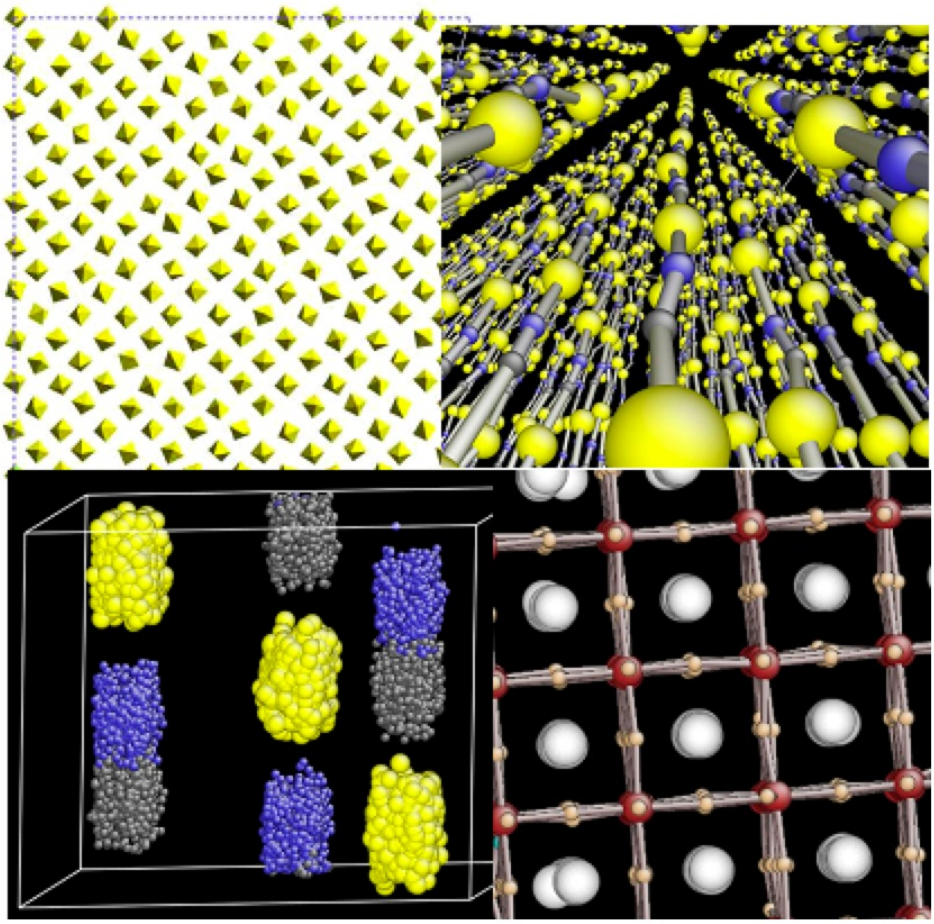
RMCProfile User Manual

Code version 6.7.9



Matt Tucker, Martin Dove, Andrew Goodwin, Anthony Phillips,
David Keen, Helen Playford, Wojciech A. Slawinski,
Igor Levin, Victor Krayzman, Maksim Eremenko, Yuanpeng Zhang

August 18, 2021

Contents

Chapter 1

Introduction

1.1 We know you don't read manuals!

Very briefly, `RMCProfile` is an implementation of the *Reverse Monte Carlo* (RMC) method [1]. `RMCProfile` is based on the original code for RMC, but significantly amended and extended to give new capabilities and to take account of more recent programming and informatics standards. If you just want get things running please just look for the `rmcprofile_tutorial.pdf` file in the tutorial folder. This helps you check everything is working and leads you through some of the common features of the program.

This manual aims to tell you most of what you need to know about `RMCProfile` program, together with appropriate background material.* 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 probably ought to read most of this manual before you start. But we know you won't want to, so hopefully we have arranged it so that this is not quite as challenging as it might seem at this point.

If you are one of those people who only want to read a manual when necessary and just want to get things running, please just look for the `rmcprofile_tutorial.pdf` file in the tutorial folder. This helps you check everything is working and leads you through some of the common features of the program.

We would point out that `RMCProfile` is still a work in progress, and we have not yet reached the stage where next versions are merely minor iterations from previous versions.

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

1.2 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 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.

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.

[†]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.

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 β-quartz [3] and β-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.

1.3 Introduction to `RMCProfile`

1.3.1 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 materials 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 simul-

taneous 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.

1.3.2 `RMCProfile` in a nutshell

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

(1) Support for large atomic configurations.

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

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

(4) Ability to fit both neutron and x-ray total scattering data, allowing the use of more than one dataset for each experiment type.

(5) Ability to fit simultaneously the real-space pair distribution function obtained by Fourier transform of neutron total scattering data.

(6) Ability to fit the Bragg profile directly as well as the total scattering.

(7) Ability to include EXAFS data in the RMC method.

(8) Use of generalised molecular potential-energy-based constraints for bond lengths and angles.

(9) On-the-fly calculation of bond-angle distribution histograms and bond-orientation maps.

(10) On-the-fly calculation of various average quantities, such as mean bond lengths, mean bond angles, mean bond-orientation spherical harmonics, and mean bond Kubic harmonic values.

(11) Ability to use bond valence sums.

(12) Use of closest approach and distance window constraints.

(13) Use of coordination number constraints.

(14) Ability to use model-specific polyhedral constraints.

(15) XML output for data visualisation and analysis.

(16) Various ancillary tools, eg tools to create starting configurations from crystallographic CIF files.

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

This manual is primarily focussed on Version 6 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 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 4.15. Version 6 is fully compatible with classic-style input files, although new features are not accessible via this route. The classic file formats are described in the Version 5 manual.

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 forever.

Chapter 2

Capabilities

2.1 Fitting neutron and X-ray total scattering data

The RMC modelling method was developed specifically to fit neutron and X-ray total scattering data and these data are still the key experimental data for `RMCProfile`. Total scattering, i.e. the Bragg and diffuse scattering collected (ideally) as an integration over all scattered energies at constant scattering vector, $Q = 4\pi \sin\theta/\lambda$, provides the information which allows `RMCProfile` to investigate deviations from the average crystal structure. These data are measured by neutron or X-ray diffraction, but their collection and treatment is more demanding than a routine powder diffraction experiment for Bragg profile analysis through, for example, Rietveld refinement. This is for a number of reasons:

(1) The data should be collected over as wide a range of scattering vectors as possible and extending to high-Q. This explains the early dominance of neutron diffraction, and in particular time-of-flight neutron diffraction where total scattering data to $Q_{max} \sim 50\,\text{Å}^{-1}$ is routinely accessible. This is achieved through the availability of short-wavelength neutrons and because the neutron-nucleus interaction does not introduce a fall-off in intensity at high-Q in contrast to X-ray diffraction where the form-factor significantly suppresses scattering at high-Q. More recently, however, diffraction instruments using high-energy X-rays from third-generation synchrotron sources have provided total scattering data to $Q_{max} \sim 35\,\text{Å}^{-1}$ and there are also some laboratory X-ray diffractometers based on silver or molybdenum anode sources that can provide data to $Q_{max} \sim 20 - 24\,\text{Å}^{-1}$.

(2) The data will typically need to be collected for longer, in order to measure the weaker, broader diffuse scattering with sufficient statistics.

(3) Sources of background should be minimised and measured in order to make a robust subtraction. This typically involves measuring, in addition to the sample, the empty sample can or capillary, empty sample environment and empty diffractometer, all in the same experimental configuration.

(4) The data need to be normalised accurately and placed on an absolute scale. This is achieved for neutron diffraction through the measurement of a vanadium sample of similar dimensions to the sample; an equivalent measurement is not possible for X-ray diffraction and a number of routines exist to facilitate accurate scaling of the data (see for example the `GudrunX` manual).

Fortunately there are a number of packages, notably `Gudrun` and `GudrunX`, which treat the experimental data for background, absorption, multiple-scattering etc. and produce normalised total scattering data so provided a careful experiment has been carried out, there should only be a small overhead in producing total scattering data over a standard powder diffraction pattern.

`RMCProfile` will fit total scattering data in both reciprocal space and in real space, i.e. both total scattering structure factors and pair distribution functions. It may initially appear curious that the 'same' data can be used in two different ways. However experience has shown that it is beneficial to fit both the reciprocal and real space data together since each function emphasises different aspects of the structure; the pair distribution function highlights structure at short distances whereas the scattering data are weighted more strongly to the longer-range structure. Details about these different functions and their forms can be found in section 4.3.

2.2 Fitting experimental pair distribution function data

As discussed above, `RMCProfile` exploits separately the information contained in the total scattering data measured in reciprocal space and the pair distribution function (PDF), which is a function defined in real space.

As shown in the equations described in section 4.3, the PDF is obtained as the Fourier transform of the scattering data. There are a number of approaches to computing this transform. If data are collected on an instrument such as GEM at ISIS (which is typical of instruments at pulsed spallation sources), the scattering data are obtained separately for different banks of detectors. The common practice within the liquids and amorphous materials communities is to merge data from all banks to produce an overall scattering function. This overall scattering function can then be Fourier transformed quite easily. The same practice can be followed for crystalline materials, and although we sometimes do this, in all honesty it is not a good idea to do so. The patterns from the different banks have different resolution functions, which can easily be seen when comparing Bragg peaks seen in several banks. Usually the Bragg peaks in the detectors with lower scattering angles are broader (there is an important $\cot\theta$ term in the function describing the resolution). Thus merging of data typically means that you slice together data from different banks of detectors (rather than adding the data when obtained in different banks). Such a approach is really being rather cavalier with the effects of resolution, but it is what is commonly done.*

Thus we recommend that resolution be taken into account within the process of converting the total scattering data into the PDF. One approach is to use the MCGR method [9], in which a model PDF is adjusted until its Fourier transform is consistent with the experimental data. It is possible to include the effects of resolution when comparing the computed Fourier transform with the experimental data, so the effects of resolution are automatically removed from the process.

*The effect of ignoring resolution is easy to understand. The experimental data represents the convolution of the actual scattering function with the function describing the resolution. Thus the Fourier transform will be the product of the PDF and the Fourier transform of the resolution function, which typically will be a function peaked at r = 0 and decaying slowing with r. Thus the features at higher-r will be damped, and the important error introduced into the analysis will be that the coordination numbers represented by the data will be lower than the true coordination numbers. This is likely to lead to the RMC method introducing some additional disorder into the models it produces.

Users need to be aware that the PDF is weighted by the strength of scattering from individual atoms in addition to a weighting by the relative concentration of each atom. In the case of neutron scattering this weighting will correspond to the product of scattering lengths of the two atoms for each contribution to the PDF. In the case of x-ray scattering, there is a complicating factor that the effects of the size of atoms are approximately accounted for via a scaling of the scattering function by the products of x-ray form factors; this is not an exact procedure however.

In passing we note that there are several definitions of the PDF in common use – this situation will not change, primarily because different definitions have their own advantages, and there is no single function that captures all the advantages. These are described in detail in section 4.3. `RMCProfile` is capable of using all definitions for input, and will convert between conventions for fitting if required. We recommend that the functions $D(r)$ or $T(r)$ be used as the function for fitting because the errors are spread evenly across all points in the function. Be warned though – not only are there several different functions, different authors use different symbols for these functions! We use the symbols described in section 4.3.

2.3 Fitting the Bragg profile

The headline reason for the existence of `RMCProfile` is its ability to utilise the information contained specifically in the Bragg diffraction pattern, rather than merely folding it into the normal total scattering pattern.

The Bragg diffraction contains information about the distribution of atoms on an absolute scale. On the other hand, the total scattering contains information about the arrangements of atoms relative to each other. A determination of the structure of a material requires both types of information. Traditionally crystallography of course focuses only on the atomic distribution, which in practice usually means obtaining the mean positions of the atoms and their associated distribution function, which is usually defined as a Gaussian function with a width in three-dimensional spaced defined by an ellipsoid function with six parameters defining its shape and orientation. Bragg diffraction alone contains no information on the correlation between neighbouring atoms, which is the information contained within the diffuse scattering component of total scattering. By explicit treatment of the Bragg scattering we give more weight to the distribution of atomic positions, and allow the treatment of the total scattering to focus more on the correlated motions of the atoms. Furthermore, since the Bragg peaks are assigned a set of Miller indices, which reflect the fact that the scattering vectors associated with the Bragg peaks are vectors in three-dimensional space. Thus inclusion of this information puts something of the three-dimensional nature of the data into the RMC model, even though the data are collected in one-dimensional mode.

The experiments are typically performed using powder diffraction. It is important that for each data set run through RMC there will have been a prior Rietveld refinement performed. There are four reasons for this:

(1) The refinement will give the best lattice parameters consistent with the data, and which should be used in generating the initial atomic configuration;

(2) The refinement will give an absolute scale factor that is required as input for the Bragg part of RMC;

(3) The refinement will give parameters for the background function to enable the Bragg peaks to be accurately subtracted from the diffraction pattern;

(4) The refinement will give parameters for the experimental resolution function, which are used in the RMC treatement of the Bragg profile.

Conventionally, the Bragg part of RMCProfile is closely tied to the functions used in the GSAS program, and thus GSAS would be recommended to be used for the Rietveld refinement. Users may of course use their preferred refinement packages for their Rietveld work, but with the old implementation of Bragg profile in RMCProfile it has been necessary to convert the output into GSAS format in order for RMCProfile to read it. For data collected on certain diffractometer, those profile functions available in GSAS (I or II) may not be appropriate enough to describe peak shapes. In this case, one may need the capability of defining new peak profile function per needed. Accordingly, the defined profile function will have be to implemented in RMCProfile . With this regards, we here take a generic approach to provide interface between Rietveld refinement and RMCProfile . Instead of implementing explicitly the functional form of peak profile, we export and tabulate the profile data for RMCProfile to read. Currently following the new approach, RMCProfile is only compatible with Topas. However, the peak profile tabulation routine itself is generic and should not be specifically bundled to any Rietveld refinement package.

The user will be required to give the Bragg diffraction profile (at the present time RMCProfile will only use data from a single bank of detectors from a spallation neutron source), background parameters, the scale parameter, the parameters for the resolution function, and the upper-Q or lower-d limit of the diffraction data to be used. With conventional and new method for Bragg pattern fitting in RMCProfile , the input entries in the main control .dat file and required parameter files are different. Users are referred to section-4.1.

2.4 Resolution correction

The instrument effect can be corrected in RMCProfile. In practice, the noise level in Q-space will inrease as we approach high-Q region. This is effectively a dampening effect in Q-space, which will then lead to the broadening effect in r-space. In Rietveld-like refinement approach, such an effect is corrected with a single parameter *Qdamp* which is usually obtained through fitting the PDF pattern of a standard sample (e.g., NIST Si). The same approach is taken here in RMCProfile, details of which can be found in Appendix-F. ***However, since the PDF calculation in RMCProfile is following a numerical approach, the atomic distances falling into a certain pre-defined bin will be treated as indistinguishable. Therefore, any broadening effect with width smaller than the bin size being used won't make any practical sense. In this case, we assume the width of the r-space Gaussian broadening function is identical to the bin size.***

Another instrument effect to be corrected is the finite resolution effect. Conventionally in Rietveld-like approach, the peak profile in Q-space is assumed to be Gaussian and accordingly a single parameter *Qdamp* is introduced to account for the dampening effect in r-space as the result of resolution effect in Q-space. Again, this parameter is usually obtained through fitting PDF data for a standard sample. Here the same approach is also implemented in RMCProfile, details of which can be found in Appendix-F. Furthermore, a generic approach beyond Gaussian assumption for peak shape is also implemented. Theoretical background and technical details are both more

complicated and they won't be covered here. Users are referred to the paper Ref. [27] for detailed discussions. Technical procedures are briefly recovered in Appendix-F.

2.5 Distance-window constraints

The distance-window constraint is an extension of the standard closest-approach constraint in RMC, with two significant enhancements. First, in addition to specifying the closest two atom types can come together, 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. Secondly, the distance window constraint is formulated such that atom neighbours are retained during an `RMCProfile` run. A network can thus be generated between atoms using the distance window constraints that will retain the overall topology or local molecular topology of the initial model. When `RMCProfile` is first run, a neighbour list of all the atoms that fit within the distance windows is generated and this list remains the same unless deleted. This means that the window sizes can be expanded or reduced later but the atoms being constrained stays the same.

To use the distance window constraint it is no longer necessary to have a separate **`.dw`** file (although this will still work). Instead, use the `DISTANCE_WINDOW ::` keyword in the main Version 6 **`.dat`** file. The required information is detailed later in 4.1.2. It is currently not possible to use this constraint with a classic format input file.

Once run RMCProfile will 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.

2.6 Interatomic potentials

In principle[†] the RMC method should be sufficient to drive a simulation without needing help from additional potential energy functions, 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 may 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.

[†]and a principle the relevant author subscribes to.

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. `RMCProfile` enables the use of two types of potentials, namely for stretching of bonds and flexing of bond angles.

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. In fact, the weighting used on the potentials within `RMCProfile` is tied with the temperature of the experiment, and if the weightings on the experimental data are associated with experimental errors the RMC method will give the correct balance between the dat and potentials.

2.7 Bond valence constraints

In the early stages of an `RMCProfile` refinement, any atomic move which broadens the δ-function like peaks in the correlation function calculated from the crystal structure will be accepted as an improvement to the fit, and this can cause regions of extreme disorder in the configuration which are subsequently very difficult to correct. The use of bond valence constraints, which maintain the bond valence sum to within a certain window of the ideal value, can help to prevent this unphysical effect.

Bond valence constraints can be particularly useful when several components in the system have very similar contributions to the scattering functions.

The implementation of bond valence sums as a soft chemical constraint in `RMCProfile` is discussed further in [11].

2.8 Modelling atomic site disorder through atom-swap moves

Many materials contain some degree of chemical disorder. For example, materials in the perovskite solid solution series $CaTiO_3$–$SrTiO_3$ may have disorder of the Ca and Sr atoms over the 12-coordinated sites, or for some compositions there may be some degree of long-range order (eg at a 50:50 composition there is the possibility for these cations to order in a NaCl-type ordering pattern). Moreover, regardless of the state of long-range order, there may be some degree of short-range order. This is exactly the sort of problem that RMC should be capable of tacking, but in crystalline materials it is virtually impossible for atoms to swap positions simply through the normal small step movements that are used in the method. Instead RMC allows moves in which two atoms, selected from a specified list of atom types, swap positions. The intensities of peaks in the pair distribution function should be sensitive to the site occupancies, given that the numbers of neighbours in any specific peak in the pair distribution function is determined by the crystal structure. The specific example of the $CaTiO_3$–$SrTiO_3$ is described in [10].

2.9 Coordination Constraints

Coordination constraints are still a functioning part of the RMCProfile code, although their functionality is largely superseded by the more versatile distance window constraints. However, if one is studying an amorphous material and is using a randomly generated starting configuration, it is highly likely that *none* of the atom pairs obey the proposed distance window and therefore this constraint will fail. In these circumstances the coordination constraints are highly useful. Options for both fixed coordination constraints (where the intent is to drive all of a certain type of atom to have a specific coordination number) and average coordination constraints (where the average coordination number should be, for example, 4, but as long as this is achieved, coordination numbers of 3 or 5 are also acceptable) are available. The usage of these constraints is discussed in section 4.1.2. It is almost certainly a bad idea to try and use them both at the same time.

2.10 Polyhedral restraints

Within RMCProfile are defined a series of system-specific polyhedral restraints. The aim of these restraints is to maintain the integrity (in terms of connectivity) of a polyhedral network during RMC refinement. This is achieved by restraints on bond lengths and angles: taking into account the atoms which neighbour each other in coordination polyhedra. A full description of these restraints can be found in section 4.10.

While the polyhedral restraints are not generic, they are not limited to systems of the named composition but can be used for any system with the same polyhedral connectivity. It is important to note that the polyhedral restraints are not compatible with the atom-swapping ability of RMCProfile and therefore you will need to use the distance window constraint (see section 2.5) to maintain the connectivity of a system in which they wish to investigate swapping.

One can envisage the application of these restraints in a system such as a zeolitic SiO_2 polymorph, wherein the SiO_2 restraint can hold the tetrahedra together, thus allowing RMCProfile to concentrate on the structural features of more interest to the experimenter.

2.11 Magnetic structure modelling

Some of this section refers to features in older versions of RMCProfile and as such is in need of updating. We hope to have an updated version ready for the next release. In the meantime, please refer to the information below and get in touch if you have any questions about the implementation of magnetism in RMCProfile .

2.11.1 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.

2.11.2 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 $B(r)$:

$$\begin{aligned} S_{\text{mag}}(Q) &= \frac{2}{3}c_{\text{M}}\left[\frac{e^2\gamma}{2m_{\text{e}}c^2}gJf(Q)\right]^2 + 4\pi\rho c_{\text{M}}\left[\frac{e^2\gamma}{2m_{\text{e}}c^2}f(Q)\right]^2 \\ &\quad \times \int r^2\left\{A(r)\frac{\sin Qr}{Qr} + B(r)\left[\frac{\sin Qr}{(Qr)^3} - \frac{\cos Qr}{(Qr)^2}\right]\right\}\text{d}r, \end{aligned} \tag{2.1}$$

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

$$I(\mathbf{Q}) = \frac{1}{N}\left|\sum_j \mathbf{q}_j p_j(Q)\langle\exp(\text{i}\mathbf{Q}\cdot\mathbf{r}_j)\rangle\right|^2, \tag{2.2}$$

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

$$\mathbf{q}_j = \frac{\mathbf{m}_j}{m_j} - \frac{\mathbf{Q}(\mathbf{Q}\cdot\mathbf{m}_j)}{Q^2 m} \tag{2.3}$$

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 $f_j(Q)$:

$$p_j(Q) = \frac{e^2\gamma}{2m_e c^2} g J_j f_j(Q). \tag{2.4}$$

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

2.11.3 Implementation

The various keywords and required parameters are listed in section 4.1 below, 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

$$f(Q) = A\exp(-aQ^2/16\pi^2) + B\exp(-bQ^2/16\pi^2) + C\exp(-cQ^2/16\pi^2) + D, \tag{2.5}$$

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 $\sigma_{\max}$ 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.

2.11.4 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. 2.1. In fact there is also a slight deviation from cubic lattice symmetry associated with this magnetic transition, but here we will ignore this effect.

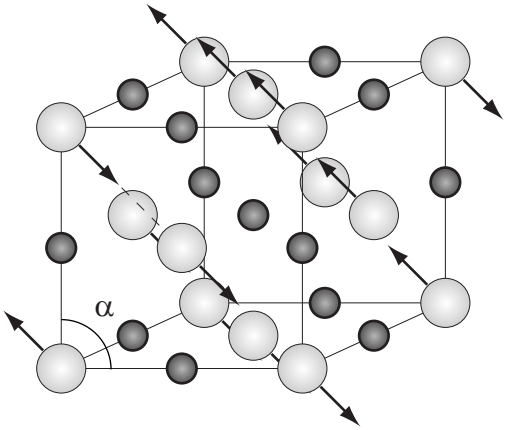


Figure 2.1: The antiferromagnetic structure of MnO: Mn and O atoms are shown as large light-grey and small dark-grey spheres, respectively.

The MnO atom and spin RMC configurations

We are going to refine the spin orientations in a 4 × 4 × 4 supercell of the unit cell shown in Fig. 2.1. 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. 2.2.

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
```

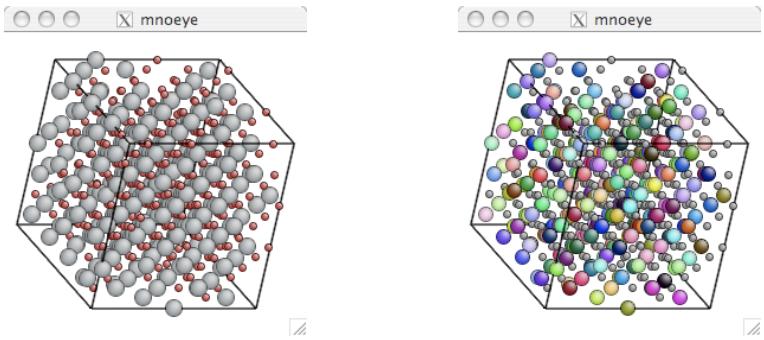


Figure 2.2: The RMC starting configuration as seen in ATOMEYE, using default colouring (left) and with Mn atoms coloured by the initial spin directions (right).

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. 2.2. 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. 2.3. 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.

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
```

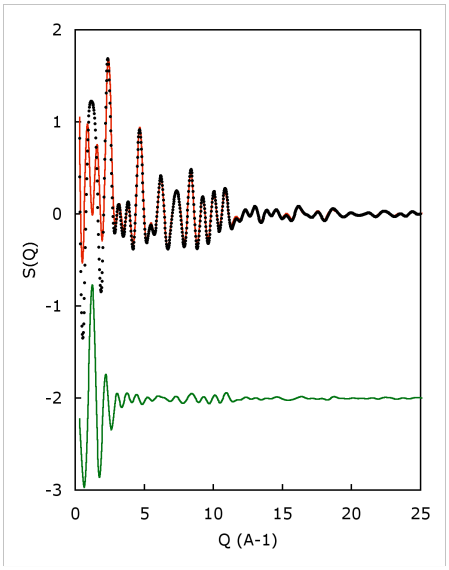


Figure 2.3: Initial RMC fit to neutron scattering data for MnO. Data are shown as solid points, the RMC fit as a red line, and the difference (data−fit) shown as a green line, shifted by 2 units.

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. 2.4. 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.

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

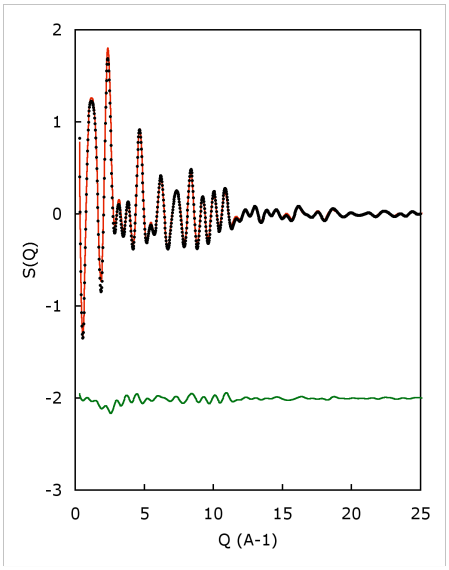


Figure 2.4: A typical equilibrium RMC fit to neutron scattering data for MnO.

```
.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. 2.5. 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.

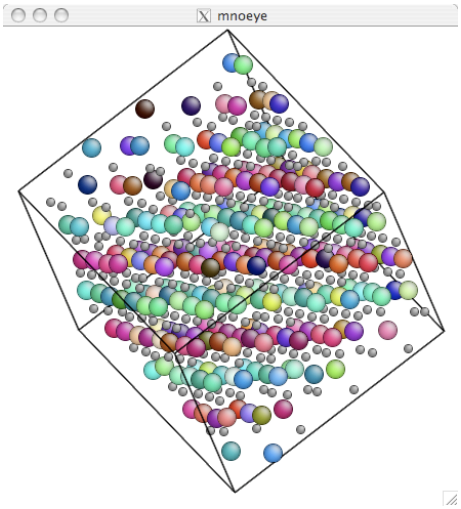


Figure 2.5: A typical RMC equilibrium configuration as seen in ATOMEYE.

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. 2.6, which in this case shows that the spins are aligned parallel and antiparallel to an axis very close 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
```

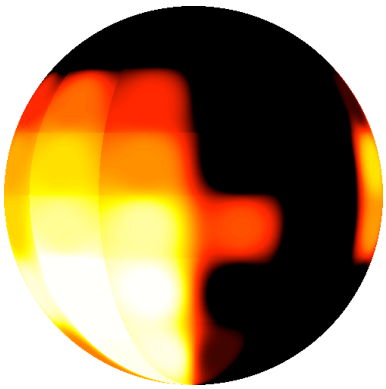


Figure 2.6: A typical spherical spin distribution (black = low probability; white = high probability). The view is taken looking down [100] with the [010] axis to the right hand side, and the [001] axis towards the top of the figure. In this particular case, alignment appears to be approximately parallel (and antiparallel) to [112].

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.

2.12 Fitting EXAFS data

The technical background for EXAFS can be found in the Appendix-C To Appendix-F in the RMCProfile tutorial documentation which comes together with current manual in the main RMCProfile distribution directory. Also the users can refer to the documentation at `http://www.rmcprofile.org/imagesFhj/5/51/Rmcprofile_exafs_manual.pdf` for more information.

2.13 Fitting 3D diffuse scattering data

The input file with diffuse-scattering intensities must have the following format:

- The 1st line contains two digits: the total number of pixels and the number of symmetrically equivalent points, *nsym*. If no symmetrization is used, nsym=1.

- The first 3 columns are digits enumerating pixels in a 3D data grid.
- The following 3 columns specify k_x, k_y, and k_z coordinates (in $\mathring{A}^{-1}$) of each pixel, which represent components of the corresponding reciprocal-lattice vector (i.e. $\vec{k} = 2\pi/\vec{a}$).
- The last column specifies the intensity on either linear or logarithmic scale.

Symmetry can be used to improve signal-to-noise ratio. Space group symmetry applies to Bragg reflections but not necessarily to diffuse scattering. If certain symmetry is assumed, it can be accounted for by inserting sets of additional $3(nsym - 1)$ columns between the 1st set of three k-coordinates and the intensity column. Each of these sets specifies the coordinates k_x, k_y, and k_z of symmetrically equivalent reciprocal-lattice points that are expected to have the same intensity. The program will calculate the average intensity over the equivalent *nsym* points and compare it with the corresponding experimental value. Since $I(\vec{k}) \equiv I(-\vec{k})$ (Laue symmetry), there is no benefit in specifying the inversion symmetry.

If certain pixels/k-points need to be excluded from the fit, their intensity should be set to zero. The program will still calculate and printout the diffuse-scattering intensities at these points but ignore their values while evaluating the residual.

If noise in the calculated signal as caused by limited statistics presents a problem, it can be smoothed using a Gaussian filter. The size of the kernel is specified in the control **.dat** file as described below.

The 3D diffuse-scattering capability is activated using the following block of keywords in the **.dat** file:

```
DIFFUSE_SCATTERING3D ::
  > FILENAME :: %% specify name of the input file
  > FITTED_SCALE ::  %% fit the scale factor
  > CONSTANT_SCALE :: %% constant scale factor
  > FITTED_OFFSET  :: %% fit the offset
  > CONSTANT_OFFSET :: %% set a constant offset
  > WEIGHT :: %% assigned weight
  > KERNEL3D :: 3 3 3 %% size of the kernel (in pixels)
                      %% for the Gaussian filter
```

The intensity scale, linear or logarithmic, is specified using flag 'LINEAR ::' or 'LOGARITHMIC ::' in the **.dat** file.

2.14 Producing starting configurations

2.14.1 Introduction

The RMC method is designed to work with atomic configurations that contain of order of a few thousand to a few tens of thousands of atoms with periodic boundaries. Unlike other techniques, it is not appropriate to consider contributions to the pair distribution function for interatomic distances

that extend beyond half-way to the the nearest replica generated by the periodic boundaries. Thus the configuration needs to be large enough for the pair distribution function to be defined to a useful distance.[‡] But against this is balanced the fact that the more atoms you have the more degrees of freedom are available to fit the data, which may not be a good thing. Moreover, the more atoms you have the longer the time the RMC simulation will take to reach convergence to a satisfactory result. Thus our experience is that a few thousand atoms is typically a good size to work with.

The starting point for `RMCProfile` is to generate an initial configuration of atoms. `RMCProfile` expects this configuration to be defined by a box with periodic boundary conditions and a given size, together with a list of constituent atoms and their positions within the box.

2.14.2 Crystalline materials

We provide a number of tools to enable the user to generate an initial configuration of atoms from a trial crystal structure. For example, we anticipate that many people will start from a crystal structure provided in the standard CIF format used by crystallographers, or perhaps with a **`.TBL`** file generated by the GSAS Rietveld refinement code. The `data2config` tool will use the information contained within these files to generate configurations containing the number of unit cells specified by the user.

2.14.3 Non-crystalline systems

Obtaining a starting structure for a non-crystalline material is a challenge that may take us outside of `RMCProfile`. There are several approaches, depending on the material and on the scope of data available. The standard RMC approach is to simply throw the atoms into a box at random, and let the RMC method arrange them appropriately. We have a tool which can do this, and can also go one step further and produce starting configurations which do not violate closest-approach constraints. This tool is called `dwbuild` and it is discussed further in section 6.1. Another approach to generate a random starting configuration is to run `RMCProfile` without data or constraints, which will randomise an initially ordered configuration (which can be generated from a real or trial crystal structure using the approach outlined above). One important point is to get the density correct, but this should already have been determined in order to convert the total scattering function to a pair distribution function. There are known problems with this approach, and the RMC method will need to be used with care.

If you have some trial interatomic potentials, you could use a molecular dynamics or Monte Carlo method to generate a starting structure. Even a crude model might be adequate to give a trial configuration that will be good enough. Our `data2config` tool can create configuration files of the right format from some configuration file formats that are generated by standard molecular dynamics simulations (and if there is demand more file formats can be incorporated into the functionality of `data2config`).

[‡]As will be explained later, when fitting against the neutron total scattering data it is necessary to take account of the finite range of the pair distribution function when performing the required Fourier transform; `RMCProfile` handles this automatically, but the shorter the range of distances in the pair distribution function the more degraded is the Fourier transform.

2.15 XML output files

RMCProfile produces a number of standard output files in text form. It 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. XML is a way of writing documents in which each piece of information is enclosed with descriptive tags, as will be demonstrated below. XML files are not designed to be read by humans – although if a human wants to read and XML file the contents should be reasonably-well self-described – but instead should be easily readable by computer programs. And herein lies the power of XML, which has enabled us to build analysis tools that will be described later in section 6.3.

Sample extracts of an RMCProfile CML file are shown in Figures 2.7 and 2.8. You can see that each item of date is properly described within the CML file. For example, in Figure 2.7 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 2.7 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 2.7 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 2.7 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 2.7 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 2.8 shows actual data generated. These are contained within <module> tags, which are defined for different tasks or roles. The examples in Figure 2.8 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).

As a prelude to the more detailed description given in section 6.3, we note that 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 is described in subsection 6.3.1, and which is bundled with `RMCProfile`. A second application called `summon`, also supplied with the `RMCProfile` package and described in subsection 6.3.2, enables extraction of information from an XML file without having to scroll through the file.[§]

[§]Both tools require Python to be available on the computer being used, and both work from the command line.

```
<?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 2.7: 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 2.8: Extracts from the latter part of an `RMCProfile` XML output file, showing the stepwise data and final data.

Chapter 3

Installing and Running RMCProfile

3.1 Installation

We strongly recommend at this point that you put this manual to one side and refer instead to `rmcprofile_tutorial.pdf` which will guide you through the installation and initial runs of `RMCProfile`. For completeness, however, the instructions are also supplied below, and on `www.rmcprofile.org`.

`RMCProfile` is provided as a single 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; simply follow the instructions below.

To install `RMCProfile` on a Windows machine:

(1) Unzip the file once it has downloaded.

(2) Copy the whole RMCProfile folder to a more permanent home, such as `c:\RMCProfile`. This folder should contain two subfolders (exe and tutorial) and a file called `RMCProfile_setup.bat`.

(3) Run the program by double clicking on the RMCProfile_setup.bat file. This should bring up a command prompt window called RMCProfile.

Note: these instructions should work for Windows XP and 7. There may be problems with file permissions on Windows Vista, and we have heard some reports of issues with the path settings in Windows 8. If you encounter these, please get in touch.

To install `RMCProfile` on a Mac OS X machine:

(1) Unzip the file once it has downloaded (your browser may have done this for you).

(2) Copy the whole RMCProfile folder to your Applications folder. This should make a folder /Applications/RMCProfile which contains three subfolders (exe, libs and tutorial) and a file called `RMCPRofile_setup.command`.

(3) Run the program by double clicking on the RMCProfile_setup.command file. This should bring up a terminal window called RMCProfile version 6.

To install RMCProfile on a Linux machine:

(1) Download the file that fits your system (i.e. 32 or 64 bit).

(2) Put the file in your top directory once it has downloaded.

(3) Type `tar -pxvzf rmcprofile_package_v6_5_1_linux_32.tgz` (change 32 to 64 if you have that version). This will unpack everything into an RMCProfile_package directory.

(4) To run the program, cd to the RMCProfile_package directory and type ./RMCProfile_setup and hit return. This should bring up an xterm window called RMCProfile version 6.

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.

3.2 Setting up the files for an RMCProfile calculation

RMCProfile requires a number of input files, that are described in more detail in the following sections but are listed here both to give a quick overview and also to act as a handy checklist. In short, the input files are the file containing main control data (**.dat**), one file containing the starting configuration (**.rmc6f**, **.his6f**, **.cfg** or **.his**), files containing the raw data (no constrains on the name), files to manage the Bragg scattering (**.bragg**, **.back**, **.inst** and **.hkl**), and files concerned with restraints (**.dw**, **.bvs**, **.bonds** and **.triplets**). Many of these files will have the same stem name but different extension tags. For example, if we were performing an RMC simulation on the crystalline phase of methane, we might have the following input files (optional unless stated otherwise):

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

methane.rmc6f containing the atomic configuration in fractional or cartesian coordinates respectively. If using classic format, the file name to be used is **methane.cfg**, and you also have the option to call this **methane.rmc3f** if using Version 6 **.dat** file with classic configuration format. (This file is compulsory unless **methane.his6f** is present, section 4.11).

methane.his6f 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 **methane.rmc6f**.[1] If using the classic file format, this file is called **methane.his** (subsection 4.11.2).

methane.bragg containing the Bragg diffraction data (section 4.13).

methane.back containing parameters to describe the shape of the background in the Bragg diffraction data (section 4.13).

[1] Actually, we will see later that this behaviour can be switched off using a keyword in the main **.dat** file

methane.inst containing information on the instrument resolution function, in the format as provided by the GSAS Rietveld refinement code (subsubsection 4.13.3c) (compulsory if using the Bragg scattering as data).

methane.hkl containing information on the range of Bragg peaks to be considered in the analysis of the Bragg diffraction data. This file is not required if the information is provided in the main **.dat** file; subsubsection 4.13.1a).

methane.dw containing information on the distance window constraint as described in section 2.5 (this has now been incorporated into the main **.dat** file but is included here as it is still possible to use the .dw file).

methane.bonds containing lists of all the bonds used in the interatomic potentials. If this file is not available, it will be automatically generated for re-use by RMCProfile; we recommend that this file be generated using a well-ordered configuration (subsection 4.7.4).

methane.triplets containing lists of all the bond angles used in the interatomic potentials. If this file is not available, it will be automatically generated for re-use by RMCProfile; we recommend that this file be generated using a well-ordered configuration (subsection 4.7.5).

The **methane** 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.

In addition to these will be files containing the scattering data and the pair distribution function. There are no constraints on the names of these files, which are given in the **.dat** file. These data files will be described in section 4.3.

These files are described in the chapters and sections indicated above

3.3 Running RMCProfile

To run RMCProfile, execute the following shell command:

```
rmcprofile methane > methane.log
```

The part of the line "> methane.log" tells the command to direct the standard output to a specified file, here called **methane.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 methane > methane.log
```

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

On Windows, if you always run `RMCProfile` by double clicking on the `.bat` file, your path should be set for you and you will not have to worry about locating the individual programs. They should run from your `RMCProfile` command prompt window.

3.4 Output files

`RMCProfile` will generate a number of output files. These include, following the methane example given above:

`methane.out` contains a lot of summary information and generated data.

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

`methane.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.

`methane.his6f` contains the configuration and the pair distribution functions generated by the run.

`methane.rmc6f` will contain an updated configuration. If you request periodic saves of the configurations, these will have a number appended as, e.g. **`methane_23.rmc6f`**.

`methane_chi2.txt` contains an evolution of all Chi2 agreement factors during the refinement.

`methane_SQpartials.csv` will contain the calculated partial scattering functions in CSV format.[2]

`methane_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.

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

`methane_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.

`methane.braggout` will contain the calculated and experimental Bragg diffraction data.

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

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

[2] 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.

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

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

methane.ylm contains lists of the mean bond spherical harmonics and Kubic harmonics.

These files will be described in subsequent chapters.

Chapter 4

Input Files

4.1 `RMCProfile` main data file

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, e.g. "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 35, 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.*[1]

`AVERAGE_COORDINATION_CONSTRAINTS ::` Requires an integer which identifies the number of average coordination constraints to be used. Is followed by a subordinate keyword to define the parameters. *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.*

`BRAGG_DUMMY ::` By providing some key parameters, such as the Bragg shape, the instrumentation file name, etc., the program will generate the simulated Bragg pattern (the so-called dummy data, for neutron only). This keyword can be provided together with the real datasets, and then the program will fit those real datasets as well as generating the corresponding simulated pattern.

`BOX_SIZE ::` Number of unit cells (values 'Nx Ny Nz' should be given following '::') in the configuration box along x,y and z directions. Required for calculating the average amplitude if fitting diffuse-scattering patterns.

`BULK_RHO ::` Specifies the bulk density for the low dimensional RMCProfile simulation. Detailed information about the theoretical background of the correction for the low D RMCProfile, refer to the paper (to be pulished).

`BVS ::` Introduces a block of subordinate keywords which define the parameters used for the bond valence constraint. *If this keyword is not present, the program will search for a .bvs file, and if neither is present the BVS constraint will not be used.*

`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 6.3.*
`COLLECT_CONFIG ::`	Specify every certain number of accepted moves to output configurations consecutively.
`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.*
`CONVOLUTION2D ::`	This keyword is optional if fitting the 2D diffuse scattering pattern. Activates convolution of the calculated 2D diffuse-scattering pattern with a specified kernel matrix (provided in a file of which the name should be given following '::'). Useful for fighting with the speckle artifacts in the calculated signal.
`CRYSTALMAKE_OUT ::`	Generates an output file with atomic coordinates in the Crystal Maker format.
`CURVATURE ::`	A restraint which suppresses artificial spikes in partial PDFs (if a problem) by minimizing the integral of the modulus of a second derivative of a given partial over a specified r-range. Parameters to following '::' are 'n r_1 r_2 w' representing number of a partial, bounds of the r-interval in Åand weight (direct).
`DATA_FILE_VERSION ::`	Requires a number that indicates a version for the data format. Allowed values are `rmc6f`, `rmc3f` and `cfg`. *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.*

DIFFUSE_SCATTERING ::	Introduces a block of subordinate keywords for fitting 2-D electron diffuse scattering patterns. Text can follow '::' but will be ignored. Each diffraction pattern requires a separate keyword followed by subordinate keywords. DO NOT include if no diffuse scattering to fit.
DISTANCE_WINDOW ::	Introduces a block of subordinate keywords which provide the distance window constraint information.
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 ::	Requires an integer which identifies the number of average coordination constraints to be used. Is followed by a subordinate keyword to define the parameters. *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.
GPU_ACCELERATOR ::	This specifies the using of GPU accelerator for RMC fitting. A single integer number should follow '::' to specify which video card to use, with the starting index of 0, i. e. 0 representing video card slot 1. As of writing, the GPU acceleration is only available on Windows release. We will be working on compiling and testing its Linux version.
IGNORE_HISTORY_FILE ::	Instructs `rmcprofile` to ignore any **`.his`** or **`.his6f`** files present in the working directory.
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`. *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.*

LAYER_THICKNESS ::	Specfies the thickness of the two dimensional layer used for the RMCProfile running.
LINEAR ::	Specifies the intensity scale (linear) for the diffuse scattering pattern.
LOGARITHMIC ::	Specifies the intensity scale (logarithmic) for the diffuse scattering pattern.
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.*[1]
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.*[2]
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 4.2).*[2]
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_REAL_SPACE_DUMMY_DATA ::	By providing some key parameters, such as the lower and upper limit of the r-space, etc., the program will generate the simulated neutron real space pattern (the so-called dummy data). This keyword can be provided together with the real datasets, and then the program will fit those real datasets as well as generating the corresponding simulated pattern.
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.*
NEUTRON_RECIPROCAL_SPACE_DUMMY_DATA ::	By providing some key parameters, such as the lower and upper limit of the Q-space, etc., the program will generate the simulated neutron reciprocal space pattern (the so-called dummy data). This keyword can be provided together with the real datasets, and then the program will fit those real datasets as well as generating the corresponding simulated pattern.
NUMBER_DENSITY ::	Requires the value of the number density. *Default units are $Å^{-3}$. Default value is the the value automatically calculated from the configuration file, and this will always override any value given with this keyword.*
PARTICLE_RADIUS ::	The radius of the nanoparticle to be used in the RMCProfile running.
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 section 4.10. *Default is to not use this constraint if this keyword is not provided.*

`POLARIZATION ::`

For perovskite structures, activates a fit of the electrical polarization to its experimental value. The calculations have been implemented for ABO3 and (A,A')BO3 perovskite structures, where A and A' are two distinct cations sharing the A-sites. Ideally, Born effective charges (BEC) should be used; however, if BECs are unavailable, formal ionic charges can be used instead. Parameters that should follow '::' include 'ψ w QB Ql Qt QA [QA']' standing for experimental polarization (C/m^2), weight for the penalty term, BEC for a B-cation, a longitudinal BEC component for oxygen, a transverse BEC component for oxygen and BEC for an A-cation, BEC for an A'-cation (only if present). The order of atomic types in the configuration should be: 1-B, 2-O, 3-A, 4-A'.

`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 75

`PRE_EDGE ::`

Activates the fitting of the average magnitude of the local off-centering for octahedrally-coordinated ions in perovskites. Parameters should be provided following '::' is: 'pre_abs_type pre_nei_type pre_int pre_weight pre_width pre_width_weight' which represents type of the octahedral atom, type of the neighbouring ligand atom, magnitude of the off-centering (Å), weight, a width of the off-centering distribution (Å) and weight for the width, respectively.

`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.*
SWAP_MULTI ::	Introduces a block of subordinate keywords that provide information about the atom swapping facility. *Default is to not use this multi 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.
TOPAS_RESOLUTION_MATRIX ::	Specify to use Topas profile for preparing resolution matrix for the purpose of resolution correction.
ITERATION_LIMIT ::	Requires generated moves limit for the job.
TITLE ::	Requires text to act as a title for the run. This is used for metadata and is not compulsory.
USE_DSHAPER ::	Use DShaper approach [25] for correcting the nano-size effect. Refer to Appendix-G for more details.

USE_RMC6F_ATOM_INFO ::	In some situations, we could possibly have atoms of the same type sitting on crystallographically different positions. In such cases, if we put all atoms of same type in the same block, we need to explicitly specify current keyword to tell RMCProfile to treat those crystallographically different atoms properly as they appear in the .rmc6f configuration file.
VALENCE ::	Specifies the valence (the value - V1 V2 ... Vn-types - should be given following '::') for all the atom types. Required for fitting the electron diffuse scattering data. By default, the value will be set to 0 for all atom types if no value is given following '::'.
WIRE_WIDTH ::	The width of the nanowire (currently the square nanowire is assumed, more shape will be supported in the future if needed) to be used in the RMCProfile running.
WEIGHT_OPTIMIZATION ::	Specify to enable an automated procedure that assigns weights by analyzing statistical correlations between changes in each residual term (corresponding to each data section or constraint) and the total residual. Refer to Appendix-E for more detailed instruction.
XRAY_REAL_SPACE_DATA ::	Introduces a block of data concerning a set of X-ray based real space 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 real space data.* Please be also aware that this part of the program could not be fairly tested yet. Nevertheless as we include it into the code we cannot guarantee it runs successfully. The feedback from the users would also be very appreciated.
XRAY_REAL_SPACE_DUMMY_DATA ::	By providing some key parameters, such as the lower and upper limit of the r-space, etc., the program will generate the simulated X-ray real space pattern (the so-called dummy data). This keyword can be provided together with the real datasets, and then the program will fit those real datasets as well as generating the corresponding simulated pattern.

XRAY_RECIPROCAL_SPACE_DATA ::	Introduces a block of data concerning a set of X-ray 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.* Starting from version 6.7.4, only one X-ray reciprocal space data block is allowed. If users want to separate the whole *Q*-range to multiple sections for fitting, please refer to the subordinate keywords section of current keyword.
XRAY_RECIPROCAL_SPACE_DUMMY_DATA ::	By providing some key parameters, such as the lower and upper limit of the Q-space, etc., the program will generate the simulated X-ray reciprocal space pattern (the so-called dummy data). This keyword can be provided together with the real datasets, and then the program will fit those real datasets as well as generating the corresponding simulated pattern.

[1]Note that the order of the atoms must be the same as the order of atoms in the configuration file.

[2]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.

AVERAGE_COORDINATION_CONSTRAINTS ::

> CAVSTR1 ::	Give, in order, the following parameters: central atom type (e.g. 1), neighbour atom type (e.g 2), the distances between which to calculate the coordination number (e.g. 0 2.0), the desired average coordination number (e.g. 4.0) and the weighting for this constraint (e.g. 0.0001, the smaller this value the stronger the weighting). The "1" refers to the 1st average coordination constraint. Duplicate this subordinate keyword and append the appropriate number (e.g. > CAVSTR2 ::) if you have more than one constraint.

BRAGG ::

> BRAGG_SHAPE ::	Give a text string to denote the type of profile line shape to use. Options are GSAS1, GSAS2, GSAS3, XRAY2, and GSAS1_ABS0, GSAS2_ABS0, GSAS3_ABS0 and XRAY2_ABS0 (the last four includes absorption correction type 0), as defined by the GSAS manual. To use Topas profile, the option will be TOPAS. *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.
> EXTENDED_PEAKS ::	(Optional) Extends peak tails to lower intensity cut-offs.
> FAST_BRAGG ::	(Optional) Activates fast calculations of Bragg profile that utilize Q-multiplicity. Currently, cannot be used with magnetism.
> 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.*
> SCATTERING_LENGTH ::	Give the scattering length (for neutron data) for each atom type, following the order specified in the main '.dat' file. For the X-ray Bragg data, this subordinate keyword does not work. We should either put the coefficients for X-ray scattering for each atom type in the '.xray' file or we don't need to worry about it and RMCProfile will take care of calculating the Q-dependent X-ray scattering factors automatically.

> 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.
> TOPAS_PARAMETER ::	Give the scale factor obtained from Rietveld refinement for Bragg pattern and an integer specifying which xdd data section in Topas refinement will be used here.
> WEIGHT ::	Parameter to weight the contribution of the Bragg profile to the Monte Carlo simulation.
> WEIGHTED_RESIDUAL ::	(Optional) Individual peak intensities are weighted in chi-sq; similar to Rwp used in Rietveld software.

BRAGG_DUMMY ::

> BANK_NO ::	Specifies the bank number for which the dummy data will be generated by providing the number following '::'.
> BRAGG_SHAPE ::	Refer to the same entry under 'BRAGG ::' keyword.
> DMIN ::	Refer to the same entry under 'BRAGG ::' keyword.
> FITTED_SCALE ::	(Optional) Refer to the same entry under 'BRAGG ::' keyword.
> INST_FILE ::	The name of the instrument file should be provided following '::'.
> SCATTERING_LENGTH ::	Refer to the same entry under 'BRAGG ::' keyword.
> SUPERCELL ::	Refer to the same entry under 'BRAGG ::' keyword.
> TOF_DATA ::	Three parameters should be provided following '::' – tof_min tof_max tof_bin, which represents the minimum, maximum and binning for the TOF, respectively. Here it should be noticed that the binning of the TOF should be given with the logrithmic scale.

BVS ::

> ATOM ::	Give the element symbols for each atom in the configuration (in the same order as defined in the configuration file - this is essentially a duplication of the ATOMS :: keyword, but unfortunately is necessary.)
> OXID ::	Give the oxidation states for the atoms in the same order as above.

> WEIGHTS ::	Give appropriate weightings for the bond valence sums for each atom type: smaller numbers give heavier weighting.
> RIJ ::	Give the bond valence parameters for each atom pair in turn (enter 0 for non-bonded pairs). These are ordered in the same way as when giving the coefficients, but in this case you do not include the like-atom pairs. For a 3 atom configuration you'd supply the parameters for pairs 1-2, 1-3 and 2-3, in that order.
> BVAL ::	Give the value of B for each atom pair. This is usually taken as a constant: 0.37. Enter 0 for non-bonded pairs.
> CUTOFF ::	Give the cut-off distances for calculating the bond valence sum for each pair in turn. Enter 0 for non-bonded pairs.
> SAVE ::	Indicate how often you want an update on the status of the bond valence sums to be printed.
> UPDATE ::	Indicate how often you want the neighbour list (which defines which atoms are bonded to each other) to be updated.

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.***

DIFFUSE_SCATTERING ::

> FILENAME ::	Filename containing the data.
> WEIGHT ::	Weight for this dataset in the total residual.
> SAVE ::	(Optional) Save the diffuse scattering fitting result to a separate file.
> NO_AVERAGING ::	(Optional) If included, no averaging over the calculated diffuse scattering patterns related by symmetry will be performed; each pattern will be fitted to experimental data separately.
> FITTED_OFFSET ::	Activates fitting of an intensity offset between the calculated and experimental diffuse-scattering patterns.
> CONSTANT_OFFSET ::	An intensity offset between the calculated and experimental diffuse-scattering patterns will be fixed at a specified value (should be given following '::').
> FITTED_SCALE ::	Activates fitting of a scale factor for the calculated diffusescattering pattern.
> CONSTANT_SCALE ::	A scale factor for the calculated diffuse-scattering will be fixed at a specified value (should be given following '::').

DISTANCE_WINDOW ::

> MNDIST ::	Give the closest approach distances for each atom pair. These are ordered as the partials, so for a 2 atom configuration three distances would be given for pair 1-1, pair 1-2 and pair 2-2 respectively.
> MXDIST ::	Exactly the same as the previous keyword but this time giving the furthest away distances.

EXAFS ::

> FILENAME ::	Filename containing the data.
> FIT_SPACE ::	Acceptable values: r, k (case-insensitive). Specifies whether a given EXAFS dataset if fitted in r- or k-space.
> START_POINT_(R_SPACE) ::	Start point for a fit in r-space (Å).
> END_POINT_(R_SPACE) ::	End point for a fit in r-space (Å).
> R_SPACING ::	r-spacing (Å) to be used in the EXAFS fitting.
> LOW_R_REGION_LIMIT ::	(Optional) The upper limit (in Å) for a lower-r part of EXAFS to be included separately in the fit. If specified, introduces an additional term to the total residual, which describes a contribution from the lower-r part of EXAFS. This subordiante keyword must be provided together with the following subordinate keyword '¿ LOW_R_WEIGHT'.
> LOW_R_WEIGHT	Weight assigned to the lower-r part of EXAFS.
> START_POINT_(K_SPACE) ::	Start point ($Å^{-1}$) of a fit in k-space (if specified in '>FIT_SPACE') or the lower limit of a k-range used in the Fourier transform if the fit is performed in r-space.
> END_POINT_(K_SPACE) ::	End point ($Å^{-1}$) of the fit in k-space (if specified in '>FIT_SPACE') or the upper limit of a k range used in the Fourier transform if the fit is performed in r-space.
> K_POWER ::	Specifies the power of n in the weight-factor k^n applied to an EXAFS signal chi(k) prior to the Fourier transform.
> WEIGHT ::	Weight assigned to a given EXAFS dataset in the total residual.
> ENERGY_OFFSET ::	(Optional) Shift of E0 (eV) in the experimental spectrum. Currently, E0=E0_true/3.807, where E0_true is the desired shift. If not given, E0=0 will be used.
> SCALE_FACTOR ::	(Optional) Scale factor for the experimental spectrum. If not specified, the default scale factor 1 will be used.
> NUM_TYPES_ABSORBING_ATOMS ::	(Optional) Specifies the number of types of absorbing atoms for a given EXAFS dataset. If not included, the number of types will be set to 1.
> TYPE(S)_OF_ABSORBING_ATOMS ::	A list of types of the absorbing atoms for a given EXAFS dataset.

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.*

LEFT_TAILS :: (RIGHT_TAILS ::)

> START_STOP ::	Bounds (Å) for the r-interval over which this tail restraint will be applied. Parameters that should follow '::' is 'rmin rmax'.
> PARTIAL ::	Number of a partial to which the restraint is applied.
> COEFFICIENTS ::	Parameters of a sigmoidal function: $(A_1 - A_2)/(1 + e^{(r-r_0)/\Delta r}) + A_2$.
> WEIGHT ::	Weight assigned to the tails penalty term.

> HARD ::	(Otional) Becomes a hard constraint; the move is rejected if the condition is not met; will override the 'weight' keyword.

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 ::

> BANK_INFO_FILE ::	This keyword should be present if 'INST_RESOLUTION_FILE' is present. The name of the file containing differential cross section for all banks should be provided as the parameter. This file will be used for weighting different banks when merging the resolution matrix corresponding to each bank.
BANK_WEIGHT_TYPE ::	If 'INST_RESOLUTION_FILE' keyword is present, current keyword could be used to specify the scheme to be used for merging resolution matrices corresponding to each bank. A single integer should be provided here as the parameter – '1' for weighting with standard deviation, '2' for weighting with inverse of standard deviation and '3' for weighting with intensity. If current keyword is not given, the value will be defaulted to option '3'.
> BROADENING_CORRECTION ::	A single value should follow to specify the parameter concerning the broadening effect in real-space, accounting for the increasing noise level in high Q region. The specified parameter is quite similar with (but not identical to) the *qbroad* parameter in PDFgui. Refer to Appendix-F for details.
> DATA_TYPE ::	Gives the type of data; options are `G(r)`, `T(r)`, `D(r)`, `G'(r)` or `G(r)P` (used in PDFGetX3 and PDFGui programs), 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.*
> CONVOLVE ::	This subordinate keyword tells RMCProfile to convolve a sinc function into the calculated PDF pattern to account for the finite Q-range effect. A parameter should be provided following '::' to specify the effect Q-range. To keep consistence with experimental data, this value is better to be identical to the actual Q-range used for Fourier transforming the $S(Q)$ data to obtain the PDF.
> END_POINT ::	End point of the data (an integer).

> END_POINT ::	RMCProfile6 can now run in dynamic mode and automatically extend the range of the **NEUTRON_REAL_SPACE_DATA**. If four values are provided those would have the following meaning: initial end point, time of the calclation of initial range, extension step in points and time intervals between following extinctions in minutes. For example the keyword END_POINT :: 250 180 100 120 would start the refinement with the range of 250 points and run for 180 minutes. Next, it will be extended by 100 points every 120 min.
> FILENAME ::	Filename containing the data.
> FIT_TYPE ::	Gives the function that is fitted; options are G(r), T(r), D(r), G′(r) or G(r)P (used in PDFGetX3 and PDFGui programs), 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 4.3.5.*
> 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.
> INST_RESOLUTION_FILE ::	If specified, the name of instrument resolution file should be provided. This file should be prepared through fitting the Bragg pattern of a standard sample. Users need to be aware of that we have two different methods for correcting the resolution effect – one based on GSAS profile and the other based on Topas profile. The instrument file provided here is therefore different in between those two methods. Refer to Appendix-F for more details.
> 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 4.2).*[1]
> 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.*
> RESOLUTION_CORRECTION ::	A single value should follow to specify the exponent accounting for the dampening in real-space (which fundamentally comes from the finite resolution effect in *Q*-space), in the form of $exp(-r * Expo)$.
> R_CUTOFF ::	As the result of convolving with a sinc function (see the 'CONVOLVE ::' subordinate keyword above), the PDF pattern will contain a lot of Fourier ripples. Here, the cutoff parameter provided following '::' specifies that the region below the cutoff in r-space will be set to the baseline, to get rid of the Fourier ripples. This subordinate keyword does not have be specified. If not given, the value will be set to be equal to the non-zero minimum among those values following the 'MINIMUM_DISTANCES ::' keyword.
> START_POINT ::	Start point of the data (an integer).
> STOG ::	If specified, the data file was generated by the STOG program for converting total scattering data to PDF data.
> WEIGHT ::	Parameter that weights the data in the simulation.

NEUTRON_REAL_SPACE_DUMMY_DATA ::

> BROADENING_CORRECTION ::	A single value should follow to specify the parameter concerning the broadening effect in real-space, accounting for the increasing noise level in high *Q* region. The specified parameter is quite similar with (but not identical to) the *qbroad* parameter in PDFgui. Refer to Appendix-F for details.
> DATA_TYPE ::	Specifies the type of the generated dummy data. The acceptable value is the same with that under 'NEUTRON_REAL_SPACE_DATA ::' keyword.
> NEUTRON_COEFFICIENTS ::	Refer to the same entry under 'NEUTRON_REAL_SPACE_DATA ::' keyword.
> RESOLUTION_CORRECTION ::	A single value should follow to specify the exponent accounting for the dampening in real-space (which fundamentally comes from the finite resolution effect in *Q*-space), in the form of $exp(-r * Expo)$.

> RMAX ::	Specifies the upper limit of the r-space in which the dummy data will be generated.
> RMIN ::	Specifies the lower limit of the r-space in which the dummy data will be generated.

NEUTRON_RECIPROCAL_SPACE_DATA ::

> BANK_INFO_FILE ::	This keyword should be present if 'INST_RESOLUTION_FILE' is present. The name of the file containing differential cross section for all banks should be provided as the parameter. This file will be used for weighting different banks when merging the resolution matrix corresponding to each bank.
BANK_WEIGHT_TYPE ::	If 'INST_RESOLUTION_FILE' keyword is present, current keyword could be used to specify the scheme to be used for merging resolution matrices corresponding to each bank. A single integer should be provided here as the parameter – '1' for weighting with standard deviation, '2' for weighting with inverse of standard deviation and '3' for weighting with intensity. If current keyword is not given, the value will be defaulted to option '3'.
> BROADENING_CORRECTION ::	In accordance with the corresponding to correction in real-space, we also need to implement the correction in reciprocal space. Refer to the same subordinate keyword under neutron real space data section.
> 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 4.3.5.*
> 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.
> INST_RESOLUTION_FILE ::	If specified, the name of instrument resolution file should be provided. This file should be prepared through fitting the Bragg pattern of a standard sample. Users need to be aware of that we have two different methods for correcting the resolution effect – one based on GSAS profile and the other based on Topas profile. The instrument file provided here is therefore different in between those two methods. Refer to Appendix-F for more details.
> 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 4.2).*[1]

> 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.*
> RESOLUTION_CORRECTION ::	In accordance with the corresponding to correction in real-space, we also need to implement the correction in reciprocal space. Refer to the same subordinate keyword under neutron real space data section.
> SQ_CORRECTION ::	Filename for a Q-space resolution-correction matrix. The file name should be provided following '::'.
> SQ_CORRECTION_M_DIM ::	Dimensions of the Q-space resolution correction matrix (N$\times$M). N and M, which should be provided following '::', are the numbers of points in r- and Q-spaces, respectively. N must be equal to or greater than the number of points in the histogram, whereas M must be equal to or greater than the number of points in the S(Q).
> START_POINT ::	Start point of the data (provided as an integer).
> STOG ::	If specified, the data file was generated by the STOG program for converting total scattering data to PDF data.
> WEIGHT ::	Parameter that weights the data in the simulation.

NEUTRON_RECIPROCAL_SPACE_DUMMY_DATA ::

> BROADENING_CORRECTION ::	In accordance with the corresponding to correction in real-space, we also need to implement the correction in reciprocal space. Refer to the same subordinate keyword under neutron real space data section.
> CONVOLVE ::	Refer to the same entry under the 'NEUTRON_RECIPROCAL_SPACE_DATA ::' keyword.
> DATA_TYPE ::	Specifies the type of the generated dummy data. The acceptable value is the same with that under 'NEUTRON_RECIPROCAL_SPACE_DATA ::' keyword.

> NEUTRON_COEFFICIENTS ::	Refer to the same entry under 'NEUTRON_RECIPROCAL_SPACE_DATA ::' keyword.
> QBIN ::	Specifies the binning of the Q-space for the dummy data generation. If not provided, the default bin 0.02 will be used.
> QMAX ::	Specifies the upper limit of the Q-space in which the dummy data will be generated.
> QMIN ::	Specifies the lower limit of the Q-space in which the dummy data will be generated.

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_HISTOGRAM ::	Requests the generation of bond-angle histograms, in steps of bond angle and the cosine of the bond angle. The required step can be selected by using the angle = <value> option.
> 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>,(the US spelling color will also work) 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).[2]

`> 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 ::`

`> 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.[3]

`> 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.*

`SWAP_MULTI ::`

`> SWAP_ATOMS ::`

List the three numbers to provide information of what types of atoms to be swapped, followed by the swap probability.[3]. The `> SWAP_ATOMS ::` keyword should be used multiple times in order to obtain swaps between several atom types. For example `> SWAP_ATOMS :: 1 2 0.1` and `> SWAP_ATOMS :: 2 3 0.2` is used to enable swapping between atom types 1 and 2 with the probability of 0.1 and between atom types 2 and 3 with the probability of 0.2.

XRAY_REAL_SPACE_DATA ::

> BROADENING_CORRECTION ::	Refer to the same subordinate keyword under neutron real space data section.
> DATA_TYPE ::	Gives the type of data; options are G(r), T(r), D(r) or G′(r).
> 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).
> FITTED_OFFSET	If specified, this instructs the program to fit the offset value provided for this set of data. *Default is not to fit.*
> NEUTRON_COEFFICIENTS ::	This is for calculating the X-ray PDF in an alternative way. No need to be provided. In fact, this is not the real neutron scattering coefficients. Instead, it should provide weightings that are calculated using Z in place of neutron scattering lengths, and then normalised to 1 (this is explained in more detail in Appendix D).
> 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.*
> RESOLUTION_CORRECTION ::	Refer to the same subordinate keyword under neutron real space data section.
> START_POINT ::	Start point of the data (an integer).
> WEIGHT ::	Parameter that weights the data in the simulation.
> QMIN ::	Minimum value of Q used to obtain real space X-Ray data.
> QMAX ::	Maximum value of Q used to obtain real space X-Ray data.
> QSTEP ::	Step size of Q in reciprocal space used to obtain real space X-Ray data.

> NORMALIZATION_TYPE ::	The normalization for X-ray dataset, which can be provided as <f^2> or <f>^2. The default value will be <f^2> if this subordinate keyword is not given.

XRAY_REAL_SPACE_DUMMY_DATA ::

> BROADENING_CORRECTION ::	Refer to the same subordinate keyword under neutron real space data section.
> DATA_TYPE ::	Refer to the same entry under the 'XRAY_REAL_SPACE_DATA ::' keyword.
> QMAX ::	Specifies the upper limit of the Q-space where the calculated histogram will be temporarily Fouriere transformed into. The reason why we need to specify this is that we need to transform the histogram into the Q-space and then take into account of the Q-dependent of the scattering form factor for X-ray. Then the pattern in Q-space will be transformed back into r-space to get the X-ray real space pattern.
> QMIN ::	Specifies the lower limit of the Q-space where the calculated histogram will be temporarily Fouriere transformed into.
> QSTEP ::	Specifies the binning of the Q-space where the calculated histogram will be temporarily Fouriere transformed into. If not provided, the default binning 0.02 will be used.
> RMAX ::	Specifies the upper limit of the r-space in which the dummy data will be generated.
> RMIN ::	Specifies the lower limit of the r-space in which the dummy data will be generated.
> NORMALISATION_TYPE ::	Refer to the same entry under the 'XRAY_REAL_SPACE_DATA ::' keyword.
> RESOLUTION_CORRECTION ::	Refer to the same subordinate keyword under neutron real space data section.
> XRAY_COEFFICIENTS ::	If this keyword is to be used, the users are supposed to provide the following value 'Z1*Z2*frac_species(ia1)*frac_species(ia2)' for each partial, where Z1 and Z2 represents the atomic number of the two atoms involved in the pair and 'frac_species(ia1)' and 'frac_species(ia2)' refer to the overall fraction of each species.

XRAY_RECIPROCAL_SPACE_DATA ::

> BROADENING_CORRECTION ::	Refer to the same subordinate keyword under neutron reciprocal space data section.
> 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.*
> DUMMY_REAL_SPACE_FIT ::	Specifies to calculate the dummy X-ray real space data.
> DUMMY_REAL_SPACE_PARAMETERS ::	Specifies the starting and ending point for the dummy X-ray real space data calculation, together with the corresponding weight. When we want to calculate dummy X-ray real space in various r- sections, just specify current sub-keyword multiple times.
> 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.*
> FORM_FACTOR_IN_DATA_FILE	Instructs the program to read in the Q-dependent form factor from the F(Q) data file. Those form factors should be provided by users in the F(Q) data file as columns to the right of data columns, in the order of atomic species as specified following the 'ATOMS' keyword.
> INST_RESOLUTION_FILE ::	If specified, the name of instrument resolution file should be provided. This file should be prepared through fitting the Bragg pattern of a standard sample. Refer to Appendix-F for more details.

> 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.*
> NORMALIZATION_TYPE ::	The normalization for X-ray dataset, which can be provided as <f^2> or <f>^2. The default value will be <f^2> if this subordinate keyword is not given.
> NORM_EXP_X ::	The exponent specifying the resolution correction, similar to that used for the neutron data.
> REAL_SPACE_FIT ::	Specifies that we want to fit the X-ray data in real space as well. Here it should be mentioned that the Fourier transform from the reciprocal space to the real space now is done internally in RMCProfile. Therefore users don't need to provide the X-ray real space datasets. Three parameters should be given following '::' – Starting and ending points (integer for both) in real space together with the number of sections (refer to next following sub-keyword), in the order they appear here. The corresponding output for X-ray real space fit then can be found in the last section of the '.out' file.
> REAL_SPACE_PARAMETERS ::	The principle here is we can separate the real space into different sections for the fitting (if it is indeed specified to refine in real space). For each section, we have three parameters – the starting point (integer), the ending point (integer) and the corresponding weight. All the three parameters should be provided following '::'. When we want to fit in more than one sections, just specify current subordinate together with the corresponding three parameters multiple times.
> RECIPROCAL_SPACE_FIT ::	Specifies the starting and ending points together with the number of sections we want to divide the whole data range into. All three parameters should be given following '::' as integers in the order - starting_point ending_point number_of_sections.

> RECIPROCAL_SPACE_PARAMETERS ::	This subordinate keyword specifies the parameters for each data section in reciprocal space. For each section, we have three parameters – the starting point (integer), the ending point (integer) and the corresponding weight. All the three parameters should be provided following '::'. When we want to fit in more than one sections, just specify current subordinate together with the corresponding three parameters multiple times.
> RESOLUTION_CORRECTION ::	Refer to the same subordinate keyword under neutron reciprocal space data section.

XRAY_RECIPROCAL_SPACE_DUMMY_DATA ::

> BROADENING_CORRECTION ::	Refer to the same subordinate keyword under neutron reciprocal space data section.
> DATA_TYPE ::	Refer to the same entry under the 'XRAY_RECIPROCAL_SPACE_DATA ::' keyword.
> QMAX ::	Specifies the upper limit of the Q-space in which the dummy data will be generated.
> QMIN ::	Specifies the lower limit of the Q-space in which the dummy data will be generated.
> QBIN ::	Specifies the binning of the Q-space in which the dummy data will be generated. If not provided, the default binning 0.02 will be used.
> NORMALISATION_TYPE ::	Refer to the same entry under the 'XRAY_RECIPROCAL_SPACE_DATA ::' keyword.
> RESOLUTION_CORRECTION ::	Refer to the same subordinate keyword under neutron reciprocal space data section.

[1]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.
[2]Additional units can be made available on request, but internally the program works in terms of kJ/mol.
[3]Note that the order of atoms is set by the configuration file.
[4]'Chimin' — the minimum desired value of chi^2 for a given dataset. Can be estimated, for example, according to statistical-noise errors. 'Dermax' — the maximum value of the chi^2 derivative for a given dataset. If set at zero, the weights will be adjusted at each save of a configuration to prevent a prolonged increase of chi^2. Setting this value to a negative number will cause gradual readjustment of weights so that the chi^2 for this dataset will decrease faster.

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.

4.2 Neutron and X-ray 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. However, the default units used by `RMCProfile` require values smaller by a factor of 10. Thus if the neutron coefficients are given in the input file, the example line would be

```
NEUTRON_COEFFICIENTS :: 0.01233 0.00346 0.05535 0.07795 0.00024
       0.00776 0.01092 0.06211 0.17496 0.12320
```

These values are calculated as follows: the first, referring to Ag-Ag pairings, is equal to $(0.5922 \times 3/16)^2$, and the second, referring to Ag-Co pairings, is equal to $2[(0.5922 \times 3/16) \times (0.249 \times 1/16)]$. It is important to remember that coefficients for unlike pairs must be multiplied by two. Note that the list of values is allowed to span more than one line. Also note that you can provide coefficients in fm^2 rather than $10^{-28}\,m^2$, as shown here, as long as your data are similarly scaled.

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).

4.3 Using experimental data

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.

4.3.1 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:[1]

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

which has limiting values

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

4.3.2 Neutron scattering function

The scattering function and the PDF $G(r)$ are related by[2]

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

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

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

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[3]

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

which has limiting values

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

4.3.3 Alternative forms of the pair distribution function

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

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

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

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

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

and

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

4.3.4 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

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

with limiting cases[4]

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

[4]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 $S_{\text{norm}}(Q \sim 0)$

4.3.5 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`'.[5]

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

Function	Input text	Eqn in Keen	Eqn here	Limiting values
$i(Q)$	`i(Q)`	25	4.6	$-\sum_j c_j b_j^2 \rightarrow 0$
$F(Q)$	`F(Q)`	11	4.6	$-\sum_j c_j b_j^2 \rightarrow 0$
$S(Q)$	`S(Q)`	–	4.9	$-\sum_j c_j b_j^2 + (\sum_j c_j b_j)^2 + \rightarrow (\sum_j c_j b_j)^2$
$Qi(Q)$	`Qi(Q)`	–	–	$-\sum_j c_j b_j^2 Q \rightarrow 0$
$QF(Q)$	`QF(Q)`	–	–	$-\sum_j c_j b_j^2 Q \rightarrow 0$
$i_{\mathrm{norm}}(Q)$	`i(Q) normalised`	–	4.19	$-1 \rightarrow 0$
$F_{\mathrm{norm}}(Q)$	`F(Q) normalised`	–	4.19	$-1 \rightarrow 0$
$S_{\mathrm{norm}}(Q)$	`S(Q) normalised`	19	4.19	$1 - \sum_j c_j b_j^2 / (\sum_j c_j b_j)^2 \rightarrow +1$
$G(r)$	`G(r)`	10	4.2,4.7	$-(\sum_j c_j b_j)^2 \rightarrow 0$
$G'(r)$	`G'(r)`	–	4.4	$0 \rightarrow +(\sum_j c_j b_j)^2$
$D(r)$	`D(r)`	26	4.11	$-4\pi\rho(\sum_j c_j b_j)^2 r \rightarrow 0$
$T(r)$	`T(r)`	27	4.12	$0 \rightarrow +4\pi\rho(\sum_j c_j b_j)^2 r$
$G_{\mathrm{norm}}(r)$	`G(r) normalised`	–	4.15	$-1 \rightarrow 0$
$G'_{\mathrm{norm}}(r)$	`G'(r) normalised`	16	4.15	$0 \rightarrow +1$
$D_{\mathrm{norm}}(r)$	`D(r) normalised`	–	4.16	$-r \rightarrow 0$
$T_{\mathrm{norm}}(r)$	`T(r) normalised`	–	4.16	$0 \rightarrow +r$

4.4 Using magnetism

You may wish to include magnetism in your refinement for several reasons. Firstly, you may wish to find out more about the magnetic structure of your material, and allow a greater degree of freedom than other techniques may allow. Alternatively you may be mainly interested in the nuclear structure of your material, and include the magnetism just to ensure that the relevant peaks in your data are

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

properly fitted. Finally, you might be interested in both the positions of the atoms and their magnetic moments. RMCProfile should be able to help you in each of these cases, just choose what aspects of your structure you wish to refine. [6]

A couple of caveats regarding the use of magnetism in your refinements, which may be obvious. Firstly, the magnetic scattering is only included for neutron data sets, so don't expect the magnetic contribution to appear in x-ray data refinements. Secondly, magnetic correlations are not included in any fits to PDF data, because of complications that arise when Fourier transforming magnetic data. The other side to this is that if you transform your experimental data into real space, there will be contributions from both the nuclear and magnetic scattering, which can make accurate fitting difficult. In some cases it may be best to avoid the use of real space functions if the magnetic scattering cannot be separated from the nuclear component.

4.4.1 The main data file

RMCProfile is instructed to include magnetic scattering by inserting the appropriate text into the version 6 **.dat** file. An example is shown below:

```
MAGNETISM ::
  > MAGNETISM_FILE_STEM :: fe3o4_spin
  > MAGNETIC_ATOMS :: 2
  > FORM_FACTOR :: 1 0.3972 13.2443 0.6295 4.9034 0.0314 0.3496 0.0044
  > FORM_FACTOR :: 2 0.3972 13.2443 0.6295 4.9034 0.0314 0.3496 0.0044
  > MAGNETIC_MOMENTS :: 4.625 4.130
  > MAX_SPIN_MOVEMENT :: 0.05
  > SPIN_MOVE_RATE :: 0.5
  > NO_VARY_SPIN_MOVE_RATE
```

Here the MAGNETISM :: major keyword indicates the beginning of the magnetism section. As with other sections, the order of the following subordinate keywords is arbitrary, and left to the whim of the user. The keywords available are described above in Section 4.1.4, but we shall discuss them here as well to avoid excessive scrolling. Taking the keywords in the order listed above, we start with > MAGNETISM_FILE_STEM :: fe3o4_spin. This instructs the program to look for a file called **fe3o4_spin.cfg** containing the spin configuration (on which we shall say more later), and to name any associated magnetism output files in that way. The > MAGNETIC_ATOMS :: 2 line tells the program that there are two magnetic atom types in the structure. It is assumed that these are the first two atom types listed in the main configuration file. If this is not the case then you will have to re-order your main configuration file!

The coefficients used to calculate the magnetic form factors from equation 2.5 are entered using the lines > FORM_FACTOR :: n A a B b C c D. This is required for every magnetic atom type. Alternatively the > FORM_FACTOR_FILE :: <name.file> keyword must be used, and an appropriate file included in the directory. The magnitudes of the magnetic moments for each atom type are entered using the > MAGNETIC_MOMENTS :: ... keyword. The numbers should be entered in the correct order.

[6] Thank you to Callum Young for writing this section.

The > MAX_SPIN_MOVEMENT :: line gives the value of σ_{max} described in Section 2.10.2; a small value $\lesssim$0.1 is appropriate. > SPIN_MOVE_RATE :: describes the fraction of moves generated that will change the spin configuration. For example, a spin move rate of 0.0 will result in no spin moves being generated, and the magnetic structure will remain unchanged. Alternatively, a spin move rate of 1.0 (assuming no swap moves are being used) will result in only magnetic moves being produced.[7] RMCProfile has the facility for the spin move rate to be automatically varied during the course of the refinement.[8] This is implemented using the > VARY_SPIN_MOVE_RATE keyword. The default is not to vary the rate, but this can be explicitly included using the > NO_VARY_SPIN_MOVE_RATE keyword.

4.4.2 The spin configuration file

The starting spin configuration is contained within one file, with the name **MAGNETISM_FILE_STEM.cfg**. This has a fairly simple structure, much like the 'classic' version 3 files. An example is shown below:

```
0.095105            576                   2
192                     384
0.00000000       0.00000000     1.00000000
0.00000000       0.00000000     1.00000000
0.00000000       0.00000000     1.00000000
...
```

The first line contains: the atomic number density of the system (here 0.095105); the total number of magnetic atoms, and hence the number of entries in this file (576); and the number of different magnetic atom types (2). The following line contains the number of atoms of each type present. These should be listed in the correct order, and should sum to give the total number of magnetic atoms (here, 192 + 384 = 576)! There then follows a list of all the magnetic spin vectors in the structure. These are unit vectors and should correspond to the atom positions listed in the main configuration file, *i.e.* the *n*th vector listed in the **compound_spin.cfg** file should correspond to the atom whose coordinates are listed in the *n*th entry in the main **compound.cfg** / **compound.rmc6f** file.

The spin configuration file, like the standard configuration file, is updated during the course of the refinement, so if you wish to keep track of your starting configuration, we suggest you keep a copy of this file separate.

4.4.3 Generated files

RMCProfile produces three new files when refining a magnetic structure. These are **mag_stem_name.his**, **mag_stem_name.ff** and **mag_stem_name.braggff**. The **.his** file performs much the same function as the equivalent histogram file that accompanies the main configuration file. The **.braggff** file lists the magnetic form factors for each atom type over the correct *Q*-range for

[7] The generation rate of 'standard' translational moves is unspecified, and will take up the remaining fraction once magnetic and swap moves have been accounted for. Common sense will tell you that the total fraction of generated moves specified by the user (spin + swap) should not exceed 1.0.

[8] This feature has not been used by the current author, but by all means try it.

the Bragg data set, while the **.ff** file lists the magnetic form factors in a suitable format for the other data sets. These will likely play little role in any analysis performed: the final spin configuration and the fits to data will be of more use.

As outlined in Section 2.10.4,the program ATOMEYE can be used to help view the refined structure. PYMOL can also be of use.

4.5 Using the Bragg profile

When the Bragg option is specified, `RMCProfile` extracts the necessary information about background and lineshape functions from a GSAS refinement. You can of course choose a different Rietveld package for your refinements, but in this case you will need to put it in GSAS format at the end in order for `RMCProfile` to read it.

In the course of an RMC refinement, `RMCProfile` calculates Bragg peak intensities and adjusts the configuration to match the measured intensities.

For more information about using Bragg data, please see section 4.13.

4.6 Using EXAFS data

The technical background for EXAFS can be found in the Appendix-C To Appendix-F in the RMCProfile tutorial documentation which comes together with current manual in the main RMCProfile distribution directory. Also the users can refer to the documentation at `http://www.rmcprofile.org/imagesFhj/5/51/Rmcprofile_exafs_manual.pdf` for more information.

4.7 Using potentials

As discussed in section 2.6, using well-parameterised interatomic potentials judiciously can play a role when the short-distance part of the pair distribution function is dominated by intramolecular distances that are not of primary interest in a a study.

4.7.1 Bond-stretching potential

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

$$E_M = D[1 - \exp(-\alpha(r - r_0))]^2 \tag{4.21}$$

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:

$$E_{\mathrm{M}} = D\alpha^2 (r - r_0)^2 \tag{4.22}$$

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

$$E_{\mathrm{M}} = \frac{1}{2} k (r - r_0)^2 \tag{4.23}$$

the association between k, D and α is clear. 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. A set of recommended values for the parameters, taken from the MM3 and MM3 protein databases are given in Table B.1 and Table B.2 respectively in Appendix B.

4.7.2 Bond angle potentials

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

$$E = \frac{1}{2} K (\cos\theta - \cos\theta_0)^2 \tag{4.24}$$

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

$$\begin{aligned} E &= 2K \sin^2 \tfrac{\theta+\theta_0}{2} \sin^2 \tfrac{\theta-\theta_0}{2} \\ &\approx \tfrac{1}{2} K \sin^2\theta_0 (\theta - \theta_0)^2 \end{aligned} \tag{4.25}$$

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 given in Table B.3 and Table B.4 taken from the MM3 and MM3 protein databases respectively in Appendix B.

4.7.3 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, zrotation=45 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.[9] 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.

4.7.4 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:[10]

[9] 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.

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

```
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
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.

4.7.5 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:[11]

```
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
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:

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

```
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.

4.7.6 Visualisation of bond orientation distribution functions

`RMCProfile` allows for easy plotting of bond orientation distribution functions.

4.7.6a 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.[12] 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.

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

Figure 4.1: Example of a stereographic representation of the bond orientation function generated by `RMCProfile`.

4.7.6b 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`.[13] 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`.[14] To produce the image from `Imagemagick`, execute the following shell command:

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

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.

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

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

Other graphics program, such as `GraphicConverter`[15] for Mac OS X, and Adobe's Photoshop, are able to read and manipulate files in the PPM format.

4.7.6c 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;
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.

4.7.7 Generation of spherical harmonic function averages

RMCProfile automatically computes the mean-value and mean-squared value (and hence the variance) on values of the spherical harmonics that describe the orientation of a bond. We convert the complex numbers to real equivalents by combining the real and imaginary components as

$$y_\ell(m) = \begin{cases} \frac{1}{\sqrt{2}}\left(Y_\ell^m + (-1)^m Y_\ell^{-m}\right) & \text{if } m > 0 \\ Y_\ell^0 & \text{if } m = 0 \\ \frac{-i}{\sqrt{2}}\left(Y_\ell^m - (-1)^m Y_\ell^{-m}\right) & \text{if } m < 0 \end{cases} \tag{4.26}$$

Definitions were taken from http://en.wikipedia.org/wiki/Table_of_spherical_harmonics but slightly rearranged in a logical order.

[15] http://www.lemkesoft.com/

Real spherical harmonics with $\ell = 1$

$$y_1(-1) = i\sqrt{\frac{1}{2}}\left(Y_1^{-1} + Y_1^1\right) = \sqrt{\frac{3}{4\pi}} \cdot \frac{y}{r} \tag{4.27}$$

$$y_1(0) = Y_1^0 = \sqrt{\frac{3}{4\pi}} \cdot \frac{z}{r} \tag{4.28}$$

$$y_1(1) = \sqrt{\frac{1}{2}}\left(Y_1^{-1} - Y_1^1\right) = \sqrt{\frac{3}{4\pi}} \cdot \frac{x}{r} \tag{4.29}$$

Real spherical harmonics with $\ell = 2$

$$y_2(-2) = i\sqrt{\frac{1}{2}}\left(Y_2^{-2} - Y_2^2\right) = \frac{1}{2}\sqrt{\frac{15}{\pi}} \cdot \frac{xy}{r^2} \tag{4.30}$$

$$y_2(-1) = i\sqrt{\frac{1}{2}}\left(Y_2^{-1} + Y_2^1\right) = \frac{1}{2}\sqrt{\frac{15}{\pi}} \cdot \frac{yz}{r^2} \tag{4.31}$$

$$y_2(0) = Y_2^0 = \frac{1}{4}\sqrt{\frac{5}{\pi}} \cdot \frac{2z^2 - x^2 - y^2}{r^2} \tag{4.32}$$

$$y_2(1) = \sqrt{\frac{1}{2}}\left(Y_2^{-1} - Y_2^1\right) = \frac{1}{2}\sqrt{\frac{15}{\pi}} \cdot \frac{zx}{r^2} \tag{4.33}$$

$$y_2(2) = \sqrt{\frac{1}{2}}\left(Y_2^{-2} + Y_2^2\right) = \frac{1}{4}\sqrt{\frac{15}{\pi}} \cdot \frac{x^2 - y^2}{r^2} \tag{4.34}$$

Real spherical harmonics with $\ell = 3$

$$y_3(-3) = i\sqrt{\frac{1}{2}}\left(Y_3^{-3} + Y_3^3\right) = \frac{1}{4}\sqrt{\frac{35}{2\pi}} \cdot \frac{(3x^2 - y^2)\,y}{r^3} \tag{4.35}$$

$$y_3(-2) = i\sqrt{\frac{1}{2}}\left(Y_3^{-2} - Y_3^2\right) = \frac{1}{2}\sqrt{\frac{105}{\pi}} \cdot \frac{xyz}{r^3} \tag{4.36}$$

$$y_3(-1) = i\sqrt{\frac{1}{2}}\left(Y_3^{-1} + Y_3^1\right) = \frac{1}{4}\sqrt{\frac{21}{2\pi}} \cdot \frac{y(4z^2 - x^2 - y^2)}{r^3} \tag{4.37}$$

$$y_3(0) = Y_3^0 = \frac{1}{4}\sqrt{\frac{7}{\pi}} \cdot \frac{z(2z^2 - 3x^2 - 3y^2)}{r^3} \tag{4.38}$$

$$y_3(1) = \sqrt{\frac{1}{2}}\left(Y_3^{-1} - Y_3^1\right) = \frac{1}{4}\sqrt{\frac{21}{2\pi}} \cdot \frac{x(4z^2 - x^2 - y^2)}{r^3} \tag{4.39}$$

$$y_3(2) = \sqrt{\frac{1}{2}}\left(Y_3^{-2} + Y_3^2\right) = \frac{1}{4}\sqrt{\frac{105}{\pi}} \cdot \frac{\left(x^2 - y^2\right) z}{r^3} \tag{4.40}$$

$$y_3(3) = \sqrt{\frac{1}{2}}\left(Y_3^{-3} - Y_3^3\right) = \frac{1}{4}\sqrt{\frac{35}{2\pi}} \cdot \frac{\left(x^2 - 3y^2\right) x}{r^3} \tag{4.41}$$

Real spherical harmonics with $\ell = 4$

$$y_4(-4) = i\sqrt{\frac{1}{2}}\left(Y_4^{-4} - Y_4^4\right) = \frac{3}{4}\sqrt{\frac{35}{\pi}} \cdot \frac{xy\left(x^2 - y^2\right)}{r^4} \tag{4.42}$$

$$y_4(-3) = i\sqrt{\frac{1}{2}}\left(Y_4^{-3} + Y_4^3\right) = \frac{3}{4}\sqrt{\frac{35}{2\pi}} \cdot \frac{(3x^2 - y^2)yz}{r^4} \tag{4.43}$$

$$y_4(-2) = i\sqrt{\frac{1}{2}}\left(Y_4^{-2} - Y_4^2\right) = \frac{3}{4}\sqrt{\frac{5}{\pi}} \cdot \frac{xy \cdot (7z^2 - r^2)}{r^4} \tag{4.44}$$

$$y_4(-1) = i\sqrt{\frac{1}{2}}\left(Y_4^{-1} + Y_4^1\right) = \frac{3}{4}\sqrt{\frac{5}{2\pi}} \cdot \frac{yz \cdot (7z^2 - 3r^2)}{r^4} \tag{4.45}$$

$$y_4(0) = Y_4^0 = \frac{3}{16}\sqrt{\frac{1}{\pi}} \cdot \frac{35z^4 - 30z^2r^2 + 3r^4}{r^4} \tag{4.46}$$

$$y_4(1) = \sqrt{\frac{1}{2}}\left(Y_4^{-1} - Y_4^1\right) = \frac{3}{4}\sqrt{\frac{5}{2\pi}} \cdot \frac{xz \cdot (7z^2 - 3r^2)}{r^4} \tag{4.47}$$

$$y_4(2) = \sqrt{\frac{1}{2}}\left(Y_4^{-2} + Y_4^2\right) = \frac{3}{8}\sqrt{\frac{5}{\pi}} \cdot \frac{(x^2 - y^2) \cdot (7z^2 - r^2)}{r^4} \tag{4.48}$$

$$y_4(3) = \sqrt{\frac{1}{2}}\left(Y_4^{-3} - Y_4^3\right) = \frac{3}{4}\sqrt{\frac{35}{2\pi}} \cdot \frac{(x^2 - 3y^2)xz}{r^4} \tag{4.49}$$

$$y_4(4) = \sqrt{\frac{1}{2}}\left(Y_4^{-4} + Y_4^4\right) = \frac{3}{16}\sqrt{\frac{35}{\pi}} \cdot \frac{x^2\left(x^2 - 3y^2\right) - y^2\left(3x^2 - y^2\right)}{r^4} \tag{4.50}$$

These quantities are averaged over each bond for any given bond type, and also averaged over steps. The resultant averages, namely $\langle Y_\ell(m)\rangle$ and $\left\langle Y_\ell^2(m)\right\rangle$, are printed in a file with name **`<file>.ylm`**.

4.7.8 Generation of Kubic harmonic function averages

If `rmcprofile` deduces that the RMC configuration is likely to be of a cubic lattice type, it will calculate the averages of the Kubic harmonics, $K_\ell(\Omega)$, of the appropriate symmetries for the molecular bonds, where Ω represents the polar coordinates describing the orientation of a bond.

The Kubic harmonics calculated are the following, where $Q = x^4 + y^4 + z^4$ and $S = x^2y^2z^2$:

$$K_0(\Omega) = 1 \tag{4.51}$$

$$K_4(\Omega) = \frac{1}{4}\sqrt{21}\,(5Q - 3) \tag{4.52}$$

$$K_6(\Omega) = \frac{1}{8}\sqrt{13/2}\,(462S + 21Q - 17) \tag{4.53}$$

$$K_8(\Omega) = \frac{1}{32}\sqrt{561}\left(65Q^2 - 208S - 94Q + 33\right) \tag{4.54}$$

$$K_{10}(\Omega) = \frac{1}{64}\sqrt{455/2}\left(7106QS + 187Q^2 - 3190S - 264Q + 85\right) \tag{4.55}$$

The value of the Kubic harmonics is that the bond orientation distribution function for a molecule located at a site of octahedral symmetry (point groups O_h or $m\bar{3}m$) can be expressed as an expansion in these functions:

$$P(\Omega) = \frac{1}{4\pi}\sum_\ell c_\ell K_\ell(\Omega) \tag{4.56}$$

where only even terms in ℓ are allowed by symmetry, and

$$c_\ell = \langle K_\ell(\Omega)\rangle \tag{4.57}$$

As with the spherical harmonic quantities above, values of the Kubic harmonics are averaged over each bond for any given bond type, and also averaged over steps. The resultant averages, namely $\langle Y_\ell(m)\rangle$ and $\langle Y_\ell^2\rangle$, are printed in the file with name **`<file>.ylm`**.

4.8 Using Bond valence sum

To access this functionality you will need to provide the `BVS::` block of keywords, as described in section 4.1.2 in the `.dat` file. An example is given below.

```
BVS ::
```

```
> ATOM :: Y Zr O
> OXID :: 3 4 -2
> WEIGHTS :: 0.055 0.055 0.140
> RIJ :: 0 2.019 1.928
> BVAL :: 0 0.37 0.37
> CUTOFF ::  0 3.2 3.2
> SAVE :: 400000
> UPDATE :: 200000
```

The first two subordinate keyword lines define your atom types and their oxidation states. These must be in the same order as specified in the **.dat** file. The next line gives the weights of the constraint for each atom type. The smaller the number, the stronger the constraint.

The subsequent lines give the relevant bond valence parameters for your atom pairs, as per Brese and O'Keefe, Acta Cryst B, **47**, 192-197 (1991). The ordering of the pairs should be, for an example 3 atom configuration, given as 1-2, 1-3, 2-3. Note that unlike when listing coefficients in the main **.dat** file, here you do not include like atom pairs. Enter zeroes for non-bonded pairs.

The final lines determine how often you want an update on the status of the BVS constraint and how often the neighbour list is recalculated.

While we recommend the keyword method for simplicity, it is also possible to provide this information via a text file with the same stem name as your **.dat** file and the extension **.bvs**. A file for the same example as given above would look like this:

```
Y   3                   // atom #1
Zr  4                   // atom #2
O  -2                   // atom #3



0.055 0.055 0.140       // chi^2 weights (for each type)



0 0 0                   // Y-Zr (Rij, B, cut-off distance)
2.019 0.37 3.2          // Y-O
1.928 0.37 3.2          // Zr-O



400000                  // "intermediate save"
200000                  // "neighbour list update"
```

4.9 Using constraints and restraints

Constraints and restraints are valuable ways to direct the `RMCProfile` run away from unphysical models. It should be born in mind however that the overriding ethos of RMC is to produce atomistic models from experimental data and as such the constraints and restraints should play a lesser role than the data in any `RMCProfile` run. The choice of the relative weighting of the constraint(s) and restraint(s) with respect to the data is very important and optimum values are often only achieved through trial and error. For example, when using polyhedral restraints (described in the following section), it is sometimes beneficial to weight them quite strongly at the start of an `RMCProfile` minimisation until the model is fitting the data reasonably and then to reduce the weighting so that the data now dominates the choice of preferred atom moves. Similarly distance window constraints should be inspected regularly during an `RMCProfile` minimisation to ensure that they are not restricting the model excessively.

4.10 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.

4.10.1 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 6.1.

4.10.2 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 6.1.

4.10.3 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 6.1.

4.10.4 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 6.1.

4.10.5 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 6.1.

4.10.6 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 6.1.

4.10.7 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 6.1.

4.10.8 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 6.1.

4.10.9 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 6.1.

4.10.10 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 6.1.

4.10.11 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 6.1.

4.10.12 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 6.1.

4.10.13 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 6.1, 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.

4.10.14 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 6.1.

4.11 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).

4.11.1 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 (Angˆ-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.

4.11.2 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`.

4.12 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.

4.12.1 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
     39.98     -0.002873718
     40.00     -0.004977553
etc...
```

4.12.2 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_REAL_SPACE_DAT
:: and NEUTRON_RECIPROCAL_SPACE_DATA :: keyword blocks.

The file format looks something like the following for the scattering function:

```
# 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.

4.13 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.

4.13.1 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.

4.13.1a 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.

4.13.2 The `.bragg`, `.back` and `.inst` files - working with GSAS

4.13.2a `.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
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 **.inst** 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.[16]

4.13.2b .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.

4.13.2c .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.

4.13.3 The .bragg#, .back# and .hkl# files - working with Topas

4.13.3a .bragg# file

'#' here represents an integer number - this should correspond to which 'xdd' section in Topas .inp file that will be used here for Bragg pattern fitting in RMCProfile. 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:

```
4983.635   4.38E-02
4987.625   4.33E-02
4991.615   4.27E-02
4995.605   4.23E-02
4999.600   4.26E-02
5003.600   4.30E-02
5007.606   4.35E-02
...
```

[16] This is the volume GSAS used to calculate the total scale for the data so has to be the same as value in the **.EXP** file. This of course should be the same as the unitcell volume used to build the configuration.

```
21275.999   2.34E-03
21292.999   1.12E-02
21310.051   2.57E-02
21327.099   2.65E-02
```

4.13.3b .back# file

'#' carries the same meaning as presented above for .bragg# file. The file contains the tabulated background pattern.

4.13.3c .hkl# file

'#' carries the same meaning as presented above for .bragg# file. The file contains tabulated peak profile data for all distinctive (in terms of position) hkl peaks.

```
Dmin = 0.5
Phase-1
Cell Vol = 586.887
Scale = 1.24412096
# of hkl = 78
h   k   l   Multiplicity    d-spacing   Start_Point End_Point   hkl_Profile->
   14    6    6   24       0.5114300         1        53       0.00004180 0.0000
   13    7    7   24       0.5123800         1        55       0.00002652 0.0000
   16    2    2   24       0.5152900         1        60       0.00001045 0.0000
   ...
```

The first line specifies the mininum *d*-spacing. The second line indicates the current file is for phase-1. Since currently in RMCProfile version-6, we only allow a single phase, the second line here does not make any difference, but it has to be there. The third line contains information about unit cell volume. The fourth line specifies the scale factor obtained from Rietveld refinement. The fifth line gives the number of distinctive hkl peaks. Starting from the sixth line, peak profile for each hkl peak will be presented.

N.B. All the three files mentioned above does not need to be manually prepared. Instead, a tool named 'Topas4RMC' could be used for preparing all the essential input files mentioned here.

4.13.4 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.

4.14 RMCProfile version 3 ('classic') files

In this section we describe the input files required when running RMCProfile in the version 3 classic mode. In most cases we recommend using the new version 6 approach, described previously, especially for the **.dat** file as this provides access to more functionality.[17]

4.14.1 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
1 3000
0.
.00165 .03942
```

[17] 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.

```
      .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 get in touch.

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.

4.14.2 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.

At this point it is worth bringing to your attention that 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 it may eventually become impossible to maintain the code in this format. Hence we do recommend using the version 6 files and methods if they meet your needs.

4.14.3 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 (4.10).

4.15 Tools to upgrade from use of classic and RMCA versions

At this point in the development of the RMCProfile code we cannot guarantee that all functionality is compatible with the 'classic' version, and we strongly recommend making the switch to Version 6. To make this easier, 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 6.1.1 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 6.1.1.

To convert the **.dat** file to the new v6 format, we provide the rmc326 command (**??**). 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 **??**.

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.

Chapter 5

Examples

5.1 Example of using total scattering data

5.1.1 Neutron total scattering data file

We consider here an RMC study of KCN in its disordered phase. This has a very simple crystal structure based on the textbook NaCl structure, with the CN molecular anion having disordered orientations. In this case we are using a merged file for the total scattering data. The start of the file has the form

```
# File: Scattering function file
# Generated using stog version 4.1, January 2010
# Title: KCN in CCR(II) at 250K I 12Wx34H chop:19302,0off,1939
# Generating user: martin
# File directory: /Users/martin/Research/rmc/KCN/2
# Data date: 17-FEB-2003 12:04:47
# stog date: 25-01-2010
# stog time: 12:11:27
# Number of points: 2469
# -----------------------------------
  0.51000000000000001      -0.51415966245250289       9.54305797480000176E-003
  0.53000000000000003      -0.50451782866197492       4.84622437520000008E-003
  0.55000000000000004      -0.50557809724236857       3.60939646880000008E-003
  0.56999999999999995      -0.51164861557939501       3.11133462599999999E-003
  0.58999999999999997      -0.51200039215334059       2.61025113720000012E-003
  0.60999999999999999      -0.50293028481557944       2.29396395120000027E-003
  0.63000000000000000      -0.50412099867804050       2.13825509799999997E-003
...
```

In this case we are using a file that contains metadata (the lines beginning with the # character).

5.1.2 Data file

In the input data file, the scattering data are handled using the following keyword block:

```
NEUTRON_RECIPROCAL_SPACE_DATA :: 1
```

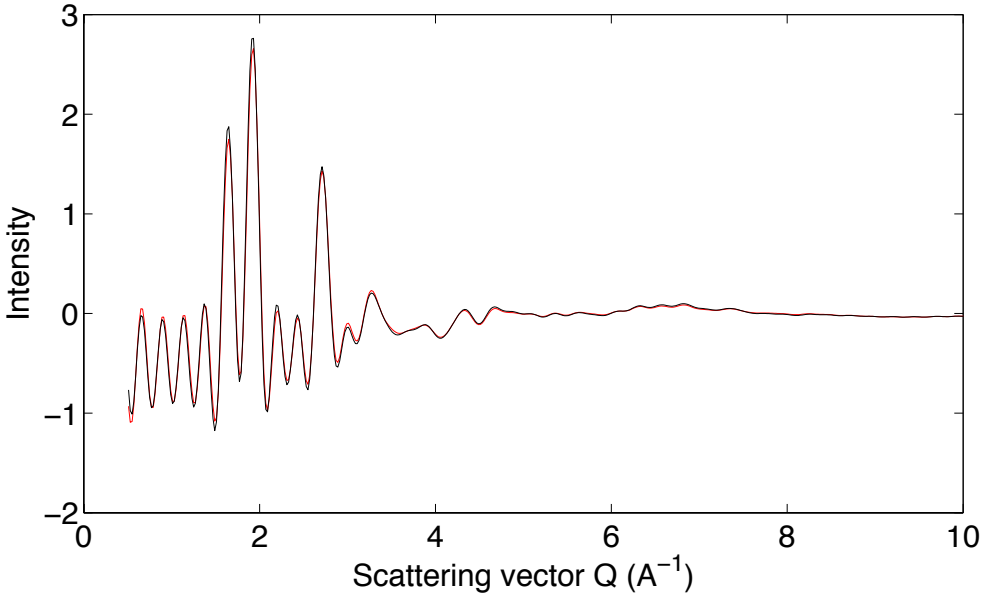


Figure 5.1: The scattering function $i(Q)$ for KCN, comparing experimental data (black line) and the fitted function from the RMC model (red line).

```
> FILENAME :: GEM12285.sq
> DATA_TYPE :: F(Q)
> FIT_TYPE :: F(Q)
> START_POINT :: 1
> END_POINT :: 2000
> CONSTANT_OFFSET ::   0.0000
> WEIGHT ::   0.005
> NO_FITTED_OFFSET
> NO_FITTED_SCALE
> STOG
> CONVOLVE ::
```

The key points from this example are

1. We are using the $F(Q)/i(Q)$ functions (both are equivalent) for both the input data format and for the RMC analysis. In this format, the errors are more-or-less equal for each point.
2. The data file was produced by the `STOG` program.
3. We have instructed `RMCProfile` to convolve the experimental data with the Fourier transform of a box function with width equal to the maximum distance in the PDF generated by `RMCProfile` from the atomic configuration.

The fitted function is shown in Figure 5.1. The effect of the convolution with the box function is shown by the strong oscillations in the function. In practice the data extend to much higher values of Q than shown in the plot, but showing the high-Q data would have squeezed the higher-intensity data into the far left-hand side of the plot.

5.2 Example of using pair distribution function data

5.2.1 Neutron PDF data file

We consider here the same RMC study of KCN in its disordered phase. The PDF data file was obtained by Fourier transform of the merged data set, and has the form

```
# File: pair distribution function file
# Generated using stog version 4.1, January 2010
# Title: KCN in CCR(II) at 250K I 12Wx34H chop:19302,0off,1939
# Generating user: martin
# File directory: /Users/martin/Research/rmc/KCN/2
# Data date: 17-FEB-2003 12:04:47
# stog date: 25-01-2010
# stog time: 12:11:27
# Number of points: 2000
# ------------------------------------
  2.0000000000000004E-002  -13.271486492010869       0.73834056742191556
  4.0000000000000008E-002  -8.5875318972803303       0.56286299993679112
  5.9999999999999978E-002  -2.9343881244971763       0.35405575737765133
  8.0000000000000017E-002   1.5829247982417609       0.20558527999260373
  0.10000000000000001        3.6272129637947250       0.16511124249615863
  0.12000000000000000        3.1324692439799078       0.15321102691071234
  0.14000000000000001        1.1215681925009062       0.12718942545392448
  ...
```

Again, we are using metadata with this file.

5.2.2 Data file

In the input data file, the scattering data are handled using the following keyword block:

```
NEUTRON_REAL_SPACE_DATA :: 1
  > FILENAME :: GEM12285.gr
  > DATA_TYPE :: G(r)
  > FIT_TYPE :: D(r)
  > START_POINT :: 1
  > END_POINT :: 3000
  > CONSTANT_OFFSET ::   0.0000
  > WEIGHT ::   0.01
  > NO_FITTED_OFFSET
  > STOG
```

The key points from this example are

1. The PDF data file gives the data in the $G(r)$ format, but we are using the $D(r)$ function for the RMC analysis. In this format, the errors are more-or-less equal for each point.

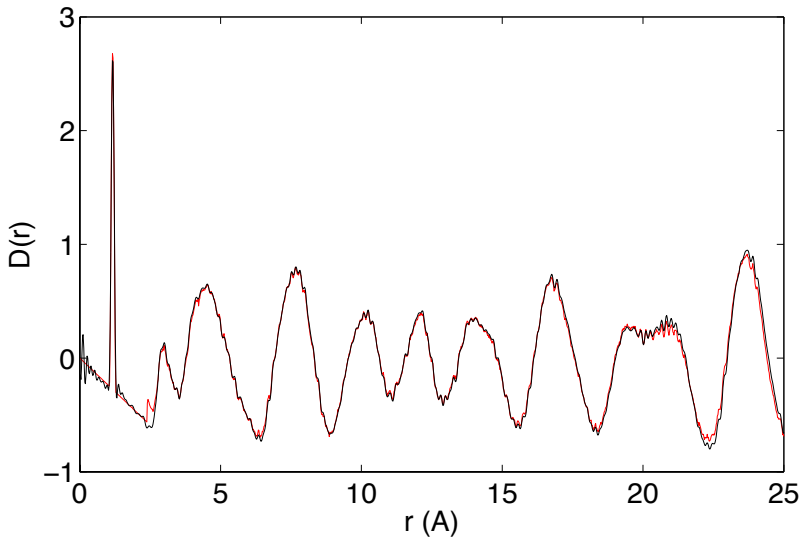


Figure 5.2: The PDF $D(r)$ for KCN, comparing experimental data (black line) and the fitted function from the RMC model (red line).

2. The data file was produced by the `STOG` program.

The fitted function is shown in Figure 5.2. The very sharp first peak represents the C–N bond, and is so sharp because this is a strong covalent bond of high stretching frequency and hence low amplitude for stretching motions. Note that the $D(r)$ function does not diverge at low r. This is a good sign; a peak at very low r is a characteristic sign of inadequate data correction.

5.3 Example of using the Bragg profile

5.3.1 Bragg profile files

Our example here will be KCN at 50 K, where the orientations of the CN molecular anions are ordered. The file containing the Bragg profile data has the form

```
        1781           2   12.38600       133.5340
Time I(obs) KCN_50K_LONG
   5.003600      3.6351800E-02
   5.007605      3.9751101E-02
   5.011610      4.3346100E-02
   5.015615      4.7388699E-02
   5.023650      5.5795401E-02
   5.027670      5.9385199E-02
   5.031691      6.1015900E-02
   5.035715      6.0725201E-02
   5.039745      5.9345201E-02
   5.043775      5.6898199E-02
```

where the file is simply a list of flight-time and intensity data. The first number given is the number of points, the second is the bank number (see next file), the third is the scale factor given by the GSAS Rietveld refinement, and the fourth is the unit cell volume (which is not used).

The Bragg data can be accompanied by two other files. The first is the file containing the instrument resolution function, extracted directly from the GSAS output file:

```
           4
           1
   1456.130     -0.3300000      -3.020000       17.98000
0.000000E+00  0.381440E+00  0.322500E+02  0.532200E+02
0.000000E+00  0.200492E+03  0.000000E+00
0.000000E+00  0.146638E+02  0.000000E+00  0.000000E+00  0.000000E+00
           2
   4883.270      -1.960000      -7.080000       63.62000
0.000000E+00  0.381443E+00  0.322488E+02  0.532215E+02
0.000000E+00  0.265473E+03  0.000000E+00
0.000000E+00  0.101245E+02  0.000000E+00  0.000000E+00  0.000000E+00
           3
   6681.000      -5.230000      -2.120000       91.37000
0.000000E+00  0.381440E+00  0.322500E+02  0.532200E+02
0.000000E+00  0.135144E+03  0.000000E+00
0.000000E+00  0.943758E+01  0.000000E+00  0.000000E+00  0.000000E+00
           4
   9087.750      -9.690000      -5.120000       154.4600
0.000000E+00  0.381443E+00  0.322488E+02  0.532215E+02
0.223010E+02  0.949869E+02  0.000000E+00
0.000000E+00  0.981154E+01  0.000000E+00  0.000000E+00  0.000000E+00
```

Here we give data for four banks, even though the Bragg diffraction data can only use one bank. The second file is the list of background parameters.

5.3.2 Data file

In the input data file, the Bragg profile is handled using the following keyword block:

```
BRAGG ::
  > BRAGG_SHAPE :: GSAS2
  > WEIGHT :: 0.01
  > RECALCULATE
  > DMIN :: 0.9
```

The fitted function is shown in Figure 5.3. In this case, agreement is not yet perfect.

5.4 Example of using potentials

5.4.1 Example of KCN

In the examples of KCN given above, we used a potential to bind the atoms in the C–N molecular anion. This was controlled in the data file using the keyword block:

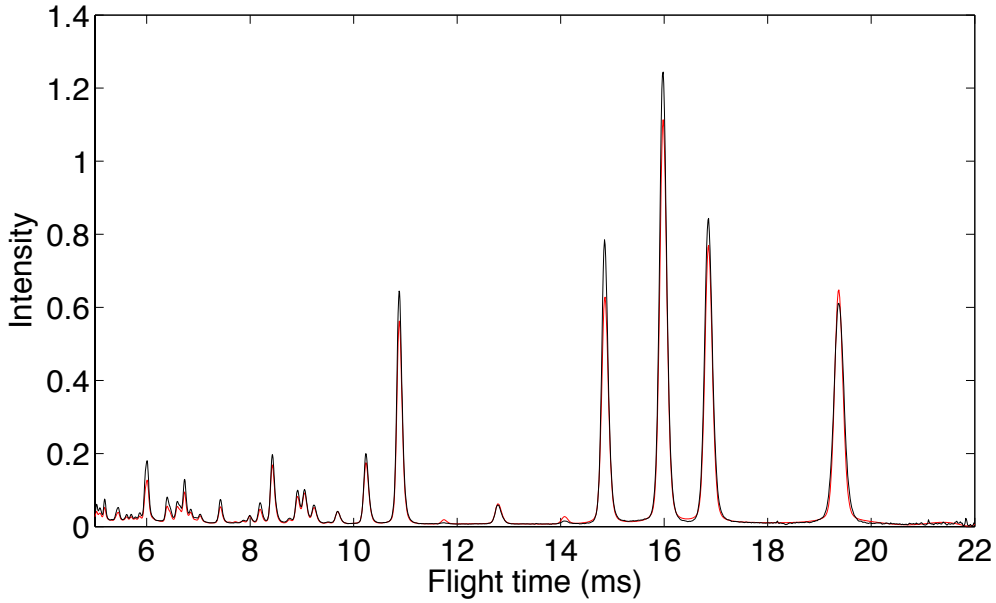


Figure 5.3: The Bragg profile for KCN at 50 K in its ordered phase, comparing experimental data (black line) and the fitted function from the RMC model (red line).

```
POTENTIALS ::
  > STRETCH :: C N 8.86 eV 1.16 Ang
  > STRETCH_SEARCH :: 20%
  > TEMPERATURE :: 250 K
```

This is a relatively simple example, because we have only one potential involving the bond stretch. In this example, the potential parameters are chosen to give the correct stretch frequency. When combined with the sample temperature, the potential energy function will give a width of the C–N peak in the PDF of the correct width.

A mode complicated example is for the crystal of ZrP_2O_7. The keyword block for the potentials is:

```
POTENTIALS ::
  > STRETCH :: P O 8.86 eV 1.52 Ang
  > STRETCH :: Zr O 8.86 eV 2.05 Ang
  > STRETCH_SEARCH :: 10\%
  > ANGLE :: P O O 20.0 eV 109.471 deg 1.45 1.45 Ang
  > ANGLE :: Zr O O 20.0 eV 90.0 deg 2.05 2.05 Ang
  > ANGLE_SEARCH :: 10 deg
  > TEMPERATURE :: 800 K
  > angle_histogram :: dangle = 5 deg
```

Here we have stretching parameters for both Zr–O and P–O bonds, and angle potentials for the O–Zr–O angles within the ZrO_6 octahedra and the O–P–O angles within the PO_4 tetrahedra. In this case also we are selecting an angle step for the construction of the bond orientation distribution functions that are automatically calculated.

Pullng together the examples discussed in the previous sections, Figure 5.4 shows a single layer of atoms taken from a configuration of the disordered phase of KCN.

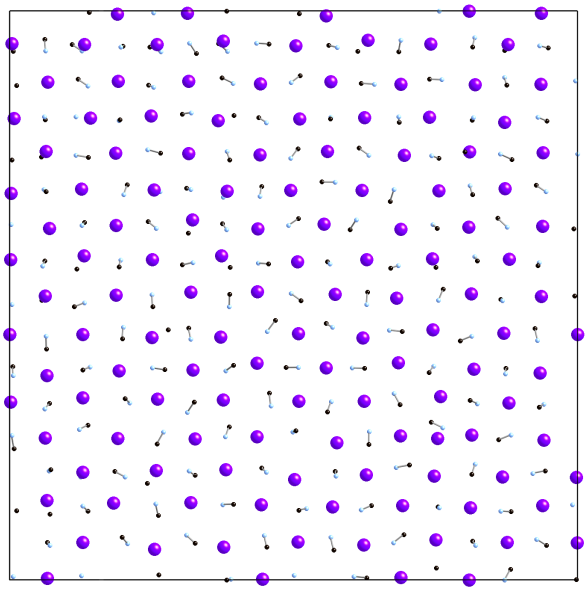


Figure 5.4: A layer of atoms in a configuration of KCN in the disordered phase. The K atoms are purple, C are black and N are blue. The C–N bonds are shown.

5.5 Example of using Bond valence sum

Using the bond valence sum (BVS) constraint requires the `BVS ::` block of keywords, as described in section 4.1.2.

Here is an example of setting up this constraint for the system yttria stabilized zirconia, $(ZrO_2)_x(Y_2O_3)_y$.

```
BVS ::
 > ATOM :: Y Zr O
 > OXID :: 3 4 -2
 > WEIGHTS :: 0.055 0.055 0.140
 > RIJ :: 0 2.019 1.928
 > BVAL :: 0 0.37 0.37
 > CUTOFF ::  0 3.2 3.2
 > SAVE :: 400000
 > UPDATE :: 200000
```

This should be fairly self explanatory, but it is worth drawing attention to the `> OXID ::` line, which contains the oxidation states of each constituent atom, and to the `> RIJ ::` line, which contains the bond valence parameters for each atom pair (zeroes for non-bonded pairs). These values can be found tabulated in Appendix C and are taken from Brese and O'Keefe, Acta Cryst B, **47**, 192-197 (1991).

The order in which the parameters are given is very important, and slightly different to how the partials and coefficients are ordered elsewhere. For BVS, you do not include like-atom pairs.

Configuration	Number of pairs	Order
AB	1	AB
ABC	3	AB AC BC
ABCD	6	AB AC AD BC BD CD
ABCDE	10	AB AC AD AE BC BD BE CD CE DE

The current maximum number of atom types is 10.

5.6 Example of using constraints

5.6.1 Distance window

A good example of the use of distance window constraints would be the work that we have carried out on $D_3Co(CN)_6$ (Hydrogen-bonding transition and isotope-dependent negative thermal expansion in $H_3Co(CN)_6$, D A Keen, M T Dove, J S O Evans, A L Goodwin, L Peters and M G Tucker, J. Phys. Condensed Matter (2010)). The structure of this material consists of three interpenetrating lattices each with a distorted cubic topology with octahedrally co-ordinated Co atoms at the corners of the cubes linked to each other via Co–C–N–D–N–C–Co linkages. The interest in this material is in the nature of the hydrogen bonding (and, by implication the location of the hydrogen atom) and how to explain the smaller negative thermal expansion of this material when contrasted with the colossal thermal expansion in the isostructural material $Ag_3Co(CN)_6$. Neutron total scattering data were obtained from $D_3Co(CN)_6$ at three temperatures and modelled using `RMCProfile` using distance window constraints to maintain the topology of the structure whilst allowing flexibility of the linkages. It was found that as the temperature increased the locations of the deuterium atoms moved from the linkage centre defined by the neighbouring N...N atoms such that the density of deuterium atoms peaked preferentially either side of the centre but with no reduction in overall symmetry. This shows that the material becomes increasingly molecular as temperature is increased, allowing the linkage vibrations to decouple and thus reduce the overall degree of thermal expansion/contraction in the material.

The keyword input for the system described above, with atoms given in the order D, Co, C, N, might look like this:

```
DISTANCE_WINDOW ::
  > MNDIST :: 2.8 0 0 0.91 0 1.72 0 2.47 0
  > MXDIST :: 3.8 0 0 1.8 0 2.06 0 2.87 0
```

The minimum and maximum distances for each pair are given sequentially in the order 1-1, 1-2, 1-3, 1-4, 2-2, 2-3,... etc. Therefore, in this case the constraint is stating that pair 1-1, two deuterium atoms, must be no closer together than 2.8 , and no further apart than 3.8 . The zero values are

used when there is no constraint applied to that pair. When using the distance window constraint it is extremely important to delete the neighbours files (`.neigh` and `.neighlist`) when you start a new run, or after changing the configuration, as these files define the connectivity of your system and must be updated when changes occur.

5.6.2 Coordination number

Either fixed or average coordination number constraints can be used. A fixed coordination number constraint will attempt to drive all (or a specified percentage) of an atom type to have the desired coordination number. The strength of this constraint can be increased by reducing the weighting number and/or by narrowing the range over which the coordination number is calculated. The weight should be within the range $1x10^{-2}$ – $1x10^{-9}$ (though values of around $1x10^{-5}$ seem to work best). Outside of this range you risk either the constraint being ignored completely or the data being ignored in favour of the constraint.

In the following example, it is specified that 100% of atom number 1 should have a coordination number of 4, and that 100 % of atom 2 should have a coordination number of 2. This sort of setup would be suitable for a tetrahedral network such as SiO_2.

```
FIXED_COORDINATION_CONSTRAINTS :: 2
  > CSTR1 :: 1 2 0.5 2.0 4 1.0 0.00001
  > CSTR2 :: 2 1 0.5 2.0 2 1.0 0.00001
```

An average coordination constraint will allow individual coordination numbers to differ from the specified value, as long as the average is as desired.

```
AVERAGE_COORDINATION_CONSTRAINTS :: 1
  > CAVSTR1 :: 1 2 0.5 2.0 4.0 0.00001
```

5.7 Example of using polyhedral restraints

This example shows what is required to use the SiO_2 polyhedral restraint. Please note that at the present time, you will need to have your *configuration* in the classic style, with extension `.cfg` for the polyhedral restraints to work. In the **`.dat`** file the following line must appear:

```
POLYHEDRAL_RESTRAINT :: 1
```

This tells `RMCProfile` that you are using polyhedral restraint number 1, the SiO_2 restraint, which maintains a network of linked tetrahedra through restraints on bond angle and length. The restraint requires several simple auxiliary files to be present in your working directory and with the same stem-name as your **`.dat`** file.

The first is the **`.poly`** file, which contains the weights for the restraint.

```
title
1.613 100 300
```

The first line is a title line which is ignored, and the second line contains the ideal Si-O bond length and the weightings for the bond restraint and the tetrahedral angle restraint. The weights are simple multipliers with larger numbers giving heavier weighting, so in this case the angle restraint is given the higher weight. The format of this file changes slightly for different restraints, please see section 4.10.

Next are the neighbours files. These differ with different restraints, again please see section 4.10. For the SiO_2 restraint, you will need two files: **`.sio`** and **`.osi`**

The **`.sio`** file contains a list of O neighbours around each Si atom and looks like this:

```
    5184
    1     4    5509  7887 10801 14392
    2     4    5510  7885 10802 14393
    3     4    5511  7886 10803 14394
    4     4    5512  7890 10804 14395
    5     4    5513  7888 10805 14396
    6     4    5514  7889 10806 14397
    7     4    5515  7893 10807 14398
    8     4    5516  7891 10808 14399
    9     4    5517  7892 10809 14400
   10     4    5518  7896 10810 14401
```

The first line is the number of Si atoms, and the subsequent lines are structured as atom number, number of O neighbours, and the atom numbers of those neighbours in the configuration file. Similarly, the **`.osi`** file contains the Si neighbours around each O.

```
   10368
 5185     2     865  1423
 5186     2     866  1424
 5187     2     867  1425
 5188     2     868  1426
 5189     2     869  1427
 5190     2     870  1428
 5191     2     871  1429
 5192     2     872  1430
 5193     2     873  1431
 5194     2     874  1405
```

5.8 Example of using magnetism

Coming soon!

5.9 Example of using EXAFS data

There is now a tutorial on using EXAFS data in the tutorials folder included with your RMCProfile download.

5.10 Example of using X-ray data

There is now a tutorial on using X-ray data in the tutorials folder included with your RMCProfile download.

Chapter 6

Tools for data preparation and analysis

6.1 Data preparation 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. Here it should be mentioned that for all the small programs mentioned here, either the version information will be directly printed out to the screen as we start the program, or we can execute the program with the '-version' argument to obtain the version information.

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. Alternatively the program Xming is freely available on the web,[1] and is very satisfactory. If you use Mac OS X or Linux/Unix, you are already set and ready to go.

6.1.1 data2config: the configuration preparation tool

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 as well as the easier to read version 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. An alternative, if slightly more labour intensive, method uses the `crystal` program that is described later.

[1] http://www.straightrunning.com/XmingNotes/

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.

For users of the Bragg component of `RMCProfile`, `data2config` will allow you to set up all the Bragg files from the original GSAS files.

6.1.1a Command syntax

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

```
data2config [optional flags] 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`.

When run without any arguments, you will get a summary of the command flags and file types. More help information can be obtained from the following command and flag

One useful first command argument is

```
data2config -help
```

Or the users can just run 'data2config' followed by nothing and the program will print out all the available available options with a brief description for each.

The various flags, all of which are prefaced with a hyphen and some of which have a following argument, are

`-analysis`	Generates simple bond analysis, requires file input.
`-angle`	Generates simple bond angle analysis.
`-atomeye`	Write configuration file in Atomeye format.
`-bank`	Provides the GSAS histogram number as argument.
`-Biso`	Applies Gaussian distribution displacement for all atoms with Biso value spacified in [...].
`-cif`	Writes the configuration to a CIF file for viewing by a crystal structure visualisation program. This enables a check that the configuration is what you expected it to be. The output file name has the same root name as the input file, but is prepended by the characters `new_`, so if the input crystal structure is called `sio2.cif`, the configuration will be written to `new_sio2.cif`.
`-cmtx`	Write CrystalMaker text input file.
`-crystal`	Writes a file (with extension '.cfgcom') in the format used by the `crystal` tool for generation of version 3 configuration files (described later).

`-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`.
`-delete`	Delete vacant sites if provided in the structure file.
`-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.
`-dlpoly`	Write DLPOLY `CONFIG` file – you get this conversion for free because it was easy to implement.
`-dw`	Generates rmc distance windows file, requires file input.
`-elabel`	Generates new element based labels.
`-field`	Write bonds part of DLPOLY FIELD file, requires file input.
`-gasp`	Write the configuration in GASP input format.
`-gridfill`	Reads size of grid cell to replace the default value of 5.0.
`-gulpc`	Writes structure in format for GULP, in cartesian coordinates.
`-gulpf`	Writes structure in format for GULP, in fractional coordinates.
`-help`	Provides help information to the computer terminal window
`-hollow`	Creates hollow nanoparticles.
`-history`	Lists version history.
`-keep`	Keeps orginal coordinates, ie so not shift atom coordinates to put into unit cell.
`-limits`	Restricts calculations to a range of target atoms specified within [...] brackets.
`-listf`	Provides a list file as an argument.
`-metadata`	Allows metadata to be supplied in a file whose name is provided as an argument to this flag. The metadata file containing keyword/value pairs following a heading line containing the single word `[metadata]` (case independent, square brackets required), and with one metadata item provided per line with keyword and value separated by an `=` sign. Allowed metadata keywords (case independent) are `investigators`, `affiliation`, `material`, `phase`, `chemical formula`, `title`, `purpose`, `keywords`, `temperature`, `pressure`, `note` and `comment`. Note that the metadata values can have several words.
`-molecules`	Finds molecules in a configuration, using file with provided name.

`-nano <value>`	Creates a spherical nanoparticle, with radius given as the essential `<value>` parameter.
`-nanobox`	Specify the multipliers for the nanoparticle box. If not provided, [4 4 4] will be used.
`-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.
`-not`	Do not generate the Bragg scattering files if the .exp and .lst files are present.
`-one`	Produces a configuration that is a single unit cell from the provided structure.
`-order`	Order the atoms as per a list supplied in response to a question.
`-output`	This will tell the program to write all output files to a directory/folder supplied as an argument immediately following this flag.
`-pdf`	Provides a calculated pdf with maximum distance as supplied argument.
`-pdfwidth`	Provides a width for broadening the pdf peaks as supplied argument.
`-reduce`	Moves all atoms in a supercell back into the original unit cell.
`-rect`	Request the program to form a *C*-centred orthorhombic cell from an initial hexgonal (or trigonal) set of axes. This will work even if your cell is not strictly with a set of hexagonal lattice parameters.
`-relabel`	Generates new integer-based atom labels.
`-reorder`	Reorders atoms without generating supercell.
`-replace`	Requests the program to generate output structure with partial occupancies. This subroutine allows you to replace some atoms at random by specified atoms, allowing for site disorder where all sites are fully occupied with fractional occupancy of two different atoms. The argent is given as a string within [...] brackets, where the string has the form "Si 0.5 Al", where we assume that the sum of occupancies is one. It is possible to use labels, eg "Ca1 0.5 Mg". Multiple substitutions are separated by commands, eg "Si 0.125 Al : Mg 0.5 Ca". For a site occupancy lower than 1.0 the an atom type or site label should be replaced by vacancy (atom type Va or va should be used).
`-rmc3`	Writes 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`	Writes RMC version 6f configuration file (a properly-annotated input file with the atoms in fractional coordinates). The output filename has the extension `.rmc6f`.
`-rmc6o`	Writes RMC v6o configuration file.

`-rsame`	Change the default test parameter to determine identical sites.
`-shape`	Provides nanoparticle shape as argument (e.g. sphere, cube, cylinder, rectangle, ellipsoid etc.).
`-shift`	Makes sure all fractional coordinate values are between 0 and 1.
`-size <value>`	Usually `data2config` will ask for the user for three integers with which to scale the starting structure in each direction to create the configuration. Using this flag with the required parameter will set the supercell to have dimensions along each direction to be as close to the provided value as possible, and avoid the need to be asked the questions.
`-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).
`-supercell`	Generate a supercell of the input crystal unit cell by multiplying each unit cell vector by an integer. The set of integers are provided within a pair of square brackets.
`-Uiso`	Applies Gaussian distribution displacement for all atoms with Uiso value specified in [...].
`-usehist`	When reading a GSAS EXP file, use only the histograms specified within [...] brackets.
`-uselabels`	Uses atom labels as provided in structure file.
`-vacancy`	Creates vacancies, provide atoms and fractions within [...] brackets as argument.
`-version`	Prints the version number.
`-writef`	In some cases where you have an option, write the output file in fractional coordinates.
`-writeo`	In some cases where you have an option, write the output file in orthogonal coordinates.
`-xtl`	Write an XTL format configuration file.

Any number of options can be selected; for example

```
data2config -cssr -rmc6f -size 50 -sort NaCl.cif
```

will produce an RMC configuration file from an initial CIF file for sodium chloride, with approximate dimension 50 Å in each direction, ordered by atom type (in order of atomic number), and also giving a CSSR file for drawing with appropriate crystal visualisation software;

```
data2config -rmc6f -sort -rect quartz.tbl
```

will produce an RMC configuration of quartz from a GSAS output file, again with atoms sorted, and with orthogonal axes rather than the crystallographic hexagonal axes, but with a supercell yet to be provided by the user in response to questions given at the terminal; and

```
data2config -rmc6f -sort -nano 40 TiO2.cif
```

will produce a configuration containing a single spherical nanoparticle of TiO_2 with radius 40 Å.

6.1.1b Allowed crystal structure file types

The 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. Please be aware that in case of using the Bragg data type in the RMCProfile refinement is recommenced to use `get_gsas_bragg` program to create all the necessary files (`.back, .bragg, .hkl, .inst and .xray`)
`.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 6.1.1e).
`.CFG` or `.cfg`	RMC v3 configuration file
`.CFGCOM` or `.cfgcom`	Input file for the `crystal` configuration generation tool (subsection 6.1.3).

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

```
data2config -diag -cssr -rmc6f nacl.cif
```

will produce the configuration files `nacl.cssr` and `nacl.rmc6f`, 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.

6.1.1c Setting up for Bragg scattering data from GSAS

If you are including the Bragg scattering data explicitly within your RMC analysis, you will need to have performed a GSAS Rietveld refinement and provide the `.back`, `.inst` and `.bragg` files (see Section 4.5).

These files will be automatically generated if your crystal structure file type is of type `.TBL` (a GSAS table file) and your working directly also contains the corresponding GSAS `.EXP` and `.LST` files.

6.1.1d 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.
- 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.

6.1.1e 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.

6.1.1f Split site and partial occupancy

From the version `3.5` a program `data2config` can generate supercell structure with split site occupancy or partially occupied sites. In the case of partial occupancy a given fraction of generated sites will be replaced by a vacancy (atom type `Va` will be used). See `-replace` option for details.

On the other hand, `data2config` can now recognise split site and partial occupancy when reading structural data from `cif` and `tbl` files. Based on the information from the structural file a proper `-replace` option will be generated and displayed.

Please be aware that only two atoms per site can be split. If you need more it is possible to do it step by step, simply calling `data2config` multiple times introducing new atom type each time.

6.1.1g 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.

6.1.1h 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.[2] Similarly, requests for new features can be sent to the author.

Also, it's worth mentioning here that since there are so many CIF file formats which may be generated by various programs. Although we try to make `data2config` working robustly with various CIF file formats, it is still expected that some format of CIF file can not be read in properly by `data2config`. If this is the case, the temporary solution is to import the CIF file into CrystalMaker and export the CIF from there. Also the users are welcome to report any problems about CIF file reading to the authors.

6.1.2 cleanrun

A little script for cleaning up all the files generated by previous RMCProfile running. File names could be provided as the arguments and the corresponding file(s) will be removed together with all the RMCProfile running files.

[2] Email contact details can be obtained from `http://web.me.com/dove_family/martin/contact.html`. Another available contact (Yuanpeng Zhang) is: zyroc1990@gmail.com.

6.1.3 crystal: tool for producing classic-format configuration files

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` is not as fully featured as `data2config` but we include it in the distribution as many people still find it useful.

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.

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`.

6.1.4 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.

6.1.5 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`.

6.1.6 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.

6.1.7 datamplot

This program can be used to plot several RMCProfile output files at once.

6.1.8 gasp

`GASP` is a tool to geometric analysis of structural polyhedra written by Stephen Wells (Royal Institution, London W1S 4BS. stephen@ri.ac.uk). For details see `GASP` manual.

6.1.9 get_gsas_bragg

This program used to be required to extract the information required by `RMCProfile` from a GSAS refinement to enable the Bragg profile to be fitted, but the same functionality is now be provided by `data2config`. 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\`.

6.1.10 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. If you use the new molecular restraints, this is not required.

6.1.11 neighbour_list2

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. If you use the new molecular restraints, this is not required. The file format `cfg` and `rmc6f` is recognised from the filename.

6.1.12 rmc_to_gasp

This program reads in the RMC configuration and writes an output file which can be than read into the `gasp` program. The file format `cfg` and `rmc6f` is recognised from the filename.

6.1.13 gaussdist

This program reads in an RMC configuration and applies a small shift to each atom to produce a Gaussian distribution around each 'crystallographic' site. It also applies a 'gaussian narrowing' to directly bonded atoms in order to produce a starting configuration with a better visual agreement with a measured PDF. It asks for the information it needs and produces both **`.cfg`** and **`.rmc6f`** files.

6.1.14 dwbuild

This program builds a non-crystalline starting configuration, based on a composition and density you provide. While it can be used to produce a "random" configuration, a significant amount of time can be saved by applying closest-approach constraints, producing a starting configuration in which none of the atoms are too close together. The program asks for the information it requires and

writes out both **.cfg** and **.rmc6f** files. Note: to obtain a ”random” configuration, the user should enter zeroes when asked for distances.

6.1.15 atomchoose

This program takes the '.cfg' structural configuration file as the input. To use this program, it is required that the number of two types of atoms to be considered is equal. Assuming we only have two types of atoms in the configuration, then it is required that the number of atoms with type 1 and type 2 should be equal to each other. After asking for the input and output file name, the program will ask which two atoms are going to be swapped, followed by asking the fraction of atom 1 (which refers to the first one of the input from last step) to be kept. Taking a configuration with 10 atoms for each of the two atom types as an example, the final number of atoms in the output configuration will be 10 (half of the original total amount). If we enter '1 2' when asked for the two atoms to be swapped, followed by entering '0.2' when asked for the fraction of atom 1, the program will first randomly pick up 2 atoms from those with atom type 1. Then the rest 8 (10 $\rightarrow$ total, 2 $\rightarrow$ type1) atoms will come from atom type 2. Assuming the 1st and 2nd atoms in the list of atoms with type 1 is picked up, then the 3rd till 10th atoms in the list of atoms with type 2 will be picked up accordingly.

6.1.16 get_gsas_bragg

Currently, it is not possible to use **data2config** to prepare the input X-ray Bragg data files. If GSAS is installed on your machine, the **get_gsas_bragg** program can then be used as an alternative. The usage of this program is like '`get_gsas_bragg FULL_PATH_TO_GSAS_EXE_FOLDER`'. For example, on Windows, if GSAS is installed under C drive, we can type '`get_gsas_bragg c:\gsas\exe\`' to start the program. Here it should be pointed out that the ending backslash is necessary. Also the full path to the GSAS executables should NOT be too long since otherwise RMCProfile will possibly fail to find those executables! To use this program, it is required that the '.gsas' and '.prm' files exist in the directory where the program is executed. The program will ask for the stem name of the '.EXP' file followed by asking for the histogram to use and the minimum *d*-spacing. After this the program will run and produce the **.bragg**, **.back** and **.inst** files as well as a **.xray** file which contains the X-ray form factor information.

6.1.17 gpar_to_tot_gen

This program can compose the partial PDF to total PDF. For example, the theoretical partial PDF generated by **grsq_gen** program (see next entry) can be transformed to total PDF using this program. Also for the latest version, the '.csv' file containing the partial PDF generated by RMCProfile is supported. After starting the program, first enter the input file containing the partial PDF. Then the program will ask for the scattering coefficients corresponding to each partial PDF. For example if we have the SF6 system, the scattering coefficients corresponding to each pair of atoms should be entered in the following order – S-S S-F F-F. Finally, the program will ask for the output file name to write the composed PDF to.

6.1.18 grsq_gen

This program can be used to generate the theoretical partial PDF and S(Q) data corresponding to the given structural configuration. Both the '.cfg' and '.rmc6f' formats are supported as the input. The usage of this program should be straightforward to following by inspecting the prompts.

6.1.19 rmc326

This program can be used to transform the version 3 input control file (the '.dat' file) for RMCProfile to version 6. For example, to transform the version 3 '.dat' file '`rmcsf6_190k_v3.dat`' to version 6, we can type '`rmc326 -o rmcsf6_190k_v6.dat rmcsf6_190k_v3.dat`' where '`rmcsf6_190k_v6.dat`' is the output version 6 '.dat' file for running RMCProfile. Typing '`rmc326`' followed by nothing will then print out the full option list of the program.

6.1.20 rebin

Currently for RMCProfile simulation with either real or reciprocal space scattering data, it is assumed the data is equal-interval. Specifically for PDF data, it is also assumed the interval of the experimental data is the same as the value specified in the '.dat' input control file. When the data sets do not meet the requirement, the **rebin** program then can be used to put the data onto certain grid points suitable for RMCProfile simulation.

6.1.21 site_split

This program takes the '.cfg' structural configuration as the input file, and it will take certain positions of specified atom type and assign them to new atom type. The program will first ask for the dimension of the supercell, followed by several other questions, which should be straightforward to follow by inspecting the prompt. When asked for the 'number of sites', we should enter the number of atomic positions where we are going to replace the original atom with the new one. This should be followed by entering the x, y and z coordinates for each of the atomic positions. Also the program will ask for the tolerance for x, y and z coordinates (fractional), which means not only the atoms located on the exact positions specified but also those near the specified positions (within the given tolerance) will be replaced with atoms of the new type.

6.1.22 stog_new

As indicated by its name, this program transforms the S(Q) data ('s' in 'stog_new') to G(r) data ('g' in 'stog_new'). To promise the data is transformed properly, it is optimal that the input S(Q) data is equal-interval. If the raw S(Q) data does not meet the optimal requirement, the '**rebin**' tool mentioned above can be used to cast the S(Q) data to an equal-interval grid. The program will start by asking for the number of data files to transform and the corresponding file names. Then for each S(Q) data set, the program will ask a series of questions concerning the transformation. Most of the questions are straightforward to follow by inspecting the prompt. There are a few of them that needs a bit of explanation. The first one is 'yscale', which is actually the divider but not a multiplier to the raw intensity (the 'y' component of the S(Q) data). The second one is about the window function. If we answer 'Y' to the question about whether we want to use the window function, the S(Q) data will be multiplied by a window function before the Fourier transform to *r*-space. This may help dealing with the problem of the artificial Fourier ripples during the transformation. The third one is the 'Add values', which actually plays the role of both scale factor and offset. The calculation formula is something like this 'y=y*(1+vadd)+vadd'. The next one is about the 'Fourier filter'. If answering 'Y' to the question about whether the Fourier filter will be considered, the program will ask for the r cut-off for the Fourier filter. This means the program will try to get rid of all the frequency component (in *r*-space) lower than the specified r cut-off. The last one to mention is the 'RMC cuttoff', which basically means the r cut-off for the G(r) data to be used for the RMC simulation.

6.1.23 stog_gudrun_offsets

This program can be used to process the S(Q) data output from GUDRUN, and the usage is quite similar to '**stog_new**' introduced above. The difference here is this program will ask for the DIFC, DIFA and ZERO parameters from GSAS refinement and also the DIFC parameter from GUDRUN '.cal' file. Therefore users are required to get those parameters ready to use this program.

6.1.24 xray_to_rmc

For X-ray scattering, we have the Q-dependent scattering factor, which we have to take into account when refining the X-ray F(Q) data. With the old version of RMCProfile, this program can be used to tabulate all the partial scattering factors (q-dependent) into the file containing the F(Q) data for RMCProfile to use during the refinement. Although current version of RMCProfile already can do this pre-refinement work automatically without using '**xray_to_rmc**', this program is kept so that users can choose whichever way for refining the X-ray F(Q) data and compare in between the outputs from both. After starting the program, first enter the scattering data file name. Then the program will ask for the constants to subtract, multiply by and offset by, respectively. Next the stem name for RMCProfile simulation will be required. For example, if we have the RMCProfile control file named as '`gapo4_xray.dat`' then the stem name is '`gapo4_xray`'. After that, the program will ask for the data normalization option, the way to to get the sample concentration and the way to deal with the low-Q range of the data, respectively. Answer to the question as needed. Then the program will ask for the way to obtain the atomic form factor information. We can manually enter the information for each element in the system in certain order. Or we can choose to read from the '.xray' file which contains all the needed information. Here the '.xray' file should be with the same stem name as we input in one of the previous steps. Using whichever way to input the atomic form factor information, the users are required to find the information by themselves. Simply searching 'atomic form factor table' on the web will guide us to the right place. Finally, give the output name for the file containing the X-ray F(Q) data together with the tabulated Q-dependent form factors.

6.1.25 Topas4RMC

This is a dedicated GUI written in Python so users have to manually install it separately from RMCProfile package. *conda* should be used to install 'Topas4RMC' tool for preparing those essential files for fitting Bragg pattern in RMCProfile using Topas profile. Once *conda* is installed, one can simply execute

conda install -c apw247 topas4rmc

The GUI looks like,

and detailed documentation can be found in the 'Help' tab of the GUI.

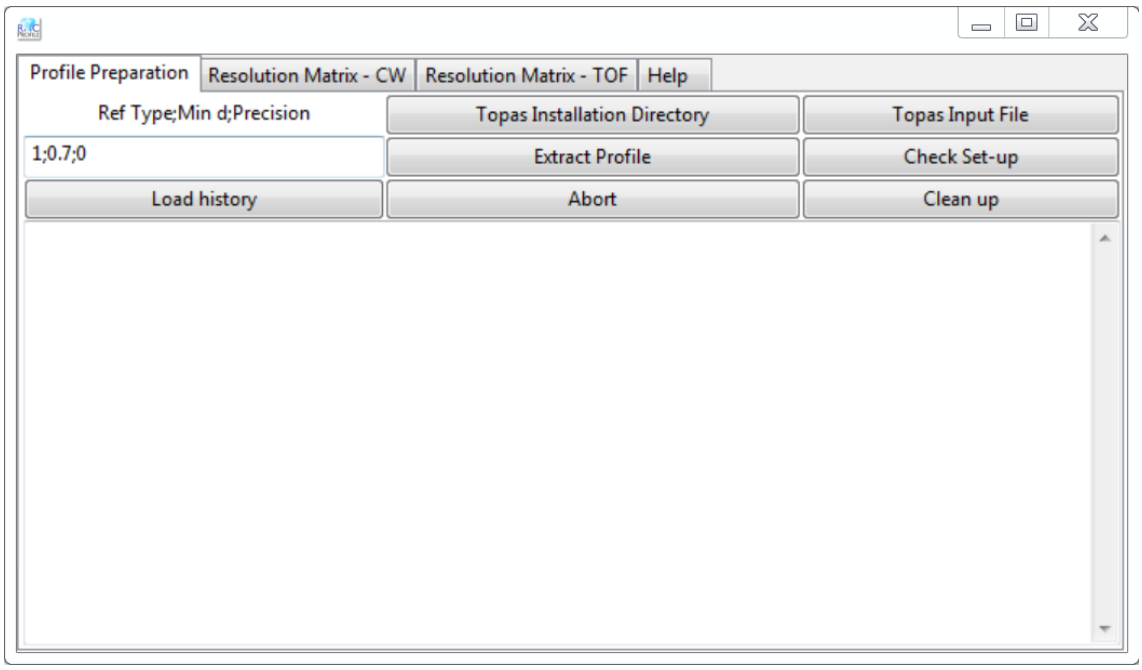


Figure 6.1: Screenshot of the Topas4RMC GUI.

6.2 Analysis tools

6.2.1 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. The file format `cfg` and `rmc6f` is recognised from the filename.

6.2.2 rmc_to_discus

This program can be used to convert `RMCProfile` configurations into the correct format to be used by the `Discus` program. It is run on the command line and asks for the information required. Only the file format `rmc6f` is recognised from the filename. Please be aware that the `rmc6f` file has to be generated with the newest version of `data2config` which also writes supplementary information about listed atoms. The same applies to `RMCProfile` program. Each atom line has to contain all the information like

```
1 Fe [1]      0.0     0.0     0.0     1 0 0 0
```

6.2.3 rmcplot

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.

It requires an X server to be running and can be run on the command line by typing "`rmcplot.bat <<stem name>>`", where `<<stem name>>` is replaced by the `RMCProfile` run name such as "`rmcsf6_190k`" in the above example. In the plot interface, left click allows you to zoom, right click resets the view and middle mouse button exits from cursor.

6.2.4 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. It can then be run on the command line by typing "`atomeye.bat <<atomeye file>>`", where 'atomeye file' is replaced by the configuration converted by the program `rmc_to_atomeye` — *i.e.*, `rmcsf6_190k.eyecfg`. 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.

6.2.5 unitcell

This program can be used to shrink the supercell `RMCProfile` configurations into a single unitcell. This allows possibly recognise simple structural distortion in details such as anisotropic temperature factor or polyhedra distortion. The file format `cfg` and `rmc6f` is recognised from the filename.

6.2.6 atom_density

This program can be used to calculate atom density in the shrieked supercell. Prior to `atom_density` the `unit_cell` should be used. The `atom_density` program can be used on a single shrieked file or on a series of files names `steam_name_1.rmc6f, steam_name_2.rmc6f...`. The program calculates the histogram of atom positions for the given subvoulme of the unit cell and number of histogram bins. As an output the *.vtk file is generates which can be visualised in ParaView program [3].

6.2.7 get_gasp_rotors

This program works together with the 'gasp' program for the geometry analysis. Detailed usage can be found in exercise 4 (step-6) of the RMCProfile tutorial coming together with the current distribution.

[3] `http://www.paraview.org`

6.2.8 getavcell_gen

This program takes the output configuration from 'unitcell' (which condenses all atoms of the supercell back into the unit cell) and it will calculate the average positions of each crystallographically non-equivalent atom in the condensed unit cell. It should be mentioned that the configuration file should be with the extension of '.uc'. Apart from stem name of the input configuration, the program will ask for the dimension of the original supercell and also the atomic symbol for each type of atoms contained in the configuration.

6.2.9 histogram_data

This program can be used to generate the histogram of a whole bunch of numbers, which basically means it can put the input data into certain bins. When started, the program can be followed by three arguments, i.e. '`histogram_data FILENAME NUMBER_OF_BINS NUMBER_OF_PARAMETERS`', where the number of parameters basically means the number of columns to be included in the histogram calculation. If no arguments are provided when starting **histogram_data**, the program will ask for the input data file name and then sets the number of bins and number of parameters to their default value, respectively. The name of the output histogram data file will be ended with 'bin.dat'.

6.2.10 triplets_new_bonds_sinth

This program will help analysing the RMCProfile structural configuration to extract the information about bondings. It will take the '.rmc6f' file as the input and output reports about the bond information between atoms (number of bonds, average distance, etc.) and bond angle data. The program will first ask for the number of theta points for bond angle analysis. Then it will ask for the number of neighbours to consider for bond angle analysis, and obviously the input number should be greater or equal than 2 to promise we really have a meaningful bond angle there. Next the program will ask for the number of configurations and the file name for each of them. This will be followed by the input requirement for maximum and minimum bond length between each pair of atom types. Finally, give the name for the files containing the average bonding data and the bond angle analysis output, respectively. Here it should be mentioned that the bond angle analysis output contains three-column data for each different type of triplet bonding (e.g. b111 represents the triplet bonding between three atoms of type1). For the three-column data, the first column is the angle in degree (θ). The second column is the number of triplet bondings with the corresponding bond angle (allowing for certain variance) divided by $sin(\theta)$. The third column is just the number of triplet bondings with the corresponding bond angle (again, allowing for certain variance).

6.2.11 triplets_new_sinth

This program is quite similar to the previous one, and the usage is nearly identical. One difference as compared to the previous one is that this program takes the '.cfg' file as the input. The second difference is that this program does not report the average data about the bondings.

6.3 CML-based analysis tools

With the use of the "`CML ::`" keyword, `RMCProfile` will generate an XML file containing a rich set of output data in the Chemical Markup Language (CML) format, as described previously. The power of XML is obtained from tools that can process XML documents. Here we describe two tools that exploit the use of CML to the RMC user's advantage.

6.3.1 The ccViz tool

`ccViz` is a python program (written by Toby White[4]) 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[5] for the mac and windows platforms, or Firefox[6] for mac, windows and linux platforms. Older browsers, or modern browsers that do not conform to 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 6.2 and 6.3. 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 experts – our view is that output files should not need to be read hand-in-hand with the manual to

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

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

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

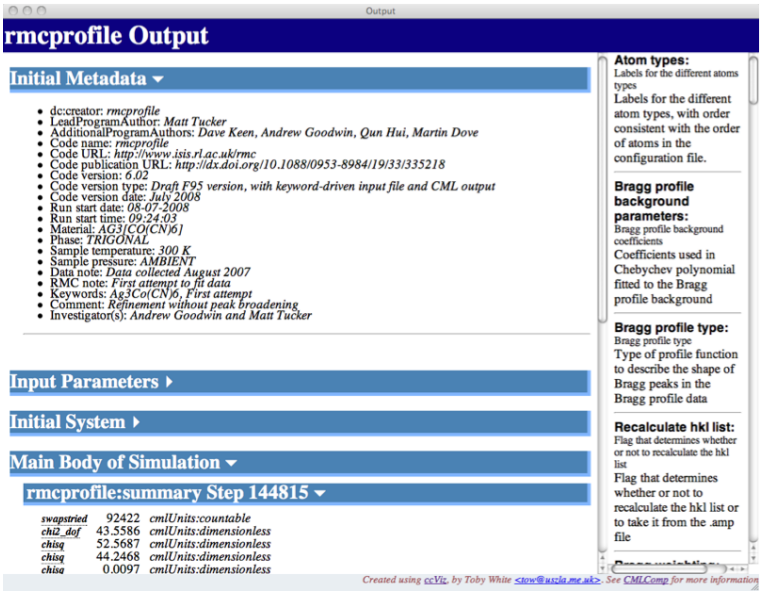


Figure 6.2: Top of the XHTML file generated by `ccViz`, showing the initial metadata in expanded form, the link to the input parameters in collapsed view, and the output form a single step in the simulation. By moving the mouse cursor over any parameter will bring up the corresponding dictionary definition.

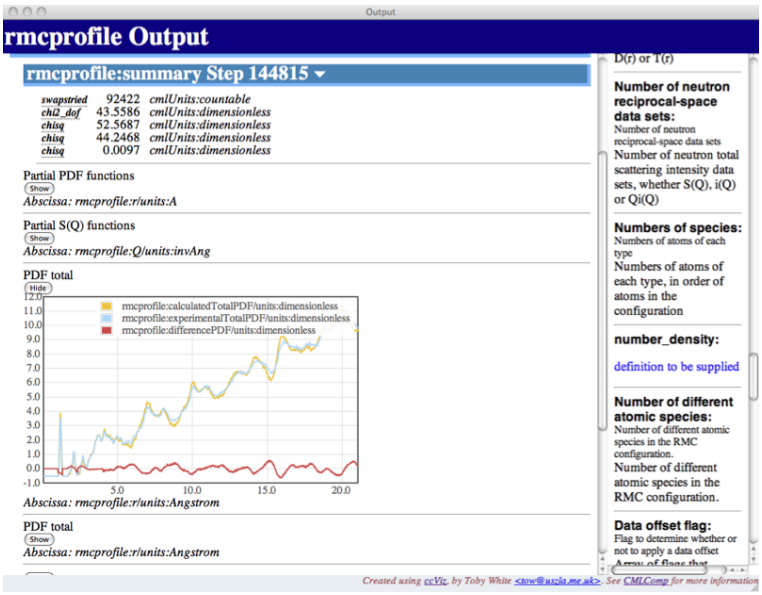


Figure 6.3: Portion of the XHTML file generated by `ccViz`, showing the data plots. Each plot can be opened or closed using the `Show`/`Hide` buttons. Selecting a portion of a plot will give a magnified view.

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.

6.3.2 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):

```
$  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 2.8. The keyword is the word in square brackets in the first line.

The result of this example invocation of the summon command is

Table 6.1: 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.

```
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 6.1. 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.

Appendix A

Basic theory of total scattering

1.1 Theoretical basics of 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, $I(\mathbf{Q})$:

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

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

$$I(\mathbf{Q}) = \frac{1}{N}\sum_{j,k}\overline{b}_k\overline{b}_k\langle\exp\left(i\mathbf{Q}\cdot[\mathbf{r}_j-\mathbf{r}_k]\right)\rangle \tag{A.2}$$

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

$$I_{\text{Bragg}}(\mathbf{Q}) = \frac{1}{N}\left|\sum_j\langle\exp\left(i\mathbf{Q}\cdot\mathbf{r}_j\right)\rangle\right|^2 \tag{A.3}$$

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

$$\langle\exp\left(i\mathbf{Q}\cdot\mathbf{r}_j\right)\rangle = \exp\left(i\mathbf{Q}\cdot\overline{\mathbf{r}}_j\right)\int p(\mathbf{r}-\overline{\mathbf{r}}_j)\exp\left(i\mathbf{Q}\cdot\mathbf{r}\right)\,\mathrm{d}\mathbf{r} \tag{A.4}$$

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

$$I(Q) = \frac{1}{N}\sum_{j,k}\overline{b}_k\overline{b}_k\frac{\sin\left(Q\left|\mathbf{r}_j-\mathbf{r}_k\right|\right)}{Q\left|\mathbf{r}_j-\mathbf{r}_k\right|} \tag{A.5}$$

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 [*]

$$I(Q) = i(Q) + \sum_m c_m\overline{b_m^2} \tag{A.6}$$

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.

$$i(Q) = \rho_0 \int_0^\infty 4\pi r^2 G(r) \frac{\sin Qr}{Qr} \, \mathrm{d}r \tag{A.7}$$

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

$$G(r) = \sum_{m,n} c_m c_n \overline{b}_m \overline{b}_n \left(g_{mn}(r) - 1\right) \tag{A.8}$$

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

$$g_{mn}(r) = \frac{n_{mn}(r)}{4\pi r^2 \rho_m \mathrm{d}r} \tag{A.9}$$

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

$$D(r) = 4\pi r \rho_0 G(r) \tag{A.10}$$

so that

$$Qi(Q) = \int_0^\infty D(r) \sin(Qr) \, \mathrm{d}r \tag{A.11}$$

The reverse transformation is then given as

$$D(r) = \frac{2}{\pi} \int_0^\infty Qi(Q) \sin(Qr) \, \mathrm{d}r \tag{A.12}$$

(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.

1.2 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

$$\langle \exp\left(i\mathbf{Q}\cdot[\mathbf{r}_j - \mathbf{r}_k]\right)\rangle = \frac{1}{4\pi}\int_0^{2\pi} \mathrm{d}\phi \int_0^{\pi} \sin\theta\,\mathrm{d}\theta \exp(iQr_{jk}\cos\theta) \quad \text{(A.13)}$$

$$= \frac{1}{2}\int_{-1}^{+1} \exp(iQr_{jk}x)\,\mathrm{d}x \quad \text{(A.14)}$$

$$= \frac{\sin(Qr_{jk})}{Qr_{jk}} \quad \text{(A.15)}$$

Thus for all atoms we obtain

$$S(Q) = \sum_{jk} \overline{b}_j \overline{b}_k \sin((Qr_{jk})/Qr_{jk} \quad \text{(A.16)}$$

$$= \sum_{j} \overline{b}_j^2 + \sum_{j\neq k} \overline{b}_j \overline{b}_k \sin((Qr_{jk})/Qr_{jk} \quad \text{(A.17)}$$

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

$$I(Q) = \sum_m c_m \overline{b_m^2} + i(Q) + S_0 \quad \text{(A.18)}$$

$$i(Q) = \rho_0 \sum_{m,n} c_m c_n \overline{b}_m \overline{b}_n \int_0^{\infty} 4\pi r^2 \left(g_{mn}(r) - 1\right) \frac{\sin(Qr)}{Qr}\,\mathrm{d}r \quad \text{(A.19)}$$

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 $Q \to 0$.

1.3 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

$$\chi^2 = \sum_j \left(y_j^{\mathrm{exp}} - y_j^{\mathrm{calc}} \right) / \sigma_j^2 \tag{A.20}$$

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

$$P = \exp\left(-\Delta\chi^2/2\right) \tag{A.21}$$

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:

$$\chi^2_{\mathrm{RMC}} = \sum_m S_m \chi^2_m \tag{A.22}$$

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

$$\chi^2_{Qi(Q)} = \sum_k \sum_j \left(Qi_{\text{calc}}(Q_j)_k - Qi_{\text{exp}}(Q_j)_k\right)^2 / \sigma^2_k(Q_j) \quad \text{(A.23)}$$

$$\chi^2_{D(r)} = \sum_j \left(D_{\text{calc}}(r_j)_k - D_{\text{exp}}(r_j)_k\right)^2 / \sigma^2(r_j) \quad \text{(A.24)}$$

$$\chi^2_{\text{profile}} = \sum_k \sum_j \left(I^{\text{calc}}_{\text{profile}}(t_j)_k - I^{\text{exp}}_{\text{profile}}(t_j)_k\right)^2 / \sigma^2_k(t_j) \quad \text{(A.25)}$$

$$\chi^2_f = \sum_\ell \left(f^{\text{calc}}_\ell - f^{\text{req}}_\ell\right)^2 / \sigma^2_\ell \quad \text{(A.26)}$$

$$\chi^2_{\text{BS}} = \frac{1}{k_{\text{B}}T} \sum_\ell D_\ell \left[1 - \exp\left(-\alpha(r - r_0)\right)\right]^2 \quad \text{(A.27)}$$

$$\chi^2_{\text{BB}} = \frac{1}{k_{\text{B}}T} \sum_\ell K_\ell \left(\cos\theta - \cos\theta_0\right)^2 \quad \text{(A.28)}$$

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.

Appendix B

Suggested values for interatomic potentials

This appendix contains recommended values for certain molecular interactions that you might need. If you can't find exactly what you need, it should be possible to adapt these values.

The tables below report values taken from the MM3 molecular mechanics database of values. For the values of the distances, r_0, you are recommended to check that these are consistent with the position of the corresponding peak in the experimental PDF, and if they appear to be significantly different you are recommended to use the distances from the PDF. Moreover, if you can deduce a bond angle from the PDF and it is not consistent with the value obtained from these tables, you are again recommended to use the value deduced from the PDF. Note, however, we caution against using distances and angles taken from a standard crystal structure refinement, because these are not true bond lengths. Crystal structure gives you the mean positions of atoms, and the distance between two mean positions is not always the same as the mean instantaneous distance between the two atoms.* One of the joys of PDF-based approaches to studying materials is to be able to identify cases where there is a difference between these quantities, because it points to the existence of disorder beyond simple thermally-induced harmonic atomic motions.

2.1 Bond-stretching terms

Table B.1: Bond stretching terms from the standard MM3 data set

	r_0 (Å)	D (eV)
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$	1.52	2.155
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$	1.50	3.024
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$	1.51	2.304

*Mathematically, this point is equivalent to the fact that $\langle (r_1 - r_2)^2 \rangle \neq (\langle r_1 \rangle - \langle r_1 \rangle)^2$.

$C_{CSP3ALKANE}$–$C_{CSPALKYNE}$	1.47	2.640
$C_{CSP3ALKANE}$–$H_{EXCEPTONN,O,S}$	1.11	2.275
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$	1.41	2.736
$C_{CSP3ALKANE}$–N_{NSP3}	1.45	2.544
$C_{CSP3ALKANE}$–N_{NSP2}	1.45	2.501
$C_{CSP3ALKANE}$–$F_{FLUORIDE}$	1.39	2.832
$C_{CSP3ALKANE}$–$Cl_{CHLORIDE}$	1.79	1.488
$C_{CSP3ALKANE}$–$Br_{BROMIDE}$	1.94	1.104
$C_{CSP3ALKANE}$–I_{IODIDE}	2.17	1.032
$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$	1.80	1.440
$C_{CSP3ALKANE}$–$S_{>S+SULFONIUM}$	1.82	1.542
$C_{CSP3ALKANE}$–$S_{>S=OSULFOXIDE}$	1.80	1.416
$C_{CSP3ALKANE}$–$S_{>SO2SULFONE}$	1.77	1.488
$C_{CSP3ALKANE}$–Si_{SILANE}	1.88	1.464
$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$	1.51	2.400
$C_{CSP3ALKANE}$–$P_{>PPHOSPHINE}$	1.84	1.411
$C_{CSP3ALKANE}$–$B_{>BTRIGONAL}$	1.58	2.160
$C_{CSP3ALKANE}$–$C^{*}_{RADICAL}$	1.50	2.496
$C_{CSP3ALKANE}$–$Ge_{GERMANIUM}$	1.95	1.305
$C_{CSP3ALKANE}$–Sn_{TIN}	2.15	1.019
$C_{CSP3ALKANE}$–$Pb_{LEAD(IV)}$	2.24	0.912
$C_{CSP3ALKANE}$–$Se_{SELENIUM}$	1.95	1.286
$C_{CSP3ALKANE}$–$Te_{TELLURIUM}$	2.14	1.296
$C_{CSP3ALKANE}$–$D_{DEUTERIUM}$	1.11	2.275
$C_{CSP3ALKANE}$–$N_{N=C-/PYR(DELOCLZD)}$	1.43	2.400
$C_{CSP3ALKANE}$–$C_{CSP2CYCLOPROPENE}$	1.50	2.112
$C_{CSP3ALKANE}$–$N_{NSP3AMMONIUM}$	1.51	2.051
$C_{CSP3ALKANE}$–$N_{NSP2PYRROLE}$	1.49	2.030
$C_{CSP3ALKANE}$–$O_{OSP2FURAN}$	1.42	2.592
$C_{CSP3ALKANE}$–$N_{=NOAZOXY(LOCAL)}$	1.48	2.496
$C_{CSP3ALKANE}$–N_{NITRO}	1.50	2.640
$C_{CSP3ALKANE}$–$C_{BENZENE(LOCALIZED)}$	1.50	3.024
$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$	1.52	2.155
$C_{CSP3ALKANE}$–$C_{CSP2CYCLOBUTENE}$	1.50	3.024
$C_{CSP3ALKANE}$–$O_{KETONIUMOXYGEN}$	1.49	3.552
$C_{CSP3ALKANE}$–$C_{KETONIUMCARBON}$	1.50	2.304
$C_{CSP3ALKANE}$–$N_{=NIMINE(LOCALZD)}$	1.44	2.400
$C_{CSP3ALKANE}$–$O_{OH,O-C(CARBOXYL)}$	1.41	2.736
$C_{CSP3ALKANE}$–$N_{N=AZOXY(LOCAL)}$	1.45	2.150

$C_{CSP3ALKANE}$–$N_{N(+)=IMMINIUM}$	1.47	3.081
$C_{CSP3ALKANE}$–$N_{N(+)=PYRIDINIUM}$	1.48	2.640
$C_{CSP3ALKANE}$–$CP_{CFERROCENEC}$	1.51	3.024
$C_{CSP3ALKANE}$–$O_{>NOHHYDROXYAMINE}$	1.37	2.808
$C_{CSP3ALKANE}$–$N_{>NOHHYDROXYAMINE}$	1.41	2.304
$C_{CSP3ALKANE}$–$N_{NSP3HYDRAZINE}$	1.44	1.824
$C_{CSP3ALKANE}$–$P_{>P=OPHOSPHATE}$	1.80	1.584
$C_{CSP3ALKANE}$–$S_{>SO2SULFONAMIDE}$	1.78	1.490
$C_{CSP3ALKANE}$–$N_{NSP3SULFONAMIDE}$	1.45	2.138
$C_{CSP3ALKANE}$–$O_{OP=OPHOSPHATE}$	1.42	2.112
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$	1.33	3.600
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$	1.35	4.080
$C_{CSP2ALKENE}$–$C_{CSPALKYNE}$	1.31	5.375
$C_{CSP2ALKENE}$–$H_{EXCEPTONN,O,S}$	1.10	2.472
$C_{CSP2ALKENE}$–$O_{CO-H,C-O-C,O-O}$	1.35	2.880
$C_{CSP2ALKENE}$–N_{NSP3}	1.37	3.033
$C_{CSP2ALKENE}$–N_{NSP2}	1.41	2.860
$C_{CSP2ALKENE}$–$F_{FLUORIDE}$	1.35	2.640
$C_{CSP2ALKENE}$–$Cl_{CHLORIDE}$	1.73	1.344
$C_{CSP2ALKENE}$–$Br_{BROMIDE}$	1.89	1.200
$C_{CSP2ALKENE}$–I_{IODIDE}	2.08	1.190
$C_{CSP2ALKENE}$–$S_{>SO2SULFONE}$	1.77	1.344
$C_{CSP2ALKENE}$–Si_{SILANE}	1.85	1.440
$C_{CSP2ALKENE}$–$C_{CYCLOPROPANE}$	1.46	2.736
$C_{CSP2ALKENE}$–$P_{>PPHOSPHINE}$	1.83	1.397
$C_{CSP2ALKENE}$–$B_{>BTRIGONAL}$	1.55	1.661
$C_{CSP2ALKENE}$–$Ge_{GERMANIUM}$	1.94	1.718
$C_{CSP2ALKENE}$–$D_{DEUTERIUM}$	1.10	2.472
$C_{CSP2ALKENE}$–$N_{N=C-/PYR(DELOCLZD)}$	1.27	4.319
$C_{CSP2ALKENE}$–$C_{CSP2CYCLOPROPENE}$	1.34	4.607
$C_{CSP2ALKENE}$–$N_{NSP3AMMONIUM}$	1.26	5.323
$C_{CSP2ALKENE}$–$N_{NSP2PYRROLE}$	1.27	5.323
$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$	1.22	5.855
$C_{CSP2ALKENE}$–$S_{SSP2THIOPHENE}$	1.54	3.442
$C_{CSP2ALKENE}$–$N_{=NOAZOXY(LOCAL)}$	1.21	1.920
$C_{CSP2ALKENE}$–N_{NITRO}	1.47	2.424
$C_{CSP2ALKENE}$–$C_{BENZENE(LOCALIZED)}$	1.43	2.534
$C_{CSP2ALKENE}$–$C_{CSP3CYCLOBUTANE}$	1.50	3.024
$C_{CSP2ALKENE}$–$C_{CSP2CYCLOBUTENE}$	1.33	3.600

$C_{CSP2ALKENE}$–$N_{=NIMINE(LOCALZD)}$	1.27	4.319
$C_{CSP2ALKENE}$–$C_{=C=OKETENE}$	1.31	5.711
$C_{CSP2ALKENE}$–$N_{=NOHOXIME}$	1.28	4.176
$C_{CSP2ALKENE}$–$N_{N(+)=IMMINIUM}$	1.28	4.799
$C_{CSP2ALKENE}$–$N_{N(+)=PYRIDINIUM}$	1.27	3.984
$C_{CSP2ALKENE}$–$N_{=NOAXOXY(DELOC)}$	0.83	1.824
$C_{CSP2ALKENE}$–$N_{N=AZOXY(DELOC)}$	1.40	2.496
$C_{CSP2CARBONYL}$–$C_{CSP2CARBONYL}$	1.22	5.399
$C_{CSP2CARBONYL}$–$H_{EXCEPTONN,O,S}$	1.12	2.097
$C_{CSP2CARBONYL}$–$O_{CO-H,C-O-C,O-O}$	1.35	2.880
$C_{CSP2CARBONYL}$–$O_{O=CCARBONYL}$	1.21	4.847
$C_{CSP2CARBONYL}$–N_{NSP2}	1.38	3.216
$C_{CSP2CARBONYL}$–$F_{FLUORIDE}$	1.38	2.016
$C_{CSP2CARBONYL}$–$Cl_{CHLORIDE}$	1.82	1.382
$C_{CSP2CARBONYL}$–$Br_{BROMIDE}$	1.99	1.344
$C_{CSP2CARBONYL}$–I_{IODIDE}	2.23	1.248
$C_{CSP2CARBONYL}$–$C_{CYCLOPROPANE}$	1.45	2.112
$C_{CSP2CARBONYL}$–$D_{DEUTERIUM}$	1.11	2.097
$C_{CSP2CARBONYL}$–$O_{CARBOXYLATEION}$	1.28	3.376
$C_{CSP2CARBONYL}$–$C_{CSP3CYCLOBUTANE}$	1.51	2.304
$C_{CSP2CARBONYL}$–$C_{CSP2CYCLOBUTENE}$	1.35	4.607
$C_{CSP2CARBONYL}$–$O_{OH,O-C(CARBOXYL)}$	1.35	2.880
$C_{CSP2CARBONYL}$–$O_{O=CC=O}$	1.21	5.183
$C_{CSP2CARBONYL}$–$O_{O=CO-H(ACID)}$	1.21	4.703
$C_{CSP2CARBONYL}$–$O_{O=CO-C(ESTER)}$	1.21	4.703
$C_{CSP2CARBONYL}$–$O_{O=CX(HALIDE)}$	1.20	5.591
$C_{CSP2CARBONYL}$–$O_{O=CC=C<}$	1.21	4.627
$C_{CSP2CARBONYL}$–$O_{O=CO-C=O}$	1.20	5.087
$C_{CSP2CARBONYL}$–$O_{O=C(C=C)(C=C)}$	1.21	4.319
$C_{CSP2CARBONYL}$–$O_{O=C(C=C)(OC=O)}$	1.20	4.607
$C_{CSP2CARBONYL}$–$O_{O=C(C=O)(C=C<)}$	1.21	5.183
$C_{CSP2CARBONYL}$–$O_{O-ANHYDRIDE(LOCL)}$	1.41	2.064
$C_{CSPALKYNE}$–$C_{CSPALKYNE}$	1.21	7.319
$C_{CSPALKYNE}$–N_{NSP}	1.16	8.317
$C_{CSPALKYNE}$–$C_{CSP2CYCLOBUTENE}$	1.31	5.375
$C_{CSPALKYNE}$–$H_{HCACETYLENE}$	1.08	2.865
$H_{EXCEPTONN,O,S}$–$S_{>S+SULFONIUM}$	1.35	1.824
$H_{EXCEPTONN,O,S}$–$S_{>S=OSULFOXIDE}$	1.37	1.521
$H_{EXCEPTONN,O,S}$–$S_{>SO2SULFONE}$	1.35	1.824

$H_{EXCEPTONN,O,S}$–Si_{SILANE}	1.48	1.272
$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$	1.09	2.438
$H_{EXCEPTONN,O,S}$–$P_{>PPHOSPHINE}$	1.42	1.471
$H_{EXCEPTONN,O,S}$–$C^{*}_{RADICAL}$	1.10	2.505
$H_{EXCEPTONN,O,S}$–$Ge_{GERMANIUM}$	1.53	1.224
$H_{EXCEPTONN,O,S}$–Sn_{TIN}	1.70	1.070
$H_{EXCEPTONN,O,S}$–$Pb_{LEAD(IV)}$	1.77	0.909
$H_{EXCEPTONN,O,S}$–$Se_{SELENIUM}$	1.47	1.521
$H_{EXCEPTONN,O,S}$–$Te_{TELLURIUM}$	1.67	1.368
$H_{EXCEPTONN,O,S}$–$C_{CSP2CYCLOPROPENE}$	1.07	2.208
$H_{EXCEPTONN,O,S}$–$C_{BENZENE(LOCALIZED)}$	1.10	2.472
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$	1.11	2.275
$H_{EXCEPTONN,O,S}$–$C_{CSP2CYCLOBUTENE}$	1.10	2.472
$H_{EXCEPTONN,O,S}$–$C_{KETONIUMCARBON}$	1.09	2.448
$H_{EXCEPTONN,O,S}$–$CH_{CFERROCENEH}$	1.10	2.472
$H_{EXCEPTONN,O,S}$–$P_{>P=OPHOSPHATE}$	1.40	1.574
$H_{EXCEPTONN,O,S}$–$S_{>SO2SULFONAMIDE}$	1.35	1.788
$O_{CO-H,C-O-C,O-O}$–$O_{CO-H,C-O-C,O-O}$	1.45	1.896
$O_{CO-H,C-O-C,O-O}$–Si_{SILANE}	1.64	2.424
$O_{CO-H,C-O-C,O-O}$–$H_{OHALCOHOL}$	0.95	3.662
$O_{CO-H,C-O-C,O-O}$–$H_{COOHCARBOXYL}$	0.97	3.432
$O_{CO-H,C-O-C,O-O}$–$P_{>PPHOSPHINE}$	1.61	1.392
$O_{CO-H,C-O-C,O-O}$–$B_{>BTRIGONAL}$	1.36	2.217
$O_{CO-H,C-O-C,O-O}$–$C_{CSP3CYCLOBUTANE}$	1.42	1.344
$O_{CO-H,C-O-C,O-O}$–$C_{CSP2CYCLOBUTENE}$	1.35	2.880
$O_{CO-H,C-O-C,O-O}$–$H_{HOENOL/PHENOL}$	0.97	3.456
$O_{CO-H,C-O-C,O-O}$–$P_{>P=OPHOSPHATE}$	1.60	2.544
$O_{O=CCARBONYL}$–$S_{>S=OSULFOXIDE}$	1.49	3.408
$O_{O=CCARBONYL}$–$S_{>SO2SULFONE}$	1.44	4.521
$O_{O=CCARBONYL}$–$N_{=NOAZOXY(LOCAL)}$	1.27	5.183
$O_{O=CCARBONYL}$–N_{NITRO}	1.22	3.600
$O_{O=CCARBONYL}$–$C_{C=OCYCLOBUTANONE}$	1.20	4.871
$O_{O=CCARBONYL}$–$C_{C=OCYCLOPROPANONE}$	1.20	5.481
$O_{O=CCARBONYL}$–$C_{=C=OKETENE}$	1.17	5.039
$O_{O=CCARBONYL}$–$P_{>P=OPHOSPHATE}$	1.49	4.271
$O_{O=CCARBONYL}$–$S_{>SO2SULFONAMIDE}$	1.46	4.164
N_{NSP3}–$H_{NHAMINE/IMINE}$	1.01	3.081
N_{NSP3}–$C_{BENZENE(LOCALIZED)}$	1.38	3.033
N_{NSP3}–$C_{CSP3CYCLOBUTANE}$	1.45	2.544

N_{NSP2}–$S_{>SO2SULFONE}$	1.66	2.928
N_{NSP2}–$H_{HN-C=OAMIDE}$	1.03	3.249
N_{NSP2}–$C_{CARBONIUMION}$	1.32	3.278
$S_{S-SULFIDE}$–$S_{S-SULFIDE}$	2.02	1.257
$S_{S-SULFIDE}$–$H_{SHTHIOL}$	1.34	1.857
Si_{SILANE}–Si_{SILANE}	2.32	0.792
Si_{SILANE}–$C_{CYCLOPROPANE}$	1.84	1.680
Si_{SILANE}–$C_{CSP3CYCLOBUTANE}$	1.88	0.624
$H_{OHALCOHOL}$–$O_{>NOHHYDROXYAMINE}$	0.97	3.600
$H_{OHALCOHOL}$–$O_{OP=OPHOSPHATE}$	0.95	3.734
$C_{CYCLOPROPANE}$–$C_{CYCLOPROPANE}$	1.49	2.400
$C_{CYCLOPROPANE}$–$Ge_{GERMANIUM}$	1.91	1.296
$C_{CYCLOPROPANE}$–$C_{CSP2CYCLOPROPENE}$	1.49	2.112
$C_{CYCLOPROPANE}$–N_{NITRO}	1.48	2.088
$C_{CYCLOPROPANE}$–$C_{CSP3CYCLOBUTANE}$	1.50	2.112
$H_{NHAMINE/IMINE}$–$N_{N=C-/PYR(DELOCLZD)}$	1.03	2.736
$H_{NHAMINE/IMINE}$–$N_{NSP2PYRROLE}$	1.03	3.120
$H_{NHAMINE/IMINE}$–$N_{=NOAZOXY(LOCAL)}$	1.04	2.649
$H_{NHAMINE/IMINE}$–$N_{=NIMINE(LOCALZD)}$	1.02	2.865
$H_{NHAMINE/IMINE}$–$N_{N=N-AZO(LOCAL)}$	1.03	2.568
$H_{NHAMINE/IMINE}$–$N_{N=AZOXY(LOCAL)}$	1.03	2.856
$H_{NHAMINE/IMINE}$–$N_{N(+)=IMMINIUM}$	1.02	3.129
$H_{NHAMINE/IMINE}$–$N_{N(+)=PYRIDINIUM}$	1.01	3.129
$H_{NHAMINE/IMINE}$–$N_{>NOHHYDROXYAMINE}$	1.02	2.952
$H_{NHAMINE/IMINE}$–$N_{NSP3HYDRAZINE}$	1.02	3.052
$H_{NHAMINE/IMINE}$–$N_{NSP3SULFONAMIDE}$	1.02	3.061
$H_{COOHCARBOXYL}$–$O_{OH,O-C(CARBOXYL)}$	0.97	3.432
$Ge_{GERMANIUM}$–$Ge_{GERMANIUM}$	2.40	0.696
$Pb_{LEAD(IV)}$–$Pb_{LEAD(IV)}$	1.94	0.984
$N_{N=C-/PYR(DELOCLZD)}$–$N_{N=C-/PYR(DELOCLZD)}$	1.25	4.559
$N_{N=C-/PYR(DELOCLZD)}$–$N_{NSP2PYRROLE}$	1.23	5.279
$N_{N=C-/PYR(DELOCLZD)}$–$N_{=NOAZOXY(LOCAL)}$	1.26	5.145
$C_{CSP2CYCLOPROPENE}$–$C_{CSP2CYCLOPROPENE}$	1.30	4.607
$N_{NSP3AMMONIUM}$–$N_{NSP3AMMONIUM}$	1.25	5.145
$N_{NSP3AMMONIUM}$–$N_{NSP2PYRROLE}$	1.23	5.279
$N_{NSP3AMMONIUM}$–$H_{AMMONIUM}$	1.05	2.947
$N_{NSP3AMMONIUM}$–$O_{AMINEOXIDEOXYGEN}$	1.36	2.208
$O_{OSP2FURAN}$–$H_{HOENOL/PHENOL}$	0.96	3.456
$O_{OSP2FURAN}$–$N_{=NOHOXIME}$	1.40	2.073

$S_{SSP2THIOPHENE}$–$H_{SHTHIOL}$	1.34	1.867
$N_{=NOAZOXY(LOCAL)}$–$O_{AMINEOXIDEOXYGEN}$	1.27	4.223
$N_{=NOAZOXY(LOCAL)}$–$N_{N=AZOXY(LOCAL)}$	1.26	3.408
$C_{BENZENE(LOCALIZED)}$–$C_{BENZENE(LOCALIZED)}$	1.39	3.148
$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$	1.50	2.155
$C_{CSP2CYCLOBUTENE}$–$C_{CSP2CYCLOBUTENE}$	1.33	3.600
$C_{C=OCYCLOBUTANONE}$–$O_{O=CO-C(ESTER)}$	1.17	5.039
$C_{C=OCYCLOBUTANONE}$–$O_{O=CN<(AMIDE)}$	1.21	3.360
$O_{AMINEOXIDEOXYGEN}$–$N_{=NOAXOXY(DELOC)}$	1.28	4.319
$O_{KETONIUMOXYGEN}$–$C_{KETONIUMCARBON}$	1.25	4.080
$N_{=NOAXOXY(DELOC)}$–$N_{N=AZOXY(DELOC)}$	1.26	4.223
$O_{>NOHHYDROXYAMINE}$–$N_{>NOHHYDROXYAMINE}$	1.41	2.160
$N_{NSP3HYDRAZINE}$–$N_{NSP3HYDRAZINE}$	1.55	1.440
$P_{>P=OPHOSPHATE}$–$O_{OP=OPHOSPHATE}$	1.60	2.736
$S_{>SO2SULFONAMIDE}$–$N_{NSP3SULFONAMIDE}$	1.70	1.893

Table B.2: Bond-stretching terms from the protein set of MM3 parameters

	r_0 (Å)	D (eV)
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$	1.52	2.155
$C_{Alkane(Csp3)}$–C_{Amide}	1.51	2.304
$C_{Alkane(Csp3)}$–$C_{Carboxyl}$	1.51	2.304
$C_{Alkane(Csp3)}$–C_{Phenyl}	1.50	3.024
$C_{Alkane(Csp3)}$–$C_{Alkene(His/TrpC=C)}$	1.50	3.024
$C_{Alkane(Csp3)}$–N_{Amide}	1.45	2.501
$C_{Alkane(Csp3)}$–$N_{Ammonium}$	1.47	1.896
$C_{Alkane(Csp3)}$–$N_{Guanidinium}$	1.45	2.501
$C_{Alkane(Csp3)}$–$O_{Alcohol/Phenol}$	1.41	2.736
$C_{Alkane(Csp3)}$–$S_{Sulfide}$	1.80	1.440
$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	1.11	2.275
C_{Amide}–N_{Amide}	1.33	3.216
C_{Amide}–O_{Amide}	1.23	4.415
C_{Amide}–$H_{Hydrogen(CH)}$	1.12	2.097
$C_{Carboxyl}$–$O_{Carboxylate}$	1.25	3.633
C_{Phenyl}–C_{Phenyl}	1.39	3.148
C_{Phenyl}–$C_{Alkene(His/TrpC=C)}$	1.48	2.304
C_{Phenyl}–$N_{Pyrrole}$	1.35	3.840

C_{Phenyl}–$O_{Alcohol/Phenol}$	1.35	2.880
C_{Phenyl}–$H_{Hydrogen(CH)}$	1.10	2.472
$C_{Guanidinium}$–$N_{Guanidinium}$	1.32	3.278
$C_{Alkene(His/TrpC=C)}$–$C_{Alkene(His/TrpC=C)}$	1.33	3.600
$C_{Alkene(His/TrpC=C)}$–$N_{Pyrrole}$	1.35	3.840
$C_{Alkene(His/TrpC=C)}$–$N_{Imidazolium}$	1.35	3.840
$C_{Alkene(His/TrpC=C)}$–$H_{Hydrogen(CH)}$	1.10	2.472
$C_{Imidazolium(NC-N)}$–$N_{Imidazolium}$	1.35	3.840
$C_{Imidazolium(NC-N)}$–$H_{Hydrogen(CH)}$	1.10	2.208
N_{Amide}–H_{Amide}	1.03	3.249
$N_{Ammonium}$–$H_{Ammonium}$	1.04	2.928
$N_{Guanidinium}$–$H_{Guanidinium}$	1.03	3.249
$N_{Pyrrole}$–$H_{Pyrrole}$	1.05	2.928
$N_{Imidazolium}$–$H_{Imidazolium}$	1.04	2.928
$O_{Alcohol/Phenol}$–$H_{Alcohol/Phenol}$	0.95	3.662
$S_{Sulfide}$–$S_{Sulfide}$	2.02	1.257
$S_{Sulfide}$–$H_{Thiol(SH)}$	1.34	1.857

2.2 Bond-bending terms

Table B.3: Bond bending terms from the standard MM3 data set

	θ_0 (deg)	K_θ (eV)
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$	109.500	8.364
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$	110.600	6.741
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$	110.600	9.987
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C_{CSPALKYNE}$	108.800	11.984
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$H_{EXCEPTONN,O,S}$	109.800	7.365
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$	107.500	10.361
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–N_{NSP3}	109.470	9.737
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–N_{NSP2}	109.480	9.362
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$F_{FLUORIDE}$	107.300	11.485
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$Cl_{CHLORIDE}$	106.200	9.987
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$Br_{BROMIDE}$	108.200	9.238
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–I_{IODIDE}	106.000	8.114

$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$	108.000	9.238
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$S_{>S+SULFONIUM}$	107.800	5.243
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$S_{>S=OSULFOXIDE}$	107.500	8.114
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$S_{>SO2SULFONE}$	103.000	10.860
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–Si_{SILANE}	109.000	4.993
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$	114.400	4.369
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$P_{>PPHOSPHINE}$	107.500	7.490
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$B_{>BTRIGONAL}$	110.920	4.731
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C^{*}_{RADICAL}$	110.000	5.617
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$Ge_{GERMANIUM}$	109.300	5.617
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–Sn_{TIN}	110.200	6.991
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$Pb_{LEAD(IV)}$	110.200	3.745
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$Se_{SELENIUM}$	108.200	6.242
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$Te_{TELLURIUM}$	108.200	6.242
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$D_{DEUTERIUM}$	110.100	7.365
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$N_{N=C-/PYR(DELOCLZD)}$	110.740	6.242
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$N_{NSP3AMMONIUM}$	103.500	6.778
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$O_{OSP2FURAN}$	107.500	11.235
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$N_{=NOAZOXY(LOCAL)}$	104.500	5.617
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–N_{NITRO}	106.000	12.483
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C_{BENZENE(LOCALIZED)}$	110.600	6.741
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$	109.500	8.364
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C_{CSP2CYCLOBUTENE}$	110.600	6.741
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$O_{KETONIUMOXYGEN}$	104.100	13.482
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$C_{KETONIUMCARBON}$	109.000	4.494
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$N_{N=AZOXY(LOCAL)}$	108.500	6.242
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$N_{N(+)=IMMINIUM}$	105.000	5.680
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$N_{N(+)=PYRIDINIUM}$	108.640	12.234
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$N_{NSP3HYDRAZINE}$	109.140	9.737
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$P_{>P=OPHOSPHATE}$	109.500	6.242
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$S_{>SO2SULFONAMIDE}$	108.154	15.454
$C_{CSP3ALKANE}$–$C_{CSP3ALKANE}$–$O_{OP=OPHOSPHATE}$	109.500	10.361
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$	113.200	5.617
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$	109.470	5.867
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$C_{CSPALKYNE}$	109.470	7.115
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$H_{EXCEPTONN,O,S}$	109.500	6.866
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$	107.000	8.738
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–N_{NSP3}	110.740	13.045
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–N_{NSP2}	109.800	6.242

$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$F_{FLUORIDE}$	109.000	9.487
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$Cl_{CHLORIDE}$	109.500	8.114
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–I_{IODIDE}	109.300	8.676
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$	107.800	8.114
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$S_{>S+SULFONIUM}$	107.800	5.243
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–Si_{SILANE}	109.500	6.242
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$	112.400	5.617
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$D_{DEUTERIUM}$	109.500	6.866
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$N_{N=C-/PYR(DELOCLZD)}$	110.510	4.744
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$N_{NSP3AMMONIUM}$	110.740	13.045
$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$–$N_{=NOAZOXY(LOCAL)}$	110.510	4.744
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$	109.470	5.867
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$H_{EXCEPTONN,O,S}$	109.490	6.741
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$	109.500	8.738
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–N_{NSP3}	110.740	13.045
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–N_{NSP2}	109.500	10.611
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$F_{FLUORIDE}$	109.200	9.487
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$Cl_{CHLORIDE}$	109.800	8.114
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$Br_{BROMIDE}$	109.100	9.362
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–I_{IODIDE}	108.900	7.490
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$	107.800	5.243
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$S_{>S+SULFONIUM}$	107.800	5.243
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$D_{DEUTERIUM}$	109.490	6.741
$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$–$N_{NSP3AMMONIUM}$	110.740	13.045
$C_{CSPALKYNE}$–$C_{CSP3ALKANE}$–$C_{CSPALKYNE}$	109.470	5.867
$C_{CSPALKYNE}$–$C_{CSP3ALKANE}$–$H_{EXCEPTONN,O,S}$	109.390	8.489
$C_{CSPALKYNE}$–$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$	109.000	11.235
$C_{CSPALKYNE}$–$C_{CSP3ALKANE}$–$F_{FLUORIDE}$	110.700	15.979
$C_{CSPALKYNE}$–$C_{CSP3ALKANE}$–$O_{OH,O-C(CARBOXYL)}$	107.500	9.987
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$H_{EXCEPTONN,O,S}$	107.600	6.866
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$	110.000	10.236
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–N_{NSP3}	109.300	10.236
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–N_{NSP2}	111.000	9.487
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$F_{FLUORIDE}$	108.500	8.988
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$Cl_{CHLORIDE}$	105.660	9.362
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$Br_{BROMIDE}$	106.500	6.366
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–I_{IODIDE}	102.800	7.740
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$	110.800	9.238
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$S_{>S+SULFONIUM}$	108.200	3.745

$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$S_{>S=OSULFOXIDE}$	105.000	8.926
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$S_{>SO2SULFONE}$	106.000	8.613
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–Si_{SILANE}	109.500	6.741
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$	109.410	7.365
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$P_{>PPHOSPHINE}$	111.000	7.115
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$B_{>BTRIGONAL}$	109.939	6.217
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$C^{*}_{RADICAL}$	110.000	7.240
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$Ge_{GERMANIUM}$	111.000	5.243
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–Sn_{TIN}	112.000	4.369
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$Pb_{LEAD(IV)}$	109.500	1.248
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$Se_{SELENIUM}$	110.500	4.119
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$Te_{TELLURIUM}$	110.500	4.119
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$D_{DEUTERIUM}$	107.600	6.866
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$N_{N=C-/PYR(DELOCLZD)}$	107.500	10.024
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$N_{NSP3AMMONIUM}$	104.700	11.235
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$N_{NSP2PYRROLE}$	109.400	4.494
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$O_{OSP2FURAN}$	110.000	11.485
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$N_{=NOAZOXY(LOCAL)}$	107.500	9.987
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–N_{NITRO}	109.000	9.487
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$C_{BENZENE(LOCALIZED)}$	109.500	6.866
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$	110.100	7.365
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$C_{CSP2CYCLOBUTENE}$	109.500	6.117
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$O_{KETONIUMOXYGEN}$	101.000	12.608
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$C_{KETONIUMCARBON}$	109.000	7.365
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$N_{N=AZOXY(LOCAL)}$	108.500	10.424
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$N_{N(+)=IMMINIUM}$	105.500	11.235
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$N_{N(+)=PYRIDINIUM}$	104.500	11.859
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$O_{>NOHHYDROXYAMINE}$	106.500	11.547
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$N_{>NOHHYDROXYAMINE}$	107.000	11.235
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$N_{NSP3HYDRAZINE}$	109.380	10.611
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$P_{>P=OPHOSPHATE}$	109.600	6.491
$H_{EXCEPTONN,O,S}$–$C_{CSP3ALKANE}$–$O_{OP=OPHOSPHATE}$	108.700	11.485
$O_{CO-H,C-O-C,O-O}$–$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$	103.100	6.741
$O_{CO-H,C-O-C,O-O}$–$C_{CSP3ALKANE}$–$Cl_{CHLORIDE}$	108.500	8.364
$O_{CO-H,C-O-C,O-O}$–$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$	107.500	10.361
$O_{CO-H,C-O-C,O-O}$–$C_{CSP3ALKANE}$–$C_{BENZENE(LOCALIZED)}$	107.000	8.738
$O_{CO-H,C-O-C,O-O}$–$C_{CSP3ALKANE}$–$C_{KETONIUMCARBON}$	104.100	14.980
N_{NSP3}–$C_{CSP3ALKANE}$–N_{NSP3}	110.740	13.045
N_{NSP3}–$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$	109.500	7.490

N_{NSP3}–$C_{CSP3ALKANE}$–$N_{NSP3AMMONIUM}$	110.740	13.045
N_{NSP2}–$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$	107.000	7.240
N_{NSP2}–$C_{CSP3ALKANE}$–$D_{DEUTERIUM}$	111.000	9.487
$F_{FLUORIDE}$–$C_{CSP3ALKANE}$–$F_{FLUORIDE}$	106.100	24.342
$F_{FLUORIDE}$–$C_{CSP3ALKANE}$–$Cl_{CHLORIDE}$	110.400	9.362
$F_{FLUORIDE}$–$C_{CSP3ALKANE}$–$Br_{BROMIDE}$	109.400	8.988
$Cl_{CHLORIDE}$–$C_{CSP3ALKANE}$–$Cl_{CHLORIDE}$	108.100	9.487
$Cl_{CHLORIDE}$–$C_{CSP3ALKANE}$–$Br_{BROMIDE}$	110.700	8.988
$Br_{BROMIDE}$–$C_{CSP3ALKANE}$–$Br_{BROMIDE}$	109.700	8.613
I_{IODIDE}–$C_{CSP3ALKANE}$–I_{IODIDE}	104.800	8.676
$S_{S-SULFIDE}$–$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$	110.000	5.243
Si_{SILANE}–$C_{CSP3ALKANE}$–Si_{SILANE}	109.500	4.369
$C_{CYCLOPROPANE}$–$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$	112.600	5.992
$C_{CYCLOPROPANE}$–$C_{CSP3ALKANE}$–$N_{N=C-/PYR(DELOCLZD)}$	110.200	4.744
$C_{CYCLOPROPANE}$–$C_{CSP3ALKANE}$–$N_{=NOAZOXY(LOCAL)}$	110.200	4.744
$D_{DEUTERIUM}$–$C_{CSP3ALKANE}$–$D_{DEUTERIUM}$	107.600	6.866
$N_{N=C-/PYR(DELOCLZD)}$–$C_{CSP3ALKANE}$–$N_{N=C-/PYR(DELOCLZD)}$	109.000	11.859
N_{NITRO}–$C_{CSP3ALKANE}$–N_{NITRO}	109.500	7.989
$C_{BENZENE(LOCALIZED)}$–$C_{CSP3ALKANE}$–$C_{BENZENE(LOCALIZED)}$	113.200	5.617
$C_{CSP2CYCLOBUTENE}$–$C_{CSP3ALKANE}$–$C_{CSP2CYCLOBUTENE}$	113.200	5.617
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$C_{CSP3ALKANE}$	117.000	6.741
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$	122.300	5.867
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$	117.000	6.242
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$C_{CSPALKYNE}$	116.600	5.867
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$H_{EXCEPTONN,O,S}$	117.500	6.117
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$O_{CO-H,C-O-C,O-O}$	120.000	6.242
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–Si_{SILANE}	120.000	4.993
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$D_{DEUTERIUM}$	117.500	6.117
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$N_{N=C-/PYR(DELOCLZD)}$	115.100	8.988
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$N_{NSP2PYRROLE}$	120.000	5.617
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$	120.000	6.242
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$S_{SSP2THIOPHENE}$	120.000	8.738
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$N_{=NIMINE(LOCALZD)}$	123.000	8.988
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$C_{=C=OKETENE}$	117.400	5.867
$C_{CSP3ALKANE}$–$C_{CSP2ALKENE}$–$N_{=NOHOXIME}$	117.830	10.860
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$	122.000	9.487
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$	115.500	6.242
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$H_{EXCEPTONN,O,S}$	120.000	6.117
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$O_{CO-H,C-O-C,O-O}$	121.900	7.490

$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–N_{NSP3}	122.000	7.490
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–N_{NSP2}	118.000	6.242
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$F_{FLUORIDE}$	115.500	10.111
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$Cl_{CHLORIDE}$	118.800	8.114
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$Br_{BROMIDE}$	118.100	5.617
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–I_{IODIDE}	118.800	5.118
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$S_{>SO2SULFONE}$	116.000	12.483
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–Si_{SILANE}	122.000	3.995
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$C_{CYCLOPROPANE}$	127.800	5.867
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$P_{>PPHOSPHINE}$	120.000	4.744
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$B_{>BTRIGONAL}$	121.457	8.801
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$Ge_{GERMANIUM}$	119.100	3.121
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$D_{DEUTERIUM}$	120.000	6.117
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$N_{N=C-/PYR(DELOCLZD)}$	121.000	12.109
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$N_{NSP2PYRROLE}$	119.000	8.239
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$	119.500	13.107
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$S_{SSP2THIOPHENE}$	117.000	7.240
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$N_{=NOAZOXY(LOCAL)}$	124.200	16.852
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–N_{NITRO}	115.200	12.483
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$C_{CSP2CYCLOBUTENE}$	121.000	9.487
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$N_{=NOHOXIME}$	120.590	12.483
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$N_{N(+)=PYRIDINIUM}$	118.000	9.362
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$N_{=NOAXOXY(DELOC)}$	119.700	8.738
$C_{CSP2ALKENE}$–$C_{CSP2ALKENE}$–$N_{N=AZOXY(DELOC)}$	120.000	7.864
$C_{CSP2CARBONYL}$–$C_{CSP2ALKENE}$–$H_{EXCEPTONN,O,S}$	117.000	6.242
$C_{CSP2CARBONYL}$–$C_{CSP2ALKENE}$–$N_{NSP2PYRROLE}$	120.000	6.242
$C_{CSPALKYNE}$–$C_{CSP2ALKENE}$–$H_{EXCEPTONN,O,S}$	119.500	3.745
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$H_{EXCEPTONN,O,S}$	119.000	5.617
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$O_{CO-H,C-O-C,O-O}$	116.400	6.741
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–N_{NSP3}	117.000	6.741
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–N_{NSP2}	109.000	3.745
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$F_{FLUORIDE}$	113.000	8.863
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$Cl_{CHLORIDE}$	112.600	6.866
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$Br_{BROMIDE}$	112.100	6.991
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–I_{IODIDE}	112.600	5.617
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$S_{>SO2SULFONE}$	106.000	9.737
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–Si_{SILANE}	119.500	6.554
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$C_{CYCLOPROPANE}$	120.000	6.117
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$Ge_{GERMANIUM}$	120.000	6.366

$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$D_{DEUTERIUM}$	119.000	5.617
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$N_{N=C-/PYR(DELOCLZD)}$	119.000	6.616
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$C_{CSP2CYCLOPROPENE}$	120.000	4.494
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$N_{NSP2PYRROLE}$	110.000	6.242
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$	108.000	9.987
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$S_{SSP2THIOPHENE}$	120.100	4.993
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$N_{=NOAZOXY(LOCAL)}$	116.500	4.494
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–N_{NITRO}	116.000	5.418
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$C_{BENZENE(LOCALIZED)}$	120.000	6.117
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$C_{CSP2CYCLOBUTENE}$	120.000	6.117
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$N_{=NIMINE(LOCALZD)}$	121.700	9.987
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$C_{=C=OKETENE}$	114.000	4.619
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$N_{=NOHOXIME}$	115.830	6.242
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$N_{N(+)=IMMINIUM}$	118.100	8.988
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$N_{N(+)=PYRIDINIUM}$	110.500	6.866
$H_{EXCEPTONN,O,S}$–$C_{CSP2ALKENE}$–$N_{=NOAXOXY(DELOC)}$	116.500	4.993
N_{NSP3}–$C_{CSP2ALKENE}$–N_{NSP3}	118.500	10.611
N_{NSP3}–$C_{CSP2ALKENE}$–$C_{CYCLOPROPANE}$	119.000	7.490
N_{NSP2}–$C_{CSP2ALKENE}$–N_{NSP2}	120.000	4.993
$F_{FLUORIDE}$–$C_{CSP2ALKENE}$–$C_{=C=OKETENE}$	120.200	11.859
$Cl_{CHLORIDE}$–$C_{CSP2ALKENE}$–$Cl_{CHLORIDE}$	119.200	19.599
$Cl_{CHLORIDE}$–$C_{CSP2ALKENE}$–$Br_{BROMIDE}$	110.700	19.599
$Cl_{CHLORIDE}$–$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$	105.000	6.242
$Cl_{CHLORIDE}$–$C_{CSP2ALKENE}$–$C_{=C=OKETENE}$	119.800	7.115
$Br_{BROMIDE}$–$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$	104.000	6.242
$Br_{BROMIDE}$–$C_{CSP2ALKENE}$–$C_{=C=OKETENE}$	111.200	6.616
I_{IODIDE}–$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$	107.000	6.242
Si_{SILANE}–$C_{CSP2ALKENE}$–Si_{SILANE}	120.000	4.993
$D_{DEUTERIUM}$–$C_{CSP2ALKENE}$–$D_{DEUTERIUM}$	119.000	5.617
$N_{N=C-/PYR(DELOCLZD)}$–$C_{CSP2ALKENE}$–$N_{N=C-/PYR(DELOCLZD)}$	120.000	14.980
$N_{N=C-/PYR(DELOCLZD)}$–$C_{CSP2ALKENE}$–$N_{NSP2PYRROLE}$	126.000	4.993
$S_{SSP2THIOPHENE}$–$C_{CSP2ALKENE}$–$S_{SSP2THIOPHENE}$	117.000	7.240
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$C_{CSP3ALKANE}$	116.800	15.604
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$C_{CSP2ALKENE}$	116.000	6.242
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$C_{CSP2CARBONYL}$	114.600	12.483
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$H_{EXCEPTONN,O,S}$	116.100	5.792
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$O_{CO-H,C-O-C,O-O}$	107.100	8.114
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$O_{O=CCARBONYL}$	123.500	10.611
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–N_{NSP2}	114.400	7.115

$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$F_{FLUORIDE}$	109.000	24.342
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$Cl_{CHLORIDE}$	110.200	18.101
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$Br_{BROMIDE}$	109.300	14.730
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–I_{IODIDE}	108.890	12.109
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$O_{CARBOXYLATEION}$	115.900	9.300
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$O_{OH,O-C(CARBOXYL)}$	110.300	19.349
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$O_{O=CC=O}$	122.100	8.988
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$O_{O=CO-H(ACID)}$	123.500	10.611
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$O_{O=CX(HALIDE)}$	121.600	3.995
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$O_{O=CO-C=O}$	123.000	7.490
$C_{CSP3ALKANE}$–$C_{CSP2CARBONYL}$–$O_{O-ANHYDRIDE(LOCL)}$	109.000	24.967
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$–$C_{CSP2ALKENE}$	114.700	10.611
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$–$C_{CSP2CARBONYL}$	120.000	9.987
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$–$H_{EXCEPTONN,O,S}$	111.500	3.745
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$–$O_{CO-H,C-O-C,O-O}$	124.300	8.738
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$–$O_{O=CCARBONYL}$	122.000	16.228
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$–$O_{OH,O-C(CARBOXYL)}$	124.300	8.738
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$–$O_{O=CC=O}$	123.400	16.228
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$–$O_{O=C(C=C)(OC=O)}$	126.500	9.987
$C_{CSP2ALKENE}$–$C_{CSP2CARBONYL}$–$N_{NSP2AMIDE(DELOC)}$	120.000	8.676
$C_{CSP2CARBONYL}$–$C_{CSP2CARBONYL}$–$H_{EXCEPTONN,O,S}$	112.400	8.489
$C_{CSP2CARBONYL}$–$C_{CSP2CARBONYL}$–$D_{DEUTERIUM}$	110.600	8.489
$C_{CSP2CARBONYL}$–$C_{CSP2CARBONYL}$–$O_{O=CC=O}$	119.200	9.362
$C_{CSP2CARBONYL}$–$C_{CSP2CARBONYL}$–$O_{O=C(C=C)(C=C)}$	121.600	8.863
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$H_{EXCEPTONN,O,S}$	115.500	8.114
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$O_{CO-H,C-O-C,O-O}$	107.000	6.866
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$O_{O=CCARBONYL}$	119.200	10.611
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–N_{NSP2}	109.300	5.493
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$F_{FLUORIDE}$	104.000	16.728
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$Cl_{CHLORIDE}$	105.150	15.230
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$Br_{BROMIDE}$	108.800	13.981
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$O_{CARBOXYLATEION}$	114.900	7.403
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$C_{CSP3CYCLOBUTANE}$	120.000	6.991
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$O_{OH,O-C(CARBOXYL)}$	107.000	6.866
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$O_{O=CC=O}$	118.900	7.989
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$O_{O=CX(HALIDE)}$	110.000	3.995
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$O_{O=CO-C=O}$	117.200	9.987
$H_{EXCEPTONN,O,S}$–$C_{CSP2CARBONYL}$–$O_{O-ANHYDRIDE(LOCL)}$	101.000	12.483
$O_{CO-H,C-O-C,O-O}$–$C_{CSP2CARBONYL}$–$O_{O=CCARBONYL}$	121.500	21.222

$O_{O=CCARBONYL}$–$C_{CSP2CARBONYL}$–N_{NSP2}	124.800	13.357
$O_{O=CCARBONYL}$–$C_{CSP2CARBONYL}$–$C_{CYCLOPROPANE}$	122.500	5.742
$O_{O=CCARBONYL}$–$C_{CSP2CARBONYL}$–$C_{CSP3CYCLOBUTANE}$	122.500	10.611
N_{NSP2}–$C_{CSP2CARBONYL}$–N_{NSP2}	112.500	11.235
$F_{FLUORIDE}$–$C_{CSP2CARBONYL}$–$O_{O=CX(HALIDE)}$	120.000	21.846
$Cl_{CHLORIDE}$–$C_{CSP2CARBONYL}$–$O_{O=CX(HALIDE)}$	120.500	15.479
$Br_{BROMIDE}$–$C_{CSP2CARBONYL}$–$O_{O=CX(HALIDE)}$	120.270	10.611
I_{IODIDE}–$C_{CSP2CARBONYL}$–$O_{O=CX(HALIDE)}$	117.400	6.242
$D_{DEUTERIUM}$–$C_{CSP2CARBONYL}$–$O_{O=CC=O}$	120.500	7.989
$O_{CARBOXYLATEION}$–$C_{CSP2CARBONYL}$–$O_{CARBOXYLATEION}$	129.800	17.876
$O_{OH,O-C(CARBOXYL)}$–$C_{CSP2CARBONYL}$–$O_{O=CO-H(ACID)}$	121.500	21.222
$O_{OH,O-C(CARBOXYL)}$–$C_{CSP2CARBONYL}$–$O_{O=CO-C(ESTER)}$	121.500	21.222
$O_{O=CO-C=O}$–$C_{CSP2CARBONYL}$–$O_{O-ANHYDRIDE(LOCL)}$	119.000	24.967
$O_{O=C(C=C)(OC=O)}$–$C_{CSP2CARBONYL}$–$O_{O-ANHYDRIDE(DELO)}$	121.000	16.228
$C_{CSP3ALKANE}$–$C_{CSPALKYNE}$–$C_{CSPALKYNE}$	180.000	4.744
$C_{CSP3ALKANE}$–$C_{CSPALKYNE}$–N_{NSP}	180.000	4.182
$C_{CSP2ALKENE}$–$C_{CSPALKYNE}$–$C_{CSP2ALKENE}$	180.000	4.993
$C_{CSP2ALKENE}$–$C_{CSPALKYNE}$–$C_{CSPALKYNE}$	180.000	5.867
$C_{CSPALKYNE}$–$C_{CSPALKYNE}$–$H_{HCACETYLENE}$	180.000	3.121
N_{NSP}–$C_{CSPALKYNE}$–$C_{BENZENE(LOCALIZED)}$	180.000	1.598
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$–$C_{CSP3ALKANE}$	107.200	10.236
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$–$C_{CSP2ALKENE}$	108.500	9.612
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$–$C_{CSP2CARBONYL}$	112.800	15.604
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$–$O_{CO-H,C-O-C,O-O}$	103.300	13.207
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$–Si_{SILANE}	117.100	7.864
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$–$H_{OHALCOHOL}$	106.800	9.362
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$–$P_{>PPHOSPHINE}$	116.000	9.612
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$–$C_{CSP3CYCLOBUTANE}$	108.900	8.613
$C_{CSP3ALKANE}$–$O_{CO-H,C-O-C,O-O}$–$P_{>P=OPHOSPHATE}$	119.600	17.227
$C_{CSP2ALKENE}$–$O_{CO-H,C-O-C,O-O}$–$H_{OHALCOHOL}$	109.000	4.494
$C_{CSP2ALKENE}$–$O_{CO-H,C-O-C,O-O}$–$P_{>PPHOSPHINE}$	118.000	9.987
$C_{CSP2ALKENE}$–$O_{CO-H,C-O-C,O-O}$–$H_{HOENOL/PHENOL}$	108.000	4.369
$C_{CSP2CARBONYL}$–$O_{CO-H,C-O-C,O-O}$–$C_{CSP2CARBONYL}$	106.800	9.612
$C_{CSP2CARBONYL}$–$O_{CO-H,C-O-C,O-O}$–$H_{COOHCARBOXYL}$	107.700	8.613
$C_{CSP2CARBONYL}$–$O_{CO-H,C-O-C,O-O}$–$C_{CSP3CYCLOBUTANE}$	110.800	15.604
$O_{CO-H,C-O-C,O-O}$–$O_{CO-H,C-O-C,O-O}$–$H_{OHALCOHOL}$	99.500	10.636
Si_{SILANE}–$O_{CO-H,C-O-C,O-O}$–Si_{SILANE}	142.900	3.121
Si_{SILANE}–$O_{CO-H,C-O-C,O-O}$–$H_{OHALCOHOL}$	117.500	6.366
$H_{OHALCOHOL}$–$O_{CO-H,C-O-C,O-O}$–$H_{OHALCOHOL}$	105.000	7.864

$H_{OHALCOHOL}$–$O_{CO-H,C-O-C,O-O}$–$B_{>BTRIGONAL}$	110.607	6.054
$H_{OHALCOHOL}$–$O_{CO-H,C-O-C,O-O}$–$P_{>P=OPHOSPHATE}$	110.200	4.744
$C_{CSP3ALKANE}$–N_{NSP3}–$C_{CSP3ALKANE}$	107.200	8.988
$C_{CSP3ALKANE}$–N_{NSP3}–$C_{CSP2ALKENE}$	106.800	7.465
$C_{CSP3ALKANE}$–N_{NSP3}–$H_{NHAMINE/IMINE}$	108.100	7.490
$C_{CSP3ALKANE}$–N_{NSP3}–$C_{BENZENE(LOCALIZED)}$	102.500	7.465
$C_{CSP2ALKENE}$–N_{NSP3}–$H_{NHAMINE/IMINE}$	110.500	7.328
$H_{NHAMINE/IMINE}$–N_{NSP3}–$H_{NHAMINE/IMINE}$	106.400	7.552
$H_{NHAMINE/IMINE}$–N_{NSP3}–$C_{BENZENE(LOCALIZED)}$	109.000	7.328
$H_{NHAMINE/IMINE}$–N_{NSP3}–$C_{CSP3CYCLOBUTANE}$	108.100	7.490
$C_{CSP3ALKANE}$–N_{NSP2}–$C_{CSP3ALKANE}$	122.500	9.487
$C_{CSP3ALKANE}$–N_{NSP2}–$C_{CSP2ALKENE}$	119.900	7.864
$C_{CSP3ALKANE}$–N_{NSP2}–$C_{CSP2CARBONYL}$	121.100	20.223
$C_{CSP3ALKANE}$–N_{NSP2}–$S_{>SO2SULFONE}$	110.000	3.745
$C_{CSP3ALKANE}$–N_{NSP2}–$H_{HN-C=OAMIDE}$	122.400	2.372
$C_{CSP3ALKANE}$–N_{NSP2}–$C_{CARBONIUMION}$	120.400	9.113
$C_{CSP3ALKANE}$–N_{NSP2}–$C_{CSP3CYCLOBUTANE}$	111.200	5.243
$C_{CSP3ALKANE}$–N_{NSP2}–$C_{C=OCYCLOBUTANONE}$	110.000	7.490
$C_{CSP2ALKENE}$–N_{NSP2}–$C_{CSP2ALKENE}$	107.000	4.993
$C_{CSP2ALKENE}$–N_{NSP2}–$C_{CSP2CARBONYL}$	115.000	7.490
$C_{CSP2ALKENE}$–N_{NSP2}–$H_{HN-C=OAMIDE}$	110.000	6.242
$C_{CSP2CARBONYL}$–N_{NSP2}–$C_{CSP2CARBONYL}$	124.000	11.235
$C_{CSP2CARBONYL}$–N_{NSP2}–$H_{HN-C=OAMIDE}$	118.500	7.240
$S_{>SO2SULFONE}$–N_{NSP2}–$H_{HN-C=OAMIDE}$	122.000	3.745
$H_{HN-C=OAMIDE}$–N_{NSP2}–$H_{HN-C=OAMIDE}$	123.000	5.118
$H_{HN-C=OAMIDE}$–N_{NSP2}–$C_{CARBONIUMION}$	120.500	7.240
$H_{HN-C=OAMIDE}$–N_{NSP2}–$C_{CSP3CYCLOBUTANE}$	118.300	14.356
$H_{HN-C=OAMIDE}$–N_{NSP2}–$C_{C=OCYCLOBUTANONE}$	119.200	12.483
$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$–$C_{CSP3ALKANE}$	95.900	10.486
$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$–$S_{S-SULFIDE}$	101.800	12.483
$C_{CSP3ALKANE}$–$S_{S-SULFIDE}$–$H_{SHTHIOL}$	96.000	8.114
$S_{S-SULFIDE}$–$S_{S-SULFIDE}$–$H_{SHTHIOL}$	92.200	9.650
$H_{SHTHIOL}$–$S_{S-SULFIDE}$–$H_{SHTHIOL}$	92.900	9.113
$C_{CSP3ALKANE}$–$S_{>S+SULFONIUM}$–$C_{CSP3ALKANE}$	94.300	6.242
$C_{CSP3ALKANE}$–$S_{>S+SULFONIUM}$–$H_{EXCEPTONN,O,S}$	94.000	4.993
$C_{CSP3ALKANE}$–$S_{>S=OSULFOXIDE}$–$C_{CSP3ALKANE}$	94.400	14.980
$C_{CSP3ALKANE}$–$S_{>S=OSULFOXIDE}$–$H_{EXCEPTONN,O,S}$	90.000	10.736
$C_{CSP3ALKANE}$–$S_{>S=OSULFOXIDE}$–$O_{O=CCARBONYL}$	105.600	9.987
$H_{EXCEPTONN,O,S}$–$S_{>S=OSULFOXIDE}$–$O_{O=CCARBONYL}$	109.600	10.736

$C_{CSP3ALKANE}$–$S_{>SO2SULFONE}$–$C_{CSP3ALKANE}$	101.600	10.611
$C_{CSP3ALKANE}$–$S_{>SO2SULFONE}$–$C_{CSP2ALKENE}$	102.000	22.470
$C_{CSP3ALKANE}$–$S_{>SO2SULFONE}$–$H_{EXCEPTONN,O,S}$	106.000	9.238
$C_{CSP3ALKANE}$–$S_{>SO2SULFONE}$–$O_{O=CCARBONYL}$	106.900	13.732
$C_{CSP3ALKANE}$–$S_{>SO2SULFONE}$–N_{NSP2}	103.000	9.987
$C_{CSP2ALKENE}$–$S_{>SO2SULFONE}$–$C_{CSP2ALKENE}$	102.100	24.717
$C_{CSP2ALKENE}$–$S_{>SO2SULFONE}$–$O_{O=CCARBONYL}$	108.100	24.592
$H_{EXCEPTONN,O,S}$–$S_{>SO2SULFONE}$–$O_{O=CCARBONYL}$	101.500	6.991
$H_{EXCEPTONN,O,S}$–$S_{>SO2SULFONE}$–N_{NSP2}	98.000	7.490
$O_{O=CCARBONYL}$–$S_{>SO2SULFONE}$–$O_{O=CCARBONYL}$	119.500	18.538
$O_{O=CCARBONYL}$–$S_{>SO2SULFONE}$–N_{NSP2}	108.000	21.222
$C_{CSP3ALKANE}$–Si_{SILANE}–$C_{CSP3ALKANE}$	109.500	5.992
$C_{CSP3ALKANE}$–Si_{SILANE}–$C_{CSP2ALKENE}$	110.200	4.993
$C_{CSP3ALKANE}$–Si_{SILANE}–$H_{EXCEPTONN,O,S}$	109.300	4.993
$C_{CSP3ALKANE}$–Si_{SILANE}–$O_{CO-H,C-O-C,O-O}$	108.000	6.866
$C_{CSP3ALKANE}$–Si_{SILANE}–Si_{SILANE}	109.000	5.617
$C_{CSP3ALKANE}$–Si_{SILANE}–$C_{CYCLOPROPANE}$	110.200	6.616
$C_{CSP3ALKANE}$–Si_{SILANE}–$C_{CSP3CYCLOBUTANE}$	109.500	6.616
$C_{CSP2ALKENE}$–Si_{SILANE}–$C_{CSP2ALKENE}$	104.500	7.490
$C_{CSP2ALKENE}$–Si_{SILANE}–$H_{EXCEPTONN,O,S}$	109.500	6.866
$C_{CSP2ALKENE}$–Si_{SILANE}–$O_{CO-H,C-O-C,O-O}$	109.500	6.242
$C_{CSP2ALKENE}$–Si_{SILANE}–Si_{SILANE}	110.200	4.993
$H_{EXCEPTONN,O,S}$–Si_{SILANE}–$H_{EXCEPTONN,O,S}$	106.500	5.742
$H_{EXCEPTONN,O,S}$–Si_{SILANE}–$O_{CO-H,C-O-C,O-O}$	109.500	8.364
$H_{EXCEPTONN,O,S}$–Si_{SILANE}–Si_{SILANE}	109.400	4.369
$H_{EXCEPTONN,O,S}$–Si_{SILANE}–$C_{CYCLOPROPANE}$	110.200	5.742
$H_{EXCEPTONN,O,S}$–Si_{SILANE}–$C_{CSP3CYCLOBUTANE}$	109.300	5.617
$O_{CO-H,C-O-C,O-O}$–Si_{SILANE}–$O_{CO-H,C-O-C,O-O}$	112.000	7.490
Si_{SILANE}–Si_{SILANE}–Si_{SILANE}	118.000	3.121
$C_{CSP3CYCLOBUTANE}$–Si_{SILANE}–$C_{CSP3CYCLOBUTANE}$	124.500	5.992
$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$–$C_{CSP3ALKANE}$	120.000	8.364
$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$–$H_{EXCEPTONN,O,S}$	117.100	7.490
$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$–$C_{CYCLOPROPANE}$	112.000	7.490
$C_{CSP3ALKANE}$–$C_{CYCLOPROPANE}$–O_{EPOXY}	115.000	6.991
$C_{CSP2ALKENE}$–$C_{CYCLOPROPANE}$–$C_{CSP2ALKENE}$	120.000	5.617
$C_{CSP2ALKENE}$–$C_{CYCLOPROPANE}$–$H_{EXCEPTONN,O,S}$	128.500	4.494
$C_{CSP2ALKENE}$–$C_{CYCLOPROPANE}$–$C_{CYCLOPROPANE}$	122.000	7.490
$C_{CSP2ALKENE}$–$C_{CYCLOPROPANE}$–O_{EPOXY}	115.000	6.991
$C_{CSP2CARBONYL}$–$C_{CYCLOPROPANE}$–$H_{EXCEPTONN,O,S}$	123.500	4.494

$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$–$H_{EXCEPTONN,O,S}$	116.500	3.121
$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$–Si_{SILANE}	124.500	2.996
$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$–$C_{CYCLOPROPANE}$	116.800	8.114
$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$–$Ge_{GERMANIUM}$	119.000	4.993
$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$–$C_{CSP2CYCLOPROPENE}$	124.500	4.494
$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$–N_{NITRO}	112.500	0.624
$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$–O_{EPOXY}	113.000	5.368
$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$–$C_{CSP3CYCLOBUTANE}$	113.000	4.494
$H_{EXCEPTONN,O,S}$–$C_{CYCLOPROPANE}$–$C_{C=OCYCLOPROPANONE}$	117.400	4.494
Si_{SILANE}–$C_{CYCLOPROPANE}$–$C_{CYCLOPROPANE}$	118.000	6.616
$C_{CYCLOPROPANE}$–$C_{CYCLOPROPANE}$–$C_{CYCLOPROPANE}$	119.500	4.494
$C_{CYCLOPROPANE}$–$C_{CYCLOPROPANE}$–$Ge_{GERMANIUM}$	117.000	6.242
$C_{CYCLOPROPANE}$–$C_{CYCLOPROPANE}$–N_{NITRO}	114.600	14.980
$C_{CSP3ALKANE}$–$P_{>PPHOSPHINE}$–$C_{CSP3ALKANE}$	95.600	9.612
$C_{CSP3ALKANE}$–$P_{>PPHOSPHINE}$–$C_{CSP2ALKENE}$	92.500	5.992
$C_{CSP3ALKANE}$–$P_{>PPHOSPHINE}$–$H_{EXCEPTONN,O,S}$	94.700	8.801
$C_{CSP2ALKENE}$–$P_{>PPHOSPHINE}$–$C_{CSP2ALKENE}$	95.000	5.992
$H_{EXCEPTONN,O,S}$–$P_{>PPHOSPHINE}$–$H_{EXCEPTONN,O,S}$	92.200	8.489
$O_{CO-H,C-O-C,O-O}$–$P_{>PPHOSPHINE}$–$O_{CO-H,C-O-C,O-O}$	99.500	5.617
$C_{CSP3ALKANE}$–$B_{>BTRIGONAL}$–$O_{CO-H,C-O-C,O-O}$	122.371	5.692
$C_{CSP2ALKENE}$–$B_{>BTRIGONAL}$–$O_{CO-H,C-O-C,O-O}$	126.986	7.540
$O_{CO-H,C-O-C,O-O}$–$B_{>BTRIGONAL}$–$O_{CO-H,C-O-C,O-O}$	119.860	9.113
$C_{CSP3ALKANE}$–$C^{*}_{RADICAL}$–$C_{CSP3ALKANE}$	119.000	6.242
$C_{CSP3ALKANE}$–$C^{*}_{RADICAL}$–$H_{EXCEPTONN,O,S}$	118.000	5.243
$H_{EXCEPTONN,O,S}$–$C^{*}_{RADICAL}$–$H_{EXCEPTONN,O,S}$	117.000	5.243
N_{NSP2}–$C_{CARBONIUMION}$–N_{NSP2}	120.000	4.993
$C_{CSP3ALKANE}$–$Ge_{GERMANIUM}$–$C_{CSP3ALKANE}$	109.500	6.242
$C_{CSP3ALKANE}$–$Ge_{GERMANIUM}$–$C_{CSP2ALKENE}$	110.900	6.242
$C_{CSP3ALKANE}$–$Ge_{GERMANIUM}$–$H_{EXCEPTONN,O,S}$	110.200	4.868
$C_{CSP3ALKANE}$–$Ge_{GERMANIUM}$–$Ge_{GERMANIUM}$	111.500	4.369
$C_{CSP2ALKENE}$–$Ge_{GERMANIUM}$–$C_{CSP2ALKENE}$	109.500	6.242
$C_{CSP2ALKENE}$–$Ge_{GERMANIUM}$–$H_{EXCEPTONN,O,S}$	110.100	4.868
$C_{CSP2ALKENE}$–$Ge_{GERMANIUM}$–$Ge_{GERMANIUM}$	109.500	6.242
$H_{EXCEPTONN,O,S}$–$Ge_{GERMANIUM}$–$H_{EXCEPTONN,O,S}$	107.500	5.280
$H_{EXCEPTONN,O,S}$–$Ge_{GERMANIUM}$–$C_{CYCLOPROPANE}$	108.800	4.993
$H_{EXCEPTONN,O,S}$–$Ge_{GERMANIUM}$–$Ge_{GERMANIUM}$	114.500	4.369
$H_{EXCEPTONN,O,S}$–$Ge_{GERMANIUM}$–$C_{CSP3CYCLOBUTANE}$	105.200	6.366
$Ge_{GERMANIUM}$–$Ge_{GERMANIUM}$–$Ge_{GERMANIUM}$	112.500	3.745
$C_{CSP3ALKANE}$–Sn_{TIN}–$C_{CSP3ALKANE}$	107.000	3.995

$C_{CSP3ALKANE}$–Sn_{TIN}–$H_{EXCEPTONN,O,S}$	110.500	2.834
$H_{EXCEPTONN,O,S}$–Sn_{TIN}–$H_{EXCEPTONN,O,S}$	109.500	1.648
$C_{CSP3ALKANE}$–$Pb_{LEAD(IV)}$–$C_{CSP3ALKANE}$	109.500	1.248
$C_{CSP3ALKANE}$–$Pb_{LEAD(IV)}$–$H_{EXCEPTONN,O,S}$	109.500	1.248
$C_{CSP3ALKANE}$–$Pb_{LEAD(IV)}$–$Pb_{LEAD(IV)}$	109.500	1.248
$H_{EXCEPTONN,O,S}$–$Pb_{LEAD(IV)}$–$H_{EXCEPTONN,O,S}$	109.500	1.248
$C_{CSP3ALKANE}$–$Se_{SELENIUM}$–$C_{CSP3ALKANE}$	94.800	8.114
$C_{CSP3ALKANE}$–$Se_{SELENIUM}$–$H_{EXCEPTONN,O,S}$	94.500	5.368
$C_{CSP3ALKANE}$–$Te_{TELLURIUM}$–$C_{CSP3ALKANE}$	95.050	8.114
$C_{CSP3ALKANE}$–$Te_{TELLURIUM}$–$H_{EXCEPTONN,O,S}$	94.600	5.368
$C_{CSP3ALKANE}$–$N_{N=C-/PYR(DELOCLZD)}$–$C_{CSP2ALKENE}$	109.000	8.988
$C_{CSP3ALKANE}$–$N_{N=C-/PYR(DELOCLZD)}$–$N_{N=C-/PYR(DELOCLZD)}$	106.500	8.613
$C_{CSP3ALKANE}$–$N_{N=C-/PYR(DELOCLZD)}$–$N_{=NOAZOXY(LOCAL)}$	99.100	7.115
$C_{CSP2ALKENE}$–$N_{N=C-/PYR(DELOCLZD)}$–$C_{CSP2ALKENE}$	112.600	14.980
$C_{CSP2ALKENE}$–$N_{N=C-/PYR(DELOCLZD)}$–$H_{NHAMINE/IMINE}$	109.800	8.364
$C_{CSP2ALKENE}$–$N_{N=C-/PYR(DELOCLZD)}$–$N_{N=C-/PYR(DELOCLZD)}$	107.500	16.228
$C_{CSP2ALKENE}$–$N_{N=C-/PYR(DELOCLZD)}$–$N_{NSP2PYRROLE}$	115.000	5.368
$H_{NHAMINE/IMINE}$–$N_{N=C-/PYR(DELOCLZD)}$–$N_{N=C-/PYR(DELOCLZD)}$	106.400	11.235
$C_{CSP3ALKANE}$–$C_{CSP2CYCLOPROPENE}$–$C_{CYCLOPROPANE}$	117.200	5.617
$C_{CSP2ALKENE}$–$C_{CSP2CYCLOPROPENE}$–$C_{CYCLOPROPANE}$	137.000	6.866
$H_{EXCEPTONN,O,S}$–$C_{CSP2CYCLOPROPENE}$–$C_{CYCLOPROPANE}$	146.000	4.494
$H_{EXCEPTONN,O,S}$–$C_{CSP2CYCLOPROPENE}$–$C_{CSP2CYCLOPROPENE}$	146.000	4.494
$C_{CSP3ALKANE}$–$N_{NSP3AMMONIUM}$–$C_{CSP3ALKANE}$	107.114	8.976
$C_{CSP3ALKANE}$–$N_{NSP3AMMONIUM}$–$H_{AMMONIUM}$	105.949	6.504
$C_{CSP3ALKANE}$–$N_{NSP3AMMONIUM}$–$O_{AMINEOXIDEOXYGEN}$	109.700	11.048
$H_{AMMONIUM}$–$N_{NSP3AMMONIUM}$–$H_{AMMONIUM}$	106.367	7.602
$C_{CSP3ALKANE}$–$N_{NSP2PYRROLE}$–$C_{CSP2ALKENE}$	120.500	6.117
$C_{CSP2ALKENE}$–$N_{NSP2PYRROLE}$–$C_{CSP2ALKENE}$	124.000	5.368
$C_{CSP2ALKENE}$–$N_{NSP2PYRROLE}$–$H_{NHAMINE/IMINE}$	118.500	6.117
$C_{CSP2ALKENE}$–$N_{NSP2PYRROLE}$–$N_{N=C-/PYR(DELOCLZD)}$	124.000	5.368
$H_{NHAMINE/IMINE}$–$N_{NSP2PYRROLE}$–$N_{N=C-/PYR(DELOCLZD)}$	116.000	6.117
$C_{CSP3ALKANE}$–$O_{OSP2FURAN}$–$C_{CSP2ALKENE}$	109.100	12.483
$C_{CSP3ALKANE}$–$O_{OSP2FURAN}$–$N_{=NOHOXIME}$	110.510	11.235
$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$–$C_{CSP2ALKENE}$	112.000	11.859
$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$–$H_{HOENOL/PHENOL}$	107.000	10.611
$C_{CSP2ALKENE}$–$O_{OSP2FURAN}$–$N_{=NOHOXIME}$	110.850	11.235
$H_{HOENOL/PHENOL}$–$O_{OSP2FURAN}$–$N_{=NOHOXIME}$	101.884	12.483
$C_{CSP2ALKENE}$–$S_{SSP2THIOPHENE}$–$C_{CSP2ALKENE}$	98.500	8.489
$C_{CSP2ALKENE}$–$S_{SSP2THIOPHENE}$–$H_{SHTHIOL}$	94.000	9.362

$C_{CSP3ALKANE}$–$N_{=NOAZOXY(LOCAL)}$–$O_{O=CCARBONYL}$	120.400	7.115
$C_{CSP3ALKANE}$–$N_{=NOAZOXY(LOCAL)}$–$N_{N=C-/PYR(DELOCLZD)}$	114.000	4.744
$C_{CSP3ALKANE}$–$N_{=NOAZOXY(LOCAL)}$–$O_{AMINEOXIDEOXYGEN}$	117.000	11.485
$C_{CSP3ALKANE}$–$N_{=NOAZOXY(LOCAL)}$–$N_{N=AZOXY(LOCAL)}$	118.000	12.109
$C_{CSP2ALKENE}$–$N_{=NOAZOXY(LOCAL)}$–$C_{CSP2ALKENE}$	117.300	15.604
$C_{CSP2ALKENE}$–$N_{=NOAZOXY(LOCAL)}$–$O_{AMINEOXIDEOXYGEN}$	117.000	21.596
$O_{O=CCARBONYL}$–$N_{=NOAZOXY(LOCAL)}$–$N_{N=C-/PYR(DELOCLZD)}$	125.000	4.744
$H_{NHAMINE/IMINE}$–$N_{=NOAZOXY(LOCAL)}$–$O_{AMINEOXIDEOXYGEN}$	108.000	9.737
$H_{NHAMINE/IMINE}$–$N_{=NOAZOXY(LOCAL)}$–$N_{N=AZOXY(LOCAL)}$	106.000	11.422
$O_{AMINEOXIDEOXYGEN}$–$N_{=NOAZOXY(LOCAL)}$–$N_{N=AZOXY(LOCAL)}$	124.200	22.720
$C_{CSP3ALKANE}$–N_{NITRO}–$O_{O=CCARBONYL}$	115.900	12.483
$C_{CSP2ALKENE}$–N_{NITRO}–$O_{O=CCARBONYL}$	116.100	17.726
$O_{O=CCARBONYL}$–N_{NITRO}–$O_{O=CCARBONYL}$	127.700	12.483
$O_{O=CCARBONYL}$–N_{NITRO}–$C_{CYCLOPROPANE}$	114.500	12.733
$C_{CSP3ALKANE}$–$C_{BENZENE(LOCALIZED)}$–$C_{BENZENE(LOCALIZED)}$	122.300	5.867
$C_{CSP2ALKENE}$–$C_{BENZENE(LOCALIZED)}$–$C_{BENZENE(LOCALIZED)}$	121.700	9.487
$C_{CSPALKYNE}$–$C_{BENZENE(LOCALIZED)}$–$C_{BENZENE(LOCALIZED)}$	121.200	5.368
$H_{EXCEPTONN,O,S}$–$C_{BENZENE(LOCALIZED)}$–$C_{BENZENE(LOCALIZED)}$	120.000	6.117
N_{NSP3}–$C_{BENZENE(LOCALIZED)}$–$C_{BENZENE(LOCALIZED)}$	123.000	7.490
$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$–$C_{CSP3ALKANE}$	109.500	8.364
$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$–$H_{EXCEPTONN,O,S}$	110.100	7.365
$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$–$O_{CO-H,C-O-C,O-O}$	109.300	9.612
$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$–N_{NSP2}	110.700	12.858
$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$–$C_{CYCLOPROPANE}$	0.000	8.364
$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$	109.500	8.364
$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$–$C_{C=OCYCLOBUTANONE}$	110.300	6.242
$C_{CSP3ALKANE}$–$C_{CSP3CYCLOBUTANE}$–$O_{OH,O-C(CARBOXYL)}$	109.300	9.612
$C_{CSP2CARBONYL}$–$C_{CSP3CYCLOBUTANE}$–$H_{EXCEPTONN,O,S}$	109.490	6.741
$C_{CSP2CARBONYL}$–$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$	109.500	8.114
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–$H_{EXCEPTONN,O,S}$	107.600	6.866
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–$O_{CO-H,C-O-C,O-O}$	108.700	5.617
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–N_{NSP3}	109.300	10.236
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–N_{NSP2}	111.000	9.487
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–$S_{S-SULFIDE}$	111.500	9.238
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–Si_{SILANE}	110.000	8.489
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–$C_{CYCLOPROPANE}$	112.000	4.494
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–$Ge_{GERMANIUM}$	110.200	4.993
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$	110.100	7.365
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–$C_{CSP2CYCLOBUTENE}$	110.000	6.117

$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–$C_{C=OCYCLOBUTANONE}$	110.490	7.864
$H_{EXCEPTONN,O,S}$–$C_{CSP3CYCLOBUTANE}$–$O_{OH,O-C(CARBOXYL)}$	108.700	5.617
$O_{CO-H,C-O-C,O-O}$–$C_{CSP3CYCLOBUTANE}$–N_{NSP2}	110.500	6.991
$O_{CO-H,C-O-C,O-O}$–$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$	109.100	13.482
N_{NSP3}–$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$	109.470	9.737
Si_{SILANE}–$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$	111.000	5.992
$Ge_{GERMANIUM}$–$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$	110.000	5.742
$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$–$C_{CSP3CYCLOBUTANE}$	109.500	8.364
$C_{CSP2CYCLOBUTENE}$–$C_{CSP3CYCLOBUTANE}$–$C_{CSP2CYCLOBUTENE}$	113.200	6.741
$C_{CSP3ALKANE}$–$C_{CSP2CYCLOBUTENE}$–$C_{CSP3CYCLOBUTANE}$	115.200	6.741
$C_{CSP3ALKANE}$–$C_{CSP2CYCLOBUTENE}$–$C_{CSP2CYCLOBUTENE}$	122.300	5.867
$C_{CSP2ALKENE}$–$C_{CSP2CYCLOBUTENE}$–$C_{CSP3CYCLOBUTANE}$	127.800	9.487
$C_{CSP2ALKENE}$–$C_{CSP2CYCLOBUTENE}$–$C_{CSP2CYCLOBUTENE}$	127.200	9.487
$H_{EXCEPTONN,O,S}$–$C_{CSP2CYCLOBUTENE}$–$C_{CSP3CYCLOBUTANE}$	117.500	6.117
$H_{EXCEPTONN,O,S}$–$C_{CSP2CYCLOBUTENE}$–$C_{CSP2CYCLOBUTENE}$	120.000	6.117
$C_{CSP2CYCLOBUTENE}$–$C_{CSP2CYCLOBUTENE}$–$C_{CSP2CYCLOBUTENE}$	121.700	9.487
$O_{CO-H,C-O-C,O-O}$–$C_{C=OCYCLOBUTANONE}$–$O_{O=CCARBONYL}$	126.000	12.483
$O_{O=CCARBONYL}$–$C_{C=OCYCLOBUTANONE}$–$C_{CSP3CYCLOBUTANE}$	135.100	6.866
N_{NSP2}–$C_{C=OCYCLOBUTANONE}$–$O_{O=CN<(AMIDE)}$	125.800	13.107
$C_{CSP3CYCLOBUTANE}$–$C_{C=OCYCLOBUTANONE}$–$O_{O=CN<(AMIDE)}$	125.000	8.114
$O_{OH,O-C(CARBOXYL)}$–$C_{C=OCYCLOBUTANONE}$–$O_{O=CO-C(ESTER)}$	126.000	12.483
$O_{O=CCARBONYL}$–$C_{C=OCYCLOPROPANONE}$–$C_{CYCLOPROPANE}$	143.600	5.742
$C_{CSP3ALKANE}$–$O_{KETONIUMOXYGEN}$–$C_{KETONIUMCARBON}$	113.900	9.113
$C_{CSP3ALKANE}$–$C_{KETONIUMCARBON}$–$C_{CSP3ALKANE}$	107.700	3.995
$C_{CSP3ALKANE}$–$C_{KETONIUMCARBON}$–$H_{EXCEPTONN,O,S}$	116.900	0.999
$C_{CSP3ALKANE}$–$C_{KETONIUMCARBON}$–$O_{KETONIUMOXYGEN}$	113.500	13.607
$H_{EXCEPTONN,O,S}$–$C_{KETONIUMCARBON}$–$H_{EXCEPTONN,O,S}$	124.500	4.993
$H_{EXCEPTONN,O,S}$–$C_{KETONIUMCARBON}$–$O_{KETONIUMOXYGEN}$	111.700	2.746
$C_{CSP3ALKANE}$–$N_{=NIMINE(LOCALZD)}$–$C_{CSP2ALKENE}$	109.600	8.988
$C_{CSP3ALKANE}$–$O_{OH,O-C(CARBOXYL)}$–$C_{CSP2CARBONYL}$	112.800	15.604
$C_{CSP2CARBONYL}$–$O_{OH,O-C(CARBOXYL)}$–$C_{CSP2CARBONYL}$	106.800	9.612
$C_{CSP2CARBONYL}$–$O_{OH,O-C(CARBOXYL)}$–$H_{COOHCARBOXYL}$	107.700	8.613
$C_{CSP2CARBONYL}$–$O_{OH,O-C(CARBOXYL)}$–$C_{CSP3CYCLOBUTANE}$	110.800	15.604
$C_{CSP2ALKENE}$–$C_{=C=OKETENE}$–$O_{O=CCARBONYL}$	180.000	8.738
$H_{NHAMINE/IMINE}$–$N_{N=N-AZO(LOCAL)}$–$N_{N=N-AZO(LOCAL)}$	106.200	11.235
$C_{CSP2ALKENE}$–$N_{=NOHOXIME}$–$O_{OSP2FURAN}$	107.973	18.725
$C_{CSP3ALKANE}$–$N_{N=AZOXY(LOCAL)}$–$N_{=NOAZOXY(LOCAL)}$	105.200	14.980
$H_{NHAMINE/IMINE}$–$N_{N=AZOXY(LOCAL)}$–$N_{=NOAZOXY(LOCAL)}$	101.500	14.356
$C_{CSP3ALKANE}$–$N_{N(+)=IMMINIUM}$–$C_{CSP3ALKANE}$	117.500	11.235

$C_{CSP3ALKANE}$–$N_{N(+)=IMMINIUM}$–$C_{CSP2ALKENE}$	124.000	5.617
$C_{CSP3ALKANE}$–$N_{N(+)=IMMINIUM}$–$H_{NHAMINE/IMINE}$	116.700	9.362
$C_{CSP2ALKENE}$–$N_{N(+)=IMMINIUM}$–$H_{NHAMINE/IMINE}$	119.400	5.617
$C_{CSP3ALKANE}$–$N_{N(+)=PYRIDINIUM}$–$C_{CSP2ALKENE}$	119.000	9.987
$C_{CSP2ALKENE}$–$N_{N(+)=PYRIDINIUM}$–$C_{CSP2ALKENE}$	120.000	11.235
$C_{CSP2ALKENE}$–$N_{N(+)=PYRIDINIUM}$–$H_{NHAMINE/IMINE}$	109.800	9.362
$H_{EXCEPTONN,O,S}$–$CH_{CFERROCENEH}$–$CH_{CFERROCENEH}$	126.000	4.993
$C_{CSP3ALKANE}$–$CP_{CFERROCENEC}$–$CP_{CFERROCENEC}$	126.000	5.867
$C_{CSP2ALKENE}$–$N_{=NOAXOXY(DELOC)}$–$C_{CSP2ALKENE}$	119.100	15.604
$C_{CSP2ALKENE}$–$N_{=NOAXOXY(DELOC)}$–$O_{AMINEOXIDEOXYGEN}$	121.700	21.596
$C_{CSP2ALKENE}$–$N_{=NOAXOXY(DELOC)}$–$N_{N=AZOXY(DELOC)}$	114.000	14.356
$O_{AMINEOXIDEOXYGEN}$–$N_{=NOAXOXY(DELOC)}$–$N_{N=AZOXY(DELOC)}$	136.000	23.469
$C_{CSP2ALKENE}$–$N_{N=AZOXY(DELOC)}$–$N_{=NOAXOXY(DELOC)}$	102.000	9.987
$C_{CSP3ALKANE}$–$O_{>NOHHYDROXYAMINE}$–$N_{>NOHHYDROXYAMINE}$	107.500	13.732
$H_{OHALCOHOL}$–$O_{>NOHHYDROXYAMINE}$–$N_{>NOHHYDROXYAMINE}$	104.000	11.547
$C_{CSP3ALKANE}$–$N_{>NOHHYDROXYAMINE}$–$C_{CSP3ALKANE}$	98.500	5.742
$C_{CSP3ALKANE}$–$N_{>NOHHYDROXYAMINE}$–$H_{NHAMINE/IMINE}$	104.500	8.738
$C_{CSP3ALKANE}$–$N_{>NOHHYDROXYAMINE}$–$O_{>NOHHYDROXYAMINE}$	102.500	15.916
$H_{NHAMINE/IMINE}$–$N_{>NOHHYDROXYAMINE}$–$H_{NHAMINE/IMINE}$	103.000	9.737
$H_{NHAMINE/IMINE}$–$N_{>NOHHYDROXYAMINE}$–$O_{>NOHHYDROXYAMINE}$	101.500	9.050
$C_{CSP2CARBONYL}$–$O_{O-ANHYDRIDE(LOCL)}$–$C_{CSP2CARBONYL}$	110.000	10.361
$C_{CSP3ALKANE}$–$N_{NSP3HYDRAZINE}$–$C_{CSP3ALKANE}$	106.980	13.732
$C_{CSP3ALKANE}$–$N_{NSP3HYDRAZINE}$–$H_{NHAMINE/IMINE}$	102.350	8.738
$C_{CSP3ALKANE}$–$N_{NSP3HYDRAZINE}$–$N_{NSP3HYDRAZINE}$	102.470	9.987
$H_{NHAMINE/IMINE}$–$N_{NSP3HYDRAZINE}$–$H_{NHAMINE/IMINE}$	102.900	8.114
$H_{NHAMINE/IMINE}$–$N_{NSP3HYDRAZINE}$–$N_{NSP3HYDRAZINE}$	104.580	10.860
$C_{CSP3ALKANE}$–$P_{>P=OPHOSPHATE}$–$C_{CSP3ALKANE}$	109.200	4.993
$C_{CSP3ALKANE}$–$P_{>P=OPHOSPHATE}$–$H_{EXCEPTONN,O,S}$	105.500	6.242
$C_{CSP3ALKANE}$–$P_{>P=OPHOSPHATE}$–$O_{CO-H,C-O-C,O-O}$	107.200	4.494
$C_{CSP3ALKANE}$–$P_{>P=OPHOSPHATE}$–$O_{O=CCARBONYL}$	117.500	9.987
$C_{CSP3ALKANE}$–$P_{>P=OPHOSPHATE}$–$O_{OP=OPHOSPHATE}$	104.700	6.866
$H_{EXCEPTONN,O,S}$–$P_{>P=OPHOSPHATE}$–$H_{EXCEPTONN,O,S}$	104.500	5.243
$H_{EXCEPTONN,O,S}$–$P_{>P=OPHOSPHATE}$–$O_{CO-H,C-O-C,O-O}$	101.200	7.989
$H_{EXCEPTONN,O,S}$–$P_{>P=OPHOSPHATE}$–$O_{O=CCARBONYL}$	116.700	10.361
$H_{EXCEPTONN,O,S}$–$P_{>P=OPHOSPHATE}$–$O_{OP=OPHOSPHATE}$	102.000	8.738
$O_{CO-H,C-O-C,O-O}$–$P_{>P=OPHOSPHATE}$–$O_{CO-H,C-O-C,O-O}$	102.900	13.732
$O_{CO-H,C-O-C,O-O}$–$P_{>P=OPHOSPHATE}$–$O_{O=CCARBONYL}$	117.200	11.734
$O_{O=CCARBONYL}$–$P_{>P=OPHOSPHATE}$–$O_{OP=OPHOSPHATE}$	116.200	14.730
$O_{OP=OPHOSPHATE}$–$P_{>P=OPHOSPHATE}$–$O_{OP=OPHOSPHATE}$	103.500	11.235

$C_{CSP3ALKANE}$–$S_{>SO2SULFONAMIDE}$–$O_{O=CCARBONYL}$	108.728	9.325
$C_{CSP3ALKANE}$–$S_{>SO2SULFONAMIDE}$–$N_{NSP3SULFONAMIDE}$	102.092	7.240
$H_{EXCEPTONN,O,S}$–$S_{>SO2SULFONAMIDE}$–$O_{O=CCARBONYL}$	107.614	9.163
$H_{EXCEPTONN,O,S}$–$S_{>SO2SULFONAMIDE}$–$N_{NSP3SULFONAMIDE}$	100.776	11.410
$O_{O=CCARBONYL}$–$S_{>SO2SULFONAMIDE}$–$O_{O=CCARBONYL}$	122.966	8.239
$O_{O=CCARBONYL}$–$S_{>SO2SULFONAMIDE}$–$N_{NSP3SULFONAMIDE}$	107.883	20.747
$C_{CSP3ALKANE}$–$N_{NSP3SULFONAMIDE}$–$C_{CSP3ALKANE}$	111.163	16.178
$C_{CSP3ALKANE}$–$N_{NSP3SULFONAMIDE}$–$H_{NHAMINE/IMINE}$	109.505	7.078
$C_{CSP3ALKANE}$–$N_{NSP3SULFONAMIDE}$–$S_{>SO2SULFONAMIDE}$	109.551	14.855
$H_{NHAMINE/IMINE}$–$N_{NSP3SULFONAMIDE}$–$H_{NHAMINE/IMINE}$	109.688	7.590
$H_{NHAMINE/IMINE}$–$N_{NSP3SULFONAMIDE}$–$S_{>SO2SULFONAMIDE}$	103.248	7.228
$C_{CSP3ALKANE}$–$O_{OP=OPHOSPHATE}$–$P_{>P=OPHOSPHATE}$	118.800	11.235
$H_{OHALCOHOL}$–$O_{OP=OPHOSPHATE}$–$P_{>P=OPHOSPHATE}$	112.000	4.744

Table B.4: Bond bending terms from the protein set of MM3 parameters

	θ_0 (deg)	K_θ (eV)
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$	109.500	8.364
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–C_{Amide}	110.600	9.987
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–$C_{Carboxyl}$	110.600	9.987
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–C_{Phenyl}	110.600	6.741
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–$C_{Alkene(His/TrpC=C)}$	110.600	6.741
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–N_{Amide}	109.480	9.362
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–$N_{Ammonium}$	103.500	7.115
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–$N_{Guanidinium}$	109.480	9.362
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–$O_{Alcohol/Phenol}$	107.500	10.361
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–$S_{Sulfide}$	108.000	9.238
$C_{Alkane(Csp3)}$–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	109.800	7.365
C_{Amide}–$C_{Alkane(Csp3)}$–N_{Amide}	109.500	10.611
C_{Amide}–$C_{Alkane(Csp3)}$–$N_{Ammonium}$	110.740	13.045
C_{Amide}–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	109.490	6.741
$C_{Carboxyl}$–$C_{Alkane(Csp3)}$–N_{Amide}	109.500	10.611
$C_{Carboxyl}$–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	109.490	6.741
C_{Phenyl}–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	109.500	6.866
$C_{Alkene(His/TrpC=C)}$–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	109.500	6.866
N_{Amide}–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	111.000	9.487
$N_{Ammonium}$–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	108.800	6.242

$N_{Guanidinium}$–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	111.000	9.487
$O_{Alcohol/Phenol}$–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	110.000	10.236
$S_{Sulfide}$–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	110.800	9.238
$H_{Hydrogen(CH)}$–$C_{Alkane(Csp3)}$–$H_{Hydrogen(CH)}$	107.600	6.866
$C_{Alkane(Csp3)}$–C_{Amide}–N_{Amide}	114.400	7.115
$C_{Alkane(Csp3)}$–C_{Amide}–O_{Amide}	123.500	10.611
N_{Amide}–C_{Amide}–O_{Amide}	124.800	13.357
N_{Amide}–C_{Amide}–$H_{Hydrogen(CH)}$	109.300	5.493
O_{Amide}–C_{Amide}–$H_{Hydrogen(CH)}$	119.200	10.611
$C_{Alkane(Csp3)}$–$C_{Carboxyl}$–$O_{Carboxylate}$	125.100	10.611
$O_{Carboxylate}$–$C_{Carboxyl}$–$O_{Carboxylate}$	134.000	9.987
$C_{Alkane(Csp3)}$–C_{Phenyl}–C_{Phenyl}	122.300	5.867
C_{Phenyl}–C_{Phenyl}–C_{Phenyl}	121.700	9.487
C_{Phenyl}–C_{Phenyl}–$C_{Alkene(His/TrpC=C)}$	121.700	9.487
C_{Phenyl}–C_{Phenyl}–$N_{Pyrrole}$	120.000	5.368
C_{Phenyl}–C_{Phenyl}–$O_{Alcohol/Phenol}$	120.000	7.490
C_{Phenyl}–C_{Phenyl}–$H_{Hydrogen(CH)}$	120.000	6.117
$N_{Guanidinium}$–$C_{Guanidinium}$–$N_{Guanidinium}$	120.000	4.993
$C_{Alkane(Csp3)}$–$C_{Alkene(His/TrpC=C)}$–C_{Phenyl}	122.300	5.867
$C_{Alkane(Csp3)}$–$C_{Alkene(His/TrpC=C)}$–$C_{Alkene(His/TrpC=C)}$	122.300	5.867
$C_{Alkane(Csp3)}$–$C_{Alkene(His/TrpC=C)}$–$N_{Imidazolium}$	120.000	5.368
C_{Phenyl}–$C_{Alkene(His/TrpC=C)}$–$C_{Alkene(His/TrpC=C)}$	122.000	9.487
$C_{Alkene(His/TrpC=C)}$–$C_{Alkene(His/TrpC=C)}$–$N_{Pyrrole}$	120.000	5.368
$C_{Alkene(His/TrpC=C)}$–$C_{Alkene(His/TrpC=C)}$–$N_{Imidazolium}$	120.000	9.987
$C_{Alkene(His/TrpC=C)}$–$C_{Alkene(His/TrpC=C)}$–$H_{Hydrogen(CH)}$	120.000	6.117
$N_{Pyrrole}$–$C_{Alkene(His/TrpC=C)}$–$H_{Hydrogen(CH)}$	113.500	4.993
$N_{Imidazolium}$–$C_{Alkene(His/TrpC=C)}$–$H_{Hydrogen(CH)}$	113.500	4.993
$N_{Imidazolium}$–$C_{Imidazolium(NC-N)}$–$N_{Imidazolium}$	120.000	4.993
$N_{Imidazolium}$–$C_{Imidazolium(NC-N)}$–$H_{Hydrogen(CH)}$	120.000	4.993
$C_{Alkane(Csp3)}$–N_{Amide}–$C_{Alkane(Csp3)}$	122.500	9.487
$C_{Alkane(Csp3)}$–N_{Amide}–C_{Amide}	121.100	20.223
$C_{Alkane(Csp3)}$–N_{Amide}–H_{Amide}	122.400	2.372
C_{Amide}–N_{Amide}–H_{Amide}	118.500	7.240
H_{Amide}–N_{Amide}–H_{Amide}	123.000	5.118
$C_{Alkane(Csp3)}$–$N_{Ammonium}$–$C_{Alkane(Csp3)}$	108.600	7.864
$C_{Alkane(Csp3)}$–$N_{Ammonium}$–$H_{Ammonium}$	109.470	6.242
$H_{Ammonium}$–$N_{Ammonium}$–$H_{Ammonium}$	104.500	6.242
$C_{Alkane(Csp3)}$–$N_{Guanidinium}$–$C_{Guanidinium}$	120.400	9.113
$C_{Alkane(Csp3)}$–$N_{Guanidinium}$–$H_{Guanidinium}$	122.400	2.372

$C_{Guanidinium}$–$N_{Guanidinium}$–$H_{Guanidinium}$	120.500	7.240
$H_{Guanidinium}$–$N_{Guanidinium}$–$H_{Guanidinium}$	123.000	5.118
C_{Phenyl}–$N_{Pyrrole}$–$C_{Alkene(His/TrpC=C)}$	124.000	5.368
C_{Phenyl}–$N_{Pyrrole}$–$H_{Pyrrole}$	118.000	4.494
$C_{Alkene(His/TrpC=C)}$–$N_{Pyrrole}$–$H_{Pyrrole}$	118.000	4.494
$C_{Alkene(His/TrpC=C)}$–$N_{Imidazolium}$–$C_{Imidazolium(NC-N)}$	115.000	5.368
$C_{Alkene(His/TrpC=C)}$–$N_{Imidazolium}$–$H_{Imidazolium}$	110.000	6.242
$C_{Imidazolium(NC-N)}$–$N_{Imidazolium}$–$H_{Imidazolium}$	110.000	6.242
$C_{Alkane(Csp3)}$–$O_{Alcohol/Phenol}$–$H_{Alcohol/Phenol}$	106.800	9.362
C_{Phenyl}–$O_{Alcohol/Phenol}$–$H_{Alcohol/Phenol}$	109.000	4.494
$H_{Alcohol/Phenol}$–$O_{Alcohol/Phenol}$–$H_{Alcohol/Phenol}$	105.000	7.864
$C_{Alkane(Csp3)}$–$S_{Sulfide}$–$C_{Alkane(Csp3)}$	95.900	10.486
$C_{Alkane(Csp3)}$–$S_{Sulfide}$–$S_{Sulfide}$	102.000	12.483
$C_{Alkane(Csp3)}$–$S_{Sulfide}$–$H_{Thiol(SH)}$	96.000	8.114

Appendix C

Bond valence parameters

The R_{ij} values in the table below are taken from Brese and O'Keefe, Acta Cryst B, **47**, 192-197 (1991). Values for anions other than O, F and Cl can be found in that paper along with a discussion of how they were determined. These values are used when supplying the `BVS ::` keyword to `RMCProfile`.

Ideal bond lengths, d, may be calculated using these values:

$$d_{ij} = R_{ij} - B \ln\left(\frac{V}{CN}\right) \tag{C.1}$$

Where $B = 0.37$, V = *valence* and CN = *coordination number* (of central atom). It should be noted that this calculation assumes a regular coordination environment, i.e. the valence of atom i would be satisfied with CN equal bonds of length d_{ij}.

Importantly, the bond valence sum for an atom can be calculated as the sum of the valences of the individual bonds:

$$V = \sum v_{ij} \tag{C.2}$$

$$v_{ij} = \exp\left(R_{ij} - d_{ij}/B\right) \tag{C.3}$$

Table C.1: Bond valence parameters, R_{ij}

Cation	O	F	Cl	Cation	O	F	Cl
Ac III	2.24	2.13	2.63	Mn IV	1.753	1.71	2.13
Ag I	1.805	1.80	2.09	Mn VII	1.79	1.72	2.17
Al III	1.651	1.545	2.03	Mo VI	1.907	1.81	2.28
Am III	2.11	2.00	2.48	N III	1.361	1.37	1.75
As III	1.789	1.70	2.16	N V	1.432	1.36	1.8
As V	1.767	1.62	2.14	Na I	1.8	1.677	2.15
Au III	1.833	1.81	2.17	Nb V	1.911	1.87	2.27
B III	1.371	1.31	1.74	Nd III	2.117	2.008	2.492
Ba II	2.29	2.19	2.69	Ni II	1.654	1.599	2.02
Be II	1.381	1.28	1.76	Os IV	1.811	1.72	2.19
Bi III	2.09	1.99	2.48	P V	1.604	1.521	1.99
Bi V	2.06	1.97	2.44	Pb II	2.112	2.03	2.53
Bk III	2.08	1.96	2.46	Pb IV	2.042	1.94	2.43
Br VII	1.81	1.72	2.19	Pd II	1.792	1.74	2.05
C IV	1.39	1.32	1.76	Pr III	2.135	2.022	2.5
Ca II	1.967	1.842	2.37	Pt II	1.768	1.68	2.05
Cd II	1.904	1.811	2.23	Pt IV	1.879	1.759	2.17
Ce III	2.151	2.036	2.52	Pu III	2.11	2	2.48
Ce IV	2.028	1.995	2.41	Rb I	2.26	2.16	2.65
Cf III	2.07	1.95	2.45	Re VII	1.97	1.86	2.23
Cl VII	1.632	1.55	2.00	Rh III	1.791	1.71	2.17
Cm III	2.23	2.12	2.62	Ru IV	1.834	1.74	2.21
Co II	1.692	1.64	2.01	S IV	1.644	1.6	2.02
Co III	1.70	1.62	2.05	S VI	1.624	1.56	2.03
Cr II	1.73	1.67	2.09	Sb III	1.973	1.9	2.35
Cr III	1.724	1.64	2.08	Sb V	1.942	1.8	2.3
Cr VI	1.794	1.74	2.12	Sc III	1.849	1.76	2.23
Cs I	2.42	2.33	2.79	Se IV	1.811	1.73	2.22
Cu I	1.593	1.60	1.85	Se VI	1.788	1.69	2.16
Cu II	1.679	1.60	2.00	Si IV	1.624	1.58	2.03
Dy III	2.036	1.922	2.41	Sm III	2.088	1.977	2.466
Er III	2.01	1.906	2.39	Sn II	1.984	1.925	2.36
Eu II	2.147	2.04	2.53	Sn IV	1.905	1.84	2.28
Eu III	2.076	1.961	2.455	Sr II	2.118	2.019	2.51
Fe II	1.734	1.65	2.06	Ta V	1.92	1.88	2.3
Fe III	1.759	1.67	2.09	Tb III	2.049	1.936	2.427

Ga III	1.73	1.62	2.07	Te IV	1.977	1.87	2.37
Gd III	2.065	1.95	2.445	Te VI	1.917	1.82	2.3
GeIV	1.748	1.66	2.14	Th IV	2.167	2.07	2.55
H I	0.95	0.92	1.28	Ti III	1.791	1.723	2.17
Hf IV	1.923	1.85	2.30	Ti IV	1.815	1.76	2.19
Hg I	1.90	1.81	2.28	Tl I	2.172	2.15	2.56
Hg II	1.93	1.90	2.25	Tl III	2.003	1.88	2.32
Ho III	2.023	1.908	2.401	Tm III	2	1.842	2.38
I V	2.00	1.90	2.38	U IV	2.112	2.034	2.48
I VII	1.93	1.83	2.31	U VI	2.075	1.966	2.46
In III	1.902	1.79	2.28	V III	1.743	1.702	2.19
Ir V	1.916	1.82	2.30	V IV	1.784	1.7	2.16
K I	2.13	1.99	2.52	V V	1.803	1.71	2.16
La III	2.172	2.057	2.545	W VI	1.921	1.83	2.27
Li I	1.466	1.36	1.91	Y III	2.014	1.904	2.4
Lu III	1.971	1.876	2.361	Yb III	1.985	1.875	2.371
Mg II	1.693	1.581	2.08	Zn II	1.704	1.62	2.01
Mn II	1.79	1.698	2.13	Zr IV	1.937	1.854	2.33
Mn III	1.76	1.66	2.14				

Appendix D

X-ray coefficients

To use `RMCProfile` with real-space X-ray data requires an approximation which involves treating the data as neutron data with different weightings. This is possible because we have removed the Q-dependent part and can treat the atoms as having a constant 'scattering length' equivalent to the atomic number, Z. There are a few points to note when using this method.

1. You will need to normalise your X-ray G(r) so that it is scaled from 0 to 1
2. Use the `NEUTRON_REAL_SPACE_DATA` keyword in the RMC .dat file to introduce these data to your refinement.
3. You will need to provide a `NEUTRON_COEFFICIENTS ::` line under in the `NEUTRON_REAL_SPACE_DATA` block.
4. The coefficients for this line can be calculated in the same way as Faber-Ziman coefficients for neutron data, but replacing b with Z and normalising the result so that the coefficients sum to 1. An example of this for the $GaPO_4$ example is given below.

Table D.1: Configuration information

Atom	Z	Concentration
Ga	31	$^1/_6$
P	15	$^1/_6$
O	6	$^4/_6$

The coefficients for this system are calculated as follows:

$$(\sum_i c_i Z_i)^2 = c_{Ga}^2 Z_{Ga}^2 + 2(c_{Ga} Z_{Ga} c_P Z_P) + 2(c_{Ga} Z_{Ga} c_O Z_O) + c_P^2 Z_P^2 + 2(c_P Z_P c_O Z_O)) + c_O^2 Z_O^2$$

Table D.2: Coefficients

Ga	P	O	
26.6944	25.8333	55.1111	Ga
	6.2500	26.6667	P
		28.4444	O

To use these coefficients with `RMCProfile` you need to divide by the total so that they add up to 1. Therefore the final values for the `NEUTRON_COEFFICIENTS` line are as shown below:

Table D.3: Normalised coefficients

Ga	P	O	
0.1580	0.1529	0.3261	Ga
	0.0370	0.1577	P
		0.1683	O

Appendix E

Automated weight optimization

In refinements that involve more than one dataset (including restraints), assigning weights manually is challenging. In RMCProfile, this task can be accomplished using an automated procedure that assigns weights by analyzing statistical correlations between changes in each residual term and the total residual. Weights are adjusted during fit so that all the datasets/restraints would have comparable contributions to the total residual while forcing all the residual components to decrease over time to their respective preset minimal values. The adjustment is performed automatically as the fit progresses. The user has an option to increase the relative weight of a specific dataset manually without stopping the fit. Also, the user must specify a desired fraction of accepted moves to be maintained during the fit and the minimal value of a residual for each dataset/restraint. Experience demonstrates that this weight-optimization option greatly facilitates conversion while significantly reducing fitting times even for problems that include only 2-3 datasets.

The algorithm of automated weight adjustments works as follows. The total residual after the n^{th} atomic move is:

$$R^{tot}(n) = \frac{1}{T}\sum_{i=1}^{N_{ds}} w_i R_i(n) \tag{E.1}$$

where N_{ds} is the number of fitted datasets, including restraints, $R_i(n)$ is the residual for a dataset i after the n^{th} move; w_i is the weight assigned to the i^{th} dataset, and T is the global weight rescaling parameter.

According to the Metropolis minimization algorithm that underlies RMCProfile, moves are accepted so that $R^{tot}(n)$ decreases over time. Therefore, the changes at each move, $\Delta R^{tot}(n) = R^{tot}(n) - R^{tot}(n-1)$, are mostly negative. The adjustment is performed after every N generated atomic moves, where N is specified by the user. During each update, the weight-adjustment routine evaluates sequences of $\Delta R_i(n)$ for each dataset/restraint collected since the last evaluation. A set of weights is sought that yields a positive Pearson correlation coefficient for each dataset i versus R^{tot}. Only those datasets/restraints are included in this analysis that have their residual values greater than the minimal values set by the user.

The optimization procedure is activated by a keyword 'WEIGHT_OPTIMIZATION ::' in the stem-name.dat file. Besides, a file 'optimization.dat' must be present in the working directory. This is a text file with the following content:

2000
0.2
.false.
.false.
.false.
.false.

1. The entry in the 1^{st} line specifies the **number of generated moves N after which the weight adjustment is to be performed**. This number should be large enough to collect adequate statistics on $\Delta R_i(n)$ but not too large as otherwise the efficacy of the procedure will be reduced. Experience shows that **2000 to 3000 moves is optimal**.

2. The value in the 2nd line specifies a desired fraction of accepted atomic moves. According to the Metropolis algorithm all moves that result in negative ΔR^{tot} are accepted unconditionally. Thus, the fraction set by the user can only affect the probability of acceptance of unfavorable moves, which lead to positive ΔR^{tot}. If weights of all the datasets/restraints are multiplied by a factor $1/T$ close to zero, nearly all unfavorable moves will be accepted. In contrast, scaling these weights by a large value of $1/T$ will result in a nearly complete rejection of such 'bad' moves. At each weight evaluation, the routine adjusts $1/T$ so, as to provide the acceptance rate equal or greater than the specified value. For example, in early stages of a fit, inevitably, nearly all moves are accepted. If the minimal acceptance rate is set too high, the fit may not converge whereas if this value is too low, few moves will be accepted, and the fit would freeze. A value of 0.2 is a reasonable choice but it can be varied according to specific needs.

3. 3.The false/true entries on the next 4 lines control the generation of diagnostic files. For a normal use, these entries should be kept as 'false' (no diagnostic files produced).

The parameters specified in the optimization.dat file can be changed by editing this file (any text editor) without interrupting the fit.

The user can also modify relative weights assigned to specific datasets/restrains by modifying the content of the file **adv_opt.dat**, which is generated by the program at the beginning of the fit. Any text editor can be used, and changes can be performed without interrupting the fit.

This **adv_opt.dat** file contains two columns of numbers followed by the third column which lists the corresponding datasets/restraints. The 1^{st} column lists the weight scale factors for each dataset/restraint included in the fit. By default, these values are set to 1. The user can increase the effect of a dataset on the total residual by increasing the scale factor applied to its weight. In the example below, setting this scale factor for the EXAFS1 data to 1.5 while leaving the rest at 1.0 will cause the residual for this EXAFS dataset to decrease faster. The 2^{nd} column specified the minimum residual values for each dataset. The default values are zero. In the absence of systematic errors, these minimum values would be determined by the counting statistics. However, in most practical cases, systematic errors prevail. Forcing the residual to zero is likely to result in overfitting. At present, the extent of systematic errors is generally unknown. Therefore, sensible minimum values of the residual for each dataset are determined by the user empirically. Once the residual for a given dataset reaches its minimum value, its weight for the next evaluation cycle is set to zero. Over time, this residual will oscillate about its preset minimum value.

Note: every time RMCProfile is restarted, the program will re-generate the **adv_opt.dat** file with the default values – unity for the scale factors and zeros for the minimum residual values. The user will have to edit this file again to set the parameters to their desired values.

EXAMPLE of **adv_opt.dat** file:

weight factor	minimum chi	experiment
0.100E+01	0.000E+00	EXPER#1
0.100E+01	0.000E+00	EXPER#2
0.100E+01	0.746E-02	FAST_BRAGG
0.150E+01	0.341E-01	EXAFS#1
0.100E+01	0.198E-01	EXAFS#2

Guideline for using the procedure:

1. Set all the weights in the stemname.dat file that controls the fit to 1.0.
2. Generate the **optimization.dat** file, like described above, and place it in the working directory.
3. Start the fit. The program will generate the **adv_opt.dat** file and the **stemname.log** file with the numbers of accepted/generated moves and values of the residuals for each dataset/restraint.
4. Monitor the progress of the fit by examining the log file. If needed, modify the **adv_opt.dat** file without stopping the fit.
5. If the fit had to be stopped and re-started, leave the initial weights in the **stemname.dat** file at 1.0. After re-starting, the program will generate new **stemname.log** and **adv_opt.dat** files (the latter with the default values). Edit **adv_opt.dat** file if needed.

Appendix F

Correction for resolution effect

6.0.1 Gaussian assumption

First, it should be pointed out that the conventional way (i. e. with *qdamp* and *qbroad* parameters, as that implemented in PDFgui) of correcting the resolution effect is based on the assumption that the peak shape in both real and reciprocal space is Gaussian. Currently, we are working on correcting the resolution effect going beyond the Gaussian assumption and will implement and document the advanced approach of conducting the resolution correction after testing for robustness.

N.B. Atomic distances falling into a certain bin means they are not distinguishable. It further means any broadening effect with width smaller than the bin size being used won't make sense. So, when the input width for the Gaussian broadening function is detected to be smaller than the bin size, we will force it to be identical to the bin size.

Here, the *qdamp* parameter holds the same meaning as that in PDFgui and therefore one can transfer the value used in PDFgui refinement into RMCProfile. The *qbroad* parameter, however, is not identical to that used in PDFgui and we will present some discussions about it as follows. To begin with, the convolution Gaussian peak width in RMCProfile is defined as:

$$\sigma = Q_{broad}^{RMC} r \tag{F.1}$$

The equation here originates from that given in section-5.4 in the manual of PDFgui. Here, all the terms irrelevant to the instrument effect, i. e. $\sigma^{'}$, δ_1/r and δ_2/r^2, are ignored, since all of these terms directly and explicitly come from atom positions in the supercell of RMCProfile and therefore there is no need for parameterization. Comparing the Eqn-F.1 and that given in section-5.4 in the manual of PDFgui, one can conclude:

$$Q_{broad}^{RMC} = \sigma^{'} \cdot Q_{broad}^{PDFgui} \tag{F.2}$$

in which $\sigma^{'}$ originally refers to the PDF peak width without correlation or instrument broadening effect. In another word, $\sigma^{'}$ purely comes from thermal parameter of atoms involved. Suppose one is using Si standard sample to obtain the Q_{broad}^{PDFgui} parameter, then

$$\sigma^{'} = \sqrt{\sigma_{Si}^2 + \sigma_{Si}^2} = \sqrt{2}\sigma_{Si} \tag{F.3}$$

where σ_{Si}^2 pops up twice simply because we are calculating pair distribution function and for each pair, two atoms are involved (see the paper by Jeong, et al. for more details: Ref-[22]). Therefore, one has:

$$Q_{broad}^{RMC} = \sqrt{2}\sigma_{Si} \cdot Q_{broad}^{PDFgui} = \sqrt{2}u_{iso}^{Si} \cdot Q_{broad}^{PDFgui} \tag{F.4}$$

Hence, to get a reasonable value of Q_{broad}^{RMC} to be used in RMCProfile, one can first fit Si standard sample in PDFgui to obtain both u_{iso}^{Si} and Q_{broad}^{PDFgui}, after which the value of Q_{broad}^{RMC} can be obtained following Eqn. F.4.

The second aspect worth pointing out here is again relevant to the equation in section-5.4 in the manual of PDFgui. From the equation therein, one may notice that the actual width of the Gaussian peaks to be convolved into the PDF pattern not only depends on r but also depends on the peak width associated with each single type of atom pairs – pulling the $\sigma_{ij}^{'}$ term inside the square and multiply to the last term will give the term $\sigma_{ij}^{'} Q_{broad}^{PDFgui} r_{ij}$. However, in RMCProfile, since the thermal effect is considered following the 'ensemble-average' approach (see, e. g. Ref-[23] by T. Proffen, et al. - top of left column in page-573), one does not have access to the thermal parameter for each single type of atom pairs during the fitting. Therefore, in RMCProfile, we have to assume the parameter Q_{broad}^{RMC} is the same across all types of atom pairs. In fact, the same assumption is also applied in DISCUS (see Eqn. 4.15 in DISCUS cookbook by Prof. Neder & T. Proffen [24]).

6.0.2 Beyond Gaussian assumption

`RMCProfile` also allows using those non-Gaussian profile functions as defined in GSAS-I or those user-defined shape functions for describing peaks. To enable the using of user-defined shape functions, we implement a generic approach to tabulate the profile data (which will be used to generate the resolution matrix) which could be extracted from suitable Rietveld program. In this way, the profile implementation in `RMCProfile` is no longer restricted to specific profile functions. Instead, users can use whatever profile function suitable for describing peaks specific to certain diffractometer and transfer the profile data to `RMCProfile` in a tabulated manner. To prepare the tabulated profile data (i.e. resolution matrix), we developed a tool named 'Topas4RMC' - see 6.1.25 for details. As follows, the normal practice of conducting resolution correction using non-Gaussian peaks shapes will be presented in a stepwise manner.

1. A standard sample (e.g., NIST Si) should be measured in the same condition as that for samples of interest to obtain the Bragg pattern. Here, by 'condition', we mean both sample environment and container.
2. Rietveld refinement should be conducted for the Bragg pattern measured for the standard sample mentioned in step-1.
3. Prepare resolution matrix.
   (a) Using genetic Topas approach.
      i. Use the GUI 'Topas4RMC' to prepare the needed resolution matrix for next step, by feeding in the Rietveld refinement for the standard sample Bragg pattern. The GUI is shown as follows,
      Detailed instruction about how-to can be found in the 'Help' tab of the GUI.
      ii. Add in a single header line in the generated resolution matrix file in step-, similar to the one presented as follows,
      6 5761 1.2 0.005 4 1440
      The first number specifies the number of banks, which should be consistent with both the number of banks used for standard sample Rietveld refinement and the number of banks used to obtain the finally merged $S(Q)$ data. The second number

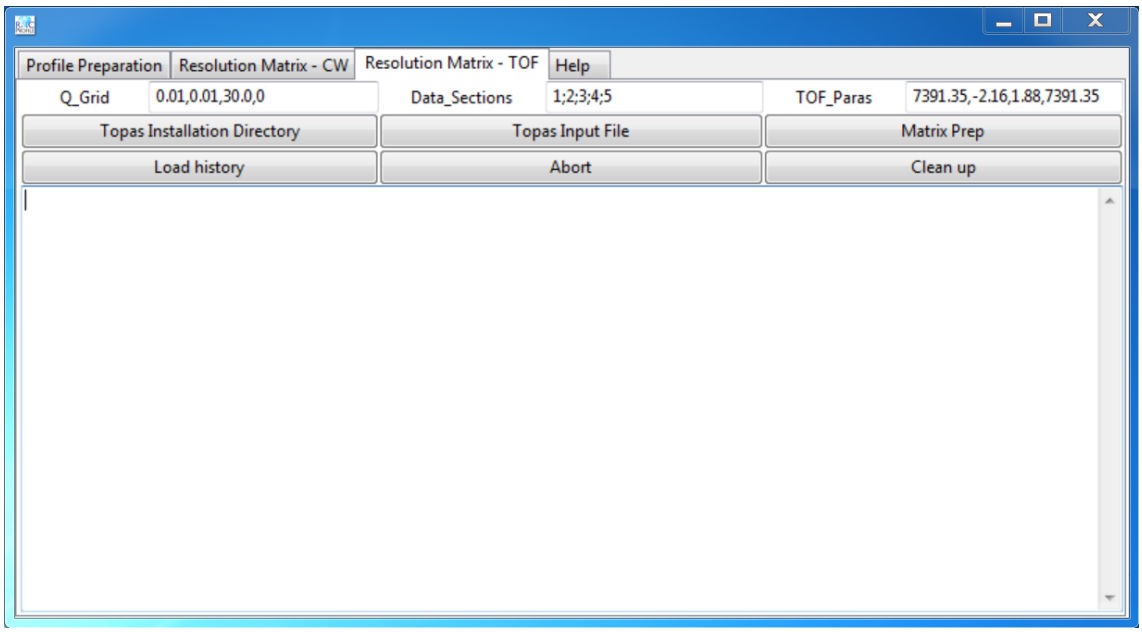


Figure F.1: Screenshot of the interface for preparing resolution matrix with Topas4RMC GUI.

specifies the number of lines for each bank in the generated resolution matrix file. Grabbing the resolution matrix file, one can see the number of lines in total. Divide this number by the number of banks (the first number here) should give the second number needed. The third number tells the starting position in Q-space experimental data. The fourth number specifies the interval (in terms of corresponding peak center position) between each line in resolution matrix file. This number should stay consistent with whatever specified in the GUI when preparing the resolution matrix file in step-. The fifth number is the ratio of step in Q-space in $S(Q)$ experimental data to the step (i.e. the fourth number) in resolution matrix file. The final number tells the number of Q-points in Q-space experimental data.

(b) Using conventional GSAS approach.

i. Once we are done with Rietveld refinement for standard sample in GSAS, we can use the get_gsas_bragg tool (refer to 6.1) to prepare the needed .inst file.

4. Prepare the differential cross section (DCS) file containing all bank data. An example is given below,

   1.2 29.00 0.02

   7391.35 -2.16 1.88

   7391.35

   8003 6

   ...

   1.72E-01 5.65E-01 0.00E+00 ...

   ...

   The first line specifies the staring, ending points and the internal in Q-space experimental data. The second and third lines are there for historic reason and general users can just leave

them as it is exactly shown above. The fourth line tells the total number of Q points in the DCS file and the total number of banks. Starting from the fifth line, each single line contains the corresponding Q position and the intensity and uncertainty for each bank. Suppose we just have 6 banks in total, we would then have in total 13 columns starting for all lines from the fifth one.

The DCS file provided here is for weighting the resolution matrix corresponding to each single bank to obtain a merged version to be used for correcting the resolution effect in Q-space.

5. The following three subordinate keywords are needed to introduce resolution correction for either neutron real space data, neutron reciprocal space data or X-ray data section.

   > INST_RESOLUTION_FILE ::

   > BANK_INFO_FILE ::

   > BANK_WEIGHT_TYPE ::

   One need to notice that the file following 'INST_RESOLUTION_FILE' is different in format when using GSAS and Topas profile for describing peak shape.

   To use Topas profile, one needs to explicitly put the following keyword in the main .dat file,

   TOPAS_RESOLUTION_MATRIX ::

Appendix G

Correction for nano-size effect

Detailed discussion for introducing correction for the nano-size effect when simulating PDF pattern for nano-system in RMCProfile is discussed by Y. Zhang, et al [26]. However, when the shape of nano-particle is irregular, one cannot derive an analytical correction as for, e. g. spherical nano-particle. In such a situation, an approximated approach is taken in RMCProfile, following the DShaper method reported by D. Olds. et al [25]. To activate DShaper-type of nano- baseline correction for calculated PDF pattern, apart from the keyword 'USE_DSHAPER ::' provided in the main '.dat' file, one should also provide a separate parameter file for DShaper, which should be with exactly the file name of 'dshaper.parm'. The content of this parameter file should be in the following format:

kernel
sinc

width
1.0

One should refer to the paper by D. Olds. et al [25]. for detailed introduction about the parameter settings here.

Appendix H

Change log

Version 6.7

v6.7.9

1. Correction for resolution effect beyond Gaussian assumption for peak shape.
2. Using Topas profile for fitting Bragg data, for both non-magnetic and magnetic systems.
3. Collect configurations every certain accepted moves.
4. Statistics about generated and accepted moves will be accumulated from previous runs.
5. Several bugs fixed, such as that relevant to the using of 'Qbroad' parameter for correction the broadening effect in *r*-space, etc.

v6.7.8

1. Correction for the finite Q-range effect is added.
2. Qdamp parameter added to account for the finite Q-space resolution effect, in both real and reciprocal space. Similar approach is taken, as it is in the PDFgui software.
3. Qbroad parameter added to account for the high noise level in the high Q ragion, in both real and reciprocal space. Similar approach is taken, as it is in the PDFgui software.
4. Correction for nano-size effect now is no longer restricted to regular shapes. The DShaper approach for estimating the nano- baseline when calculating PDF for nano-systems is introduced.

v6.7.7

The magnetic implementation in this version has been debugged and tested to be working robustly. Another major update is a new plot routine based on Python is included to replace the old 'rmcplot'. To use the new 'rmcplotpy' routine, one needs to have Python (>= 3.0) installed with matplotlib (>= 3.0). Users are strongly recommended to install Python with Anaconda and activate conda base environment by simply executing 'conda activate' from RMCProfile command window.

v6.7.6

The major change in this update is about the resolution correction. Now a new routine has been implemented and the resolution correction has been tested robustly on data collected from various diffractometers with different peak shapes. A few bugs fixed. Previously, when dealing with the lattice structure calculation with data2config for unit cell with alpha not equal to 90 degrees, we have a serious bug - both data2config and RMCProfile gives the wrong lattice vectors which is crucial for atomic distances calculation. Now this bug has been fixed. For all the other lattice types, previous implementation works perfectly. Other bugs fixed include the using of potential constraint, Bragg hkl indeces search, etc.

v6.7.5

Automated weight optimization introduced for RMCProfile to work with multiple data section and constraints. A few bugs fixed, concerning fitting X-ray data with fast Bragg approach, potential constraint working with weight optimization, etc.

v6.7.4

The way to fit the X-ray data is changed as compared to previous version. Users are recommended to read the corresponding section in the manual for properly setting up the keywords for fitting X-ray data.

v6.7.3

1. The 'XRAY_REAL_SPACE_DATA' keyword is now working in a robust way. It allows user to define the minimum, maximum and step in q-space for the temporary Fourier transform from r-space to q-space. Also alternative way of calculating the X-ray PDF is also available. Previously, this relies on the reading-in of 'neutron coefficients' from the '.dat' file for the calculation. Now the program can calculate the coefficients needed automatically.
2. RMCprofile now can work with both 9 or 11 coefficients X-ray scattering factors. If the '.xray' file exists, RMCProfile will work in the 9 coefficients diagram. Otherwise RMCProfile will use the embedded 11 coefficients for calculating the q-dependent X-ray scattering factors. The embedded 11-coefficients uses the following paper:

   > New Analytical Scattering-Factor Functions for Free Atoms and Ions
   >
   > By D. Waasmaier and A. Kirfel
   >
   > Acta Cryst. (1995). A51,416-431

   Also, since we now have the embedded coefficients for calculating the q-dependent scattering factor for X-ray, we do not need the tool 'xray_to_rmc' any more to refine the X-ray reciprocal data.
3. Resolution correction now is available to account for the limited resolution in q-space.
4. RMCProfile now can check the r-step in the input experimental data file. If the r-spacing of the experimental data is not the same as that given in the '.dat' file, warning messages will be printed out to screen.

5. This version of data2config can deals with partial occupancy. However, now it only allows atoms of the same type to have the same occupancy. Also data2config can now deal CIF files with various formats robustly. Furthermore, with the new flag '-Uiso' or '-Biso', data2config can introduce Gaussian distribution displacement for all atoms.
6. Now the gaussdist program will ask users whether to use the correlated moving of atoms, and if the answer is no then the program will deal with the zero correlation automatically. Also gaussdist can now deal with the situation where we have atoms of the same type sitting on crystallographically different atomic sites.
7. Version information can now be printed out to screen for all the tools with the '-version' flag.
8. The 'gpar_to_tot_gen' tool can now deal with the '.csv' file storing the partial PDF data.
9. Bugs fixed $\rightarrow$ variables initialization if corresponding keyword not provided, segmentation fault for transformation in between PDF functions, segmentation fault with CML, reading in the old-formatted structure configuration file ('.cfg' file), random number generator used in some tools does not generates number randomly, symmetry operators reading for data2config. rmc_to_discus does not work properly when no 'supercell' information is given in the '.rmc6f' file.

Version 6.5

v6.5.2

1. Helen started so things back on track and the new changes are..
2. Incorporation of bond valence and distance window parameters into the main v6 `.dat` file (`.bvs` and `.dw` files no longer needed)
3. Fixed and average coordination constraints resurrected and added to the v6 `.dat` file.
4. Triplets angle search fixed
5. Move counter no longer resets after 100000 moves
6. Various other bugs fixed...
7. X-ray example added to tutorial
8. Improvements to the manual (while still not complete it's a lot closer!)

v6.5.1

1. Sorry lost track of changes but I think EXAFS was first added here. We still need to improve the documentation but this will be in a later release
2. Probably other bug fixes...

Version 6.4

v6.4.8

1. Computes the bond Kubic harmonic function average values up to l = 10.
2. Computes the angle distribution function for the bond angles defined in the input data.
3. Calculates the mean bond distances and angles.
4. A small number of bug fixes.
5. Writes more data to the output CML file.

v6.4.7

1. Ability to read directly data files generated by the `STOG` program used for Fourier transform of the scattering data.
2. Generates spherical harmonics of the bond orientations and gives mean and mean-square values as output files.
3. Inclusion of `IGNORE_HISTORY_FILE ::` keyword

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).

Appendix I

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] Q. Hui, M. T. Dove, M. G. Tucker, S. A. T. Redfern, D. A. Keen. Neutron total scattering and reverse Monte Carlo study of cation ordering in $Ca_xSr_(1-x)TiO_3$. Journal of Physics: Condensed Matter. **19** art no 335214 (2007)

[11] S. T. Norberg, M. G. Tucker, S. Hull, Bond valence sum: a new soft chemical constraint for RMCProfile. Journal of Applied Crystallography **42** 179-184 (2009)

[12] 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).

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

[14] 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).

[15] 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).

[16] 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).

[17] 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).

[18] 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).

[19] 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).

[20] 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).

[21] A. L. Goodwin, S. A. T. Redfern, M. T. Dove, D. A. Keen and 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).

[22] I. Jeong, T. Proffen, F. Mohiuddin-Jacobs and S. J. L. Billinge, Measuring Correlated Atomic Motion Using X-ray Diffraction. J. Phys. Chem. A. **103**, 7, 921–924 (1999).

[23] T. Proffen and S. J. L. Billinge, PDFFIT, a program for full profile structural refinement of the atomic pair distribution function. J. Appl. Cryst. **32**, 572–575 (1999).

[24] R. Neder and T. Proffen, Diffuse Scattering and Defect Structure Simulations, Oxford University Press (2008).

[25] D. Olds, H.-W. Wang and K. Page, DShaper: an approach for handling missing low-Q data in pair distribution function analysis of nanostructured systems. J. Appl. Cryst. **48**, 1651–1659 (2015).

[26] Y. Zhang, M. McDonnell, W. Liu and M. G. Tucker, Reverse Monte Carlo modeling for low-dimensional systems. J. Appl. Cryst. **52**, 1035–1042 (2019).

[27] Y. Zhang, M. Eremenko, V. Krayzman, M. G. Tucker and I. Levin, New capabilities for enhancement of RMCProfile: instrumental profiles with arbitrary peak shapes for structural refinements using the reverse Monte Carlo method. J. Appl. Cryst. **53**, 1509–1518 (2020).