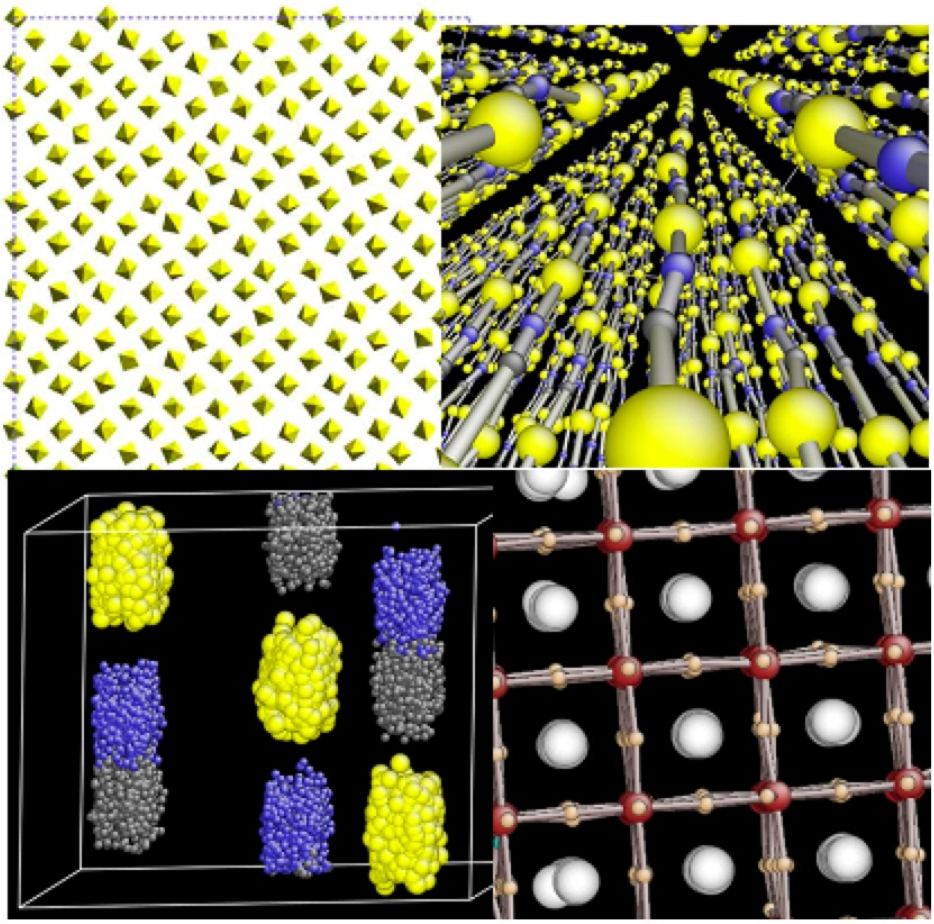
RMCProfile User Manual

Code version 6.4.6



Matt Tucker, Andrew Goodwin and Martin Dove

January 15, 2010

Contents

Chapter 1

Introduction

1.1 Preface

This manual aims to tell you most of what you need to know about our implementation of the Reverse Monte Carlo (RMC) method in the `RMCProfile` program.* RMC is an approach that will require some investment on the part of the user, at both the data collection and analysis stages. Accordingly you will probably want to read most of this manual before you start.

This manual undertakes to describe the RMC method (with some background information contained within Appendices), how to install and run the `RMCProfile` program, and how to prepare the input files. The manual contains a number of examples, with a couple of tutorials.

`RMCProfile` is still a work in progress, and we have not yet reached the stage where next versions are just minor iterations from previous versions. You will quickly appreciate that this manual reflects this fact, in particular with the juxtaposition of two types of file formats (v6 and 'classic').

1.2 Introduction: the "what is" and "why" of the Reverse Monte Carlo method

1.2.1 The "what is" in a nutshell

The *Reverse Monte Carlo* (RMC) method [1] will give you a unique view of the atomic structure of matter that is derived directly from experimental data. The mainstream usage of RMC is to analyse neutron and x-ray total scattering data from disordered materials, which are materials for which other probes can only give limited data. Examples include liquids and amorphous materials (which provided the original motivation for the development of the RMC method), magnetically disordered

*But because this is a work in progress, regrettably there will be some things that are not yet properly documented.

materials, and crystals with significant thermal disorder, rotational disorder of molecular groups, and site occupancy disorder.

At its heart, the RMC method is easy to understand. Essentially a configuration of atoms is modified by successive steps until properties calculated from it are in best agreement with experimental data. In the most common application, the property is either the pair distribution function or its Fourier transform as measured in a neutron or x-ray total scattering experiment. These data provide information about the short-range atomic structure of matter (bond lengths, numbers of atomic neighbours, layering of shells of neighbours about a central atoms, etc) and fluctuations in this structure. Some of this information can be derived directly from calculations based on the pair distribution function – for example, the first 1–3 peaks will give you information about the molecular fragments that any disordered matter is built from – but this information doesn't lead directly to a model. Crystallographers have an advantage here, because they can go from their diffraction data directly (and nowadays often quite quickly) to a model of the atomic structure that extends throughout space. RMC was designed to fill this void for liquids and amorphous materials, but it became clear that the method could provide unique information about disordered crystalline materials too.

The name of the RMC method gives away the fact that the process of building the atomic model relies on a Monte Carlo algorithm. This is an iterative approach in which successive changes to the atomic configuration are proposed at random, and then tested to see whether they improve or degrade the agreement that computed properties have with experiment data. If the proposed change improves the agreement with data, it is accepted, and the algorithm moves forward one cycle in which a subsequent random change is proposed. On the other hand, if the proposed change degrades the agreement with data, a probability algorithm is used to determine whether to accept or reject the proposed change. By accepting what appear to be bad changes prevents the method getting stuck in a state whereby the agreement with experiment can no longer be improved, even though there may be other configurations that are better. Thus the Monte Carlo method enables you to explore a wide range of possible configurations. The Monte Carlo method is based on statistical thermodynamics, which means that there are good reasons to expect it to produce configurations that are in best agreement with data.

Thus the RMC method is a computer simulation approach, but it differs from traditional simulations in that it is driven by experimental data rather than from parameterised equations. There is a huge world-wide industry in using models of atomic forces to construct models of disordered materials, where experimental data are used at the outset to tune the models and at the end to validate the model through comparison between data and predictions, but the actual modelling stage is divorced from experimental data. The RMC method takes a radically different approach, in that experimental data are used directly to drive the development of the model at all stages. There are no equations or parameters that drive the model.[†]

1.2.2 The "why" in a nutshell

Having briefly summarised the "what is" of RMC, the chances are that you are now impatient to know more about the "why". Like many techniques, RMC is not the sort of thing you can just dabble with, and so you need to be convinced that the pay-off is worthwhile.

[†]This should be qualified by noting that later on we will see how addition of some equations can help guide the simulation to focus on the more interesting features.

Let us go back to our first statement, namely that RMC will give you a unique view of the atomic structure of matter, and let us illustrate this by thinking about liquids and amorphous materials. Their atomic structure is described by the pair distribution function (PDF), which is effectively a histogram of interatomic distances. A typical PDF will contain a small number (typically 1–3) sharp peaks at small distances, followed by a structured distribution containing overlapping peaks on increasing distances. The first peaks can tell you about the relative positions of a small group of less than 10 atoms, but no more. The RMC method enables you to exploit the data in the overlapping peaks to build models that give you information about the atomic structure that extends to distances further than the first 3 peaks in the PDF.

Now is a good time to add a caveat. A sharp peak in the PDF will tell you three main things. First, the position of the peak will tell you about the average bond length between the two atoms it represents, and secondly, its width will tell you about the temporal and spatial variations in the bond length (which may arise from thermal fluctuations). Thirdly, the integral of the peak will tell you how many neighbours a single atom has. Let us illustrate this with the case of silica, chemical formula SiO_2. The first peak, at a distance of around 1.6 Å, corresponds to the Si–O bond. Its integral gives an average coordination number of 4. The obvious interpretation is that each Si atom has 4 O neighbours, exactly as found in all ambient-pressure crystal structures containing Si and O. However, the integral of the peak in the PDF only tells you about the average coordination number, and an average value of 4 neighbours does not exclude the possibility that there are large numbers of Si atoms with 3 and 5 O neighbours. And if you leave RMC to its own devices , it will generate configurations with 3 and 5 neighbours.

In light of this, it can often be advantageous to add additional constraints to the RMC method based on other experimental knowledge, such as the constraint that all Si atoms will prefer to have 4 O neighbours. This constraint is consistent with the position and integral of the first two peaks in the PDF for silica, corresponding to the Si–O and O–O nearest neighbour distances with four oxygen atoms arranged in a tetrahedral arrangement about the central siiicon atom. The third peak corresponds to the shortest Si–Si distance, and analysis of this peak tells you that neighbouring SiO_4 tetrahedra are linked at corners with an Si–O–Si angle of around 145°. Although this is nice information, it doesn't tell you much that you couldn't have previously guessed, given that this is exactly as found in crystal structures. What RMC will give you is information on how pairs of tetrahedra are linked together in a three-dimensional structure to form a larger connected network, and there is no other technique that can provide this information.

The RMC method gives a snapshot view of $\sim 10^4$ atoms, and if the fit is consistent with all the data available then the configuration will have the characteristics of the sample being studied. Of course, this number of atoms is a lot smaller than you would find in an experimental sample, and throughout the RMC simulation the positions of the atoms will fluctuate in a way that resembles thermal motions across the larger sample. Thus if an RMC simulation is run again from scratch or for a different length of time at equilibrium, a slightly different snapshot will be produced, where the atoms will have moved to different positions albeit within the constraints of the data being fitted. By collecting many independent configurations it becomes possible to do some powerful analysis in the same way that many configurations are analysed in any simulation based on thermodynamics. For example, recently RMC has been used to extract dynamical information from a collection of static configuration snapshots.

The authors of this manual have a strong interest in disordered crystals [2]. You might think that crystals are already catered for by standard crystallography tools based on x-ray and neutron

diffraction, but things are not this simple. Traditional Bragg scattering provides information about the distribution of atomic positions. In most cases, this means that diffraction will tell you about the average positions of atoms and their mean-square displacements due to thermal motions. However, Bragg diffraction contains no information about the correlated motions of atoms. This is illustrated with a vengeance in the case of high-temperature crystal polymorphs of silica such as *beta*-quartz [3] and *beta*-cristobalite [4]. If you take the crystal structure and calculate the distance between the average positions of the closest silicon and oxygen atoms, you get a value of around 1.55 Å. Often this distance is associated with the bond length. However, the PDF, which is a true measure of the average instantaneous bond length, gives an Si–O distance of 1.61 Å. The difference between the 'distance between mean positions' and the 'mean distance between instantaneous positions' reflects the existence of considerable disorder on a short length scale that arises from large-amplitude fluctuations of the structure. What tools do we have to study this disorder in a way that leads to atomic models? You guessed, RMC, and only RMC.

1.2.3 How `RMCProfile` fits into the picture

The original code for RMC was `RMCA`, and was designed for the study an amorphous and fluid matter. We started to develop `RMCProfile` as a significant extension of the original `RMCA` code in order to add support for new functionality, particularly to model crystals [5–7]. In principle we could simply have used `RMCA` for this, but we wanted to exploit the information that is specifically contained in the Bragg peaks separately from its contribution to the total scattering. Since then we have added a lot of new functionality, and we have also converted the code from Fortran77 to Fortran95 which has led to significant changes deep in the heart of the program. `RMCA` is no longer being developed and is not supported. Thus `RMCProfile` [7] should be seen as the evolutionary successor to `RMCA` and used in preference to `RMCA`; an alternative code developed by one of the core developers of `RMCA` is `RMC++` [8], and if you are not interested in crystalline you might like to take a look at this code.

As we have just said, the key feature of `RMCProfile` when applied to crystalline materials is that it separately handles both the total scattering and the Bragg scattering. Thus it make simultaneous use of the different information contained within these data concerning the distribution of the atomic positions in three-dimensional space and their correlations. For liquids and amorphous materials, the information about the distribution of atomic positions is of no value, and the structure is described entirely by the correlations between atomic positions. Traditional crystallography is focussed only on the distribution of atomic positions. The `RMCProfile` approach captures the best of both worlds, particularly when applied to the study of disordered crystalline materials in which there are significant local fluctuations from the average structure.[‡] In short, `RMCProfile` can produce a configuration of atoms that is simultaneously consistent with both the long-range and short-range order of a material as reflected in the information contained in the data: a truly holistic representation of the structure.

1.2.4 `RMCProfile` in a nutshell

In more detail, `RMCProfile` offers the following features:

[‡] But we stress that it works perfectly well for liquids and amorphous materials.

(1) Capacity to fit the Bragg profile directly as well as the total scattering.

(2) Capacity to fit both the scattering data and their derived pair distribution functions.

(3) Able to take account of instrument resolution.

(4) New to version 6.4, the incorporation of generalised molecular constraints for bond lengths and angles.

(5) Also new to version 6.4, the ability to correct the experimental data for the effects of finite configuration size.

(6) Incorporation of model-specific data-based polyhedral constraints, eg when studying materials containing molecular species (such as $NaNO_3$) or framework materials containing structural polyhedra (such as phases of silica).

(7) Ability to model spin configurations for magnetic materials.

(8) Ability to study systems with site disorder, such as cation disorder or vacancy inclusion.

(9) Ability to use a wide range of experimental data, including both x-ray and neutron diffraction, and XAFS data.

(10) Ability to use bond valence sums.

(11) XML output for data visualisation and analysis.

This manual is primarily focussed on Version 6.4 of `RMCProfile`, which represents a significant departure from the previous version, Version 5, here called the 'classic' version. The classic version of `RMCProfile` retained the same form of input file as was used in `RMCA`, and the code base was essentially that of `RMCA` with additional subroutines for new functionality. Although Version 6.4 is not a complete rewrite of the code, it was nevertheless a major revision, including conversion to Fortran95, which as previously mentioned required a lot of reorganisation and rewriting. New functionality has been added, together with the introduction of completely new input file formats. At the present time `RMCProfile` can still handle classic version file formats, but we strongly recommend that existing users switch to the new formats.[§] To help this switch, we have provided a some new tools to perform the file conversion that are described in section 4.6.

The aim of this manual is to explain the practical issues of how to run `RMCProfile`, and to describe the various files you need and how to format them. The theory on which the method is based is described in the Appendices.

[§]Not least because we do not guarantee support for classic file formats for ever.

Chapter 2

Installing and Running RMCProfile

2.1 Installation

At this point it is important to note that `RMCProfile` is provided as a singe executable that is designed to be run as a shell command (otherwise known as the command line or prompt if you are are using a Windows computer). Thus no special installation is required; just put it somewhere you can easily access it (eg in `/usr/local/bin` or in a folder that is accessed from your command path if you are using Linux or a unix-based system such as Mac OS X; or somewhere like `C:/rmcprofile` if you are using a Windows computer).

There are various other programs supplied as tools to help with preparation and analysis of the `RMCProfile` files. As with the main program most of these are standalone programs, however some are supplied in an archive (`.zip`) form. In this case all the files in the folder need to be kept together and the program run from its own folder.

2.2 Running `RMCProfile`

2.2.1 Input files

`RMCProfile` requires a number of input files, that are described in the following sections. Some of them will have the same stem name but different extension tags. For example, if we were performing an RMC simulation on the disordered crystalline material KCN, we might have the following input files (optional unless stated otherwise):

`KCN.dat` containing the key data required to control the RMC simulation (compulsory, section 4.1).

`KCN.rmc6f` or **`KCN.rmc6c`** containing the atomic configuration in fractional or cartesian coordinates respectively. If using classic format, the file name to be used is **`KCN.cfg`**, and you also have the option to call this **`KCN.rmc3f`** if using Version 6 **`.dat`** file with classic configuration format. (This file is compulsory unless **`KCN.his6f`** or **`KCN.his6c`** is present, section 4.2).

KCN.his6f or **KCN.his6c** contains the latest configuration and the pair distribution functions as generated by the most recent run, and if this file is present it will be used in preference to **KCN.rmc6f** or **KCN.rmc6c**. If using the classic file format, this file is called **KCN.his** (subsection 4.2.3).

KCN.bragg containing the Bragg diffraction data (section 4.4).

KCN.inst containing information on the instrument resolution function, in the format as provided by the GSAS Rietveld refinement code (subsubsection 4.4.2c).

KCN.hkl containing information on the range of Bragg peaks to be considered in the analysis of the Bragg diffraction data (this shortly to be incorporated into the main **.dat** file; subsubsection 4.4.1a).

KCN.dw containing information on the distance window constraint as described in section 3.3 (this will shortly be incorporated into the main **.dat** file).

The **KCN** stem part of the file names can be anything you like, but as in this example this group of files must have the same stem name.

These files are described in the chapters and sections indicated above

2.2.2 Running RMCProfile

To run RMCProfile, execute the following shell command:

```
rmcprofile KCN > KCN.log
```

The part of the line "> KCN.log" tells the command to direct the standard output to a specified file, here called **KCN.log** but which actually could be called anything you like. If this part of the line is not included, the standard output goes directly to your screen.

In this example we have assumed that the program file, here called **rmcprofile** but which could be called anything you want to rename it to, will be automatically picked up by the command interpreter, which means that it will either be in a special directory that is included in your PATH or is in your working directory which is associated within your PATH. If the program is in your working directory but your PATH is not set up to detect programs within it, you need to modify the command to

```
./rmcprofile KCN > KCN.log
```

Otherwise you need to give the full name of the program file, including all the directory information.

2.2.3 Output files

RMCProfile will generate a number of output files. These include, following the KCN example:

KCN.out contains a lot of summary information and generated data.

KCN.xml contains a lot of summary information and generated data if you have specified a request for CML data.

KCN.xhtml contains a nice web page summary of the results of the RMC simulation, provided you have requested CML data and the use of the ccviz program.

KCN.his6f or **KCN.his6c** contains the configuration and the pair distribution functions generated by the run.

KCN.rmc6f or **KCN.rmc6c** will contain an updated configuration. If you request periodic saves of the configurations, these will have a number appended as, for example, **KCN_23.rmc6f**.

KCN_SQpartials.csv will contain the calculated partial scattering functions in CSV format.*

KCN_SQ1.csv will contain the calculated and experimental data for the first file of scattering data in CSV format. If more than one file of scattering data is used, the number **1** will be replaced by the subsequent number of the data set.

KCN_PDFpartials.csv will contain the calculated partial pair distribution functions in CSV format.

KCN_PDF1.csv will contain the calculated and experimental data for the first file of PDF data in CSV format. If more than one file of PDF data is used (not a common case), the number **1** will be replaced by the subsequent number of the data set.

KCN.braggout will contain the calculated and experimental Bragg diffraction data.

KCN_bragg.csv will contain the calculated and experimental Bragg diffraction data in CSV format.

KCN.cssr which contains the configuration in a form that can be read by several visualisation programs, including CrystalMaker which is endorsed by the authors.

hkls contains information used by the Bragg modules and which can be retained to be used by a subsequent run of RMCProfile.

KCN.amp contains information used by the Bragg modules and which can be retained to be used by a subsequent run of RMCProfile.

These files will be described in subsequent chapters.

*Comma Separated Values, which is exactly as described by the name, and is excellent because this format can easily be read into many analysis programs, such as Microsoft Excel.

Chapter 3

Capabilities

3.1 Interatomic potentials

3.1.1 Why interatomic potentials?

In principle* the RMC method should be sufficient to drive a simulation, and in particular adding interatomic potentials to help drive the simulation moves us away from a purely data-driven approach. However, we have found that there are cases where the use of interatomic potentials – called "polyhedral restraints" in the classic version of `RMCProfile` – have an important role. For example, small errors in the data can cause the configuration to distort locally, and some form of restraint can be useful to minimise this effect. Moreover, the RMC method has no means to definitely associate any feature in the data with specific features in the configurations. For example, a peak with an area corresponding to a mean coordination number of 4 doesn't actually preclude the formation of structures with coordination numbers of 3 and 5 provided that the average remains as 4. Thus the use of potentials for restraints on the configuration can have a role in *preventing* bad things happening to the configuration, which is not quite the same as *forcing* some desired behaviour.

In some cases, particularly with molecular crystals, the PDF at lower distances can be dominated by the contribution from intramolecular distances. This may not be very helpful, since there is often little of scientific interest in the shape of the molecule as compared to how separate molecules interact. Using potentials is a way of enabling the simulation to place more emphasis on the interatomic contacts that are not included in the potential.

With proper weighting of the contribution of the potential with respect to the weighting given to the agreement with data, the role as a simple restraint works well. Only when the data are underweighted will the potentials be the primary driving force in the simulation.

*and a principle the relevant author subscribes to.

3.1.2 The available interatomic potentials

3.1.2a Bond-stretching potential

We assume that the energy required to stretch a bond can be described using a Morse potential:

$$E_{\mathrm{M}} = D[1 - \exp(-\alpha(r - r_0))]^2 \tag{3.1}$$

The function is written in this way so that the energy is zero at the origin, as it is for simple polynomial expansions of the bond-stretching energy. Aside from the r_0 parameter, The Morse function has three adjustable parameters, namely r_0 to specify the length of the bond, D to specify the energy required to break the bond, and α which plays a role in specifying the curvature of the potential energy function around the minimum. Following the MM3 molecular mechanics force field model, we set α = 2.55 Å^{-1} for all atom pairs.

We can write a simple expansion of the Morse potential to lowest order:

$$E_{\mathrm{M}} = D\alpha^2 (r - r_0)^2 \tag{3.2}$$

If you prefer to work with this type of quadratic expression but expressed in the following way:

$$E_{\mathrm{M}} = \frac{1}{2} k (r - r_0)^2 \tag{3.3}$$

the association between k, D and α is clear.

We recommend the following realistic values for the parameters, taken from the MM3 database (units of D are eV, units of r_0 are in Å):

C_{sp^3}–C_{sp^3}	D = 3.148, r_0 = x
C_{bz}–C_{bz}	D = 2.155, r_0 = x
C_{bz}–H	D = 2.472, r_0 = x
C_{carb}–O_{carb}	D = 3.48, r_0 = x
C_{carb}–C_{bz}	—
C_{sp^2}–C_{sp^2}	D = 3.60, r_0 = x
C_{sp^2}–H	D = 2.472, r_0 = x
C_{im}–N_{im}	D = 8.840, r_0 = x
C_{im}–H	D = 2.208, r_0 = x

We recommend using realistic rather than artificial potentials, provided that the weighting you use for the data is derived from the errors on the data.

3.1.2b Bond angle potentials

We use a simple harmonic cosine potential to describe the energy associated with bending of bonds:

$$E = \frac{1}{2}K\left(\cos\theta - \cos\theta_0\right)^2 \tag{3.4}$$

where θ is the instantaneous bond angle, and θ_0 is the set angle. This equation can be expanded to yield

$$\begin{aligned} E &= 2K\sin^2\tfrac{\theta+\theta_0}{2}\sin^2\tfrac{\theta-\theta_0}{2} \\ &\approx \tfrac{1}{2}K\sin^2\theta_0\left(\theta-\theta_0\right)^2 \end{aligned} \tag{3.5}$$

where the angles are defined in units of radians rather than degrees. As with the bond-stretching potential, we recommend the use of realistic values of the parameter K. Values of $K/\sin^2\theta_0$ for some common bonds are (units of K are eV, and angles are given in degrees):

C_{bz}–C_{bz}–C_{bz}	$K/\sin^2\theta_0 = 39.71, \theta_0 = 120$
C_{bz}–C_{bz}–H	$K/\sin^2\theta_0 = 25.61, \theta_0 = 120$

3.1.3 The main data file

Consider the following example from an input **`.dat`** file:

```
POTENTIALS ::
 > STRETCH :: C C 4.0 eV 1.15 Ang
 > STRETCH :: Zn C 2.0 eV 1.95 Ang
 > STRETCH_SEARCH :: 10%
 > ANGLE :: Zn C C 4.0 eV 90.0 deg 1.95 1.95 Ang
 > ANGLE :: C Zn C 2.0 eV 180.0 deg 1.95 1.15 Ang
 > ANGLE_SEARCH :: 10 deg
 > TEMPERATURE :: 300 K
> PLOT :: pixels = 400, colour = charcoal, zangle = 90.0, zrotation = 45.0 deg
```

Here we have defined two bonds, for C–C and Zn–C bonds, which are given in the "`> STRETCH ::`" lines. The order in which the information on the two bonds is provided will lead to the association of numbers 1 and 2 to the two bonds (which you need to know about for the bonds file below). Here we have requested that `RMCProfile` search for bonds of distance 1.15 and 1.95 Å respectively, with a tolerance of 10%. We could have specified the tolerance in absolute units instead.

The example from which this snippet was taken consisted of ZnC_6 octahedra. We have therefore defined a bond-bending potential (the first "`> ANGLE ::`" line) to describe the C–Zn–C angle of

equilibrium value 90°. The octahedra are connected via C–C bonds, which form part of a linear Zn–C–C–Zn linkage. We might want to add a potential to ensure the RMC simulation doesn't allow large departures from this linear configuration, and this is the role of the second angle constraint. Note that the equilibrium angle is actually used in the initial search of atom triplets, and the tolerance on the angles for the initial search is provided by the `ANGLE_SEARCH :: 10` line. As with the bond stretching potential, the ordering in which the information on the two bond angles is provided with lead to the association of numbers 1 and 2 to the two triplets (which you need to know about for the triplets file below). In the first case we have requested that `RMCProfile` search for bond angle triplets with bond lengths of distance 1.95 Å for both Zn–C bonds, with a tolerance of 10% on the bond lengths and with a tolerance of 10° on the bond angle. We could have specified the tolerance on the bond angle in terms of a percentage instead.

The temperature provided plays a role that is related to weighting of the data, and in the polyhedral restraints method this is seen as a parameter to tune in concert with the weighting on the data. However, when using realistic numbers in the interatomic potentials, the sample temperature should be able to play a quantitative role in the modelling, at least for the *D* parameter, in that it should give rise to a peak in the PDF of correct width.[†] The exact value of r_0 can be tuned to correspond with the position of the peak in the PDF (which may not exactly match the recommendations).

The "`> PLOT ::`" line enables the bond orientation distribution function to be plotted as a coloured stereographic projection: this is discussed below.

3.1.4 The bonds file

The bonds file will be generated by the first run of `RMCProfile`, using the information contained within the "`POTENTIALS ::`" keyword block, and will have the root name of the simulation with the **`.bonds`** extension. When this file exists, the data contained within it will supersede the bonds data provided in the input file. This is a good thing since often the bonds are established on a nicely ordered structure with no fluctuations that are hard for an automatic method to detect. If you don't want to use an existing bonds file, then rename or delete it.

A typical example has the form:[‡]

```
Metadata file type :: Bonds file for RMC simulation
Metadata creation date :: 12-08-2009
Metadata material :: ZnC4
Number of atoms = 10
Number of bonds = 2
.............................................
1 Zn 1 ::  0
2 Zn 1 ::  0
3 C 1 ::  8 C ; 1
<snip>
10 C 1 ::  5 C ; 1
```

[†] This might not work at low temperature because this discussion ignores the effect of quantum zero-point motions, in which case it might be necessary to use a higher temperature.

[‡] The `<snip>` lines replace similar data, removed for the sake of brevity

```
1 Zn 2 ::  3 C ; 4 C ; 6 C ; 10 C ; 4
2 Zn 2 ::  5 C ; 7 C ; 8 C ; 9 C ; 4
3 C 2 ::  1 Zn ; 1
4 C 2 ::  1 Zn ; 1
<snip>
9 C 2 ::  2 Zn ; 1
10 C 2 ::  1 Zn ; 1
```

The structure of this file is important. The `Metadata` lines are important to link this file with the actual simulation, but will be ignored by `RMCProfile`. They are their for your benefit, so don't delete them! The two lines containing the number of atoms and number of bonds are important and must be accurate. The line of dots is used to divide the header from the data.

So now we consider the actual bonds data. For each bond there is a list of all the atoms in the same order as the configuration – if you break this order, that will be recognised and the program will abort. We look at the data on any one line, and consider the following example line:

```
1 Zn 2 ::  3 C ; 4 C ; 6 C ; 10 C ; 4
```

The first number specifies the atom number in the configuration file, and this is followed by the chemical element symbol. The third number gives the bond number. So in this example, we are looking at the first atom in the configuration, which happens to be a zinc atom, and this lines is concerned with the second bond specified in the input file.

The double colons are important, because they separate the description of the data (to the left) from the actual bond data (on the right). Consider first the last number. This gives the number of bonds, which in this case is 4 denoting a tetrahedral coordination. Note that there are four semicolons; these separate the data for each bond. And the data for each bond merely consists of an atom number and its chemical symbol. So in this example, the zinc atom is bonded to four carbon atoms, which are numbers 3, 4, 6 and 10 in the configuration.

Ideally you should not need to edit this file, but if something in your system is more complicated, then sadly you will have to do some hand work.

3.1.5 The triplets file

The triplets file, which contains the information about bond angles, will have the root name of the simulation with the `.triplets` extension. As with the bonds file, the triplets file is created automatically if it doesn't exist.

A typical example has the form:[§]

```
Metadata file type :: Bonds file for RMC simulation
Metadata creation date :: 12-08-2009
Metadata material :: ZnC4
```

[§]The `<snip>` lines replace similar data, removed for the sake of brevity

```
Number of atoms = 10
Number of bonds = 2
Number of triplets = 2
.............................................
1 Zn 1 :: 101 C 102 C; 101 C 103 C ; 101 C 104 C ; 101 C 105 C ; 4
1 Zn 1 :: 0
<snip>
101 C 1 :: 0
101 C 1 :: 1 Zn 102 C ; 1 Zn 103 C ; 1 Zn 104 C ; 1 Zn 105 C ; 4
<snip>
101 C 2 :: 1 Zn 201 C ; 1
101 C 2 :: 201 C 2 Zn ; 1
<snip>
```

First we note that this file will be generated automatically by `RMCProfile` at the first run when the `POTENTIALS ::` keyword block is used. Thereafter its existence will be recognised, and the data within the file will supersede the triplets data provided in the input file. This is a good thing since often the bonds are established on a nicely ordered structure with no fluctuations that are hard for an automatic method to detect. If you don't want to use an existing triplets file, then rename or delete it.

The structure of this file is important. As in the `.bonds` file, the `Metadata` lines are important to link this file with the actual simulation, but will be ignored by `RMCProfile`. The two lines containing the number of atoms and number of triplets are important and must be accurate. The line of dots is used to divide the header from the data.

So now we consider the actual triplets data. For each triplet type there is a list of all the atoms in the same order as the configuration – if you break this order, that will be recognised and the program will abort. Unlike the `.bonds` file, we have two lines for each atom. This is because an atom can be either at the centre of a triplet or at the end. The first line describes the atomic connectivity when the atom is at the centre of a triplet, and the second line describes the atomic connectivity when the atom is at the end of a triplet.

Consider the following pair of lines for one atom:

```
1 Zn 1 :: 101 C 102 C; 101 C 103 C ; 101 C 104 C ; 101 C 105 C ; 4
1 Zn 1 :: 0
```

The first line describes the connectivity when the atom, Zn in this case, is at the centre of a C–Zn–C triplet. The first number specifies the atom number in the configuration file, and this is followed by the chemical element symbol (Zn in this case). The third number gives the number of the triplet type, as determined by the order in which the `> ANGLE ::` line occurs in the data file. In this example, we are looking at the first atom in the configuration, which happens to be a zinc atom, and this pair of lines is concerned with the first triplet specified in the input file.

The double colons are important, because they separate the description of the data (to the left) from the actual bond data (on the right). Consider the last number. This gives the number of triplets which involve the atom. In this example, the Zn atom occurs as the centre of 4 triplets (from the first

line) and is never at the end of the triplet (as specified by the zero in the second line). The triplets are all of the form C–Zn–C.

Now consider a second example:

```
101 C 1 :: 1 Zn 201 C ; 1
101 C 1 :: 201 C 2 Zn ; 1
```

These lines describe the second triplet defined in the data file, namely C–C–Zn linkages, and are concerned with carbon atom that is atom number 101 in the configuration. The first line describes the way that this atom is at the centre of a triple and bonded to Zn atom 1 and C atom 201. This atom is the centre of only one triplet. The second line describes the way that this atom is at the end of the triplet bonded to C atom 201as the centre of the triplet and Zn atom 2 at the other end.

Ideally you should not need to edit this file, but if something in your system is more complicated, then sadly you will have to do some hand work.

3.1.6 Visualisation of bond orientation distribution functions

RMCProfile allows for easy plotting of bond orientation distribution functions.

3.1.6a Histogram file

RMCProfile produces histograms of the number of bonds over the non-uniform grid of θ and ϕ polar coordinates, defined using the convention used in the physics community.[¶] This file is given a name of the form <name>.bondodf_n, where <name> is the root name of the RMC simulation, and n corresponds to the bond number. It is important to note that this file is no more than a dump of the numbers of bonds that lie within the bit of solid angle defined by θ and ϕ, and is not normalised for the size of the solid angle. The grid is in uniform steps of $\Delta\theta$ and $\Delta\phi$, with grid size of 40×80 cells.

3.1.6b PPM plot file

By using the > PLOT :: subordinate keyword, RMCProfile will produce data in a form suitable for plotting as a stereographic projection, as per this example:

The file is produced in the Portable PixMap format, with name <name>_bondplot_n.ppm.[‖] To convert to a more-standard format such as png, jpg, gif, tiff, eps or pdf, one option is to install a tool such as Imagemagick.[**] To produce the image from Imagemagick, execute the following shell command:

```
convert -transparent black <name>.ppm <name>.<extension>
```

[¶] Note that the mathematics community switches the meanings of these symbols; here θ gives the angle a vector makes with the *z* axis ($0 \leq \theta \leq 180^\circ$), and ϕ gives the angle that the vector is rotated about the *z* axis ($0 \leq \phi \leq 360^\circ$).

[‖] The PPM format is described at http://en.wikipedia.org/wiki/Portable_pixmap

[**] which is available from http://www.imagemagick.org/

Figure 3.1: Example of a stereographic representation of the bond orientation function generated by `RMCProfile`.

where the two placeholders indicated by `<name>` denote the main file name and the name of the file you want to generate, and `<extension>` gives the file type (examples are `png`, `gif`, `pdf` and `eps`). The modifier `-transparent black` will convert black to transparent; the background is written using a black colour, and this allows the background to be made transparent.

If you get an error message of the type:

```
convert: unable to access configure file 'colors.xml'.
```

you are probably able to ignore it; the message is telling you that it has used an internal colour map.

Other graphics program, such as `GraphicConverter`[††] for Mac OS X, and Adobe's Photoshop, are able to read and manipulate files in the PPM format.

3.1.6c Converting the histogram file to a PPM file

We provide a utility called `bondplot`, a Fortran 90 program, to allow you to convert from the histogram file to a PPM file. This gives the user some control over various options which have assumed values when the file is generated by `RMCProfile`. You run `bondplot` as a simple command within the shell or command interface, with no parameters. The program will, in order, ask for the following information

(1) The name of the histogram file;

[††] http://www.lemkesoft.com/

(2) The name of the required output file (which much have extension `.ppm` for the graphics conversion programs to work);

(3) The required number of pixels along an edge of the plot (which always has square shape);

(4) Values of angle θ and ϕ throughwhich to rotate the sphere;

(5) Option to change the maximum and minimum values of the bond odf that correspond to the extreme colours being used;

(6) Background colour;

(7) Option to rotate the plot in order to produce an animated gif file.

This generates the PPM file which you can then convert to a graphics file in a standard format using the methods described in the previous section.

3.2 Magnetic structure modelling

3.2.1 Introduction

The use of `RMCProfile` to refine magnetic structures is based on the notion of pairing each atomistic configuration with a corresponding supercell spin configuration. The positions of the magnetic moments in the spin configuration are then determined by the positions in the atomistic configuration. Each RMC move involves either a change in atomic positions or magnetic moment orientations. The relative frequency of each choice is left as a user-defineable parameter. Naturally, displacement moves of non-magnetic species do not affect the magnetic scattering functions, nor do spin displacement moves of magnetic species affect the nuclear scattering functions. Consequently the only significant additional computational cost suffered is involved in the translations of magnetic atoms, whereupon changes in both nuclear and magnetic scattering functions must be calculated.

3.2.2 Algorithm

Spin orientation moves are implemented within `RMCProfile` as follows. The orientation of each spin (*i.e.* the normalised spin vector) is treated as a point $\mathcal{P}$ on the surface of a sphere. A random spin move vector $\mathcal{M}$, whose magnitude σ_{max} determines the maximum change in spin orientation and is determined by the user, is added to $\mathcal{P}$ and the resultant vector projected back onto the surface of the sphere to give the new spin orientation $\mathcal{P}'$. The probability distribution associated with this algorithm has its maximum at a move size of σ_{max}, and so it is usually appropriate to limit the size of this parameter to relatively modest values (*ca* 0.1).

The magnetic contribution $S_{\text{mag}}(Q)$ to the scattering factor is calculated from the RMC configurations via two real-space correlation functions $A(r)$ and $B(r)$:

$$\begin{aligned} S_{\text{mag}}(Q) &= \frac{2}{3}c_{\text{M}}\left[\frac{e^2\gamma}{2m_{\text{e}}c^2}gJf(Q)\right]^2 + 4\pi\rho c_{\text{M}}\left[\frac{e^2\gamma}{2m_{\text{e}}c^2}f(Q)\right]^2 \\ &\quad \times \int r^2\left\{A(r)\frac{\sin Qr}{Qr} + B(r)\left[\frac{\sin Qr}{(Qr)^3} - \frac{\cos Qr}{(Qr)^2}\right]\right\}\text{d}r, \end{aligned} \tag{3.6}$$

where c_M is the concentration of the relevant magnetic specices, ρ is the number density of magnetic atoms, e, γ, m_e and c carry their usual meanings, gJ is the magnetic moment and $f(Q)$ the magnetic Q-dependent scattering form factor. The real-space functions $A(r)$ and $B(r)$ essentially measure the magnitude of spin-spin correlations perpendicular to and parallel to the vector that joins each pair of magnetic atoms. They can be calculated directly from the RMC configurations, and function as magnetic analogues of the nuclear pair distribution functions.

The magnetic contribution to the Bragg intensities is calculated using what is also a standard approach. The key equation involved is

$$I(\mathbf{Q}) = \frac{1}{N}\left|\sum_j \mathbf{q}_j p_j(Q)\langle \exp(i\mathbf{Q}\cdot\mathbf{r}_j)\rangle\right|^2, \tag{3.7}$$

where the magnetic interaction vector $\mathbf{q}_j$ is given by

$$\mathbf{q}_j = \frac{\mathbf{m}_j}{m_j} - \frac{\mathbf{Q}(\mathbf{Q}\cdot\mathbf{m}_j)}{Q^2 m} \tag{3.8}$$

and $\mathbf{m}_j$ is the spin vector of the magnetic species j. The magnetic scattering amplitudes $p_j(Q)$ are related to the magnetic form factors $f_j(Q)$:

$$p_j(Q) = \frac{e^2\gamma}{2m_e c^2} g J_j f_j(Q). \tag{3.9}$$

The Bragg intensities calculated in this way can be converted into a Bragg profile function in precisely the same manner as for nuclear scattering.

3.2.3 Implementation

The various keywords and required parameters are listed in section 4.1 above, and are not repeated here. The basic idea is that all spin-related files run from a slightly different (user-specified) stem to that used for the standard RMC files. For example, one might use `mno_spin` as the stem name for the spin files associated with `mno.cfg, mno.his`, etc. There are a few additional points to note:

(1) The spin configurations given in the spin `.cfg` file are normalised — the actual magnitude of a spin comes from the values given in the `MAGNETIC_ATOMS` keyword in the input file.

(2) The magnetic form factors can be calculated by `RMCProfile` using the standard analytical formula

$$f(Q) = A\exp(-aQ^2/16\pi^2) + B\exp(-bQ^2/16\pi^2) + C\exp(-cQ^2/16\pi^2) + D, \tag{3.10}$$

where A, a, B, b, C, c, D are empirical coefficients as defined in *e.g.* Acta Cryst. A**27**, 545 (1971). Alternatively, it is possible for the user to provide their own form factors as a separate file, so long as these are given for precisely the same Q values as in the neutron scattering data. The relevant flags are discussed in section 4.1 above.

(3) The order of the magnetic atoms in the RMC configurations is important, in that all the magnetic atoms *must* be given first. Naturally, the order of the atom types in the spin configuration files must be the same as the order of the magnetic atom types in the nuclear RMC configurations.

(4) In general, magnetic structure refinement takes a substantially longer time than nuclear refinement, and there is a significant degree of interplay between the nuclear and magnetic structures. A reasonable general approach appears to attempt refinement of the nuclear structure first, using random spin orientations (but not refining these). Once the nuclear refinement appears to have reached equilibrium, one might then consider either refining the spin orientations while keeping atom positions constant, or proceeding with a dual refinement. From experience, a spin move rate of about 0.2 seems to work well in the latter approach.

(5) The value entered as the input parameter `MAX_SPIN_MOVEMENT` is the variable σ_{max} described above. In practice a value of about 0.1 seems to work well, but it is important to keep an eye on the number of accepted spin moves and to adjust this value accordingly. During its periodic updates to screen, the program will allow the user to assess how many spin moves are accepted (also relative to the number of displacement moves, if these are allowed); additional information comes from the change in χ^2 for each spin move, which gives the user an idea of how strongly the data are driving magnetic structure refinement.

3.2.4 Example: refinement of magnetic structure in MnO

Introduction

This exercise focusses on the use of RMCProfile to refine magnetic structure in magnetic materials. Just as RMC can be used to refine the crystal structure of a material in terms of the positions of atoms in a large supercell, we can refine magnetic structures in terms of the orientations of spins in similar atomistic configurations. One of the advantages of an RMC approach to magnetic structure refinement is that it is often possible to solve the magnetic structure even when starting from a completely random ensemble of spin orientations.

The example we will work through concerns the magnetic structure of the well-known antiferromagnet MnO. At temperatures below 120 K the $S = \frac{5}{2}$ magnetic moments of the Mn^{2+} ions align ferromagnetically within (111) planes of the rocksalt crystal lattice. The magnetisation direction within planes then reverses from one plane to the next, giving the overall antiferromagnetic structure shown in Fig. 3.2. In fact there is also a slight deviation from cubic lattice symmetry associated with this magnetic transition, but here we will ignore this effect.

The MnO atom and spin RMC configurations

We are going to refine the spin orientations in a $4 \times 4 \times 4$ supercell of the unit cell shown in Fig. 3.2. The file `mno.cfg` contains the positions of 512 atoms—256 Mn and 256 O atoms. The positions of these atoms have been displaced slightly from their average positions. A version of this configuration file in the format readable by ATOMEYE is given as `mnoeye.cfg`. It is worth taking a look at the structure in ATOMEYE at this stage, just to familiarise oneself with the atom positions. A picture of the configuration is shown in Fig. 3.3.

A set of 256 random spin orientations are given in the file `mno_spin.cfg`. One method of visualising these orientations is to colour the Mn atoms in our ATOMEYE configuration according to the individual spin directions. There is a program ATOMEYEPREP provided that helps prepare the relevant files. If we run the command

```
atomeyeprep < atomeyeprep.in
```

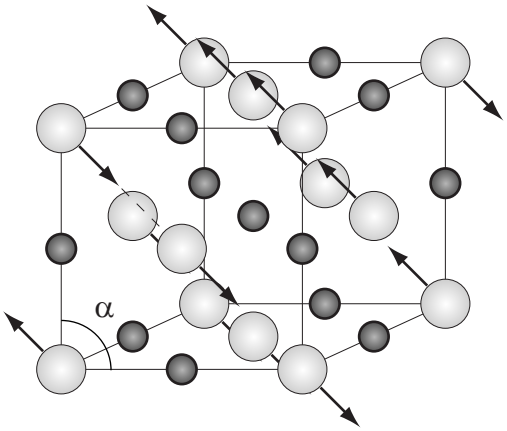


Figure 3.2: The antiferromagnetic structure of MnO: Mn and O atoms are shown as large light-grey and small dark-grey spheres, respectively.

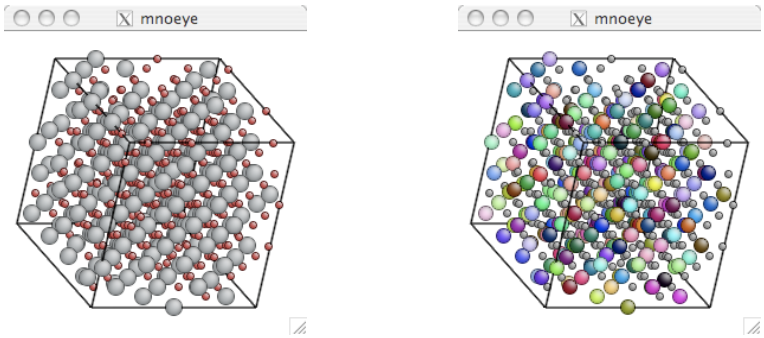


Figure 3.3: The RMC starting configuration as seen in ATOMEYE, using default colouring (left) and with Mn atoms coloured by the initial spin directions (right).

then a new file, `mnoeye.clr` is produced, which tells ATOMEYE how to colour each atom. Re-launching ATOMEYE via

```
atomeye.bat mnoeye.cfg
```

shows the same configuration as we saw previously, but the O atoms are now coloured grey, and the Mn atoms are coloured according to the spin orientations in `mno_spin.cfg`. A typical view is shown in Fig. 3.3. The main point is that things look quite random!

It is instructive to see what the diffraction pattern looks like when calculated from this combination of atomistic and spin configurations, and how it differs from the experimental data. The RMCProfile parameter file `mno.dat` is set up ready to be used for this magnetic refinement. The experimental data are stored within the file `mno_10k_sq.dat`, and we are using a Q range of approximately $0 < Q < 25\,\text{Å}^{-1}$. These data have already been convoluted with a box function of width $8.88\,\text{Å}^{-1}$, which is half the box size of our configuration. Running the program using the command

```
rmcprofile mno
```

we obtain the output file `mno.out`, which includes the fit-to-data shown in Fig. 3.4. There is reasonable agreement over most values of Q, except between $0 < Q < 3\,\text{Å}^{-1}$, where the key magnetic structure reflection is not modelled well at all.

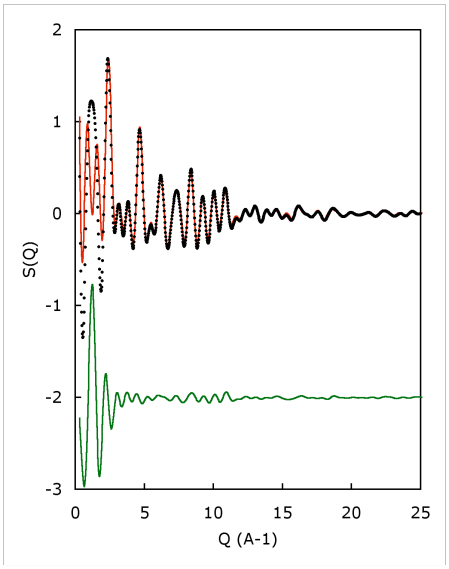


Figure 3.4: Initial RMC fit to neutron scattering data for MnO. Data are shown as solid points, the RMC fit as a red line, and the difference (data−fit) shown as a green line, shifted by 2 units.

RMC refinement

In order to proceed with a refinement of the magnetic structure, we must edit the `mno.dat` file to tell RMCProfile how long to run the refinement, and also how often to save. The relevant line in the parameter file is

```
0 0                          ! Time limit, step for saving
```

which we change to

```
5 1                          ! Time limit, step for saving
```

It so happens that five minutes is sufficient in this simple case to arrive at a reasonable fit to the magnetic structure. More complicated structures will take longer! We run the program as before, using the command

```
rmcprofile mno
```

The fit should have converged within five minutes with a goodness-of-fit reducing from its initial value χ^2 = 395.9 to $\chi^2 \simeq 12$. If the value of χ^2 is significantly larger than this value, it might be worth running RMCProfile once more.

Analysis

First, we can take a look at the new fit-to-data, once again by plotting the values given in `mno.dat`; a representative plot is given in Fig. 3.5. The key difference with the previous fit (*i.e.*, before refinement of the magnetic structure) is that the (111) magnetic peak is now well modelled. Note that our refinement only involved moving the magnetic moments (all the atom positions remain the same), so it really is the magnetic structure that accounts for this peak.

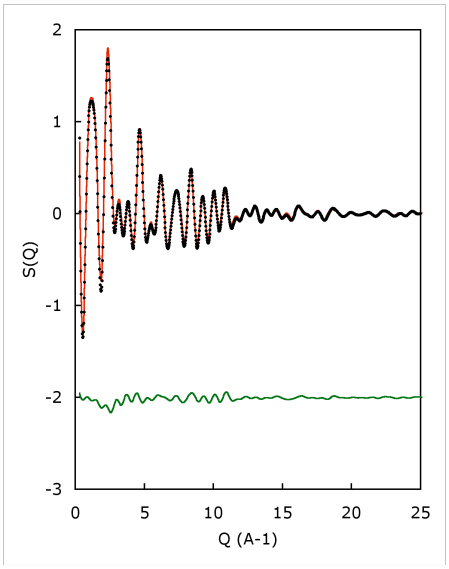


Figure 3.5: A typical equilibrium RMC fit to neutron scattering data for MnO.

Before looking at the actual spin configuration, we are going to set up RMCProfile to produce a series of equilibrium configurations. This will help us produce smoother distribution functions by increasing sample size. Again, we need to amend the `mno.dat` file, this time changing

```
.false.                 ! number of configurations to collect
500                     ! step for printing
5 1                     ! Time limit, step for saving
```

to

```
.true.                  ! number of configurations to collect
500                     ! step for printing
10 5                    ! Time limit, step for saving
```

Again, we run RMCProfile using the command

```
rmcprofile mno
```

but this time it will save a copy of the configuration file after every 500 generated moves, numbering the files sequentially. In order to generate 50 new configurations, we will need to leave the program to run for approximately 10 minutes.

Meanwhile, what we really want to do is to see what the newly-refined magnetic structure looks like. We will use ATOMEYE for this purpose, so we need to run the command

```
atomeyeprep < atomeyeprep.in
```

which will produce a new `mnoeye.clr` file from the equilibrium spin configuration. Launching ATOMEYE with

```
atomeye.bat mnoeye.cfg
```

will now show the new structure. Because there are four different symmetry-equivalent [111] axes, the actual direction of the magnetic ordering will differ from run to run. Also, the absolute direction of the spins will vary as well, and so the colours may also be different. Nevertheless a typical configuration is shown in Fig. 3.6. What should be clear, after some playing with the orientation, is that the magnetic structure is now composed of ferromagnetic layers, as described in the introduction.

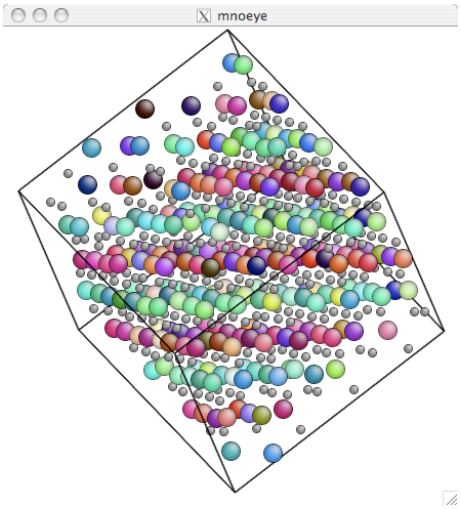


Figure 3.6: A typical RMC equilibrium configuration as seen in ATOMEYE.

Finally, we return to the series of equilibrium configurations produced by RMCProfile. If fewer than 50 configurations have been prepared in the 10 minutes, then run the program once again (it will automatically resume its sequential numbering from the correct point), until sufficiently many have been saved. We are going to use these configurations to look at the actual distribution of spin orientations (rather than the broad ordering pattern, which we observed with ATOMEYE). To do so, we extract a distribution histogram using the command

```
spindist < spindist.in
```

which produces a new file, namely `mno_spins.out`. The numbers in this file correspond to a logarithmic probability of observing a spin pointing in a specific direction.

An intuitive method of viewing these distributions is a projection onto the surface of a sphere. There is a program provided that converts `mno_spins.out` into a sphere projection; we execute it with the command

```
spinplot < spinplot.in
```

This produces a picture in the `.ppm` format, which we convert using

```
convert mno_spins.ppm mno_spins.bmp
```

A typical distribution is shown in Fig. 3.7, which in this case shows that the spins are aligned parallel and antiparallel to an axis very close to [112].

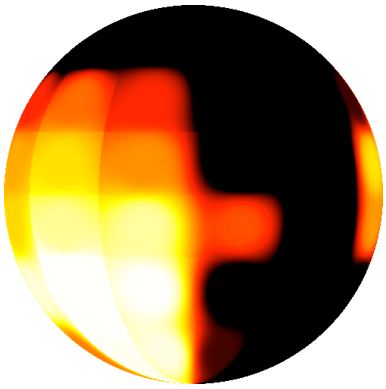


Figure 3.7: A typical spherical spin distribution (black = low probability; white = high probability). The view is taken looking down [100] with the [010] axis to the right hand side, and the [001] axis towards the top of the figure. In this particular case, alignment appears to be approximately parallel (and antiparallel) to [112].

We can use the same program to look at this distribution from arbitrary angles, and even produce an animated `.gif`. We will do this quickly here, by editing the file `spinplot.in`, changing the last entry from `F` to `T`. This part of the file is a flag that instructs the program to prepare a series of views that can be assembled to form an animation. Re-running the command

```
spinplot < spinplot.in
```

will now give a series of 24 files numbered `01_mno_spins.ppm` to `24_mno_spins.ppm`. Finally, we combine these into a single animated `.gif` with

```
convert -delay 20 -loop 0 *_mno_spins.ppm mno_spins.gif
```

The animation can then be viewed in a web browser.

Further notes

Other methods of analysing the RMC output include the generation of spin-spin correlation functions, which measure the the degree of correlation between spin orientations as a function of distance. Also we can extract some information about the spin excitations—*i.e.*, the magnons—from the sharpness of the peaks in these correlation functions.

3.3 Distance-window constraints

The distance-window constraint is an extension of the standard closest-approach constraint in RMC. In addition to specifying the closest two atom types can come you can also specify the furthest away two atom types are allowed to move apart. In this way you define windows or configuration space in which the atoms are allowed to move. When `RMCProfile` is first run, a neighbour list of all the atoms that fit within the distance windows is generated and then remains the same unless deleted. This means that the window sizes can be expanded or reduced but the atoms being constrained stays the same.

To use the distance windows constraint you do not need to change anything in the **`.dat`** file.[‡‡] Instead a **`.dw`** file must be present. This file contains two lines, the first specifies the closest approach distance for each atom pair as in the **`.dat`** file (*i*.e., ordered as the partials *eg* 1–1, 1–2, 1–3, 2–2, 2–3, 3–3 for a system of three atom types) and the second line has the furthest away distance in the same format. Once run RMCProfile will then write a **`.neigh`** file containing all the neighbour lists being used and a **`.neighlog`** file containing a history of what has been done. If you want to change the linkage or the configuration of atoms then the **`.neigh`** file should be deleted before running `RMCProfile` again.

[‡‡] This will shortly change, and it will become possible to include the data described here in the Version 6 **`.dat`** file.

Chapter 4

Input Files

4.1 `RMCProfile` version 6 input files

4.1.1 Introduction

Version 6 of `RMCProfile` introduces some new input file formats, which we recommend should be used in preference to the 'classic' formats (described later). The main data files, particularly the files containing the scattering data and the pair distribution function data, so far remain the same as in the 'classic' version, but the main **`.dat`** file and the configuration file have more transparent and more flexible formats (and hence are easier both to create and analyse). The histogram files also have new formats. Actually, by using the `data2config` program described in a later chapter on tools, users will never need to create the configuration file by hand.

4.1.2 The main .dat file

The main **`whatever.dat`** file, which is the file that controls the RMC simulation, is designed to be both flexible and readable. The format is simple and has just a couple of basic rules. These are illustrated in the following example:

```
%% RMC refinement of Ag3Co(CN)6

TITLE :: Ag3CoCN6_300K
MATERIAL :: Ag3[Co(CN)6]

NUMBER_DENSITY :: 0.052612 NUMBER/ANG^3
MAXIMUM_MOVES :: 0.0149 0.0200 0.0445 0.0411 ANG
R_SPACING :: 0.020 ANG
PRINT_PERIOD :: 10000 STEPS
TIME_LIMIT :: 0.0 MINUTES
SAVE_PERIOD :: 0.0 MINUTES
ATOMS :: Ag Co C N

NEUTRON_REAL_SPACE_DATA :: 1
```

```
  > FILENAME :: ag3cocn6_300k_tr.dat
  > START_POINT :: 1
  > END_POINT :: 2000
  > WEIGHT :: 0.02

FLAGS ::
  > SAVE_CONFIGURATIONS

END ::
```

You immediately notice the first main rule, namely the use of "%%", "::" and ">" symbols. These are used to define the type of data:

%% Anything after these characters is treated as a comment on that line.

:: Having this character on a line indicates that the line is a keyword; that is, it defines the type of data about to be provided. There are three main cases:

a) where the keyword stands alone, eg "END ::". You still need the :: symbol in the major keywords in this case in order for the parser to work; we will see an exception with the subordinate keywords below.

b) where the :: are followed by some data, for example "MATERIAL :: Quartz". The parser will know what to expect (whether characters or numbers);

c) where the major keyword is followed by a block of multi-line data. In this case, the subordinate keywords are designated by the preceding ">" character (described next).

> This denotes a subordinate keyword, and is tied to a major keyword. The order of the subordinate keywords is completely arbitrary within the block, but they have to follow immediately after the associated major keyword. In the examples shown on page 27, the "FLAGS ::" keyword line is followed by a set of lines for which each subordinate keyword is merely a directive (in this case the subordinate keyword doesn't need the :: symbol; hopefully this isn't too inconsistent for people, but at least an inconsistency is offset by the fact that there are very few rules). On the other hand, the "NEUTRON_REAL_SPACE_DATA :: 1" major keyword is followed by a set of subordinate keywords which in turn have associated data following the rule for the use of :: as above. Thus the line "> WEIGHT :: 0.02" is the way to input a value for the weighting parameter associated with this neutron data set.

The only other rule is the use of the "END ::" line. This tells the input parser to ignore anything below this line, which is useful when playing around.

Subject to these two rules, there are many ways in which the format is quite flexible. For example, the order of the "::" lines is completely arbitrary (apart from the constraints on the use of the "END ::" line), and the input is case independent. The input files also reduce the need to provide redundant data, whilst allowing you to do so if you wish. You are also allowed to include blank lines.

4.1.3 Major keywords

This is the first of two boring – but essential and comprehensive – sections of information associated with the input data file format. We recommend that you skim-read this now, and return to it after you have seen an example (given below). Note that the word 'requires' used in this list indicates the required parameters, not that this keyword is actually required.

`ATOMS ::`	Requires labels for the atoms in the configuration file. *This keyword is not required when Version 6 configuration files are being used.**
`AVERAGE_COORDINATION_CONSTRAINTS ::`	I can guess what this is supposed to do, but don't know enough about this to be able to say what is needed. *Default is to not use this constraint if this keyword is not provided.*
`BRAGG ::`	Introduces a block of subordinate keywords that provide information about the Bragg scattering data. *Do not include if you have no Bragg profile data.*
`CML ::`	Indicates that XML output in the language of the Chemical Markup Language (CML) is required (the default in the absence of this keyword is not to create a CML file). CML is described in a later section. This keyword can be followed by a block of subordinate keywords. Although optional, we recommend the use of this keyword because XML opens up a whole new world of possibilities for data analysis. *CML is discussed in section 5.1.1.*
`COMMENT ::`	Requires text that acts as any comment you want to make. An example might be that the specific run is part of a larger study. *This is used for metadata and is not compulsory.*
`DATA_FILE_VERSION ::`	Requires a number that indicates a version for the data format. Allowed values are `rmc6f`, `rmc3f` and `cfg`. *The use of `rmc6c` will be available soon, the speed of which will be determined by the time it takes for someone to first ask for this. There is no default version.*
`DATA_NOTE ::`	Requires text that allows the user to attach a one-line note about the data. For example, you could note that the data quality is good or of lower-quality just for testing purposes. *This is used for metadata and is not compulsory.*

`END ::`	Indicates that this is the last line to be read. It is useful when it enables you to move lines below this line rather than delete them should you possibly want to use them again.
`FIXED_COORDINATION_CONSTRAINTS ::`	I can guess what this is supposed to do, but don't know enough about this to be able to say what is needed. *Default is to not use this constraint if this keyword is not provided.*
`FLAGS ::`	Introduces a block of subordinate keywords that switch on/off various options.
`INPUT_CONFIGURATION_FORMAT ::`	Requires a number that indicates a version for the format of the input configuration file. Allowed values are `rmc6f`, `rmc3f` and `cfg`. *The use of `rmc6c` will be available soon, the speed of which will be determined by the time it takes for someone to first ask for this. There is no default version.*
`INVESTIGATOR ::`	Requires text concerning the name of the person running the simulation or experiment. This is designed for your future benefit. *This is used for metadata and is not compulsory.*
`KEYWORDS ::`	Requires text that act as keywords for your possible future benefit. *This is used for metadata and is not compulsory.*
`MAGNETISM ::`	Introduces a block of subordinate keywords that define the parameters associated with a magnetic simulation. *Default is to not use magnetic spins if this keyword is not provided.*
`MATERIAL ::`	Requires text to act as a title for the material being studied. *This is used for metadata and is not compulsory.*
`MAXIMUM_MOVES ::`	Requires values of the maximum move of each atom. *Currently the units are not interpreted; default units are Ångstroms.*[*]
`MINIMUM_DISTANCES ::`	Requires numbers of the minimum approach distances between pairs of atoms. *Default values are zero if this keyword is not provided. Currently the units are not interpreted; default units are Ångstroms.*[†]

`NEUTRON_COEFFICIENTS ::`

Requires list of neutron scattering coefficients (products of scattering lengths and number concentrations) for all atom pairs. The data are allowed to straddle several lines of the file. *Default units are 10^{-30} m^2. The default values are calculated by the code if this keyword is not given. It can be overwritten within the neutron data block of subordinate keywords (see section 4.1.6).*†

`NEUTRON_REAL_SPACE_DATA ::`

Introduces a block of data concerning a set of neutron-derived real-space data (*i.e.* a pair distribution function weighted by the neutron scattering lengths). Text can follow the `::` but will be ignored. This keyword must be followed by a block of subordinate keywords. *Do not provide if you have no neutron real space data.*

`NEUTRON_RECIPROCAL_SPACE_DATA ::`

Introduces a block of data concerning a set of neutron scattering data. Text can follow the `::` but will be ignored. This keyword must be followed by a block of subordinate keywords. *Do not provide if you have no neutron reciprocal space data.*

`NUMBER_DENSITY ::`

Requires the value of the number density should you want to over-ride the value in the configuration file. *Default units are $Å^{-3}$. Default value is the the value in the configuration file if this keyword is not provided.*

`PHASE ::`

Requires text to act as a description of the phase for the material being studied. For example, if you are studying quartz, you can define whether this is the low-temperature or high-temperature phase. *This is used for metadata and is not compulsory.*

`PRINT_PERIOD ::`

Requires number of steps between printing to the log file.

`POLYHEDRAL_RESTRAINT ::`

Requires the number label of the polyhedral constraint (see the main manual). *Default is to not use this constraint if this keyword is not provided.*

POTENTIALS ::	Introduces a block of data concerning the use of interatomic potentials. See *Default is to not use this constraint if this keyword is not provided. Use of this keyword means that use of the* POLYHEDRAL_RESTRAINT :: *keyword is ignored, because the two keywords provide access to similar functionality.* See page 10
PRESSURE ::	Requires text to denote the pressure of the material being studied. *This is used for metadata and is not compulsory. Units are not parsed as such.*
R_SPACING ::	Requires the value of the spacing used in the pair distribution functions. *Currently the units are not interpreted; default units are Ångstroms.*
RMC_NOTE ::	Requires text that allows the user to attach a one-line note about the specific simulation. For example, you could add a note to say that this is a test of one of several trial configurations. *This is used for metadata and is not compulsory.*
SAVE_CONFIGURATION_FORMAT ::	Requires a number that indicates a version for the format of the saved configuration file. *At the present time this is line is not required but is reserved for future use.*
SAVE_PERIOD ::	Requires time period (in minutes) between saving the configuration.
SORT_ATOMS ::	Gives a flag to sort the atoms in in the input configuration (for version 6 configuration files) into the order of atom type. Is useful when the generator program doesn't do this. *Default is not to sort*.
SWAP ::	Introduces a block of subordinate keywords that provide information about the atom swapping facility. *Default is to not use this swapping if this keyword is not provided.*
TEMPERATURE ::	Requires text to denote the temperature of the material being studied. *This is used for metadata and is not compulsory. Units are not parsed.*
TIME_LIMIT ::	Requires time limit (in minutes) for the job.
TITLE ::	Requires text to act as a title for the run. This is used for metadata and is not compulsory.

XRAY_RECIPROCAL_SPACE_DATA ::	Introduces a block of data concerning a set of neutron scattering data. Text can follow the :: but will be ignored. This keyword must be followed by a block of subordinate keywords. *Do not provide if you have no X-ray reciprocal space data.*

*Note that the order of the atoms must be the same as the order of atoms in the configuration file.

†Note that the order of the atom pairs is set by the order of atoms in the configuration file. This is illustrated by the example of 4 atoms labelled 1,2,3,4, with the order of pairs being 1–1, 1–2, 1–3, 1–4, 2–2, 2–3, 2–4, 3–3, 3–4, 4–4.

4.1.4 Subordinate keywords

Like the previous section, this is also boring but essential and comprehensive. Each block of subordinate keywords is given under the corresponding major keyword. Note that not all keywords require the :: characters; these are only used if they are to be followed by data on the line. Also note that some keywords (particularly those beginning with the characters "NO_" merely replicate the default behaviour, but can be useful as a record of the user's explicit intentions.

BRAGG ::

> BRAGG_SHAPE ::	Give a text string to denote the type of profile line shape to use. Options are GSAS1, GSAS2, XRAY1 and XRAY2, as defined by the GSAS manual. *The case doesn't matter.*
> DMIN ::	Give the minimum *d*-spacing value to be used in the analysis of the Bragg diffraction data. Users can use the QMAX subordinate keyword instead. The use of this subordinate keyword will replace the need for the **.hkl** file.
> HKL_RANGE ::	Give, in order, the maximum values of the Miller indices *h*, *k* and ℓ. *This is not yet implemented but will be available soon. The use of this subordinate keyword will replace the need for the **.hkl** file.*
> NO_RECALCULATE	Do not recalculate the list of reflections to use, but use the ones provided in the **hkls** file by a previous run. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> QMAX ::	Give the maximum value of *Q* to be used in the analysis of the Bragg diffraction data. Users can use the DMIN subordinate keyword instead. The use of this subordinate keyword will replace the need for the **.hkl** file.

> RECALCULATE	Recalculate the list of reflections to use. *The default setting is not to recalculate.*
> SUPERCELL ::	Three numbers to denote the supercell of the basic unit cell used to generate the atomic configuration. This information can also be in the .rm6f file instead of here.
> WEIGHT ::	Parameter to weight the contribution of the Bragg profile to the Monte Carlo simulation.

CML ::

> CCVIZ ::	Instructs the code to run the ccviz program that converts the xml file to an xhtml file. The parameter gives the path and name of the ccviz program. Examples are ./ccviz if the program is in the same location as you are running RMCProfile from, and /use/local/bin/ccviz if you are running ccviz from a specific stored version. No parameter is required if the ccviz command is already installed in a location picked up in your search path. For this to work it is required that mktemp is installed on the computing running the code. *Default is not to perform this action.*
> INPUT_CONFIGURATION	Instructs the code to read the configuration from a CML file. *Default is to read from a standard configuration file; not yet implemented.*
> REPORT_CONFIGURATION ::	Instructs the code to save the configuration in the main CML file for subsequent visualisation using the ccViz tool (described later). This is not recommended except for the use of small configurations. *Default is not to save the configuration in the main xml file; not yet implemented.*
> OUTPUT_CONFIGURATION	Instructs the code to write the configuration into a CML file. *Default is to write to a standard configuration file; not yet implemented.*

FLAGS ::

> CSSR	Instructs the program to write a copy of the final atomic configuration in cssr format, suitable for viewing in a molecular plotting program such as CrystalMaker. *Default is not to produce this file.*

> MOVEOUT	Instructs the RMC simulation to attempt to adjust the initial configuration to best match the minimum distances. *Default setting is not to do this.*
> NO_MOVEOUT	Instructs the simulation not to use the MOVEOUT option. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> NO_RESOLUTION_CONVOLUTION	Instructs the to not program use the convolution of the experimental neutron reciprocal space data with the experimental resolution function. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> NO_SAVE_CONFIGURATIONS	Instruct the program not to output a separate configuration file at each save point during the run. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> RESOLUTION_CONVOLUTION ::	Instruct the program to use the convolution of the experimental neutron reciprocal space data with the experimental resolution function. *The default is not to use the convolution.*
> SAVE_CONFIGURATIONS	Instruct the program to outputs a separate configuration file at each save point during the run. This is used in cases where it is necessary to collect multiple configuration files at equilibrium for analysis. The different files are numbered, e.g. as rmc6f_01, .rmc6f_02 etc. *The default is not to output these files.*

MAGNETISM ::

> FORM_FACTOR ::	Atom number followed by the seven coefficients for the expression for the magnetic form factor. *The user will need to provide either this keyword line or the* FORM_FACTOR_FILE :: *keyword line.*
> FORM_FACTOR_FILE ::	File containing the magnetic form factors. *The user will need to provide either this keyword line or the* FORM_FACTOR ::*keyword line.*
> MAGNETISM_FILE_STEM ::	Stem name for the files associated with magnetic spins.
> MAGNETIC_ATOMS ::	List of atoms that have an associated magnetic spin. *It is essential that these atoms are the first atoms in the configuration file, so that a list of non-sequential atoms would be invalid.*

> MAX_SPIN_MOVEMENT ::	Parameter that gives the maximum rotation of the magnetic spin in any RMC step.
> NO_VARY_SPIN_MOVE_RATE	Do not vary the rate of the spin move relative to the atomic displacements. *Default is not to allow the spin move rate to vary if this keyword or the* VARY_SPIN_MOVE_RATE *keyword is not given.*
> SPIN_MOVE_RATE ::	Rate at which the spins are moved relative to the atomic displacement moves. A value of 0.5 means that the number of attempted spin moves will equal the number of atom displacement moves; a value less than 0.5 means that there will be more atom displacement moves than attempted spin moves.
> VARY_SPIN_MOVE_RATE	Allow the rate of the spin move relative to the atomic displacements to be varied during the simulation. *Default is not to allow the spin move rate to vary if this keyword is not given.*

NEUTRON_REAL_SPACE_DATA ::

> DATA_TYPE ::	Gives the type of data; options are G(r), T(r), D(r) or G′(r), together with the word normalised or normalized for functions that are scaled by the neutron scattering coefficient $\sum c_i c_j b_i b_j$ to give limiting values of ± 1 depending on the function. *This is explained in more detail in Section X.*
> CONSTANT_OFFSET ::	If specified, it allows the user to provide an offset that applies to the PDF data before fitting. *Default value is 0.0 if this keyword is not provided.*
> END_POINT ::	End point of the data (an integer).
> FILENAME ::	Filename containing the data.
> FIT_TYPE ::	Gives the function that is fitted; options are G(r), T(r), D(r) or G′(r), together with the word normalised or normalized for functions that are scaled by the neutron scattering coefficient $\sum c_i c_j b_i b_j$ to give limiting values of ± 1 depending on the function. *This is explained in more detail in Section X.*
> FITTED_OFFSET	If specified, this instructs the program to fit the offset value provided for this set of data. *Default is not to fit.*
> GUDRUN	If specified, the data file was generated by the Gudrun data reduction suite, and contains errors on each point.

> NEUTRON_COEFFICIENTS ::	Requires list of neutron scattering coefficients (products of scattering lengths and number concentrations) for all atom pairs, particular to this data set only. The data are allowed to straddle several lines of the file. *Default units are 10^{-30} m^2. The default values are calculated by the code if this keyword is not given (see section 4.1.6).**
> NO_CONSTANT_OFFSET	Instructs the code that no offset is to be applied to the $T(r)$ data before fitting. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> NO_FITTED_OFFSET	Instructs the program not to fit the offset value provided for this set of data. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> START_POINT ::	Start point of the data (an integer).
> WEIGHT ::	Parameter that weights the data in the simulation.

NEUTRON_RECIPROCAL_SPACE_DATA ::

> DATA_TYPE ::	Gives the type of data; options are F(Q), QF(Q), i(Q), Qi(Q) or S(Q), together with the word normalised or normalized for functions that are scaled by the neutron scattering coefficient $\sum c_i c_j b_i b_j$ to give limiting values of ± 1 depending on the function. *This is explained in more detail in Section X.*
> CONSTANT_OFFSET ::	If specified, it allows the user to provide an offset that applies to the scattering data before fitting. *Default value is 0.0 if this keyword is not provided.*
> CONVOLVE ::	If specified, this will cause the program to convolve the reciprocal space data with a sinc function to model the effects of having a finite sample size. *Default is not to perform this operation, but in such a case the user will need to do this for himself before running RMCProfile using one of our tools. We strongly recommend using this subordinate keyword. In this case, the '::' is required because we plan to add a compulsory parameter.*
> END_POINT ::	End point of the data (an integer).
> FILENAME ::	Filename containing the data.

> FIT_TYPE ::	Gives the function that is fitted; options are F(Q), QF(Q), i(Q), Qi(Q) or S(Q), together with the word normalised or normalized for functions that are scaled by the neutron scattering coefficient $\sum c_i c_j b_i b_j$ to give limiting values of ± 1 depending on the function. *This is explained in more detail in Section X.*
> FITTED_OFFSET	If specified, this instructs the program to fit the offset value provided for this set of data. *Default is not to fit.*
> FITTED_SCALE	Instructs the program to fit a scale factor for the scattering data. *Default is not to fit.*
> GUDRUN	If specified, the data file was generated by the Gudrun data reduction suite, and contains errors on each point.
> NEUTRON_COEFFICIENTS ::	Requires list of neutron scattering coefficients (products of scattering lengths and number concentrations) for all atom pairs, particular to this data set only. The data are allowed to straddle several lines of the file. *Default units are 10^{-30} m^2. The default values are calculated by the code if this keyword is not given (see section 4.1.6).**
> NO_CONSTANT_OFFSET	Instructs the code that no offset is to be applied to the scattering data before fitting. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> NO_FITTED_OFFSET	Instructs the program not to fit any offset value on the scattering data. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> NO_FITTED_SCALE	Instructs the program not to fit a scale factor for the data. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> START_POINT ::	Start point of the data (provided as an integer).
> WEIGHT ::	Parameter that weights the data in the simulation.

POTENTIALS ::

`> ANGLE ::`

This gives the parameters in the bond-angle potential energy function. You give, in this order, the element symbol of the central atom label followed by the element symbols of the two bonded atoms, followed by the value of the force constant, K, in units of `eV` or `kJ/mol` (which need to be stated), the equilibrium bond angle θ_0 (in units of `deg`), followed by the two bond lengths in units of `Ang` (which must be stated).

`> ANGLE_SEARCH ::`

Give the range of values of angles around the equilibrium angle θ_0 that will be searched in the initial bond-search task. The value can be in a percentage (in which case give the units as `%` or `deg`).

`> PLOT ::`

This provides information for producing stereographic plots of bond orientation distribution functions. Give the following information using the words: `pixels = <value>`, with the default value being 400; `colour = <value>`,* where allowed colours are `charcoal` (*default*), `red`, `green`, `maroon` and `pink`; `zangle = <value>` and `zrotation = <value>` to give the angle that the plot is rotated away from and about the z axis, with the choice of units being `deg` or `rad`.

`> STRETCH ::`

This gives the parameters in the bond-stretch potential energy function. You give, in this order, the atom labels, followed by the value of the energy of the bond, D, in units of `eV` or `kJ/mol` (which need to be stated), and finally the equilibrium bond length, r_0, in units of `Ang` (which must be stated).†

`> STRETCH_SEARCH ::`

Give the range of values of distances around the equilibrium distance r_0 that will be searched in the initial bond-search task. The value can be in a percentage (in which case give the units as `%` or Å (in which case give the units as `Ang`).

`> TEMPERATURE ::`

Sample temperature in units of Kelvin or °C (in which case specify the units as `K` or `C` respectively). The default units are K. *Note that this value will be overridden by the value of sample temperature provided in the metadata input.*

`SWAP ::`

*The US spelling `color` will also work.

> NO_SWAP_TUNE	Don't tune the swapping probability. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> SWAP_ATOMS ::	List the numbers of atoms to be swapped.‡
> SWAP_PROBABILITY ::	Probability that a pair of atoms will attempt a swap. A value of 0.5 means that the number of attempted swap moves will equal the number of atom displacement moves; a value less than 0.5 means that there will be more atom displacement moves than attempted swap moves.
> SWAP_TUNE	Tune the swapping probability. *Default is not to tune.*

XRAY_RECIPROCAL_SPACE_DATA ::

> DATA_TYPE ::	Reserved for the type of data; the only option today is F(Q).
> CONSTANT_OFFSET ::	If specified, it allows the user to provide an offset that applies to the scattering data before fitting. *Default value is 0.0 if this keyword is not provided.*
> CONVOLVE ::	If specified, this will cause the program to convolve the reciprocal space data with a sinc function to model the effects of having a finite sample size. *Default is not to perform this operation, but in such a case the user will need to do this for himself before running RMCProfile using one of our tools. We strongly recommend using this subordinate keyword. In this case, the '::' is required because we plan to add a compulsory parameter.*
> END_POINT ::	End point of the data (an integer).
> FILENAME ::	Filename containing the data.
> FIT_TYPE ::	Reserved for the function that is fitted; the only option today is F(Q).
> FITTED_OFFSET	If specified, this instructs the program to fit the offset value provided for this set of data. *Default is not to fit.*
> FITTED_SCALE	Instructs the program to fit a scale factor for the data. *Default is not to fit.*
> NO_CONSTANT_OFFSET	Instructs the code that no offset is to be applied to the scattering data before fitting. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*

> NO_FITTED_OFFSET	Instructs the program not to fit any offset value on the scattering data. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> NO_FITTED_SCALE	Instructs the program not to fit a scale factor for the scattering data. *This is the default setting; although not necessary, this keyword can be useful as a record for the user.*
> START_POINT ::	Start point of the data (provided as an integer).
> WEIGHT ::	Parameter that weights the data in the simulation.

*Note that the order of the atom pairs is set by the order of atoms in the configuration file. This is illustrated by the example of 4 atoms labelled 1,2,3,4, with the order of pairs being 1–1, 1–2, 1–3, 1–4, 2–2, 2–3, 2–4, 3–3, 3–4, 4–4.
†Additional units can be made available on request, but internally the program works in terms of kJ/mol.
‡Note that the order of atoms is set by the configuration file.

4.1.5 Example files

This is an example of a simulation of $Ag_3Co(CN)_6$.

```
%% RMC refinement of Ag3Co(CN)6
%% Using new input format

TITLE :: Ag3CoCN6_300K
MATERIAL :: AG3[CO(CN)6]
PHASE :: TRIGONAL
TEMPERATURE :: 300 K
PRESSURE :: AMBIENT
DATA_NOTE :: Data collected August 2007
RMC_NOTE :: First attempt to fit data
KEYWORDS :: Ag3Co(CN)6, First attempt
COMMENT :: Refinement without peak broadening
INVESTIGATOR :: Andrew Goodwin and Matt Tucker

NUMBER_DENSITY :: 0.052612 Angstrom^(-3)
MAXIMUM_MOVES :: 0.0149 0.0200 0.0445 0.0411 Angstrom
R_SPACING :: 0.020 Angstrom
PRINTING_PERIOD :: 10000
TIME_LIMIT :: 0.0 MINUTES
SAVE_PERIOD :: 0.0 MINUTES
ATOMS :: Ag Co C N

NEUTRON_REAL_SPACE_DATA :: 1
```

```
  > FILENAME :: ag3cocn6_300k_tr.dat
  > DATA_TYPE :: T(r)
  > FIT_TYPE :: D(r)
  > START_POINT :: 1
  > END_POINT :: 2000
  > CONSTANT_OFFSET :: 0.0
  > WEIGHT :: 0.02
  > NO_FITTED_OFFSET

NEUTRON_RECIPROCAL_SPACE_DATA :: 1
  > FILENAME :: ag3cocn6_300k_sq.dat
  > DATA_TYPE :: S(Q)
  > FIT_TYPE :: Qi(Q)
  > START_POINT :: 1
  > END_POINT :: 2471
  > CONSTANT_OFFSET :: 0.0
  > WEIGHT :: 0.5
  > NO_FITTED_OFFSET
  > NO_FITTED_SCALE
  > CONVOLVE ::

CML ::
  > CCVIZ :: ./ccviz

FLAGS ::
  > NO_MOVEOUT
  > SAVE_CONFIGURATIONS
  > NO_RESOLUTION_CONVOLUTION
  > CSSR

BRAGG ::
  > BRAGG_SHAPE :: gsas2
  > RECALCULATE
  > SUPERCELL :: 6 4 6
  > WEIGHT :: 0.01

POTENTIALS ::
  > STRETCH :: C N 8.5 eV 1.15 Ang
  > STRETCH :: Co C 5.0 eV 2.0 Ang
  > STRETCH :: Ag N 6.0 eV 1.5 Ang
  > STRETCH_SEARCH :: 20%
  > ANGLE :: Ag N N 10 eV 180 deg 1.5 1.5 Ang
  > ANGLE :: Co C C 10 ev 90 deg 1.8 1.8 Ang
  > ANGLE_SEARCH :: 10 deg
  > TEMPERATURE :: 300 K
  > PLOT :: pixels = 400, colour = charcoal, zangle = 90.0, zrotation = 45.0 deg

END ::
```

This file is hopefully self-explanatory in light of the keyword descriptions above, but some points might help:

1. An extensive set of metadata is provided below the initial two comment lines. Although none of these lines are essential, they will provide the user with a good source of information when returning to the input or output files. These data are reproduced within the CML file.
2. The atom list here is for the example $Ag_3Co(CN)_6$, where the configuration file orders the atoms within an initial block of all the Ag atoms first, followed by all the Co, C and N atoms in blocks.
3. This example is using one file of neutron $T(r)$ data and one file of neutron $S(Q)$ data. The example is providing the request that a different representation of the data are used in the fitting.
4. No neutron coefficients are provided; these will be calculated from the internal table of atomic scattering lengths and the computed number concentrations.
5. The `CML ::` line without subordinate keywords leads to the production of a report CML file that does not contain the configuration, and the input and output configurations are in standard format.
6. The dataset includes a Bragg profile, fitted with the `gsas2` profile shape. The RMC configuration is specified to be a 6×4×6 supercell of the unit cell. The indexing of the Bragg reflections is to be recalculated.

4.1.6 Neutron coefficients

The neutron coefficients are defined as $f_{ij} = c_i c_j b_i b_j$, where i and j label two atom types, c_i is the number concentration of atom type i, and b_i is the coherent scattering length of atom type i. This is illustrated with the example of $Ag_3Co(CN)_6$. The atoms in the configuration are ordered as Ag, Co, C and N. The corresponding values of c are $^3/_{16}$, $^1/_{16}$, $^6/_{16}$ and $^6/_{16}$ respectively. The coherent scattering lengths for these elements are 5.922, 2.49, 6.646 and 9.36 fm respectively. Thus if the neutron coefficients are given in the input file, the example line would be

```
NEUTRON_COEFFICIENTS :: 1.2329 0.3859 5.5347 7.7948325 0.0302
   0.8661 1.219725 6.2113 17.495595 12.3201 fm^2
```

where clearly the first value, by way of example, is equal to $(5.922 \times 3/16)^2$, and the second value is equal to $5.922 \times 2.49 \times 3/16^2$. Note that the list of values is allowed to span more than one line. Also note that `RMCProfile` actually uses the default units for the scattering lengths of 10^{-14} m rather than fm (10^{-15} m), and the classic mode input file would therefore require the input values to be lower than this example by a factor of 100. The program can detect which of the two units is being used, although providing the units as "`fm^2`" or as "`10^-28 m^2`" is useful.

As noted above, if this keyword is not provided, `RMCProfile` will calculate all the values of the coefficients from default values of the neutron coherent scattering lengths and the calculated number concentrations. For many applications, this will be exactly what is required. But consider the case

where you have data from samples with different isotopes. Then each data set will require a different set of neutron coefficients. In this case, you are able to use the "> NEUTRON_COEFFICIENTS ::" subordinate keyword within any block of data keywords where required (you might not bother for the data sets using the natural isotopes).

4.1.7 Data normalisations

One unfortunate aspect of the world of total scattering is that different communities have developed different terminologies and definitions. Our colleague David Keen did us a great service by publishing a comparative review of these, and we recommend that you keep a copy of this paper beside you. RMCProfile will accept and work with several of the conventions and definitions used for total scattering and PDF functions. Here we review the key equations that RMCProfile supports.

4.1.7a The pair distribution function

We start with the basic definition of the partial PDF, namely that the number of atoms of type j lying within a shell of radius r and thickness dr centred on an atom of type i is given as $4\pi r^2 \rho_j g_{ij}(r)\mathrm{d}r$, where ρ_j is the number of atoms of type j per unit volume. We have the following limiting cases:

$$g_{ij}(r \sim 0) = 0 \quad ; \quad g_{ij}(r \to \infty) = 1 \tag{4.1}$$

It will be convenient to introduce the number concentration, c_j, where $\rho_j = c_j\rho$, where ρ is the total number of atoms per unit volume. Clearly $\sum_j c_j = 1$.

We define an overall PDF by merging all the partial PDFs with appropriate weighting consistent with the concentration and neutron scattering power as

$$G(r) = \sum_{i,j} c_i c_j b_i b_j \left(g_{ij}(r) - 1\right) \tag{4.2}$$

where b_j is the scattering factor of atom type j. The limiting values are

$$G(r \sim 0) = -\left(\sum_j c_j b_j\right)^2 \quad ; \quad G(r \to \infty) = 0 \tag{4.3}$$

The equation for $G(r)$ can be expressed in a form with a constant offset:[†]

$$G'(r) = \sum_{i,j} c_i c_j b_i b_j g_{ij}(r) = G(r) + \left(\sum_j c_j b_j\right)^2 \tag{4.4}$$

which has limiting values

[†]Note that this differs from the definition by Keen by a normalisation factor; we will discuss scaling by normalisation factors later.

$$G'(r \sim 0) = 0 \quad ; \quad G'(r \to \infty) = \left(\sum_j c_j b_j \right)^2 \tag{4.5}$$

4.1.7b Neutron scattering function

The scattering function and the PDF $G(r)$ are related by[‡]

$$i(Q) = F(Q) = \rho \int_0^\infty 4\pi r^2 G(r) \frac{\sin Qr}{Qr} \mathrm{d}r \tag{4.6}$$

and

$$G(r) = \frac{1}{2\pi\rho} \int_0^\infty 4\pi Q^2 i(Q) \frac{\sin Qr}{Qr} \mathrm{d}Q \tag{4.7}$$

For dense materials the scattering function $F(Q)$ has limiting values

$$i(Q \to 0) = -\sum_j c_j b_j^2 \quad ; \quad i(Q \to \infty) = 0 \tag{4.8}$$

The scattering factor can be written in a form with a constant offset, to give[§]

$$S(Q) = i(Q) + \left(\sum_j c_j b_j \right)^2 \tag{4.9}$$

which has limiting values

$$S(Q \to 0) = -\sum_j c_j b_j^2 + \left(\sum_j c_j b_j \right)^2 \quad ; \quad S(Q \to \infty) = \left(\sum_j c_j b_j \right)^2 \tag{4.10}$$

4.1.7c Alternative forms of the pair distribution function

It is common to use two other definitions of the PDF functions:

$$D(r) = \frac{2}{\pi} \int_0^\infty Q i(Q) \sin Qr \, \mathrm{d}Q = 4\pi r \rho G(r) \tag{4.11}$$

[‡] According to the definitions discussed by Keen, the functions commonly written as $i(Q)$ and $F(Q)$ are synonymous. The authors of this manuscript tend to use both functions.

[§] Keen defines this in normalised form, which we call $S_{\mathrm{norm}}(Q)$ in equation 4.19 below.

and

$$T(r) = D(r) + 4\pi r\rho \left(\sum_j c_j b_j \right)^2 = 4\pi r\rho G'(r) \tag{4.12}$$

The link between equations 4.11 and 4.7 should be clear. The interesting point is that $D(r)$ is the transform of the function $Qi(Q)$, so often these two functions are considered together. In these two functions the data are scaled by r and Q to give increased weighting to the high-r and high-Q parts of the data respectively. This can be useful when viewing effects beyond the first peaks, not least because it is often these effects that are particularly interesting.

The new PDF functions have limiting values

$$D(r \to 0) = -4\pi r\rho \left(\sum_j c_j b_j \right)^2 \quad ; \quad D(r \to \infty) = 0 \tag{4.13}$$

and

$$T(r \sim 0) = 0 \quad ; \quad T(r \to \infty) = 4\pi r\rho \left(\sum_j c_j b_j \right)^2 \tag{4.14}$$

4.1.7d Normalised functions

RMCProfile also allows the use of normalised functions. For the PDFs we can defined these as

$$G_{\text{norm}}(r) = G(r) / \left(\sum_j c_j b_j \right)^2 \quad ; \quad G'_{\text{norm}}(r) = G'(r) / \left(\sum_j c_j b_j \right)^2 \tag{4.15}$$

$$D_{\text{norm}}(r) = D(r) / 4\pi\rho \left(\sum_j c_j b_j \right)^2 \quad ; \quad T_{\text{norm}}(r) = T(r) / 4\pi\rho \left(\sum_j c_j b_j \right)^2 \tag{4.16}$$

with limiting values

$$G_{\text{norm}}(r \sim 0) = -1 \quad ; \quad G_{\text{norm}}(r \to \infty) = 0 \quad ; \quad G'_{\text{norm}}(r \sim 0) = 0 \quad ; \quad G'_{\text{norm}}(r \to \infty) = +1 \tag{4.17}$$

and

$$D_{\text{norm}}(r \sim 0) = -r \quad ; \quad D_{\text{norm}}(r \to \infty) = 0 \quad ; \quad T_{\text{norm}}(r \sim 0) = 0 \quad ; \quad T_{\text{norm}}(r \to \infty) = +r \tag{4.18}$$

Normalised scattering functions can be defined as

$$i_{\mathrm{norm}}(Q) = F_{\mathrm{norm}}(Q) = i(Q) / \sum_j c_j b_j^2 \quad ; \quad S_{\mathrm{norm}}(Q) = S(Q) / \left(\sum_j c_j b_j \right)^2 \tag{4.19}$$

with limiting cases[¶]

$$\begin{gathered} i_{\mathrm{norm}}(Q \to 0) = -1 \quad ; \quad i_{\mathrm{norm}}(Q \to \infty) = 0 \\ S_{\mathrm{norm}}(Q \sim 0) = 1 - \sum_j c_j b_j^2 / \left(\sum_j c_j b_j \right)^2 \quad ; \quad S_{\mathrm{norm}}(Q \to \infty) = +1 \end{gathered} \tag{4.20}$$

4.1.7e Implementation within `RMCProfile`

`RMCProfile` allows all the above functions to be used to describe both the form of the input data and the functions to be fitted (and hence used in the output). In addition, `RMCProfile` also allows $Qi(Q)$ (and synonymously $QF(Q)$) to be used as well. Normalised functions are indicated by the use of the word '`normalised`'.[‖]

For clarification, the data or fit types for the various functions to be used in the input data file are

[¶]Note that equation 21 in Keen, which is equivalent to second line of this equation, is missing the value of 1 in the term for $S_{\mathrm{norm}}(Q \sim 0)$

[‖]The spelling '`normalized`' works just as well.

Function	Input text	Equation from Keen	Equation here
$i(Q)$	`i(Q)`	25	4.6
$F(Q)$	`F(Q)`	11	4.6
$S(Q)$	`S(Q)`	–	4.9
$Qi(Q)$	`Qi(Q)`	–	–
$QF(Q)$	`QF(Q)`	–	–
$i_{\text{norm}}(Q)$	`i(Q) normalised`	–	4.19
$F_{\text{norm}}(Q)$	`F(Q) normalised`	–	4.19
$S_{\text{norm}}(Q)$	`S(Q) normalised`	19	4.19
$G(r)$	`G(r)`	10	4.2,4.7
$G'(r)$	`G'(r)`	–	4.4
$D(r)$	`D(r)`	26	4.11
$T(r)$	`T(r)`	27	4.12
$G_{\text{norm}}(r)$	`G(r) normalised`	–	4.15
$G'_{\text{norm}}(r)$	`G'(r) normalised`	16	4.15
$D_{\text{norm}}(r)$	`D(r) normalised`	–	4.16
$T_{\text{norm}}(r)$	`T(r) normalised`	–	4.16

4.2 RMC Version 6 format configuration files

Version 6 of `RMCProfile` not only brings a new data format, but it also brings several new formats for configuration files – described generically as “Version 6 format files” – which reflect a different approach to managing configuration files. These are somewhat richer than the ‘classic’ Version 3 format files (described later).

4.2.1 Version 6f format configuration file

This format is best described with reference to an example file:

```
(Version 6f format configuration file)
Metadata tile:      NaCl in a small box
Metadata owner:     Martin Dove
Metadata date:      15-02-2009
Metadata material:  NaCl
Metadata comment:   This is a test configuration
Metadata source:    Solid State Physics text book
Number of moves generated:           0
Number of moves tried:               0
```

```
Number of moves accepted:            0
Number of prior configuration saves: 0
Number of atoms:                     64
Supercell dimensions:                2 2 2
Number density (Ang^-3):                 0.043656
Cell (Ang/deg):    11.360000   11.360000   11.360000   90.000000   90.000000   90.000000
Lattice vectors (Ang):
   11.360000   -0.000000   -0.000000
    0.000000   11.360000   -0.000000
    0.000000    0.000000   11.360000
Atoms:
     1   Na   [1]    0.000000    0.000000    0.000000     1   0   0   0
     2   Na   [1]    0.250000    0.250000    0.000000     2   0   0   0
     3   Na   [1]    0.250000    0.000000    0.250000     3   0   0   0
     4   Na   [1]    0.000000    0.250000    0.250000     4   0   0   0
     5   Na   [1]    0.000000    0.000000    0.500000     1   0   0   1
     6   Na   [1]    0.250000    0.250000    0.500000     2   0   0   1
...
    58   Cl    0.500000    0.750000    0.000000     6   1   1   0
    59   Cl    0.500000    0.500000    0.250000     7   1   1   0
    60   Cl    0.750000    0.750000    0.250000     8   1   1   0
    61   Cl    0.750000    0.500000    0.500000     5   1   1   1
    62   Cl    0.500000    0.750000    0.500000     6   1   1   1
    63   Cl    0.500000    0.500000    0.750000     7   1   1   1
    64   Cl    0.750000    0.750000    0.750000     8   1   1   1
```

A number of points can be seen from this example:

1. The first line must contain the phrase '`Version 6f`' (case sensitive).
2. All other keywords are case insensitive.
3. The metadata lines are not essential, but given that the `data2config` program (see section on utilities) makes their use fairly easy, you are recommended to use them. Unless you keep very good notebooks and are equally good at keeping your file system in good order, then it is folly not to avail yourself of the option to use metadata. Remember the motto, "*Don't cry later* / *if you don't use metadata*".
4. And if you wisely use the metadata lines, the colons are essential.
5. The various lines beginning '`Number of`' are not necessary, but will be automatically generated by `RMCProfile` in subsequent writes of the configuration file.
6. The '`Number density`' line is also not required, but will be written by `RMCProfile` in subsequent writes of the configuration file.
7. The '`Cell (Ang/deg):`' line is essential if you don't include the '`Lattice vectors (Ang)`' data. It defines the size and shape of the configuration in terms of the conventional crystallographic lattice parameters, and will be used to create the lattice vectors. If you include both the lattice parameters and lattice vectors, the '`Lattice vectors (Ang)`' data will take precedence. Both will be generated in subsequent writes of the configuration file.
8. The '`Lattice vectors (Ang):`' line and the following three lines containing the vectors describing the shape and dimensions of the configuration. These lines are not required if

you include the 'Cell (Ang/deg):' line, but will take precedence if both are included. If these data are not include, they will be generated by RMCProfile in subsequent writes of the configuration file.

9. The 'Atoms:' line is required. It must precede the block of lines containing the information about each atom.

10. The 'supercell' line is not required, but is generated by data2config and its inclusion means that it is not required within the main **.dat** file.

11. Each atom line follows the format with the following data:

    (a) An *optional* number can be provided before the atom symbol to label the atom within the configuration.

    (b) The atomic element symbol (*compulsory*). This requirement is different from 'Version 3' classic configuration files, which do not contain any information about the atom types. The advantage of containing this information is that it makes the file self-contained; you can run analysis programs without needing to obtain this information from other sources. This element must be one of the real elements. You are allowed to use D for deuterium, and you can use Va for a vacant site.

    (c) An *optional* numerical atom label (integer) immediately after the atomic symbol, given within square brackets. This integer can have any value.

    (d) Three fractional coordinate values. Unlike Version 3 configuration files, the assumption is that the origin of the box has coordinate $0, 0, 0$, and that x, y, and z values range from $0 \rightarrow 1$.

    (e) A single *optional* integer that is appropriate if the configuration is a supercell of a fundamental crystallographic unit cell. In this case, the integer will correspond to the atom in the crystallographic unit cell that this atom corresponds to.

    (f) A set of three *optional* integers that are appropriate if the configuration is a supercell of a fundamental crystallographic unit cell. In this case, these integers denote the position of the origin of the unit cell containing this atom relative to the origin of the configuration, with the origin cell denoted by integers 0 0 0, and for a supercell having dimensions $N_x \times N_y \times N_z$ the three integers range in value from $0 \rightarrow (N_x - 1)$, $0 \rightarrow (N_y - 1)$ and $0 \rightarrow (N_z - 1)$ respectively. Note that if the optional integer giving the label of the atom in the origin cell is included (previous input quantity), these three integers *must* follow that integer.

12. There are as many atoms lines as there are atoms. There is no termination line. If the number of atoms is given in the 'Number of atoms' line, the lines will be counted as they are read, but if the 'Number of atoms' line is not given RMCProfile will presume that the set of atom lines will be the last lines in the configuration file.

It will be assumed that Version 6f configuration files will have names with a .rmc6f extension.

We have provided the data2config tool to allow users to generate this file from a number of different types of crystal structure files, such as those produced by the GSAS Rietveld refinement program, or standard CIF files. data2config also allows you to convert classic format configuration files into the Version 6f format.

4.2.2 Version 6c format configuration file

This file format is very similar to that of Version 6f, but uses Cartesian coordinates rather than fractional coordinates. An example will illustrate the point:

```
(Version 6c format configuration file)
Metadata tile:      NaCl in a small box
Metadata owner:     Martin Dove
Metadata date:      15-02-2009
Metadata material:  NaCl
Metadata comment:   This is a test configuration
Metadata source:    Solid State Physics text book
Number of moves generated:           0
Number of moves tried:               0
Number of moves accepted:            0
Number of prior configuration saves: 0
Number of atoms:                     64
Number density (Ang^-3):                 0.043656
Cell (Ang/deg):    11.360000   11.360000   11.360000   90.000000   90.000000   90.000000
Lattice vectors (Ang):
   11.360000   -0.000000   -0.000000
    0.000000   11.360000   -0.000000
    0.000000    0.000000   11.360000
Atoms:
     1   Na    0.000000    0.000000    0.000000     1   0   0   0
     2   Na    2.840000    2.840000   -0.000000     2   0   0   0
     3   Na    2.840000   -0.000000    2.840000     3   0   0   0
     4   Na    0.000000    2.840000    2.840000     4   0   0   0
     5   Na    0.000000    0.000000    5.680000     1   0   0   1
     6   Na    2.840000    2.840000    5.680000     2   0   0   1
...
    58   Cl    5.680000    8.520000   -0.000000     6   1   1   0
    59   Cl    5.680000    5.680000    2.840000     7   1   1   0
    60   Cl    8.520000    8.520000    2.840000     8   1   1   0
    61   Cl    8.520000    5.680000    5.680000     5   1   1   1
    62   Cl    5.680000    8.520000    5.680000     6   1   1   1
    63   Cl    5.680000    5.680000    8.520000     7   1   1   1
    64   Cl    8.520000    8.520000    8.520000     8   1   1   1
```

The explanation of the Version 4f format should be sufficient to describe most of the features of this file format, with only a few exceptions:

1. The first line must contain the phrase `Version 6c`.
2. Here the `Lattice vectors (Ang):` set of lines is *required*, and the `Cell (Ang/deg):` line is both *optional* and not used by `RMCProfile` (but will be generated in successive writes of the configuration file).
3. It stands to reason that the coordinates in the atom lines are now in Cartesian form with units of Å.

It will be assumed that Version 6c configuration files will have names with a `.rmc6c` extension. As above, this file can be produced using `data2config`.

4.2.3 Version 6f format histogram file

Normally users do not create the histogram files themselves, but instead they are generated by successive runs of `RMCProfile`. However, it is useful to understand the format of this file to enable it to be used for subsequent data analysis.

This file format is best illustrated with an example:

```
RMCProfile v6f intermediate (histogram) file
Metadata owner:     Martin Dove
Metadata date:      17-02-2009
Metadata material:  Ag3Co(CN)6
Metadata source:    Generated from RMC runs starting with our own GSAS structure
Number of moves generated:           92422
Number of moves tried:               92343
Number of moves accepted:            144815
Number of prior configuration saves: 0
Number of atoms:                     4608
Number density (Ang^-3):                 0.052612
Cell (Ang/deg):    42.150540   48.671252   42.692412   90.000000   90.000000   90.000000
Lattice vectors (Ang):
   42.150540    0.000000    0.000000
    0.000000   48.671252    0.000000
    0.000000    0.000000   42.692412
Atoms (fractional coordinates):
     1   Ag    0.082764    0.999551    0.080983     1   0   0   0
     2   Ag    0.084347    0.998030    0.248759     2   0   0   0
     3   Ag    0.083515    0.998656    0.415444     3   0   0   0
     4   Ag    0.081875    0.000937    0.583213     4   0   0   0
...
  4605   N     0.938087    0.920807    0.452244   13   5   5   5
  4606   N     0.940870    0.914326    0.618485   14   5   5   5
  4607   N     0.951886    0.912453    0.789664   15   5   5   5
  4608   N     0.940857    0.927571    0.956517   16   5   5   5
Number of points in pair distribution functions: 1053
Step size in pair distribution function:   2.00000000000000004E-002

 step    AgAg    AgN     N,N
           1          0          0          0
           2          0          0          0
           3          0          0          0
...
         172         85         71         47
         173         88         68         45
         174         84         95         34
         175        117         70         42
         176        112         90         47
         177         93         78         55
         178         96         74         56
         179         70         71         58
         180         77         73         55
...
```

In many ways this looks very similar to the configuration file format. Extensions to the configuration file are mostly the information concerned with the additional data for the histograms from which the pair distribution functions are calculated.

Since this file is generated automatically by `RMCProfile`, there is no need to describe some of the information as optional. Note that if you generate this file from a previous version 3 `.his` file (eg using `data2config`) there will be no means to generate the 4 integer indices at the end of each atom line, and thus these will not be given (merely replacing each integer by zero will not achieve very much; the default is not to include them).

This file usually has a filename with extension `.his6f`.

4.2.4 Version 6c format histogram file

This file is virtually identical to the `.his6f` format, but the atomic coordinates are given in Cartesian format (units Å). The files have a name with extension `.his6c`.

4.3 Experimental data files

The main experimental data, whether the scattering or pair distribution function data, all have the same basic formats. Moreover, unlike other files you are free to chose any names for these files, because the names are provided within the main **`.dat`** file. However, to avoid confusion you might like to use an extension that indicates the type of file, such as `.gr` for a $G(r)$ PDF file, or `.sq` for the neutron scattering $S(Q)$ file.

4.3.1 Traditional RMC files

These files have a very simple format. The first line contains the number of points in the file, and the second line is a title line. These are followed by one line per data point, each containing first the x values and the second the y values of the data. For example, a scattering data file[**] might have the form:

```
1975
KCN 250K with offset 0.01
      0.52       -0.4730834
      0.54       -0.4659694
      0.56       -0.4678035
      0.58       -0.4740225
      0.60        -0.474752
      0.62       -0.4666348
      0.64       -0.4686959
      0.66       -0.4683059
      0.68       -0.4674031
      0.70       -0.4667906
      0.72       -0.4660346
...
     39.96     2.608086e-05
```

[**] Chances are that the PDF file will begin with too many zero values for it to be an interesting example.

```
     39.98      -0.002873718
     40.00      -0.004977553
etc...
```

4.3.2 Data files generated by GUDRUN

GUDRUN is the ISIS data correction and transformation toolkit for total scattering data. The files generated by GUDRUN can now directly be read into `RMCProfile` rather than converted into the traditional format. The subordinate keyword `GUDRUN` needs to be provided with the `NEUTRON`$_R$*EAL*$_S$*PACE*$_D$*ATA* *and*

```
# GEM12234.mdcs01
# KCN in CCR at 20K I                                    12Wx34H chop:19302,0off,1939
#     1  2502
#    10
# MGT QH DAK MTD  /ach
# 15-FEB-2003 02:14:23
# spec.bad
# groups_def.dat
#     6
#    -1
#  0.1700000E+02
#  0.0000000E+00  0.1456720E+01  0.0000000E+00
#  0.0000000E+00  0.5096614E+02  0.0000000E+00
#  0.0000000E+00  0.9269717E+02  0.0000000E+00
   0.3000000E-01  0.0000000E+00  0.0000000E+00
   0.5000000E-01  0.0000000E+00  0.0000000E+00
...
   0.8900000E+00 -0.4750513E+00  0.1049418E-02
   0.9100000E+00 -0.4789928E+00  0.1090569E-02
   0.9300000E+00 -0.4766961E+00  0.1046322E-02
   0.9500000E+00 -0.4731495E+00  0.1022877E-02
   0.9700000E+00 -0.4702018E+00  0.9821923E-03
etc...
```

The lines beginning with the `#` are treated as comments, but the third comment line is important because it contains the start and end point numbers that are read.

4.4 Bragg scattering

The ability of `RMCProfile` to handle separately the information from Bragg scattering enables the RMC simulation to reproduce the spatial distribution of atom positions. The program uses a Rietveld-like approach in that it fits the Bragg peaks in the diffraction profile using background and

lineshape functions obtained from the GSAS program[††] and adjusts the configuration to match the intensities of the Bragg peaks.

4.4.1 The essential data files

The information that is required to control the Bragg fitting was described in the section on the Version 6 data file above. We remark here that the configuration is a supercell of the crystal unit cell, of relative size $N_x \times N_y \times N_z$. The "> SUPERCELL ::" line within the "BRAGG ::" keyword block contains the three integers N_x, N_y, N_z. When we now consider the Miller indices h, k, ℓ, these refer to the fundamental unit cell rather that the configuration supercell, and it is the supercell integers than enable RMCProfile to know how to set up the Bragg peaks.

4.4.1a The optional .hkl file

RMCProfile will calculate the Bragg profile for all Bragg peaks for d-spacings down to a minimum value and for h, k, ℓ values up to a maximum value. These limiting values are provided in a file with extension **.hkl**, which will have the form:

```
0.8
-10 10
-8 8
-12 12
```

The first line gives the minimum value of the d-spacing, and the remaining three lines give the limiting values of h, k and ℓ respectively.

This file is not necessary if either of the > DMIN :: or > QMIN :: subordinate keywords are supplied within the keyword block under the BRAGG :: keyword.

RMCProfile will compute the Bragg intensities for all reflections, including those that are supposed to be systematically absent due to symmetry. Whilst in a crystal structure refinement program this might be considered to be a waste of time, in RMCProfile this is useful because it acts to ensure that the data drive the configuration into the appropriate long-range symmetry rather than any symmetry being imposed from the outset.

4.4.2 The .bragg, .back and .inst files

4.4.2a .bragg file

The Bragg scattering data is contained within a file with extension .bragg. It has the format appropriate for time-of-flight diffractometers, an example being:

```
1818 2  6.7798  274.9949
Sample data file
4.983635   4.38E-02
```

[††] which clearly we are anticipating the user will have done.

```
4.987625    4.33E-02
4.991615    4.27E-02
4.995605    4.23E-02
4.999600    4.26E-02
5.003600    4.30E-02
5.007606    4.35E-02
...
21.275999    2.34E-03
21.292999    1.12E-02
21.310051    2.57E-02
21.327099    2.65E-02
```

The first line contains four numbers. The first gives the number of data points. The second gives the number of the detector bank, which is used to extract the correct instrument profile from the **.instr** file described below. The third parameter is the scale factor as calculated by GSAS, and the fourth parameter is the volume of the unit cell.[‡‡]

4.4.2b .back file

As in Rietveld refinement, the backgound is modelled using Chebychev polynomails. This file contains the number of polynomials on the first line, followed by the coefficients one per line.

4.4.2c .inst file

This file is extracted from the GSAS files. It has the form

```
4
1
1456.13      -0.33      -3.02    17.98
0.000000E+00    0.381440E+00    0.322500E+02    0.532200EE+02
0.000000E+00    0.204539E+03    0.000000E+00
0.000000E+00    0.126026E+02    0.000000E+00    0.000000E+00    0.000000E+00
...
```

The first line contains the number of detector banks in the instrument. Then follows a block of five lines. The first gives the bank number, and the remaining four lines contain data that you as a user need not worry about. This block is repeated for each data bank.

4.4.3 Generated files

RMCProfile automatically generates two files, namely a **.amp** file and one called **hkls**. These are used for subsequent restarts and the user need not be concerned with the details.

[‡‡] In principle you might expect not to have to give this information since it is easily computed. In a future version we will remove the need for this parameter.

4.5 RMCProfile version 3 ('classic') files

In this section we describe the input files required when running RMCProfile in the version 3 classic mode. Unless you like to work harder than you need to, we can't find any obvious reason to recommend this approach in preference to the new version 6 approach described previously.[§§] If you find this daunting (which one of the authors does), then skip past this to the next chapter.[¶¶]

4.5.1 The .dat file

This is main control file, from here everything you want RMCProfile to do is defined. An example is given below, the text after the ! explains what that line is for.

```
SF6_at_190K
0.0685951           ! number density
4.0 1.2 1.8         ! cut offs
0.05 0.1            ! maximum move
0.020               ! r spacing
.false.             ! whether to use moveout option
.false.             ! collect configurations
1000                ! step for printing
2400 60             ! Time limit, step for saving
1 3 0               ! No. of g(r), neutron, X-ray
sf6_190k.gr
1 1500
0.0
.00165 .03942
      .23486
0.05
.false.
sf6190kbank1conv29p42rmc.dat
1 3000
0.
.00165 .03942
      .23486
0.01
.false.
.false.
sf6190kbank2conv29p42rmc.dat
```

[§§] You may wonder what happened to versions 4 and 5. The answer is that as the code developed through versions 4 and 5, the input file format remained unchanged.

[¶¶] Said author recommends you jump there immediately.

```
1 3000
0.
.00165 .03942
      .23486
0.01
.false.
.false.
sf6190kbank3conv29p42rmc.dat
1 3000
0.
.00165 .03942
      .23486
0.01
.false.
.false.
0                       ! no. of coordination constraints
0                       ! no. of average coordination constraints
.true.                  ! whether to use a polyhedra restraint
4                       ! which restraint to use
.true.                  ! whether to use bragg intensities
gsas2                   ! Which gsas profile function to use
10 10 10                ! Number of unit cell in each direction
S                       ! Symbol for first atom (use spaces to !)
F                       ! Symbol for second atom(use spaces to !)
y                       ! whether to calculate from cfg at start
0.001                     ! weighting factor
.false.                  ! Convolute with profile function
.false.                  ! Calculate the magnetic scattering
0.0 .false.              ! Probability of swap moves, auto-tune
                           percentage if greater than 0.0
1 2                      ! Atom types to swap during swap moves
```

The first line is a title, then the number density of the configuration is given. The next line contains the closest-approach cut-offs. These are not necessarily the atomic radii, since the crystal structure will define the nearest neighbours etc. The closest-approach constraint is a powerful constraint and caution should be used to ensure the cut-offs are not set too high, as this may prevent the atoms moving enough to produce a true representation of the structure.

On the next line the *r* spacing to be used for histogram calculation is supplied, this number should be the same as the *r* spacing of any *G*(*r*) data supplied.

Then a value of `.true.` or `.false.` is supplied to define whether to use the move out option. This should be avoided for crystalline systems where the starting configuration should not have atoms that violate the closest approach cut-offs. It has been left in to be consistent with the forerunner `RMCA` program and for use with amorphous systems.

The next logical defines whether a configuration should be written at each print/summary cycle. This can be used to collect many configurations once a suitable equilibrium has been reached. It can also be used to collect configuration up to equilibrium for movie making purposes.

The number of generated moves between each print/summary statements is then supplied. This is not the same as accepted moves, since these depend on the data weighting and other constraints.

Then next two numbers are the total run time and time between saves (in minutes). The last number determines how often RMCProfile saves a copy of the files, this will determine how much time will be lost if the program or the computer crashes and how often the results are updated. Setting this number too low will put a heavy work load on the machines hard disk, typically for long runs 60 minutes is a good value to use.

On the following line the type and number of data sets to be used are defined. Currently three types of data are supported: neutron $G(r)$, neutron total scattering $i(Q)$ and x-ray total scattering. The file format is described in the RMCA manual and for the time being only a single x-ray data set is allowed and up to five neutron $G(r)$ and $i(Q)$ data sets.

The information on the following lines in the **.dat** files depends on the data specified in the previous line. For each $G(r)$ data set the following information is specified: the filename of the data, the first and last data points to be used, a constant to be subtracted from the data, the Faber-Zimmer partial weighting factors (*i.e.*, the components of $\sum(c_i b_i)^2$), the sigma weighting of the data and a logical determining whether RMCProfile should rescale the data to give the best fit.

The same information is required for each neutron or x-ray $i(Q)$ data set, but also an additional logical is required to define whether RMCProfile should calculate an offset of the data to give the best fit.

With all data types these rescaling and offset options should not normally be used, only when the fit seems to have a scale problem should this option be used and then only so the program can assess the rescaling required. The data should be corrected before continuing with RMCProfile refinements.

The next two lines in the .dat file define the number of coordination constraints and number of average coordination constraints. These tend not to be used for crystalline system so are set to zero in the example given above and will not be discussed here. If you would like to use them please refer to the RMCA manual.

The next two lines define the polyhedral restraint option, first a logical tells RMCProfile whether to use a restraint and only if this is set to .true. then the following line contains the number of the restraint type to use. The allowed values and addition files required are described in the restraints section below.

The next seven lines in the example **.dat** file define the Bragg profile fitting options. The first line is a logical indicating whether to fit the Bragg profile. If this were set to .false. then the follow six lines would be omitted. Here it is set to .true. so it is followed by the code for GSAS profile function to be used. At the moment only time-of-flight functions 2 and 3 are supported (so gsas2 or gsas3). More profile functions will be adding in the next release of RMCProfile. The following line defines the number of unit cells in each direction within the RMCProfile super cell. This is explained in more detail in the following .cfg section. Then the chemical symbol for each atomic species in the configuration is listed one per line. The next line contains either y or n to tell RMCProfile whether to re-calculate the Bragg scattering from the .cfg each time the program is started. This should always be set to y, except when debugging runs and the time taken for the program to initialise is an issue. Finally the sigma weight for the Bragg profile is supplied.

The next line in the .dat file is logical to determine if RMCProfile will convolute the neutron $F(Q)$ data with its profile function. This is only useful for time of flight neutron data and at the moment is at the 'experimental' stage. So for now it should be set to .false. and will not be described further here. It will hopefully be a functioning option in the next release of the program and if you are interested in this option please email me.

The next line tells RMCProfile if it should calculate the magnetic scattering if the system being studied has a magnetic component. In the example here the system is not magnetic so it is set to .false.; the various options for magnetic systems will be discussed in the magnetic scattering section below once it has been completed.

The last two lines in the **.dat** file tell RMCProfile the probability of swap moves and, if not zero, whether to auto tune this probability. If the probability is not zero then the next line contains the two atom types that are to be swapped. In the example .dat file above the first and second atom types have been specified, however since the probability is set to zero this line is not required but given by way of example. Typically RMCProfile just translates a single atom during each Monte Carlo step, however for systems with site disorder, such as cation ordering or vacancies this is not completely suitable. So a swap move enables RMCProfile to switch the position of one atom with another during a Monte Carlo step. The balance between how many swap moves and then how many translations are required to relax the surrounding structure is non-trivial. So with RMCProfile there are two approaches, the first is trial and error by just setting the swap probability (probably the most common option), the second option is to set the autotuning to true. This will then try to alter the swap move probability to maintain an appropriate level of accepted moves.

4.5.2 The .cfg file

The .cfg file contains the information about the supercell of atoms RMCProfile will use to fit the data. This file together with the .dat file are the minimum required files, RMCProfile can be run without data but not without a configuration of atoms to move. The format of this file is the same as with RMCA and consists of some header information about the number of atoms, the configuration dimensions, the atom types and then a list of the atomic positions (in fractional coordinates from -1 to 1). The order of the information and indeed the overall format is fixed so the easiest way to produce this file is to use the crystal program supplied with RMCProfile. A description of how to use this program is given later, if however this does not meet your needs then use it to produce a template .cfg file for your system and then modify this as you need to.

If you got this far, you are probably pretty committed to using this rather antiquated way of doing things in preference to the friendly version 6 approach described earlier. You have our respect at least, if not our sympathy. However, it is worth giving you a warning at this point. In order to preserve backwards compatibility with the 'classic' mode there are all sorts of things in the RMCProfile code that only exist because of this, and one day we might want to tidy up and get rid of this stuff. Thus there might be a day when we have to give up support for the 'classic' mode, so perhaps now is the time to switch versions. Why not use the dat2config utility to convert your configuration files into the new v6 format whilst waiting for your 'classic' jobs to run?

4.5.3 The .poly, .sf and .fs files

These three files are required since the flag to use a polyhedral restraint is set to true in the **.dat** file and then restraint "4" has been specified. At the moment there are 14 different types of polyhedral

restraints available with different combinations of polyhedra. For the full list of options and the files required please see the polyhedral restraints section below. The **.sf** and **.fs** files contain a list of neighbours for sulphur and fluorine respectively, the neighbours file required, their file extension and how they are produced is also described in the polyhedral restraints section below.

4.6 v6.4 RMCProfile for classic and RMCA users: how to upgrade

Let's get straight to the point; if you are tempted to stick with the 'classic' version – or even worse, with RMCA – we have one word of advice: *don't*. And to make the switch easy, we have provided some tools to help you upgrade.

There are basically two things you have to change. First is the **.dat** file, and second is the format of the configuration file. Let's deal with the configuration file first. We have created the data2config tool (subsection 5.2.1 to make this simple. You start with a classic **.cfg** file, and run the command:

```
data2config -rmc6f <whatever>.cfg
```

You answer a few questions, and out pops the new format configuration file. The key questions concern the types of atoms you have in the configuration, and when asked about the supercell type "1 1 1" if you don't want to make the configuration larger. data2config can do much more than this for you – for example, it will help you generate configurations from crystallographic CIF files. Executing the command without arguments will give you a list of options. For more information, turn to subsection 5.2.1.

To convert the **.dat** file to the new v6 format, we provide the rmc326 command (subsection 5.2.2). You simply need to run the command

```
rmc326 -o <newfile>.dat <oldfile>.dat
```

There are a few more options, that you can find from running the command with no arguments or looking at subsection 5.2.2.

Hopefully this makes switching to v6.4 as easy as possible. In case you are still not sure, here are some reasons to upgrade sooner than later:

- v6 files are designed to be easy to use, whereas the classic and RMCA formats were designed with more-or-less little regard for the user.
- v6 files contain information (metadata) that will better enable you to keep track of your data.
- v6 formats make available all of the new features being introduced into RMCProfile, which you can't access from the classic format files. These include different data formats, intramolecular potentials, use of XML output with the tools these makeavailable to you, and any new developments we will be adding.
- We cannot guarantee that support for classic format files will continue for ever; indeed when it becomes too much effort we will make a decision to stop supporting the classic format.

- And if you are still using `RMCA` you need to be aware that there is now no support for this code. Indeed, the source code for the later versions has disappeared.
- You know it makes sense.

Hopefully a combination of providing upgrade tools and berating the reluctant will be sufficient to persuade you to switch before you run any more RMC jobs. To switch will only take you a few minutes, time that you will more than recoup once your job has run.

4.7 Polyhedral restraints

As mentioned above, there are 14 different polyhedral restraints available in RMCProfile. The large number is mainly due to the fact that each restraint is for a specific type of system rather than being a generic restraint definable by the user; this has been done for simplicity of coding, use of legacy code and for speed. The 14 options are listed below and a brief description of each follows.

1. SiO_2
2. $SrTiO_3$
3. CD_4
4. SF_6
5. $AlPO_4$
6. PZT
7. ZrP_2O_7
8. ZrW_2O_8
9. Na_3PO_4
10. $NaNO_3$
11. KCN
12. AgCN
13. $Zn(CN)_2$
14. C_4F_8

The names are derived from the first system to use that restraint, but the restraint is suitable for any similar system. For example, the SiO_2 restraint defines a set of linked tetrahedra so could be used for any system where the first two atom types form a network of linked tetrahedra.

The weight for each restraint is set in the `.poly` file. The weightings are simple multipliers, so a larger number means a heavier weighting. These weightings should be chosen carefully such that

the data weighting is always higher, if this is not done then the restraint will become a constraint and the data will be ignored and the resulting configuration biased.

Please note none of the restraints support swapping moves of the atoms linked by the restraint at the moment. This feature will be added in a later release of the program if required.

4.7.1 The SiO_2 restraint

This restraint defines a network of linked tetrahedra formed from the first two atoms in the configuration, as the name suggests it was first used for phases of silica. To use this restraint option "1" needs to be specified in the `.dat` file and `.poly`, `.sio` and `.osi` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal Si–O bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the tetrahedral angle restraint (*i.e.*, 300). The `.sio` file contains a list of oxygen neighbours around each silicon atom and the `.osi` file contains a list of the silicon neighbours around each oxygen atom, both of these files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.2 The $SrTiO_3$ restraint

This restraint defines a network of linked octahedra formed from the first two atoms in the configuration, as the name suggests it was first used for studying strontium titanate. Since the octahedra are formed from the first two atoms in the configuration it must be set up to actually represent TiO_3Sr. In this way the same constraint can be used to study $Ca_xSr_{1-x}TiO_3$ or any similar system. To use this restraint option "2" needs to be specified in the `.dat` file and `.poly`, `.tio` and `.oti` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal Ti–O bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the tetrahedral angle restraint (*i.e.*, 300). The `.tio` file contains a list of oxygen neighbours around each titanium atom and the `.oti` file contains a list of the titanium neighbours around each oxygen atom, both of these files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.3 The CD_4 restraint

This restraint defines a configuration of unlinked tetrahedra formed from the first two atoms in the configuration. As the name suggests it was first used for the study of deuterated methane. To use this restraint option "3" needs to be specified in the `.dat` file and `.poly`, `.cd` and `.dc` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal C–D bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the tetrahedral angle restraint (*i.e.*, 300). The `.cd` file contains a list of deuterium neighbours around each carbon atom and the `.dc` file contains a list of the carbon neighbours around each deuterium atom, both of these files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.4 The SF_6 restraint

This restraint defines a configuration of unlinked octahedra formed from the first two atoms in the configuration, as the name suggests it was first used for studying the molecular crystal SF_6. To use this restraint option "4" needs to be specified in the `.dat` file and `.poly`, `.sf` and `.fs` files supplied.

The `.poly` file contains a title line which is ignored and the next line must contain the ideal S–F bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the octahedral angle restraint (*i.e.*, 300). The `.sf` file contains a list of fluorine neighbours around each sulphur atom and the `.fs` file contains a list of the sulphur neighbours around each fluorine atom, both of these files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.5 The $AlPO_4$ restraint

This restraint defines a network of linked tetrahedra formed from the first three atoms in the configuration, as the name suggests it was first used for phases of aluminium phosphate. This differs from the SiO_2 restraint in that the network consists of tetrahedra of two sizes: one for the AlO_4 and one for PO_4. To use this restraint option "5" needs to be specified in the `.dat` file and `.poly`, `.alo`, `.oal`, `.po` and `.op` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal Al–O bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the tetrahedral angle restraint (*i.e.*, 300). The following line must contain the ideal P–O bond distance to be used, the weighting for this bond restraint and then the weighting for the tetrahedral angle restraint. The `.alo` file contains a list of oxygen neighbours around each aluminium atom and the `.oal` file contains a list of the aluminium neighbours around each oxygen atom. Similarly, the `.po` file contains a list of oxygen neighbours around each phosphorous atom and the `.op` file contains a list of the phosphorous neighbours around each oxygen atom. All of these neighbour files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.6 The PZT restraint

This restraint defines a network of linked octahedra formed from the second, third and fourth atoms in the configuration, as the name suggests it was first used for phases of lead zirconium titanate ($PbZr_xTi_{1-x}O_3$). This differs from the $SrTiO_3$ restraint in that the network consists of octahedra of two sizes: one for the ZrO_6 and one for TiO_6. To use this restraint option "6" needs to be specified in the `.dat` file and `.poly`, `.zro`, `.ozr`, `.tio` and `.oti` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal Zr–O bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the octahedral angle restraint (*i.e.*, 300). The following line must contain the ideal Ti–O bond distance to be used, the weighting for this bond restraint and then the weighting for the octahedral angle restraint. The `.zro` file contains a list of oxygen neighbours around each zirconium atom and the `.ozr` file contains a list of the zirconium neighbours around each oxygen atom. Similarly, the `.tio` file contains a list of oxygen neighbours around each titanium atom and the `.oti` file contains a list of the titanium neighbours around each oxygen atom. All of these neighbour files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.7 The ZrP_2O_7 restraint

This restraint defines a network of linked octahedra and tetrahedra formed from the first three atoms in the configuration, as the name suggests it was first used for phases of ZrP_2O_7. To use this restraint option "7" needs to be specified in the `.dat` file and `.poly`, `.zro`, `.ozr`, `.po` and `.op` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal Zr–O bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and

then the weighting for the octahedral angle restraint (*i.e.*, 300). The following line must contain the ideal P–O bond distance to be used, the weighting for this bond restraint and then the weighting for the tetrahedral angle restraint. The `.zro` file contains a list of oxygen neighbours around each zirconium atom and the `.ozr` file contains a list of the zirconium neighbours around each oxygen atom. Similarly, the `.po` file contains a list of oxygen neighbours around each phosphorous atom and the `.op` file contains a list of the phosphorous neighbours around each oxygen atom. All of these neighbour files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.8 The ZrW_2O_8 restraint

This restraint defines a network of linked octahedra and tetrahedra formed from the first three atoms in the configuration, as the name suggests it was first used for phases of ZrW_2O_8. This restraint differs from the ZrP_2O_7 version since one of the tetrahedral oxygens is non-bridging. To use this restraint option "8" needs to be specified in the `.dat` file and `.poly`, `.zro`, `.ozr`, `.wo` and `.ow` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal Zr–O bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the octahedral angle restraint (*i.e.*, 300). The following line must contain the ideal W–O bond distance to be used, the weighting for this bond restraint and then the weighting for the tetrahedral angle restraint. The `.zro` file contains a list of oxygen neighbours around each zirconium atom and the `.ozr` file contains a list of the zirconium neighbours around each oxygen atom. Similarly, the `.wo` file contains a list of oxygen neighbours around each tungsten atom and the `.ow` file contains a list of the tungsten neighbours around each oxygen atom. All of these neighbour files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.9 The Na_3PO_4 restraint

This restraint defines a configuration of unlinked tetrahedra formed from the second and third atoms in the configuration, as the name suggests it was first used for studying the molecular crystal Na_3PO_4. To use this restraint option "9" needs to be specified in the `.dat` file and `.poly`, `.po` and `.op` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal P–O bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the octahedral angle restraint (*i.e.*, 300). The `.po` file contains a list of oxygen neighbours around each phosphorous atom and the `.op` file contains a list of the phosphorous neighbours around each oxygen atom, both of these files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.10 The $NaNO_3$ restraint

This restraint defines a configuration of unlinked triangular molecules formed from the second and third atom types in the configuration, as the name suggests it was first used for studying the molecular crystal $NaNO_3$. To use this restraint option "10" needs to be specified in the .dat file and .poly, .no and .on files supplied. The .poly file contains a title line which is ignored and the next line must contain the ideal N–O bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the angle restraint (*i.e.*, 300). The `.no` file contains a list of oxygen neighbours around each nitrogen atom and the `.on` file contains a list of the nitrogen neighbours around

each oxygen atom, both of these files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.11 The KCN restraint

This restraint defines a configuration of unlinked binary molecules formed from the second and third atom types in the configuration, as the name suggests it was first used for studying the molecular crystal potassium cyanide. To use this restraint option "11" needs to be specified in the `.dat` file and `.poly`, `.cn` and `.nc` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal C–N bond distance to be used and the weighting for this bond restraint (*i.e.*, 100). The `.cn` file contains a list of nitrogren neighbours around each carbon atom and the `.nc` file contains a list of the carbon neighbours around each nitrogen atom, both of these files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.12 The AgCN restraint

This restraint defines a configuration of atoms linked in a chain structured formed from the first three atom types in the configuration, as the name suggests it was first used for studying the molecular crystal AgCN. To use this restraint option "12" needs to be specified in the `.dat` file and `.poly`, `.agc`, `.cag`, `.cn`, `.nc`, `.nag`, and `.agn` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal Ag–C bond distance to be used and the weighting for this bond restraint (*i.e.*, 100). The following line must contain the ideal C–N bond distance to be used and the weighting for this bond restraint. The next line must contain the ideal N–Ag bond distance to be used and the weighting for this bond restraint. The `.agc` file contains a list of carbon neighbours around each silver atom and the `.cag` file contains a list of the silver neighbours around each carbon atom. The `.cn` file contains a list of nitrogren neighbours around each carbon atom and the `.nc` file contains a list of the carbon neighbours around each nitrogen atom. The `.nag` file contains a list of silver neighbours around each nitrogen atom and the `.agn` file contains a list of the nitrogren neighbours around each silver atom. All of these neighbour files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

4.7.13 The $Zn(CN)_2$ restraint

This restraint defines a network of linked tetrahedra formed from the first three atoms in the configuration, as the name suggests it was first used for phases of zinc cyanide. This differs from the SiO_2 restraint in that the network consist of tetrahedra link by two bridge atoms, in this case carbon and nitrogen with zinc at the centre of the tetrahedra. To use this restraint option "13" needs to be specified in the `.dat` file and `.poly`, `.zncn`, `.czn`, `.nzn`, `.cn` and `.nc` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal Zn–C bond distance to be used, the weighting for this bond restraint (*i.e.*, 100) and then the weighting for the tetrahedral angle restraint (*i.e.*, 300). The following line must contain the ideal C–N bond distance to be used and the weighting for this bond restraint. The next line must contain the ideal N–Zn bond distance to be used and the weighting for this bond restraint. The `.zncn` file contains a list of carbon and nitrogren neighbours around each zinc atom, the `.czn` file contains a list of the zinc neighbours around each carbon atom and the `.nzn` file contains a list of the zinc neighbours around each nitrogen atom. The `.cn` file contains a list of nitrogren neighbours around each carbon atom and the `.nc` file contains a list of the carbon neighbours around each nitrogen atom.

Most of these neighbour files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2, however the `.zncn` file should be produced using the `neighbour_list_two` program since both carbon and nitrogen atoms need to be specified as neighbours.

4.7.14 The C_4F_8 restraint

This restraint defines a configuration of unlinked C_4F_8 molecules formed from the first two atom types in the configuration. This is a very specific restraint that defines a molecule of four carbon atoms arranged in a square with two fluorine atoms attached to each corner. Only bond distance restraints are applied to hold the molecule together. To use this restraint option "14" needs to be specified in the `.dat` file and `.poly`, `.cc`, `.cf` and `.fc` files supplied. The `.poly` file contains a title line which is ignored and the next line must contain the ideal C–F bond distance to be used and the weighting for the bond restraints (*i.e.*, 100) and then the next line contains the ideal C–C bond distance to be used. The `.cc` file contains a list of carbon neighbours around each carbon atom, the `.cf` file contains a list of the fluorine neighbours around each carbon atom and the `.fc` file contains a list of the carbon neighbours around each fluorine atom. All of these neighbour files should be generated using the `neighbour_list` program supplied with RMCProfile and described in section 5.2.

Chapter 5

Tools

5.1 Version 6 analysis tools

5.1.1 CML files

RMCProfile produces a number of standard output files in text form. With the use of the "CML ::" keyword RMCProfile will also generate a file containing a rich set of output data in the Chemical Markup Language (CML) format. CML is an XML language designed to represent chemical data, and adapted for atomistic simulation. CML files are not designed to be read by humans, but are readable because each item of data is enclosed within descriptive tags.

Sample extracts of an RMCProfile CML file are shown in Figures 5.1 and 5.2. You can see that each item of date is properly described within the CML file. For example, in Figure5.1 you can see that there is a block of metadata items contained within the <metadataList> tags, with general format:

```
<metadataList>
  <metadata name=... content=... />
</metadataList>
```

Examples in Figure 5.1 include the metadata items that are provided in the input file (such as the item given in the "MATERIAL ::" keyword line), plus other metadata items generated by the code itself (such as the code name and version items). In Figure 5.1 you can also see a set of input parameters that are nested within the <parameterList> and <parameter> tags with the following format:

```
<parameterList>
  <parameter dictRef=... name=...>
    <item ...>...</item>
  </parameter>
</parameterList>
```

where in Figure 5.1 the quantity specified by `<item>` is either `<scalar>` or `<array>` depending on data type, and there is specific information contained within the tags. The examples in Figure 5.1 include parameters that are set by the user input, such as the list of atom types, and others that are deduced from the input file, such as the number of data types or number of atomic species. Note that each parameter item contains a `dictRef` (a reference to a dictionary item describing the parameter) and a name. The data items are accompanied by a description of the units.

Figure 5.2 shows actual data generated. These are contained within `<module>` tags, which are defined for different tasks or roles. The examples in Figure 5.2 illustrate two specific roles, one called `role="step"` for step by step output (the serial number gives the actual step number), and another called `role="plottable"` which contains final graphs (*e.g.* partial pair distribution functions, and comparison of data and calculated functions) for plotting with the `ccViz` tool (described below).

There are several applications that stem from using XML data representation. In general terms, XML files can easily be transformed to other representations, including XHTML. This is exploited in the `ccViz` tool that will be described below, and which is bundled with `RMCProfile`. A second application called `summon` enables extraction of information from an XML file without having to scroll through the file. These are supplied with the `RMCProfile` package; both require Python to be available on the computer being used, and both work from the command line.

CML within `RMCProfile` has been enabled through Toby White's FoX package.[1] We note here that the incorporation of CML within `RMCProfile` is still new, and any changes or additions can easily be added on request. We anticipate shortly adding the opportunity to include CML representation of the atomic configurations.

The CML files generated by `RMCProfile` have the stem name with the `.xml` extension. We now describe two tools that exploit the use of CML to the user's advantage.

5.1.2 The ccViz tool

`ccViz` is a python program (written by Toby White[2]) that will produce and information-centric report in XTHML format (*i.e.* a report that can be read using a web browser) from CML files that follow a standard document model. `ccViz` is run as a standard shell command, with the name of the CML/XML file as the argument, as per this example:

```
$ ccViz filename.xml
```

It transforms the XML file to an XHTML file, with the same root name, that can be viewed with a modern web browser – we recommend Safari[3] for the mac and windows platforms, or Firefox[4] for mac, windows and linux platforms. Older browsers, or modern browsers that do not conform to

[1] FoX reference

[2] http://uszla.me.uk/space/about

[3] http://www.apple.com/safari

[4] http://www.mozilla.com/firefox

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="http://www.eminerals.org/XSLT/display.xsl" type="text/xsl"?>
  <metadataList name="Metadata">
    <metadata name="dc:creator" content="rmcprofile"/>
    <metadata name="Code name" content="rmcprofile"/>
    <metadata name="Code version" content="6.02"/>
    <metadata name="Material" content="AG3[CO(CN)6]"/>
    <metadata name="Sample temperature" content="300 K"/>
    <metadata name="Data note" content="Data collected August 2007"/>
  </metadataList>
  <parameterList>
    <parameter dictRef="rmcprofile:number_of_species" name="Number of species">
      <scalar dataType="xsd:integer" units="cmlUnits:Ang">4</scalar>
    </parameter>
    <parameter dictRef="rmcprofile:atoms" name="Atom types">
      <array size="4" delimiter=" " dataType="xsd:string" units="cmlUnits:dimensionless">Ag Co C  N </array>
    </parameter>
    <parameter dictRef="rmcprofile:numbers_of_species" name="Numbers of species">
      <array size="4" dataType="xsd:integer" units="cmlUnits:countable">864 288 1728 1728</array>
    </parameter>
    <parameter dictRef="rmcprofile:number_density" name="Number density">
      <scalar dataType="fpx:real" units="cmlUnits:Ang">5.261200000000e-2</scalar>
    </parameter>
    <parameter dictRef="rmcprofile:ngr" name="Number of T(r) functions">
      <scalar dataType="xsd:integer" units="cmlUnits:countable">2</scalar>
    </parameter>
    <parameter dictRef="rmcprofile:nsq" name="Number of neutron S(Q) functions">
      <scalar dataType="xsd:integer" units="cmlUnits:countable">2</scalar>
    </parameter>
  </parameterList>
```

Figure 5.1: Extracts from the first part of an `RMCProfile` XML output file, showing the metadata and parameter list portions.

```
  <module serial="144815" dictRef="rmcprofile:summary" role="step">
    <propertyList>
      <property dictRef="rmcprofile:swapstried" title="Atom swaps tried">
        <scalar dataType="xsd:integer" units="cmlUnits:countable">92412</scalar>
      </property>
      <property dictRef="rmcprofile:chi2_dof" title="Chi^2/dof">
        <scalar dataType="fpx:real" units="cmlUnits:dimensionless">4.355859668604e1</scalar>
      </property>
      <property dictRef="rmcprofile:chisq" title="Chi^2/points1">
        <scalar dataType="fpx:real" units="cmlUnits:dimensionless">5.256872352697e1</scalar>
      </property>
      <property dictRef="rmcprofile:chisq" title="Chi^2/points2">
        <scalar dataType="fpx:real" units="cmlUnits:dimensionless">4.424679046320e1</scalar>
      </property>
      <property dictRef="rmcprofile:chisq" title="Chi^2/points3">
        <scalar dataType="fpx:real" units="cmlUnits:dimensionless">9.728364387699e-3</scalar>
      </property>
    </propertyList>
  </module>
  <module title="Partial PDF functions" role="plottable">
    <property dictRef="rmcprofile:differenceBraggProfile">
      <array size="1766" dataType="fpx:real" units="units:dimensionless">4.251451910081e-3 2.871865471529e-3 ...
      ... 2.211384967670e-3 3.466896392559e-3</array>
    </property>
  </module>
  <metadata name="dc:contributor" content="FoX-4.0.1 (http://www.uszla.me.uk/FoX)"/>
</cml>
```

Figure 5.2: Extracts from the latter part of an `RMCProfile` XML output file, showing the stepwise data and final data.

HTML/XHTML v5 standards, will not give the desired rendering.

The XHTML file contains a readable report of the RMC simulation, including graphs to report the progress of the simulation and plots of fitted data. It contains a comprehensive set of the metadata, input parameters, step-wise data and final parameters. If the CML file is directed to include the configuration, the report also contains an interactive three-dimensional view of the configuration rendered using the Jmol tool, but this may not be a good option for large files. In short, the resultant web report comprehensively extracts the essential information in a form that is easy to read.

Sample screen shots are shown in Figures 5.3 and 5.4. In the first shot, the blue bars act as buttons to open up the report to show subsidiary data. Graphs can then be displayed from the `Show` buttons.

In addition to providing an information-centric view of the data obtained from the course of the RMC simulation, `ccViz` provides a dictionary with definitions for all the terms. By placing the mouse pointer over any term, the relevant dictionary reference is displayed at the top of the right hand frame. The idea is that RMC users should be able to share the reports with colleagues who are not expects – our view is that output files should not need to be read hand-in-hand with the manual to make sense of them!

`ccViz` is supplied with the `RMCProfile` v6 package. It does not need to be installed; instead, it should simply be placed in a location from which it can be treated as a shell command. `ccViz` can also be downloaded from `http://uszla.me.uk/space/software/ccViz` and compiled from source.

To conclude this section, we note that `ccViz` can be used with a variety of different atomic-scale simulation outputs; examples include `CASTEP`, `SIESTA` and `DL_POLY3`.

5.1.3 The summon tool

`summon` is another python program that can be used to extract desired data from a CML file without having to browse through it with a text editor (or do the same through one of the standard output text files). It can be compared with `grep` for text files. One key application of this is to extract information from a group of files when many jobs have been run as part of a single study. It replaces the standard approaches of writing bespoke applications of scripts to parse a file and extract the desired information, or the even more depressing approach of cutting-and-pasting between the text view of the output file and the spreadsheet. Not only will summon save a lot of effort, it will also prevent mistakes that can occur when using bespoke programming or cut-and-paste approaches.

The `summon` package needs to be installed on your computer. It wraps up a number of required packages and installs summon as a shell command using the make install command.

To explain how `summon` works, we should start with an example of using summon as a shell command with simple parameters (note that "`$ summon --help`" or "`$ summon -h`" will provide a summary of the required parameters):

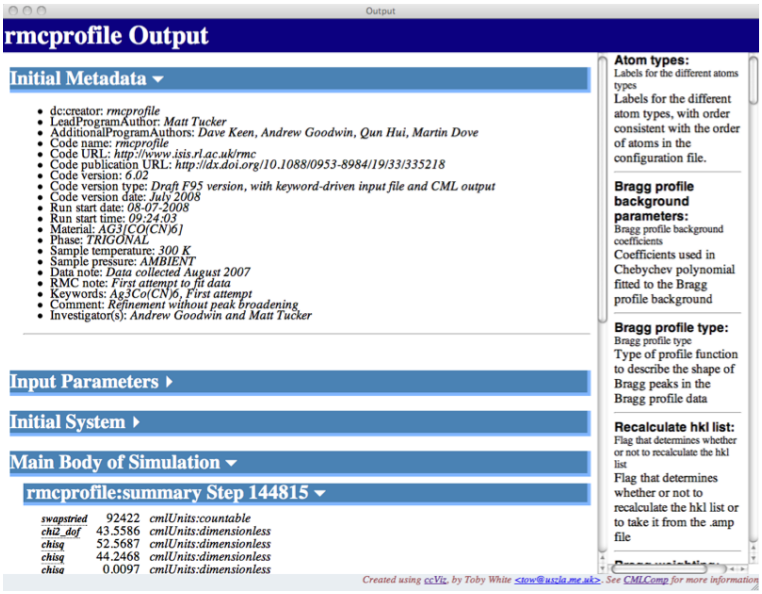


Figure 5.3: Top of the XHTML file generated by `ccViz`, showing the initial metadata in expanded form, the link to the input parameters in collapsed view, and the output form a single step in the simulation. By moving the mouse cursor over any parameter will bring up the corresponding dictionary definition.

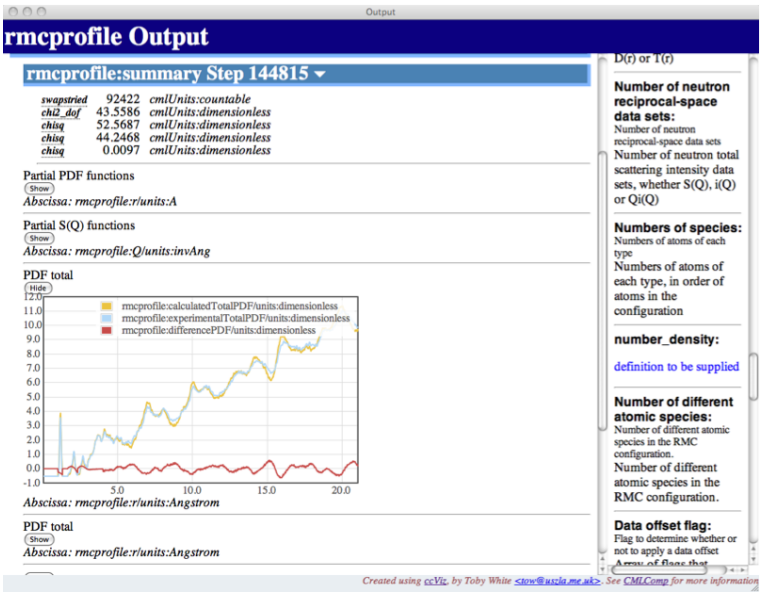


Figure 5.4: Portion of the XHTML file generated by `ccViz`, showing the data plots. Each plot can be opened or closed using the `Show`/`Hide` buttons. Selecting a portion of a plot will give a magnified view.

```
$  summon -t number_of_species -c rmcprofile.config ag3cocn6.xml
```

The specific task from this command is to extract the number of species from the XML file (here the example is called **`ag3cocn6.xml`**) in the user's directory. The first parameter, *i.e.* the one following "`-t`", is a keyword that denotes the property to be extracted. The keywords are defined as XPath

expressions in the file called **rmcprofile.config**, which in the example command is found in the same directory as the XML file. In case you want to know (but you don't actually need to know), this specific case appears in **rmcprofile.config** as the set of lines

```
[number_of_species]
type: scalar
xpath: //cml:parameterList
cml:parameter[@dictRef="rmcprofile:number_of_species"]
```

The last two lines are actually on the same line in the file; in several of the following examples we have to allow single lines to flow over two. You can see how the XPath instruction relates to the parameter "number_of_species" in the example XML file shown in Figure 5.2. The keyword is the word in square brackets in the first line.

The result of this example invocation of the summon command is

```
number_of_species
4
```

This example is probably not, *per se*, a particularly useful use of the summon command, because the user is likely to already knows how many atomic species there are, but it is illustrative. A more useful command of this kind might be to ask about the atom list and the number of atoms of each type. This would have the form:

```
$ summon -t atoms -t numbers_of_species -c rmcprofile.config
                            ag3cocn6_300k.xml
```

Another usage might be:

```
$ summon -t dSpacing -t diff_Bragg -c rmcprofile.config *.xml
```

which will extract the arrays of *d*-spacing and the difference Bragg profile (*i.e.* difference between experiment and fitted function) as comma-separated arrays, with each array enclosed within square brackets. Use of "-o filename" will cause the output to be written to a file, which is probably more sensible for this particular case than listing to the screen. In case it helps, we give another similar example:

```
$ summon -t r -t partial2 -t partial3 -c rmcprofile.config
                            -o output.txt *.xml
```

This example extracts two partial PDF functions in order of distance followed by the two PDF functions, placing the output in the file called **output.txt**.

It should be noted that arrays are written as comma-separated values contained within square brackets. Thus, for example, the command

```
$  summon -t numbers_of_species -c rmcprofile.config
                              ag3cocn6_300k.xml
```

might return

```
numbers_of_species
"[864, 288, 1728, 1728]"
```

This might not be, at least at first site, very useful for Fortran or Excel users, but it is in a form recognised by many languages (such as Python) and can be transformed into something more useful. Watch this space!

In some of the examples above, we used a wildcard to specify the XML file, ie "`*.xml`". In so doing, if this points to several XML files the result will be a tabulation (collation) of values from each file. This is extremely useful when merging data from many files.

A list of keywords for summon provided in **`rmcprofile.config`** is given in Table 5.1. In fact this file can easily be edited to change the keywords, or better still, to add new keywords for existing quantities; each quantity can be defined with an indefinite number of keywords. It should be noted that for keywords with numbers, if for a specific application there are not enough datasets defined in **`rmcprofile.config`** it is an easy matter to edit the file to add new numbers.

5.2 Tools

Together with the main program a range of small programs are also supplied as tools to help with `RMCProfile`. These tools fall into two main categories: those to help setup the `RMCProfile` files before the program is run and those to help analyse the output from `RMCProfile`. This section describes the programs supplied and how they can be used.

For Windows users, the graphical programs supplied have been compiled using Cygwin — a version of linux for windows. This should not concern you apart from the files in the programs folder must all be kept together and an X server needs to be running before the graphics will be displayed. An X server handles the graphics for unix in the same way that windows does on most PC. There are many commercial versions available, such as Exceed and Xwin32; however if you dont have one of these then we have supplied a basic version from the Cygwin distribution called `basic_X_server` and this should be sufficient for the `RMCProfile` tools. If you use Mac OS X or Linux/Unix, you are already set and ready to go.

5.2.1 data2config

This program will enable you to create a configuration file for `RMCProfile` in any of the acceptable formats starting from a number of popular crystal structure files. Examples of the types of input starting files include the `.cif` and `.tbl` files known to crystallographers, and outputs include the 'classic' generation 3 configuration files (which we do not recommend) and the nicer version

Table 5.1: Default list of `summon` keywords for RMCProfile output XML files.

Keyword	**Description of data**	Data type
`atoms`	List of atom types	Array
`Bragg_background`	Coefficients for background function in Bragg profile	Array
`Bragg_profile_type`	Bragg profile function used	Scalar
`Bragg_weighting`	Weight applied to Bragg profile fitting	Scalar
`close_approach`	Closest approaches allowed in simulation	Array
`Comment`	Metadata comment	Text
`DataNote`	Metadata note about data	Text
`diff_Bragg`	Difference between fitted and experiment Bragg profile	Array
`diff_PDF1`	Difference between fitted and experiment PDF 1*	Array
`diff_QiQ1`	Difference between fitted and experiment scattering function 1*	Array
`dSpacing`	*d*-spacing for Bragg profile	Array
`exp_Bragg`	Experimental Bragg profile data	Array
`exp_PDF1`	Experimental total PDF for dataset 1*	Array
`exp_QiQ1`	Experimental scattering data for dataset 1*	Array
`hkl_recalculate`	Flag (Y/N) to denote whether hkl array was recalculated	Scalar
`Investigator`	Metadata item on the investigators	Text
`Keywords`	Metadata keywords	Text
`lattice_vectors`	Matrix of lattice vectors	Matrix
`maximum_distances`	Maximum distances in distance windows	Array
`maximum_moves`	Maximum move allowed per atom type	Array
`minimum_distances`	Minimum distances in distance windows	Array
`neutron_coeffs_PDF1`	Neutron coefficients for PDF dataset 1*	Array
`neutron_coeffs_SQ1`	Neutron coefficients for scattering dataset 1*	Array
`nnpdf`	Number of neutron PDF datasets	Scalar
`nnsq`	Number of neutron scattering datasets	Scalar
`number_avcoord_constraints`	Number of average coordination constraints	Scalar
`number_coord_constraints`	Number of coordination constraints	Scalar
`number_density`	Number density	Scalar
`number_of_species`	Number of different atomic species	Scalar
`numbers_of_species`	Numbers of atoms of each type	Array
`nxsq`	Number of X-ray scattering datasets	Scalar
`offset_flag`	Y/N values to denote application of offsets to data	Array
`offsets`	Constant offsets applied to data	Array
`partial1`	Partial PDF for atom pair 1*	Array
`Phase`	Metadata item on sample phase	Text
`Pressure`	Metadata item on sample pressure	Text
`print_period`	Number of steps between printing	Scalar
`Q`	Array of scattering vectors for scattering data	Array
`QiQ1`	Partial $iQ(Q)$ data for atom pair 1*	Array
`r`	Array of distances used in PDF functions	Array
`renorm_flag`	Y/N values to denote automatic data renormalisation	Array
`RMC_Bragg`	Fitted Bragg profile	Array
`RMC_PDF1`	Fitted PDF function for dataset 1*	Array
`RMC_QiQ1`	Fitted scattering function for dataset 1*	Array
`RMCNote`	Metadata note about the RMC simulation	Text
`save_period`	Time between configuration saves	Scalar
`supercell`	Multiplication of three unit cell edges to create configuration	Array
`Temperature`	Metadata item on sample temperature	Text
`time_limit`	Time limit set for job	Scalar
`weights`	Relative weights for each dataset	Array

* Use numbers 2,3 etc for subsequent datasets.

6 configuration files using either fractional or orthogonal atomic coordinates. `data2config` is designed to make this stage as easy as possible, requiring very little effort from you as the user. If you prefer tools that gratuitously give you extra work, then we can recommend the `crystal` program that is described later.*

Not only will `data2config` create new configuration files from starting crystal structures, but it will also allow you to convert existing 'classic' version 3 configuration and histogram files into the new version 6 format, thereby helping you to gain maximum advantage from `RMCProfile` version 6.

5.2.1a Command syntax

The `data2config` program is run in the command or shell window by typing a command of the form:

```
data2config [optional arguments] filename
```

(If you don't have your path set to search your current working directory, then you need to use the command in the way that specifies the path, i.e. as `./data2config`.The optional arguments, all of which are prefaced with a hyphen, are

`-cssr`	write the configuration to a CSSR (Daresbury Laboratory's Cambridge Structure Search and Retrieval file format) file, which can be read by the CrystalMaker crystal visualisation program. This has the advantage that it provides an easy way to check that the configuration is what you expected it to be. This flag is switched off if the input file is itself a CSSR file (because the action would be to overwrite the file). The output filename has the extension `.cssr`.
`-dlpoly`	write DLPOLY `CONFIG` file – you get this conversion for free because it was easy to implement.
`-rmc3`	write the 'classic' RMC version 3 configuration file (output filename has the extension `.cfg`). This flag is switched off if the input file is itself a `.cfg` file (because the action would be to overwrite the file).
`-rmc6f`	write RMC version 6f configuration file (a properly-annotated input file with the atoms in fractional coordinates). The output filename has the extension `.rmc6f`.
`-rmc6c`	write RMC version 6c configuration file, which has the same properties as the version 4f file except that the lattice and coordinates are given in cartesian coordinates (units of Å). The output filename has the extension `.rmc6c`.
`-crystal`	writes a file in the format used by the `crystal` tool for generation of version 3 configuration files (described later). The output filename has the extension `.cfgcom`.

*To be fair, there was a time when `crystal` was all you could have, but there were times when the only means of transport besides foot was horseback.

`-sort`	sort the atoms by atom type before being written to the configuration file (this is selected by default when the `-rmc3` option has been selected).
`-diag`	write a diagnostics file; it isn't worth reading because there is a lot of duplicate data, but is useful if you have problems.
`-noannotate`	request no annotations to be added to the configuration files. It is set as a long flag because, in the view of the code's author, you have to be somewhat reckless not to properly annotate your file with useful metadata.

Any number of options can be selected; for example

Allowed input file types are denoted by the name extensions, and these are

`.TBL` or `.tbl`	files produced by `GSAS`; we anticipate that many users of `RMCProfile` will have run a prior Rietveld refinement.
`.CIF` or `.cif`	CIF (crystal information file) format, the format favoured by many databases (such as the Inorganic Crystal Structure Database)
`.SFF` or `.sff`	our own simple file format (subsubsection 5.2.1c).
`.CFG` or `.cfg`	RMC v3 configuration file
`.CFGCOM` or `.cfgcom`	Input file for the `crystal` configuration generation tool (subsection 5.2.3).

The output files all have the same seed name. For example, the command

```
data2config -diag -cssr -rmc4f nacl.cif
```

will produce the configuration files `nacl.cssr` and `nacl.rmc4f`, and the diagnostics file `nacl.out`.

Simply typing the command `data2config` will produce a list of allowed options and input file types, which means that you don't need to have the manual at hand when running the program.

5.2.1b Usage

When you run `data2config` you will be asked a small number of questions. These include

- The size of the supercell, defined by three indices that specify the expansion of the unit cell along each of the three lattice vectors.
- Various metadata items, such as the owner of the data (useful when you want to come back to a configuration file later), the name of the material, a title, a comment etc. Some of these metadata items are collected from the input file, or else cannot be included in any of the configuration files, and thus the specific metadata queries asked will depend on the context. You are allowed to give blank replies, in which case the metadata will be ignored. Such a course of action would be naughty and should make you feel ashamed. Remember, saving 10 seconds now may cause you hours of grief in a few months time.

- If your input file is a classic version 3 configuration file, you will be asked for the names of the atoms.
- If `data2config` cannot successfully extract the element extracted from the atom labels given in the input file, you will be asked for the name of the atom.

Input files will typically contain atom labels (such as `C2` to represent the second carbon atom in the file), from which `data2config` will attempt to extract the chemical symbol for the element. If labels do not contain the element symbol in an unambiguous way, `data2config` will either extract an element symbol regardless, or else will ask you to provide the element symbol. This labels of the form `C2` or `C1N2` will produce the element carbon, but a label of the form `12C` will ask for the element. To assist cases where there is ordering of vacant sites, the symbols `Va` or `VA` will be interpreted as vacant sites.

We would remark that not only can `data2config` be used to produce configuration files from the raw crystal structure data files in order to initiate a new RMC run, but it can also be used to convert legacy classic version 3 configuration files into the newer version 4 configuration files.

5.2.1c Simple file format files

The `SFF` format is designed to allow you to construct an input file with minimum of effort. Unlike the input file for the `crystal` program, you do not have to do much work to produce an `SFF` file. The format of the `SFF` file is most easily described with reference to an example:

```
cell
5.68 5.68 5.68 90.0 90.0 90.0

symmetry 4
x,y,z
1/2+x,1/2+y,z
1/2+x,y,1/2+z
x,1/2+y,1/2+z

atoms 2
Na 0.0 0.0 0.0
Cl 0.5 0.0 0.0
```

There are basically three blocks of data, one specifying the unit cell dimensions, one providing the minimum amount of symmetry information required to generate the whole crystal, and one containing the list of atoms in the basis (with atomic symbol and fractional coordinates). The blocks can be in any order, and the blank lines are not required. On the other hand, the numbers of symmetry operators or atoms are required, as are the lines giving the keywords.

5.2.1d Warning

Users should be warned that in making the generation of configurations easy, `data2config` also makes it easy for a lack of care to lead to problems further down the line. In particular, if your input file reflects a disordered structure – for example one with partially occupied sites – the disorder will

not be handled automatically by `data2config`. This should not come as a surprise; `RMCProfile` can simulate atomic site disorder but will only do so with real discrete atoms (and vacancies), and not using fractions of atoms as given by partial site occupancies.

One way to handle site disorder would be to generate a single unit cell with some artificial structure (for example, if you have Ca and Mg disordered over a single site, merely call this site Ca with full occupancy for now). Run `data2config` to generate a CSSR file containing single unit cell with several sites. Then edit the configuration to change some of the Ca site labels by Mg, and use this CSSR file as the input file for a second run of `data2config` designed to create a supercell. If your disorder involves vacancies, then replace some of the atom labels by the `Va` symbol.

5.2.1e Further developments

`data2config` has been written in a way that should make expansion of its functionality relatively easy. For example, different types of input file should be easy to add (e.g. from other Rietveld structure output codes). Moreover, we have plans for added functionality such as the ability to convert hexagonal lattices to the corresponding non-primitive orthogonal lattice. Finally, being a new tool there is a strong chance that there will be bugs in the code. Bugs can easily be fixed by sending the code author a brief report by email and the input file that triggered a problem.[†] Similarly, requests for new features can be sent to the author.

5.2.2 rmc326

The `rmc326` tool is provided to enable you to produce version 6 data files from old files in the 'classic' format.

The tool is used as a shell command, in the manner of

```
$ rmc326 -o <Name of output data file> [options] [file to convert]
```

The tool will ask a few questions, and then produce the basis of a version 6 data file to which you can add other features that are not found in the 'classic' version.

The options are

`-no_metadata` suppresses requests for metadata.

`-no_atoms` suppresses requests for atom information.

`-no_features` suppresses requests for v6 features.

`-rmca` use when you have an RMCA input file, and disregard all following options.

`-old_poly <number>` use when the data file uses the old polyhedral input, and give the key number of the polyhedral restraint.

`-old_bragg <resolution type>` use when the data file uses the old bragg input style without defining the type of resolution function, and give the type of resolution function.

[†] Email contact details can be obtained from `http://web.me.com/dove_family/martin/contact.html`

-no_magnetism use when data file doesn't include magnetism.

-no_swap use when the data file doesn't include swapping.

You are unlikely to want to use the last two options, but you may be tempted to avoid the use of the metadata facility. Whilst properly annotating your data may be a slight irritation at the time if you are in a hurry, cutting corners with metadata is both careless and reckless. How many times have you put something down carelessly in a hurry and then spent longer looking for it than the time it would have taken to put it somewhere sensible? Similarly, sensible people exploit metadata facilities to the full, and you would like us to think you are sensible, surely? The bottom line is that you should use the metadata features we have provided.

5.2.3 crystal

This program is an alternative to `data2config` for the generation of a starting configuration of atoms for `RMCProfile`, building a supercell of a specific crystal structure and writing out a 'classic' version 3 `.cfg` file. As you will see below, `crystal` has many disadvantages compared to `data2config` – it requires more effort from the user, it only generates version 3 files, and it gives you nothing else – but hey, it costs us nothing to give it away. And if you like nostalgia, this is for you.

The structure must be defined in *P*1 symmetry with the lattice parameter defined in a vector format. An example input file for the BCC structure of SF_6 is shown below.

```
 10   10   10
 5.88677   0.00000   0.00000
 0.00000   5.88767   0.00000
 0.00000   0.00000   5.88767
  2
  2
 0.00000   0.00000   0.00000
 0.50000   0.50000   0.50000
 12
 0.25102   0.00000   0.00000
-0.25102   0.00000   0.00000
 0.00000   0.25102   0.00000
 0.00000  -0.25102   0.00000
 0.00000   0.00000   0.25102
 0.00000   0.00000  -0.25102
 0.75102   0.50000   0.50000
 0.24898   0.50000   0.50000
 0.50000   0.75102   0.50000
 0.50000   0.24898   0.50000
 0.50000   0.50000   0.75102
 0.50000   0.50000   0.24898
 0
sf6_190k.cfg
```

The first line contains the number of unit cells required in each direction. Since this example is cubic the number in each direction is the same, however, for more complex systems the numbers should be chosen to produce a configuration box with approximately equal sides. This is due to the fact the `RMCProfile` calculates the partial radial distributions to a distance of the minimum box edge divided by two, so very different box edges will just produce wasted information.

The next three lines contain the unit cell parameters in vector form. Again for a cubic system or indeed any orthogonal system this is very straightforward with the diagonal of the matrix being *a*, *b* and *c* respectively. For non-orthogonal systems this is not so straightforward and the `lattice_vectors` program supplied with `RMCProfile` can be used to produce the vectors in the correct format for the crystal program.

The next line contains the number of different atom types. For each atom type the subsequent lines then define the number of atoms of that time in the unit cell and then the fractional coordinates (between 0 and 1) for each atom. In the example shown there are two sulphur atoms and then 12 fluorine atoms.

The following line contains a zero, which defines the number of Euler angles is now an obsolete throwback to a time when the crystal program was used for molecular systems.

The final line contains the name of the file to which you wish the configuration to be written.

And yes, you have to work all these numbers out for yourself by hand and then type them in, or else you can write a program to generate them yourself (but the extra step to create the configuration is so trivial that you might as well do it all yourself).

The `crystal` program is run in a command prompt window in two ways. The simplest way is to just type "`crystal`" on the command line and hit return (assuming the program is in your path or the same folder as the command prompt window) and then enter the information above. Alternatively the information can be saved to a file, say `sf6_190k.cfgcom`, and then run by typing "`crystal < sf6_190k.cfgcom`" and hitting return (again it is a assumed here the `sf6_190k.cfgcom` is in the current directory or path).

Note that the `data2config` tool can both read and generate files in the format used by `crystal`. Not entirely sure why.

5.2.4 convol_norm_new

This program is used to convolve the neutron structure factor data with the `RMCProfile` configuration box size function. This used to be necessary with all RMC methods to ensure a fair comparison of calculated to measured data, but now the same task can be achieved directly within `RMCProfile` v6. However, if you still prefer to use the classic mode, this is one more task that you have to do.[‡]

`convol_norm_new` is run on the command line, and it asks for the information required. It also has options to multiple the data by a constant and subtract a constant in case the data needs to be rescaled for use with `RMCProfile`.

[‡] And one more reason to decide to switch to v6 format files.

5.2.5 data_rescale

This program can be used to rescale data to make it suitable for RMCProfile, even including adding a *x*-offset if required. For example, if your data has the points defined at the bin edge and RMCProfile requires it to be defined at the bin centre then you could add or subtract half a bin width, whichever is appropriate. It is run at the command line and requests the information needed.

5.2.6 get_gsas_bragg

This program is used to extract the information required by RMCProfile from a GSAS refinement to enable the Bragg profile to be fitted. It requires that GSAS is installed on the same machine you are running it on. Again it is run on the command line in the same directory as the GSAS refinement files and requires at least the .EXP and required data file to be present. It will then run GSAS through once cycle of powpref and genles if this has not already been done. To run the program simply type "get_gsas_bragg" on the command line and hit return(assuming the program is in the current path or directory) it will then ask for the required information and produce the .bragg, .back, .hkl and .inst files. If the GSAS executable are not in C:\gsas\exe\ then their location needs to be entered after the program name before hitting return; *i.e.*, get_gsas_bragg d:\my_gsas_exe\.

5.2.7 lattice_vectors

This program outputs the lattice vectors required for the crystal program if supplied the lattice parameters from a GSAS refinement or elsewhere. As with the other programs it is run on the command line and asks for the required information.

5.2.8 neighbour_list

This program should be used to produce the neighbour files required if a polyhedral restraint is being used. Again it is run on the command file and will request the information required. The output should be given the extension described in the relevant restraint section above and should have the same 'stem' name as the other RMCProfile run files.

5.2.9 neighbour_list_two

This program is the same as the neighbour_list program apart from the feature that two surrounding atoms can be supplied at the same time, as required by some of the polyhedral restraints.

5.2.10 rmc_to_atomeye

This program can be used to convert RMCProfile configurations into the correct format to be displayed by the atomeye program described below. It is run on the command line and asks for the information required.

5.2.11 basic_x_server

As the name suggests this program provides a basic X server to enable the graphical programs supplied with RMCProfile to function on a Windows computer. It is supplied zipped in a folder and should be unpacked and stored in its own folder. This is because it is actually a collection of programs from the cygwin distribution (www.cygwin.com). To run the program double click the

"rmc_startX.bat" file. Windows security may ask for confirmation to run the program since it provides a service that other programs can access, so please say yes to running it. A command prompt window will appear and will remain open as long as the X server is running, this can be minimised as it will not be need again.

This is a very cut down version of an X server and we would recommend you use a full version such as eXceed, Xwin32 or equivalent if you have them available.[§]

5.2.12 rmcplotm

This a basic plotting program that can be used to plot information contained in the .out and .braggout file, produced by RMCProfile provided at least one save cycle has been completed.

Once again it is actually a collect of programs that must all be stored in the same folder. It requires an X server to be running and can be run in two ways. The first way is on the command line by typing "c:\rmcprofile\tools\rmcplotm\rmcplotm.bat <<stem name>>", where stem name is replaced by the RMCProfile run name such as "rmcsf6_190k" in the above example (again this assumes the program has be stored in c:\rmcprofile\tools\rmcplotm\, and if not then the correct path should be specified). The second way to run the program is to right click on the .out file, select "open with" and then find and select the rmcplotm.bat file. A dos prompt window will then open and can be used to select the required plot.

In the plot interface, left click allows you to zoom, right click resets the view and middle mouse button exits from cursor.

5.2.13 Atomeye

This program has not been written by ourselves or any of the other RMC developers, but is supplied with RMCProfile because we think it is very useful. It was written by J. Li at Ohio State University and can be used to display the RMCProfile configuration once it has been converted into the correct format using the rmc_to_atomeye program described above. For a full description of the program and to download the latest version if required please go to http://mt.seas.upenn.edu/Archive/Graphics/A/.

The version supplied here requires an X server to be running and again all the files should be stored in the same folder; *i.e.*, c:\rmcprofile\tools\atomeye. It can then be run in two ways: the first way is on the command line by typing "c:\rmcprofile\tools\atomeye\atomeye.bat <<atomeye file>>", where 'atomeye file' is replaced by the configuration converted by the program rmc_to_atomeye — *i.e.*, rmcsf6_190k.eyecfg (again this assumes the program has been stored in c:\rmcprofile\tools\atomeye\, and if not the correct path should be specified); the second way to run the program is to right click on the "atomeye file" — *i.e.*, rmcsf6_190k.eyecfg — then to select "open with" and then find and select the atomeye.bat file. A terminal window and a graphics window will then appear. If you select the graphics window and press "F1" then a help file will be displayed in the terminal widow. For full instructions please look at the web address given above.

[§] Actually we would recommend that you consider getting yourself a Linux or Mac OS X system, where you can run scientific codes rather more easily.

Chapter 6

Examples

6.1 Overview

RMCProfile comes with a directory of example files, each example consists of a folder containing the folders `gsas`, `data` and `rmc`. The aim is to supply all the information required to get RMCProfile running on the example system. The `gsas` folder has the GSAS refinement and data to provide all the Bragg profile information, it also includes a `.cfgcom` file containing the information to produce a starting configuration with the crystal program.

The `data` folder contains time-of-flight neutron structure factor $F(Q)$ and radial distribution $g(r)$ data. These two folders provide all the information required to run RMCProfile and produce a configuration of the system. The `rmc` folder contains a completed run and all the output files illustrate what the result will hopefully look like and for you to modify to help run RMCProfile on your own data.

6.2 Example 1: SF_6

SF_6 is a disordered molecular crystal made up of SF_6 octahedra arranged over the crystal lattice. Molecular crystals are often the most disordered crystalline systems and as can be seen from plotting the example data often show a large amount of diffuse scattering. The example data and files supplied are from the GEM diffractometer at ISIS and were collected at 190 K. At this temperature the crystal structure has body centred cubic symmetry with one molecule on the corner of the unit cell and one in the centre. However this average structure puts the fluorine atoms of neighbouring molecules too close together, so on a local level the system tries to minimise the contact distance of these fluorine atoms by rotation of the octahedral molecules. So the system is termed a frustrated system with fluorine-fluorine repulsion driving the motion of the molecules. With the data supplied you should be able to start with the ideal ordered structure and use RMCProfile to model the local deviation within the long range average BCC structure.

6.2.1 The GSAS refinement

As with most RMCProfile modelling, the process actually starts with a GSAS refinement. A reasonable GSAS refinement is contained in the `gsas` folder and can be used to supply RMCProfile with all the information required to model the Bragg profile. It also contains a `.cfgcom` file with the BCC cell written out in *P*1 symmetry. At the moment I do not have a simple program to go from the GSAS atom positions to this *P*1 cell; however, GSAS supplies all the information required and it is a good exercise to try with your own system to ensure you start with a good understanding of the long range average structure. To extract the rest of the information RMCProfile needs run the `get_gsas_bragg` program on the `.EXP` file and select histogram "3" to be extracted for fitting with RMCprofile.

At this point you should have five files for RMCProfile in the `gsas` folder: namely, the `.bragg`, `.back`, `.inst`, `.hkl` and `.cfgcom` files. Copy all of these files out of the `gsas` folder into the main folder using a common stem name; *i.e.*, `sf6_190k.bragg` etc.

Now in the main SF_6 folder run the crystal program and enter the information in the `.cfgcom` file or use the pipe command (*i.e.*, `crystal < sf6_190k.cfgcom`). This should produce a $10\times10\times10$ supercell of the BCC structure that will be saved in a file called `sf6_190k.cfg`.

6.2.2 The total scattering data

The next stage is to prepare the total scattering data for RMCProfile. How to collect and produce total scattering data is beyond the scope of this simple example and so will not be covered here. The data folder contains three $F(Q)$ files from three of the detector banks of the GEM diffractometer at ISIS. It is quite common to combine these into one data set but here I have left them separate to illustrate RMCProfile can fit many $F(Q)$ data sets simultaneously. The data is scaled from -1 to 0, so need to be multiplied by 0.27593 (the sum of the neutron partial scattering factors) to be suitable for RMCProfile. The data also need to be convolved with the RMCProfile configuration box function to enable a fair fitting comparison. Both of these operations can be performed using the `convol_norm_new` program. For a $10\times10\times10$ super cell the truncation distance is 29.42, so you can run the program to produce the new files or use the prepared files `*conv29p42rmc.dat` supplied. You should reproduce at least one of these files to check you follow the procedure and know how the programs work.

Once you have produced all the data files, copy them into the main SF_6 directory together with the `sf6mcgr190k.dat` file, which contains the Fourier transform of the $F(Q)$ files — namely, the $G(r)$. Once again the preparation of the $G(r)$ data will not be described here since it is normally obtained during the preparation of the total scattering data.

6.2.3 The RMCProfile files

The next file to prepare is the main `.dat` file, this can be copied from the example in section 4.5.1 or from the `rmc` folder. You should check that the names of the data files in the `.dat` file match the names you have chosen. Finally the file for the polyhedral constraint needs to be prepared and since the SF_6 molecules form a octahedron, polyhedral restraint "4" can be used. The `.poly` file can be copied from the `rmc` folder or produced with your favourite text editor; the appropriate format is describing in section 4.7.4 and the ideal S–F bond should be set to 1.56 Å. The `.sf` and `.fs` files can be produced using the `neighbour_list` program; when asked for "a maximum distance"

enter “1.9”, since this is bigger than the nearest S–F distance but not far enough to include the next nearest neighbours. When producing the `.sf` file the program should report that it has found an octahedra and that the average coordination number is 6. If this is not the case something is wrong with the `.cfg` file.

You should now have all the files you need to run RMCProfile on this system as described at the start of section 4.5. To run the full refinement using a $10\times10\times10$ super cell will take many hours, so if you dont want to wait this long check things run and that the χ^2 values start to reduce. Once you are happy everything is working you can copy the final `.cfg` file from the `rmc` folder to get an idea of a final set of fits.

6.2.4 Other things to try

Once you are happy that RMCProfile is correctly set up and you can get it running you might like to try one or more of these suggestions:

- Try producing a smaller configuration — say $4\times4\times4$ — and the appropriate total scattering data. This will run quicker and demonstrate the minimisation process, although the configuration may not be big enough to give a good representation of the thermal distribution of atoms.
- Try fitting just a subset of the data; *i.e.*, just the Bragg profile or Bragg profile or $g(r)$, etc. Again this will run more quickly and once χ^2 seems to be approaching a minimum you could addanother data set back in.
- Try replacing the polyhedral restraint with a distance window constraint. This should produce a very similar final configuration but will illustrate the different functionality.
- Obviously the best thing is to try your own data if possible and just use the `.dat` file as a template.

6.3 Example 2: $SrTiO_3$

Strontium titanate has the ABO_3 perovskite structure and as such its crystal structure is made up of a network of corner sharing TiO_6 octahedral with strontium atoms in the interstices. The example data and files supplied are from the GEM diffractometer at ISIS and were collected at 5 K. At this temperature the crystal structure is tetragonal with the octahdera tilted from their ideal positions in the cubic high temperature structure. With the data supplied in the three folders (`gsas`, `data` and `rmc`) you should be able to start with the ideal ordered structure and use RMCProfile to model the local deviation within the long range average tetragonal structure. Unlike SF_6 this system does not have a high level of local disorder, so the final configuration will be quite ordered in comparison. The procedure required to run RMCProfile on this data is the same as described in section 6.2 but obviously substituting the $SrTiO_3$ files where required.

Chapter 7

Appendices

7.1 Appendix A: Total scattering

The phrase “total scattering experiment” refers to a measurement of the scattering of radiation by matter that covers all scattering vectors (i.e. all values of $\sin\theta/\lambda$) and includes scattering with all possible changes of energy of the radiation. It therefore encompasses elastic scattering, such as from Bragg peaks, which arises from the static or mean atomic scattering, and inelastic scattering, which arises from dynamic processes. The Fourier transform of the total scattering measurement provides information about the relative positions of atoms, which can usually only be interpreted over short distances (Billinge and Thorpe, 1998; Dove, 2002).

Until recently, total scattering experiments were mostly associated with studies of fluids or glasses (Chieux, 1978; Wright, 1993, 1994, 1997). In contrast, diffraction studies of crystalline materials tend to be primarily focused on the measurements of the Bragg peaks, with little concern for the shape of the background provided that it could be fitted by an appropriate polynomial. The Bragg peaks give information about the distributions of positions of atoms within the unit cell, and for many purposes this is exactly all the information that is required. Since fluids and glasses do not have long-range periodic order, there are no Bragg peaks. One of the exciting developments in crystallography over recent years has been the application of total scattering methods to crystalline materials (Billinge and Thorpe, 1998), particularly for crystalline materials that have a high degree of structural disorder. The subsequent coupling of total scattering measurements to modelling through the RMC method has extended the opportunities for studying disordered crystalline materials at an atomistic level.

The information contained within the Bragg scattering and total scattering can be appreciated by considering the basic scattering equations. The starting point is the static scattering function, $I(\mathbf{Q})$:

$$I(\mathbf{Q}) = \frac{1}{N}\sum_{j,k}\langle b_j b_k \exp\left(i\mathbf{Q}\cdot[\mathbf{r}_j - \mathbf{r}_k]\right)\rangle \tag{7.1}$$

where b_j is the scattering factor for atom labelled j, $\mathbf{r}_j$ is the instantaneous position of this atom, and N is the number of atoms in the sample. $\mathbf{Q}$ is the scattering vector, defined as the change in wave vector of the neutron beam associated with the scattering process (note that the wavelength of the

scattered beam can change through the scattering process). For coherent scattering (i.e. where all atoms of the same type scatter the same way), this can be rewritten as

$$I(\mathbf{Q}) = \frac{1}{N}\sum_{j,k}\overline{b}_k\overline{b}_k\langle\exp\left(i\mathbf{Q}\cdot[\mathbf{r}_j-\mathbf{r}_k]\right)\rangle \tag{7.2}$$

where the overline represents the average over all atoms of the same type. The main point of this equation is that it shows how the intensity of scattering is determined directly by the instantaneous distances between atom positions, and does not directly contain information about the actual positions of individual atoms. For periodically ordered systems, the information about the positions of individual atoms is contained within the Bragg peaks. The equation for Bragg scattering is

$$I_{\text{Bragg}}(\mathbf{Q}) = \frac{1}{N}\left|\sum_j\langle\exp\left(i\mathbf{Q}\cdot\mathbf{r}_j\right)\rangle\right|^2 \tag{7.3}$$

If we consider an atom to have a mean position, $\overline{\mathbf{r}}_j$, we can write the average as

$$\langle\exp\left(i\mathbf{Q}\cdot\mathbf{r}_j\right)\rangle = \exp\left(i\mathbf{Q}\cdot\overline{\mathbf{r}}_j\right)\int p(\mathbf{r}-\overline{\mathbf{r}}_j)\exp\left(i\mathbf{Q}\cdot\mathbf{r}\right)\,\mathrm{d}\mathbf{r} \tag{7.4}$$

where p is a probability distribution function, and for a harmonic crystal it is a simple Gaussian function (Willis & Pryor, 1975), with a Fourier transform (i.e. the term in the integral) that is also a Gaussian.

The objective of total scattering experiments is to determine the distribution of interatomic distances. Indeed, the atomic structures of fluids and glasses can only reasonably be described in terms of the interatomic distances. Since fluids and glasses are isotropic, the scattering function is independent of the direction of $\mathbf{Q}$. The scattering function is therefore better described by averaging over all orientations of $\mathbf{Q}$, leading to

$$I(Q) = \frac{1}{N}\sum_{j,k}\overline{b}_k\overline{b}_k\frac{\sin\left(Q\left|\mathbf{r}_j-\mathbf{r}_k\right|\right)}{Q\left|\mathbf{r}_j-\mathbf{r}_k\right|} \tag{7.5}$$

This result is derived in Appendix B. It is useful to recast the formalism in terms of the distribution of interatomic positions rather than as a sum over all pairs of atoms. First we subtract out the terms where $j = k$ to give *

$$I(Q) = i(Q) + \sum_m c_m\overline{b_m^2} \tag{7.6}$$

where the first term will describe pairs of atoms and will be considered in more detail below, and where c_m is the proportion of atoms of type m ($\sum_m c_m = 1$). The first term can be written as

*Note that `RMCProfile` treats the functions $i(Q)$ and $F(Q)$ as synonymous; the program authors have the bad habit of using both interchangeably in their publications, but that reflects that they are individuals after all.

$$i(Q) = \rho_0 \int_0^\infty 4\pi r^2 G(r) \frac{\sin Qr}{Qr} \, \mathrm{d}r \tag{7.7}$$

where the new function $G(r)$ describes the distribution of interatomic distances:

$$G(r) = \sum_{m,n} c_m c_n \overline{b}_m \overline{b}_n \left(g_{mn}(r) - 1\right) \tag{7.8}$$

and ρ_0 is the number of atoms of any type per unit volume. The individual pair distribution functions are defined as

$$g_{mn}(r) = \frac{n_{mn}(r)}{4\pi r^2 \rho_m \mathrm{d}r} \tag{7.9}$$

where $n_{mn}(r)$ is the number of atoms of type n lying within the range of distances between r and $r + \mathrm{d}r$ from any atom of type m, and $\rho_m = c_m \rho_0$. It is common to define the pair distribution function in terms of the new function

$$D(r) = 4\pi r \rho_0 G(r) \tag{7.10}$$

so that

$$Qi(Q) = \int_0^\infty D(r) \sin(Qr) \, \mathrm{d}r \tag{7.11}$$

The reverse transformation is then given as

$$D(r) = \frac{2}{\pi} \int_0^\infty Qi(Q) \sin(Qr) \, \mathrm{d}r \tag{7.12}$$

(Wright, 1993, 1994, 1997; Chieux, 1978; Dove, 2002). This transform provides the means by which the information about structure over short length scales, as encapsulated in the function D(r), can be extracted from the experimental measurements.

The experimental task (Wright, 1993, 1994, 1997) is to obtain the best measurements of $Qi(Q)$ from the total scattering data. It is not within the purpose of this Appendix to explain the experimental details; these have been documented elsewhere (Wright, 1993; Howe et al., 1989; Dove et al., 2002). However, it is essential to appreciate that it is important to determine $Qi(Q)$ to a high value of Q (typically of order 30–50 Å^{-1}) in order to achieve the best possible resolution in $D(r)$: the resolution Δr is given as $2\pi/Q_{\mathrm{max}}$, where Q_{max} is the maximum value of Q achieved in the measurement of $Qi(Q)$. It is also important to appreciate that it is necessary to have an absolute measurement of $Qi(Q)$ if the resultant $D(r)$ is to be interpreted quantitatively. It is essential that all sources of additional scattering and all sources of signal attenuation can be independently determined and taken into account in the treatment of the data (Wright, 1993; Howe et al., 1989; Dove et al., 2002).

It should be noted at this stage that there is a confusion in the literature in that different authors use different sets of symbols for the quantities discussed in this article, including the use of $G(r)$

for what we have called $D(r)$. This is a long-standing historic problem; Keen (2001) gives a good comparison of the different ways of labelling the fundamental quantities.

7.2 Appendix B: Isotropic averaging of the scattering function

In this appendix we derive the equation for the scattering of radiation from an isotropic material. This means that we assume that any interatomic vector $\mathbf{r}$ is found for all orientations, which in turn means that we need to average over all relative orientations of $\mathbf{r}$ and $\mathbf{Q}$. We write $r_{jk} = |\mathbf{r}_j - \mathbf{r}_k|$, $Q = |\mathbf{Q}|$, and $x = \cos\theta$, and calculate the orientational average for one vector as

$$\langle \exp\left(i\mathbf{Q}\cdot[\mathbf{r}_j - \mathbf{r}_k]\right)\rangle = \frac{1}{4\pi}\int_0^{2\pi} \mathrm{d}\phi \int_0^{\pi} \sin\theta\,\mathrm{d}\theta \exp(iQr_{jk}\cos\theta) \tag{7.13}$$

$$= \frac{1}{2}\int_{-1}^{+1} \exp(iQr_{jk}x)\,\mathrm{d}x \tag{7.14}$$

$$= \frac{\sin(Qr_{jk})}{Qr_{jk}} \tag{7.15}$$

Thus for all atoms we obtain

$$S(Q) = \sum_{jk} \overline{b}_j \overline{b}_k \sin((Qr_{jk})/Qr_{jk} \tag{7.16}$$

$$= \sum_{j} \overline{b}_j^2 + \sum_{j\neq k} \overline{b}_j \overline{b}_k \sin((Qr_{jk})/Qr_{jk} \tag{7.17}$$

where we separate the terms involving the same atoms (the *self terms*) and those involving interference between different atoms.

We can express the equation using pair distribution functions rather than perform a summation over all atom pairs. We define $g_{mn}(r)\,\mathrm{d}r$ as the probability of finding a pair of atoms of types m and n with separation between r and $r + \mathrm{d}r$. This function will have peaks corresponding to specific sets of interatomic distances. For example, in a material containing SiO_4 tetrahedra there will be a peak corresponding to the Si–O bond at $\sim$ 1.6 Å and a peak corresponding to the O–O bond at $\sim$ 2.3 Å. Each partial $g(r)$ will be zero for all r below the shortest interatomic distances, and will tend to a value of 1 at large r. Thus we can rewrite $I(Q)$ as

$$I(Q) = \sum_m c_m \overline{b_m^2} + i(Q) + S_0 \tag{7.18}$$

$$i(Q) = \rho_0 \sum_{m,n} c_m c_n \overline{b}_m \overline{b}_n \int_0^{\infty} 4\pi r^2 \left(g_{mn}(r) - 1\right) \frac{\sin(Qr)}{Qr}\,\mathrm{d}r \tag{7.19}$$

where c_m and c_n are the proportions of atoms of type m and n respectively, and ρ_0 is the number density. S_0 is determined by the average density, and gives scattering only in the experimentally inaccessible limit $Q \to 0$.

7.3 Appendix C: A primer on the Reverse Monte Carlo method

The main task of the Reverse Monte Carlo method is to generate configurations of atoms from which computed properties most closely match experimental measurements, with the primary experimental data being total scattering data (McGreevy & Pusztai, 1988; McGreevy, 1995; Mellergård & McGreevy, 1999, 2000; McGreevy, 2001). The starting point is some configuration of atoms that has the correct density, and, in the case of a crystalline system, a confining box that has the dimensions that are some integral multiple of the experimental lattice parameters.

During an RMC simulation, the atomic coordinates are varied in a random manner in order to improve the best agreement with experimental data. We write any experimental quantity as y^{exp}, and the corresponding calculated quantity as y^{calc}. We then define an agreement factor as

$$\chi^2 = \sum_j \left(y_j^{\text{exp}} - y_j^{\text{calc}} \right) / \sigma_j^2 \tag{7.20}$$

where we sum over all data points (labelled by j), and σ_j is a weighting factor which may correspond to the experimental uncertainty on y_j. Clearly the best final configuration is that for which the value of χ^2 is a minimum, as in any data fitting technique. In the Monte Carlo approach, the calculated values of y are changed through the random changes in the configuration. If the change lowers the value of χ^2, the change to the configuration is accepted. On the other hand, if the change to the configuration causes χ^2 to increase by an amount $\Delta\chi^2$, the change is not automatically rejected, but accepted with the probability

$$P = \exp\left(-\Delta\chi^2/2\right) \tag{7.21}$$

This ensures that the model does not get trapped in a local minimum, and enables the model to converge towards the global minimum.

In the RMC method, the experimental total scattering data, corresponding to the values of y in the equation for χ^2, can be $Qi(Q)$ or $D(r)$. In fact, in our work we use both at the same time. It is also possible to include additional data, such as the Bragg scattering profile or XAFS data, and to also include a contribution from the use of restraints. Accordingly we write the RMC χ^2 in the following form:

$$\chi^2_{\text{RMC}} = \sum_m S_m \chi^2_m \tag{7.22}$$

where $S_m = 0, +1$, and we define a separate χ^2 for each set of data:

$$\chi^2_{Qi(Q)} = \sum_k \sum_j \left(Qi_{\text{calc}}(Q_j)_k - Qi_{\text{exp}}(Q_j)_k\right)^2 / \sigma^2_k(Q_j) \quad (7.23)$$

$$\chi^2_{D(r)} = \sum_j \left(D_{\text{calc}}(r_j)_k - D_{\text{exp}}(r_j)_k\right)^2 / \sigma^2(r_j) \quad (7.24)$$

$$\chi^2_{\text{profile}} = \sum_k \sum_j \left(I^{\text{calc}}_{\text{profile}}(t_j)_k - I^{\text{exp}}_{\text{profile}}(t_j)_k\right)^2 / \sigma^2_k(t_j) \quad (7.25)$$

$$\chi^2_f = \sum_\ell \left(f^{\text{calc}}_\ell - f^{\text{req}}_\ell\right)^2 / \sigma^2_\ell \quad (7.26)$$

$$\chi^2_{\text{BS}} = \frac{1}{k_{\text{B}}T} \sum_\ell D_\ell \left[1 - \exp\left(-\alpha(r - r_0)\right)\right]^2 \quad (7.27)$$

$$\chi^2_{\text{BB}} = \frac{1}{k_{\text{B}}T} \sum_\ell K_\ell \left(\cos\theta - \cos\theta_0\right)^2 \quad (7.28)$$

The term χ^2_f corresponds to the polyhedral constraints (Keen, 1997, 1998), where f may be a bond length or bond angle, with the required value obtained from the low-r peaks in $D(r)$. The profile term was introduced by ourselves for the study of crystalline materials (Tucker et al., 2001b, 2002a, 2002b). The terms χ^2_{BS} and χ^2_{BB} are the new molecular constraints, where the parameters are designed to be realistic so that the weighting term is now related to the experiment temperature. The function $I_{\text{profile}}(t)$ describes the Bragg diffraction pattern. Our work is mostly based on time-of-flight neutron sources, so $I_{\text{profile}}(t)$ a function of neutron flight time t. This is the same χ^2 that is minimised by a least-squares technique in Rietveld refinement.

Part of the reason for including as large a range of experimental data as possible in the RMC analysis is associated with the fact that the RMC method is effectively a method based in statistical mechanics. As a result, an RMC simulation will evolve to maximise the amount of disorder (entropy) in the configurations. Thus the RMC simulation will give the most disordered atomic configurations that are consistent with the experimental data. There may be a range of configurations that match the data, with different degrees of disorder. Only by maximising the range of experimental data can this problem be minimised.

7.4 Change log

Version 6.4

v6.4.6

1. Preparation work for v6.5, particularly with regards to handling per-point errors on the data,
2. Ability to read directly data files generated by the Gudrun data reduction and transformation package.
3. Ability to read supercell information from the `.rmc6f` file.

4. Ability to read the range of *d*-spacings from the **.dat** file and for the code to generate the range of *h, k,* ℓ values of the Bragg peaks used in the analysis of the diffraction data, thereby removing the requirement for a separate **.hkl** file.
5. Some internal workings to avoid array dimension overflow associated with the Bragg analysis.

v6.4.5

1. You can now automatically generate a ppm file from the bond orientation distribution function, from which it is easy to generate a file in png or gif format.

v6.4.4

Bug fixes only

v6.4.2

1. The ability to add molecular constraints, specifically bond-stretching and bond-bending. The details are documented in the manual.

Version 6.3

v6.3.5

1. Fixed the input of NEUTRON_COEFFICIENTS for the case where the list does not run over two lines (the previous version allowed the list to run over more than the first line, but I didn't check for the case where it didn't).
2. Added the CCVIZ subordinate keyword to the CML :: keyword block, allowing rmcprofile to automatically generate the xhtml file from the xml file. There are certain requirements for this to work.
3. Added the CSSR subordinate keyword to the FLAGS :: keyword block. This allows automatic creation of a CSSR file for easy viewing in CrystalMaker.
4. Added some missing code from the output section, which I had forgotten needed to be done. Problem was that the code was printing out PDFs and scattering functions that differed from what was being requested
5. Fixed some bugs on the way
6. Tracked down the causes of occasional bus errors / segmentation faults
7. Added data_type and fit_type to the xml file output
8. Tweaked some of the output messages

v6.3.3

1. Some small bug fixes, eg getting the wrong name for the new configuration
2. Completed the writing of **`.his6f`** files (I hadn't appreciated that they weren't completed)
3. Written the ability to read **`.his6f`** files (not having done this before was an oversight)
4. Some small rearrangements with the storage of variables to make coding easier
5. Some new subroutines and one new file exists which are not used by v6.3.3 but which will be used by v6.4, and I put them in here because I am aiming at editing my development version only when possible.

v6.3.1

1. The headline change is that it reads and writes rmcf6f configuration files. This may now sound like a big deal, but took quite a bit of programming to get there.
2. I have added a number of new subroutines to make this work.
3. I have the (undocumented) option to resort the configuration, but this doesn't quite work so I will fix it soon (one of the planned tweaks). You don't need to use this tool with the example configuration. It will use the keyword `SORT ::` in the input file. I think it is an easy fix to make but I want to start getting the testing of the RMC procedure working.
4. The program now creates **`.csv`** files for the partial $g(r)$ and partial $S(Q)$ functions, with headers that tell you which partials you have. I also put headers into the **`.out`** files. It has long been a pet dislike of mine that you have to deduce which column refers to which partial from external information, so I am pleased to have fixed this one!
5. I think that the code still generates a **`.cfg`** file when you don't want one, so I need to look at this (another tweak).
6. I doubt that restarting from **`.his6f`** works, because I need to copy some fixes from the .rmc6f code. I would like to see that the **`.rmc6f`** stuff works first (another tweak).
7. I would like to put the experimental data into the .csv format. I realise writing this that I haven't thought about when you have several data files (another tweak).
8. I also don't have some of this output stuff working with x-rays, so need to chat about this (another tweak.
9. I now write out the PDF functions with the same value of r as used in the code. I want to check again on how it handles Q (I did check this once, but will do this again as a tweak).

Chapter 8

References

[1] R. L. McGreevy, Reverse Monte Carlo modelling. Journal of Physics: Condensed Matter **13**, R877–R913 (2001).

[2] D. A. Keen, M. G. Tucker and M. T. Dove, Reverse Monte Carlo modelling of crystalline disorder. Journal of Physics: Condensed Matter **17** S15–S22 (2005).

[3] M. G. Tucker, M. T. Dove and D. A. Keen, Simultaneous analyses of changes in long-range and short-range structural order at the displacive phase transition in quartz. Journal of Physics: Condensed Matter **12** L723–L730 (2000).

[4] M. G. Tucker, M. D. Squires, M. T. Dove and D. A. Keen, Dynamic structural disorder in cristobalite: Neutron total scattering measurement and Reverse Monte Carlo modelling. Journal of Physics: Condensed Matter **13** 403–423 (2001).

[5] M. G. Tucker, M. T. Dove and D. A. Keen, Application of the Reverse Monte Carlo method to crystalline materials. Journal of Applied Crystallography **34** 630–638 (2001).

[6] M. T. Dove, M. G. Tucker and D. A. Keen, Neutron total scattering method: simultaneous determination of long-range and short-range order in disordered materials. European Journal of Mineralogy **14** 331–348 (2002).

[7] M. G. Tucker, D. A. Keen, M. T. Dove, A. L. Goodwin and Q. Hui, RMCProfile: Reverse Monte Carlo for polycrystalline materials. Journal of Physics: Condensed Matter **19** art no 335218 (16 pp) (2007).

[8] G. Evrard and L. Pusztai, Reverse Monte Carlo modelling of the structure of disordered materials with RMC++: a new implementation of the algorithm in C++. Journal of Physics: Condensed Matter **17** S1–S13 (2005).

[9] M. G. Tucker, M. T. Dove and D. A. Keen, MCGRtof: Monte Carlo $G(r)$ with resolution corrections for time-of-flight neutron diffractometers. Journal of Applied Crystallography **34** 780–782 (2001).

[10] SA Wells, M. T. Dove, M. G. Tucker and K. O. Trachenko, Real-space rigid unit mode analysis of dynamic disorder in quartz, cristobalite and amorphous silica. Journal of Physics: Condensed Matter **14** 4645–4657 (2002).

[11] An introduction to the use of neutron scattering methods in mineral sciences. M. T. Dove, European Journal of Mineralogy **14** 203–224 (2002).

[12] A. L. Goodwin, M. G. Tucker, M. T. Dove and D. A. Keen, Phonons from powder diffraction: A quantitative model-independent evaluation. A. L. Goodwin, M. G. Tucker, M. T. Dove, and D. A. Keen, Physical Review Letters **93** art no 075502 (4 pp) (2004); Erratum: Phonons from powder diffraction: A quantitative model-independent evaluation [Phys Rev Lett **93** 075502 (2004)]. Physical Review Letters **95** art no 119901 (4 pp) (2005).

[13] S. A. Wells, M. T. Dove and M. G. Tucker, Reverse Monte Carlo with geometric analysis – RMC+GA. Journal of Applied Crystallography **27** 546–544 (2004).

[14] Q. Hui, M. G. Tucker, M. T. Dove, S. A. Wells and D. A. Keen, Total scattering and reverse Monte Carlo study of the 105 K displacive phase transition in strontium titanate. Journal of Physics: Condensed Matter **17** S111–S124 (2005).

[15] M. G. Tucker, A. L. Goodwin, M. T. Dove, D. A. Keen, S. A. Wells and J. S. O. Evans, Negative thermal expansion in ZrW_2O_8: Mechanisms, rigid unit modes, and neutron total scattering. Physical Review Letters **95** art no 255501 (4 pp) (2005).

[16] M. G. Tucker, D. A. Keen, J. S. O. Evans and M. T. Dove, Local structure in ZrW_2O_8 from neutron total scattering. Journal of Physics: Condensed Matter **19** art no 335215 (16 pp) (2007).

[17] A. L. Goodwin, M. G. Tucker, M. T. Dove, E. R. Cope and D. A. Keen, Model-independent extraction of dynamical information from powder diffraction data. Physical Review B **72** art no 214304 (15 pp) (2005).

[18] A. L. Goodwin, M. T. Dove, M. G. Tucker, and D. A. Keen, MnO spin-wave dispersion curves from neutron powder diffraction. Physical Review B **75** art no 075423 (2007).

[19] A. L. Goodwin, S. A. T. Redfern, M. T. Dove, D. A. Keen, M. G. Tucker, Ferroelectric nanoscale domains and the 905 K phase transition in $SrSnO_3$: A neutron total-scattering study. Physical Review B **76** art no 174114 (11 pp) (2007).