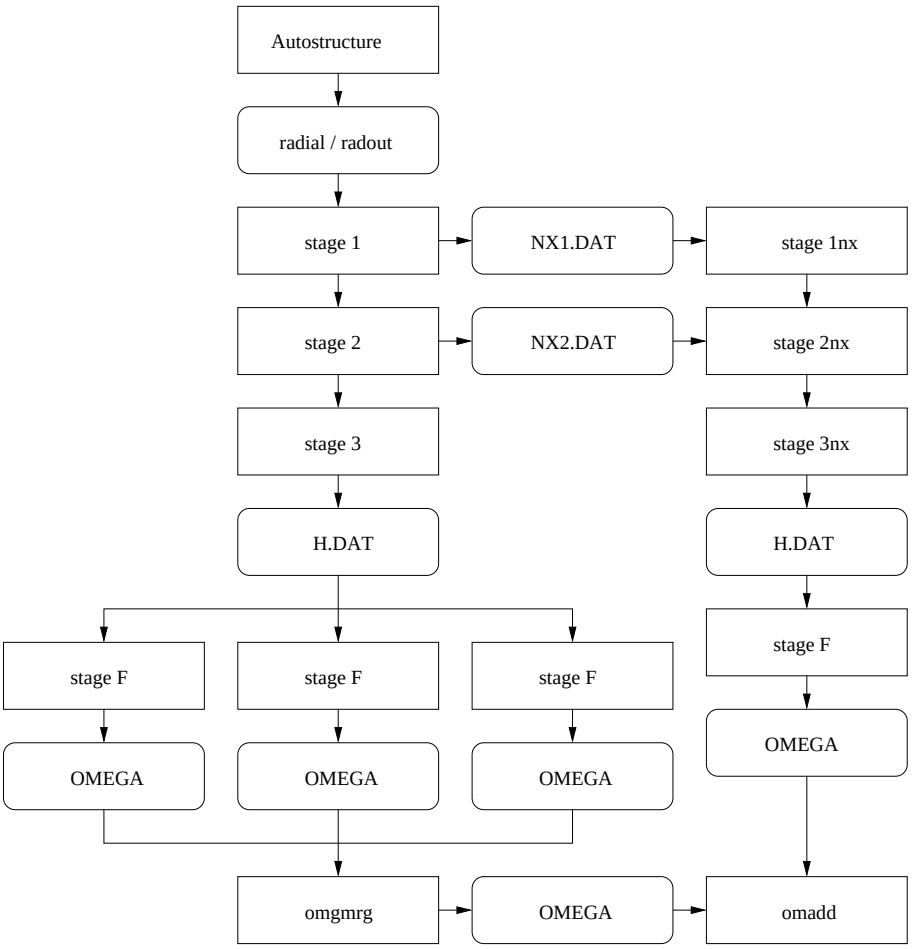
15th October 2009

Contents

1 Introduction

These notes give a rough guide to starting out with R-matrix codes to generate data for Na-like Fe in LS coupling. The examples here should be viewed as a starting point for more advanced R-matrix calculations which could include other physical effects, IC coupling, more levels or different ions.

2 Code layout



.....

3 Structure calculation

The structure is calculated using the code AUTOSTRUCTURE, other structure codes can be used if necessary.

For sodium-like iron, an example input file would look like:

```
A.S.
 &SALGEB RAD='ALL' CUP='IC' MXVORB=3 MXCONF=3 KORB1=1
                            KORB2=3 KUTSO=0 BORN='INF' &END
  3 0  3 1  3 2
   1    0    0
   0    1    0
   0    0    1
 &SMINIM NZION=26 RADOUT='YES' &END
```

MXVORB specifys three valance orbitals will be specified, these are 3s, 3p and 3d.

MXCONF specifys three configurations will be specified, these are $1s^2 2s^2 2p^6 3s$, $1s^2 2s^2 2p^6 3p$ and $1s^2 2s^2 2p^6 3d$

RAD="ALL" implies we also wish the code to calculate all types of radiation, this is not necessary for the resulting R-matrix calculation but for checking an atomic structure it is often useful to compare radiative quantities.

CUP="IC" means we wish to perform the calculation in intermediate coupling.

KORB1=1 KORB2=3 means that we want the first three orbitals to be closed, i.e. all of our confiruations are based on a $1s^2 2s^2 2p^6$ closed core and we only explicitly specify electrons outside of this.

KUTSO=0 is to turn off the spin orbit interaction, we do this because the R-matrix codes do not include this term so in order to maintain consistency between AUTOSTRUCTURE and R-matrix we turn it off in AUTOSTRUCTURE.

NZION=26 is specifiying the ion charge, in this case 26 (iron!).

RADOUT="YES" causes AUTOSTRUCTURE to produce a file called "radial" which contains radial wavefunctions to be used by the R-matrix codes. This is the only data which AUTOSTRUCTURE passes to R-matrix.

BORN="INF" causes AUTOSTRUCTURE to produce infinite energy Born limit points (required later). IC versions end up in a file called OMGINFIC and LS versions in a file called OMGINFLS.

Here we have included the configurations $1s^2 2s^2 2p^6 3s$, $1s^2 2s^2 2p^6 3p$, $1s^2 2s^2 2p^6 3d$. We specify our three configurations in using occupation numbers, recalling that using the KORB parameters we've already set a $1s^2 2s^2 2p^6$ (21522563) core.

At the end of our structure calculation, it is useful to look at the olg file. In particular the energy levels calculated, an extract of the file shows that

K	K*CM	2*S+1	L	2J	CF	(EK-E1)/RY
1	0.	2	0	1	1	0.00000000
2	276562.	-2	1	1	2	2.52021562
3	296656.	-2	1	3	2	2.70333129
4	677729.	2	2	3	3	6.17592452
5	681019.	2	2	5	3	6.20590102

These are five levels, the first corresponding to configuration (CF) 1 (i.e. $1s^2 2s^2 2p^6 3s$), the second and third to configuration two and the fourth and fifth to configuration five.

NIST quotes the four excited energies as 2.52596, 2.71688, 6.15544 and 6.18198. We can conclude that our structure is reasonable but perhaps some optimisation can be required. For our example, we will continue with this default structure regardless.

4 Inner region exchange calculation

The inner region is run in three stages, interestingly called stage 1, stage 2 and... wait for it... stage 3.

For Na-like iron, an example stage 1 input file would look like:

```
S.S.
 &STG1A &END
 &STG1B MAXLA=2 MAXLT=12 MAXC=20 MAXE=100 &END
```

MAXLA=2 is the maximum angular momentum of the target.

MAXLT is the maximum total angular momentum of the system, typically we make this a factor of 3 or 4 times MAXLA.

MAXC is the number of basis orbitals to be used, this limits how high in energy we can scatter.

An example stage 2 input file would look like:

```
S.S.
 &STG2A &END
 &STG2B MAXORB=6 NELC=11 NAST=3 INAST=0 MINLT=0 MAXLT=12
          MINST=1 MAXST=3 &END
 1 0 2 0 2 1 3 0 3 1 3 2
  3
  2 2 6 0 0 0 0 0 0 0
  2 2 6 1 1 1 1 1 1 1
  2 2 6 1 0 0 0 0 0 0  0
  2 2 6 0 1 0 0 0 0 0  0
  2 2 6 0 0 1 0 0 0 0  0
 2 0 0
 2 1 1
 2 2 0
 1
  2 2 6 0 0 0 0 0 0 0
  2 2 6 2 2 2 2 2 2 2
  2 2 6 1 0 0 0 0 0 0  2
```

MAXORB=6 are the maxium number of orbitals, in this case 6 (1s,2s,2p,3s,3p,3d).

NELC=11 specifies the number of electrons in the system.

NAST=3 is the number of atomic states, we are working in LS at the moment and there are 3 terms in the system we are looking at.

MINLT=0 is the minimum orbital angular momentum we are interested in, this should usually be zero.

`MAXLT=12` is the maximum orbital angular momentum, this should correspond to the MAXLT parameter passed into stage 1.

`MINST=1` and `MAXST=3` are the minimum and maximum spins of the N+1 system. In this case we get singlets and triplets for the intermediate 3 electron system.

We then specify our orbitals in the line which looks like:

```
 1 0 2 0 2 1 3 0 3 1 3 2
```

We give our 6 orbitals as the n of the orbitals followed by the l (e.g. '2 1' $\equiv$ 2p).

Next we specify how many configurations we have and then list them by occupation numbers, note that the fact we have 10 orbitals and 10 configurations is just a conincidence.

```
 3
 2 2 6 0 0 0 0 0 0 0
 2 2 6 1 1 1 1 1 1 1
 2 2 6 1 0 0 0 0 0 0  0
 2 2 6 0 1 0 0 0 0 0  0
 2 2 6 0 0 1 0 0 0 0  0
```

We specify that there are three configurations, on the next line we give the minimum number of electrons which can be in each orbital and on the next the maximum. On the following ten lines we give each configuration. For the moment, ignore the zeros on the end of each line.

Following this we give the 3 allowed terms, we specify multiplicity, orbital angular momentum and parity (0 for even parity and 1 for odd).

After this we list each possible intermediate configurations corresponding to the N+1 electron system:

```
 2 2 6 0 0 0 0 0 0 0
 2 2 6 2 2 2 2 2 2 2
 2 2 6 1 0 0 0 0 0 0  2
```

Again we first specify the minimum number of electrons in each orbital, then the maximum. The line which follows gives a starting configuration and the line on the end the number of promotions from this configuration (so long as we follow the minimum and maximum rules give above it) we allow two promotions. Simple combinatorial analysis will quickly show that these rules generate all intermediate configurations.

The stage 3 input data is somewhat simpler:

```
S.S.
 &STG3A &END
 &STG3B INAST=0 NAST=0 &END
```

At the end of our inner region exchange runs we have an `H.DAT` file. This is the file which is later used by the outer region codes and consists of a series of diagonlised hamiltonians for each symmetry.

5 Outer region exchange runs

The outer region codes are split into two parts, stage f and stage icf. Stage f solves the problem in LS coupling and then stgicf performs a frame transformation and makes use of the term coupling coefficients to resolve the results in an IC picture. We will concentrate here on stage f.

An example input file for stgf would be:

```
&STGF IMESH=1 IQDT=2 PERT='YES' LRGLAM=-12 IPRINT=-1 ELAS='NO'
      IPRKM=4 IBIGE=0 &END
&MESH1 MXE=9000 E0=0.01 EINCR=0.00001 &END
```

`IMESH=1` tells the code how we wish to specify our energy mesh, in this case "1" means we will later specify a starting energy, an energy increment and a number of energies.

`IQDT=2` tells the code to use K-matrices, these are later used by stgicf.

`PERT='YES'` turns on long range coupling potentials, these are usually necessary for accurate results.

`LRGLAM=-12` tells the code to calculate collision strengths up to an L of 12, this can not be higher than the maximum L specified in stage 1 and stage 2. We specify a negative number to suppress top-up since this will be handled with the non-exchange codes.

`IPRINT=-1` controls how much information is printed to the human readable file.

`ELAS='NO'` tells the code we are not interested in elastic transitions.

`IPRKM=4` tells the code to print out K-matrices so that can be used in the following stgicf code.

The energy grid is then specified, in this case we are asking for 9000 points starting at a Z-scaled energy of 0.01 and incrementing 0.00001 Z-scaled rydbergs at a time.

The output of stage f which we are interested in is an `OMEGA` file. This contains the collision strength for each transition as a function of energy.

It is normal to do a number of stage f runs, with a fine mesh to span the resonance region (i.e. up to threshold) and a coarser mesh through the non-resonance region. The input file for the coarser mesh might look like:

```
&STGF IMESH=1 IQDT=2 PERT='YES' LRGLAM=-12 IPRINT=-1 ELAS='NO'
      IPRKM=4 IBIGE=0 &END
&MESH1 MXE=1000 E0=0.1 EINCR=0.001 &END
```

identical to the first file except that the starting energy is where the last file left off and the energy increments is much larger.

6 Inner region non-exchange runs

So far, we have only calculated collision strengths up to a total angular momentum of 12. In order to genereate a complete cross-section we must extend this up to infinity, this is done via the use of the non-exchange codes. They will use a simpler approximation up to another value of L and then apply top-up rules after this.

There are three non-exchange inner region codes, they are all driven by the same input file. A typical one would look like:

```
CONTinuation run
 &STGNX  MINLT=13 MAXLT=60  &END
```

Here we tell it the minimum and maximum values of L it should work for, the rest of the information it can derive from files coming from the exchange codes (NX1.DAT, NX2.DAT and the stage 3 input file).

The three non exchange codes are then run with this input file, producing an `H.DAT` file which can be similarly processed with stage f.

7 Outer region non-exchange runs

These are the same as the exchange codes but a much coarser energy grid can be used since we don't get resonances for the non-exchange calculations.

A typical input files would look like

```
&STGF IMESH=1 IQDT=2 PERT='YES' LRGLAM=60 IPRINT=-1 ELAS='NO'
      IPRKM=4 IBIGE=0 MINLT=13 &END
&MESH1 MXE=1090 E0=0.01 EINCR=0.001 &END
```

This will produce an OMEGA file. For partial waves from 13 to 60 calculated explicitly and partial waves of 61 to infinity calculated using 'top-up' rules. We could have turned the top-up off by specifying LRGLAM as a negative value.

8 Merging omega files

By the time you are finished you will have a number of omega files, these will be limited in energy or angular momentum. There are two operations we can perform on two omega files, merge or add.

If we wish to put two omega files together with different energy grids, we merge them using a utility code called OMGMRG.

If we wish to put two omega files together with different angular momenta ranges, we add them using a utility code call OMADD.

We have produced three seperate `OMEGA` files in the examples above, we firstly merge the two exchange runs with OMGMRG and then add the exchange calculations to the non-exchange calculations using OMADD.

9 Results

From our final omega file, we can extract transitions using the `xtrct` program, this will produce an ascii list of collision strength against energy.

Looking at the $3s-3p$ transition, the different angular momenton contributions are shown in figure 1.

Comparison with the work of Mann is shown in figure 2, the results are showing the correct behaviour at high energy but are perhaps not as close as one would like, the limited inclusion of states and lack of optimisation of the structure could be responsible for the discrepancy.

10 Producing an adf04 file

To do this we use a program called adasex, this works on an OMEGA file and an input file called "adasex.in" which looks like:

```
 &ADASEX NTERM=  3 FIPOT=3946280 IONTRM='1S' IEL='FE' &END
         3s(2S)          0.    1
         3p(2P)     271831.    2
         3d(2D)     649665.    3
NAME: Your name here
DATE: Date here
Free-form comments here
.
```

`NTERM` is the number of levels. We also give the ionisation potential and the term specification of the adjacent ion stage.

Following the namelist we give a temperature grid and then a list of what each level is.

Using this information, an adf04 file can be constructed.

However, we also need to tag on infinite energy limit points, these were generated by autostructure and the file is called `OMGINFLS`, these should be omega merged on to the final `OMEGA` file before ADASEX is run.

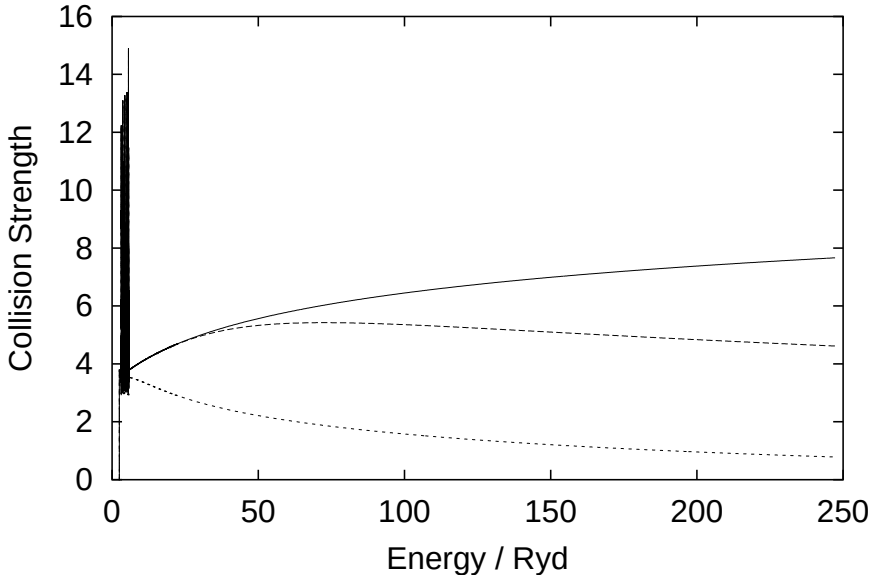


Figure 1: Contribution from different angular momenta regimes, the lower curve shows the inclusion of only the exchange data, the middle curve the inclusion of non-exchange data and the upper curve includes top-up.

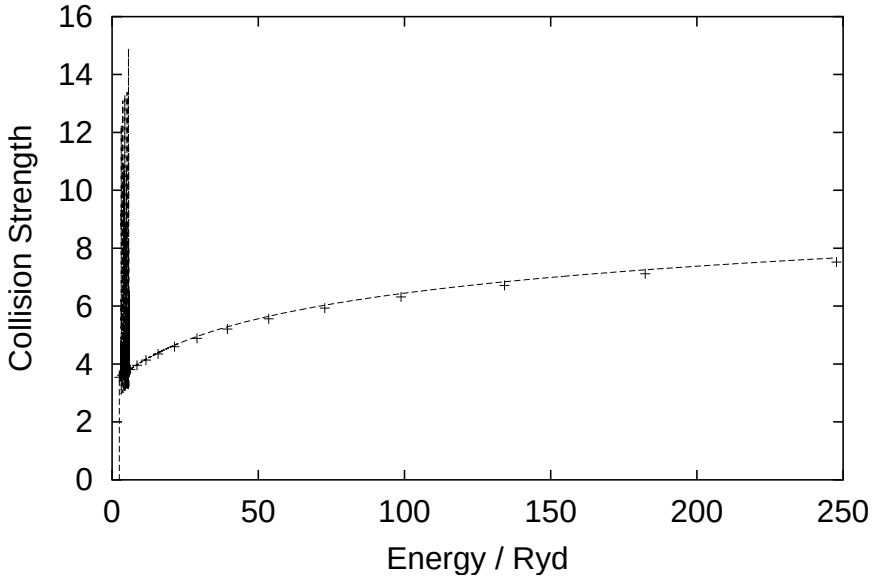


Figure 2: Comparison with the results of Mann

Downloading and compiling R-matrix codes

A D Whiteford and N R Badnell

15th October 2009

1 Downloading

For the examples were are running here, you will need to download the following files:

```
http://amdpp.phys.strath.ac.uk/rmatrix/ser/asy/PARAM
http://amdpp.phys.strath.ac.uk/rmatrix/ser/asy/stgf.f

http://amdpp.phys.strath.ac.uk/rmatrix/ser/ipbp/PARAM
http://amdpp.phys.strath.ac.uk/rmatrix/ser/ipbp/stg1r.f
http://amdpp.phys.strath.ac.uk/rmatrix/ser/ipbp/stg2r.f
http://amdpp.phys.strath.ac.uk/rmatrix/ser/ipbp/stg3r.f
http://amdpp.phys.strath.ac.uk/rmatrix/ser/ipbp/stglib.f

http://amdpp.phys.strath.ac.uk/rmatrix/ser/misc/omadd.f
http://amdpp.phys.strath.ac.uk/rmatrix/ser/misc/omgmrg.f
http://amdpp.phys.strath.ac.uk/rmatrix/ser/misc/adasex.f
http://amdpp.phys.strath.ac.uk/rmatrix/ser/misc/xtrct.f

http://amdpp.phys.strath.ac.uk/rmatrix/ser/nxls/PARAM
http://amdpp.phys.strath.ac.uk/rmatrix/ser/nxls/stg1nx.f
http://amdpp.phys.strath.ac.uk/rmatrix/ser/nxls/stg2nx.f
http://amdpp.phys.strath.ac.uk/rmatrix/ser/nxls/stg3nx.f

http://amdpp.phys.strath.ac.uk/autos/ver/asdeck.f
http://amdpp.phys.strath.ac.uk/autos/ver/PARAM
```

To prevent PARAM files from clashing, it's advised that you create different directories for the different codes in the same was as they are on the web.

2 Compling

This will vary from platform to platform, but a rough guide is given here. It is advised that if you don't have one already make a binary directory for yourself via

```
mkdir $\sim$/bin
```

Compile the codes by typing (in whatever directories you created):

```
f90 -O3 -o ~/bin/autos.x asdeck.f

f90 -O3 -o ~/bin/stgf.x stgf.f

f90 -O3 -c -o ~/bin/stglib.o stglib.f
f90 -O3 -o ~/bin/stg1r.x stg1r.f
f90 -O3 -o ~/bin/stg2r.x stg2r.f stglib.o
f90 -O3 -o ~/bin/stg3r.x stg3r.f stglib.o

f90 -O3 -o ~/bin/omadd.x omadd.f
f90 -O3 -o ~/bin/omgmrg.x omgmrg.f
f90 -O3 -o ~/bin/adasex.x adasex.f
f90 -O3 -o ~/bin/xtrct.x xtrct.f

f90 -O3 -o ~/bin/stg1nx.x stg1nx.f
f90 -O3 -o ~/bin/stg2nx.x stg2nx.f
f90 -O3 -o ~/bin/stg3nx.x stg3nx.f
```

Note that as you run the codes, they may need to be redimensioned, these is usually done by changing the appropriate PARAM file, when a code stops it is usually quite helpful about telling you what to change.

Worked example for Na-like Fe

A D Whiteford and N R Badnell

15th October 2009

1 Getting the example input files

Copy the file `allanw/na-fe-example.tar` to your chosen directory.

```
cp ~whitefor/na-fe-example.tar .
```

Untar it.

```
tar -xf na-fe-example.tar
```

You will now have a number of files as explained in the next section.

2 Directory Strucutre

- `str` - Structure run
  - `autos.dat` - Autostrucutre input file
  - `radout` - placeholder for radial wavefunctions
- `ex` - Exchange runs
  - `dstg1` - stg1 input
  - `dstg2` - stg2 input
  - `dstg3` - stg3 input
  - `H.DAT` - placeholder for stg3 output
  - `NX1.DAT` - placeholder for NX data
  - `NX2.DAT` - placeholder for NX data
  - `radial` - symbolic link back to structure run radial wavefunctions
  - `fine/` - sub directory for outer region runs
    - `dstgf` - stgf input [Damping: stgfdamp]
    - `H.DAT` - symbolic link back to stg3 output

  - `coarse/` - as for above

- `nx`- non exchange
  - `dstg3` - copy of stg3 input file
  - `dstgf` - stgf input file
  - `dstgnx` - stg1nx, stg2nx, stg3nx input file
  - `NX1.DAT` - symbolic link back to NX output from exchange run
  - `NX2.DAT` - symbolic link back to NX output from exchange run
- `tot` - merging of all omega files
  - `script` - script to merge all the omega files together
  - `adasexj.in` - input file to generate adf04 file

3 Brief summary

Step	Description	Directory
1	Structure calculation	str
2	Inner region exchange runs	ex
3	Outer region exhcange runs	ex/fine, ex/coarse
4	Inner region non-exchange runs	nx
5	Outer region non-exchange runs	nx
6	Merge omega files	tot
7	Generate adf04 file	tot

4 Running the codes

Starting from the directory na-fe-example, type:

```
cd str
~/bin/autos.x < autos.dat

cd ../ex
~/bin/stg1r.x < dstg1
~/bin/stg2r.x < dstg2
~/bin/stg3r.x < dstg3

cd fine
~/bin/stgf.x < dstgf

cd ../coarse
~/bin/stgf.x < dstgf
```

```
cd ../../nx
~/bin/stg1nx.x < dstgnx
~/bin/stg2nx.x < dstgnx
~/bin/stg3nx.x < dstgnx
~/bin/stgf.x < dstgf

cd ../tot

cp ../ex/fine/OMEGA ./omgmrg1
cp ../ex/coarse/OMEGA ./omgmrg2
~/bin/omgmrg.x

mv omgmrgt omadd1
cp ../nx/OMEGA ./omadd2
~/bin/omadd.x

mv omaddt omgmrg1
cp ../str/OMGINFLS ./omgmrg2
~/bin/omgmrg.x

mv omgmrgt OMEGA
~/bin/adasex.x
```

Be sure that you know what each stage is doing, refer to the notes for further details as to what each program does and what the input files mean.

5 Looking at the results

In your `tot` directory you have two important files, `OMEGA` and `adf04`.

To look at individual collision strengths use the program xtrct.x which you compiled earlier. To look at the level 1 to level 2 collision strength, run xtrct.x then type "1 -2" and press return (note: this is '1' followed by a space, followed by '-2'. A file called xout will be produced which is in ascii format and can be interrogated with your favourite graphing package.