Reverse Monte Carlo modelling of the structure of disordered materials with RMC++ : a new implementation of the algorithm in C++

2005 J. Phys.: Condens. Matter 17 S1

(http://iopscience.iop.org/0953-8984/17/5/001)

INSTITUTE OF PHYSICS PUBLISHING JOURNAL OF PHYSICS: CONDENSED MATTER

J. Phys.: Condens. Matter **17** (2005) S1–S13 doi:10.1088/0953-8984/17/5/001

Reverse Monte Carlo modelling of the structure of disordered materials with RMC++: a new implementation of the algorithm in C++

Guillaume Evrard and László Pusztai

Research Institute for Solid State Physics and Optics, Hungarian Academy of Sciences, Budapest PO Box 49, H-1525, Hungary

E-mail: lp@szfki.hu

Received 10 December 2004
Published 21 January 2005
Online at stacks.iop.org/JPhysCM/17/S1

Abstract

The basic reverse Monte Carlo algorithm, as applied primarily for the study of disordered systems, is introduced, using an example of a new reverse Monte Carlo computer code. RMC++ is a new implementation of the RMC algorithm in C++. Its main purpose is to provide the community with a fast, flexible and documented code for RMC simulations, compatible with the `rmca` distribution. The source code, the documentation and the executable files are made available through the Internet. The flexibility of the code is exemplified by the implementation of a 'molecular move' step in the Metropolis algorithm. This feature, as well as a performance comparison, is illustrated with simulations performed for molecular liquids such as CCl_4 and C_2Cl_4.

(Some figures in this article are in colour only in the electronic version)

1. Introduction

Reverse Monte Carlo (RMC) is a tool for structural modelling which makes it possible to generate three-dimensional structural models, containing thousands of particles, that are consistent (i.e., agree within errors) with experimental (primarily, diffraction) data. RMC makes use of available experimental data in a direct, interactive and quantitative manner, via 'modelling' the measured data over their entire (useful) range; this is in contrast with an *a posteriori* comparison with measured data (details of the modelling—or, in a way, 'fitting'—procedure are given below). The structural models ('particle configurations') provided by the RMC procedure are consistent not only with all available experimental data but also with constraints which reflect our knowledge about the system in question (such constraints are, for example: density, particle sizes, molecular structure). The application of RMC does not require interatomic potentials, and therefore the technique is completely general; the information provided by the method, on the other hand, depends heavily on the type and quality of the experimental data being modelled.

The reverse Monte Carlo method has now been used for 15 years [1], and for a number of applications ranging from liquids, glasses to investigations of disorder in crystals (see [2] for a review). The RMC scheme can also be applied to the study of magnetic disorder, and may possibly be adapted to tackle dynamics [3].

Although the recent review mentioned above [2] touches the most relevant areas of applications, for an update, a couple of further examples may be mentioned here. Over the past few years, interest in metallic glasses has been renewed: this is mainly due to the fact that now, a number of such materials can be prepared as bulk materials. Following the early papers on metallic glasses (see, e.g., [4–7]), no new RMC-related applications have come up for quite a few years. Very recently, new studies on bulk metallic glasses have appeared [8]; it seems that a lot more will follow soon.

There has been a continuous interest in the microscopic structure of various chalcogenide glasses. Amorphous selenium appears to be the 'evergreen' prototype of these materials [9, 10], and Se and Te based amorphous alloys are also being looked at currently [11, 12].

Molecular liquids have been at the focus of our interest lately. Over the past five years, simple systems like carbon disulfide [13], carbon tetrachloride and its XCl_4 analogues [14] have been considered by reverse Monte Carlo modelling, as well as the most notorious of molecular liquids: water [15]. With the purpose-built features of the new computer code (see below), partially (or even, fully) rigid molecules, like amino acids, will be much more accessible.

Another type of systems where RMC modelling seems to be setting foot firmly nowadays is colloids. After an early (standard) study on the structure of polymer latices [16], RMC-like computational methods have been developing separately, with the aim of being able to consider more and more general small-angle scattering signals [17–19].

At the core of most of these applications lies the computing program that performs the simulations. For the 'standard' case of disordered materials, the Fortran `rmca` distribution[1] has been available for about ten years. Its source code includes features made necessary by the computer technology of the late 1980s, which are now obsolete (e.g. a specific array implementation to minimize memory usage). Besides, the original code yielded later modified versions that include new constraints (e.g. fixed neighbour constraints —FNC— [20]); however, only the original code remains properly documented and easily accessible. Consequently, the development of an updated, optimized RMC implementation was undertaken in Budapest. The RMC algorithm will be introduced here using an example of the new code, RMC++. The original list of specifications for the new software included clarity, portability, flexibility, upgradability, inclusion of a documentation and compatibility with `rmca`. In the following, the resulting code is briefly described. We particularly address specific issues concerning the computation of calculated data and the relevance of RMC results.

The recent extensive review article of McGreevy [2] tackles most issues relevant to the applications of reverse Monte Carlo modelling, and therefore only a few further examples are appropriate here; these include molecular liquids and some 'theoretical' aspects of the method. Instead of providing a multitude of examples, this work focuses on a somewhat different approach to the numerical background of RMC, as well as on the specific features of the new code.

2. The basic algorithm

On an abstract level, the structure modelling problem can be seen as a particular case of an inverse problem: given some experimental data and a theoretical modelling relation, one has

[1] Code and documentation available at www.studsvik.uu.se.

to infer (and eventually to select) a model that is compatible with the data (within some defined uncertainty). In the case of RMC, the relevant experimental data are overwhelmingly those of diffraction measurements, although any experimental signal may be used that can be calculated directly from particle positions.

Experimental diffraction data are assumed to be given in the form of the usual (static) total structure factor (TSF) $S_{\mathrm{T}}(Q)$. The model space can be defined as the set of possible partial pair correlation functions (PPCFs) $g_{\alpha\beta}(r)$, with α and β defining different atom types. The partial structure factors (PSFs) $S_{\alpha\beta}$ are then defined by

$$Q[S_{\alpha\beta}(Q) - 1] = 4\pi\rho \int_0^\infty r[g_{\alpha\beta}(r) - 1] \sin(Qr)\, \mathrm{d}r \tag{1}$$

where ρ is the atom number density of the system. These PSFs are weighted by the scattering lengths $\bar{b}_\alpha$, $\bar{b}_\beta$ and the (molar) concentration c_α, c_β of each atomic species to yield the TSF:

$$S_{\mathrm{T}}(Q) = \sum_{\alpha\beta} c_\alpha c_\beta \bar{b}_\alpha \bar{b}_\beta [S_{\alpha\beta}(Q) - 1]. \tag{2}$$

Usual *direct* methods perform the inverse sine Fourier transform from the TSF to obtain the PPCF. In the case of a polyatomic material, that requires isotopic substitution to separate each contribution. One alternative to sine Fourier transform inversion is MCGR [21]—Monte Carlo determination of $g(r)$—which samples the space of PPCFs until it finds a solution in agreement with the experimental data.

RMC gets closer to the real physical system by defining a model (a *configuration*) as a set of N virtual atoms put into a box at the density corresponding to that of the material under study (see figure 1). Figure 2 shows the flowchart of the basic algorithm; in short ('text format'), the algorithm can be summarized as follows (for a more detailed description, see [1, 2]).

(i) Start with a simulation box of sidelength L, containing N particles at the correct density ('starting particle configuration').
(ii) Calculate distance histograms for the partial PPCFs from the particle coordinates (see section 3).
(iii) Calculate the total structure factor(s), via equations (1) and (2) (see section 4).
(iv) Determine χ^2, the difference between model and experimental TSFs (χ^2 is the sum of squared differences between calculated and measured TSFs).
(v) Generate a new particle configuration by moving one particle at random.
(vi) Calculate χ^2 for the new configuration.
(vii) If the new χ^2 is smaller than previously, accept the move (the new configuration is saved); if not then the move still may be accepted with probability $\exp[-\Delta\chi^2]$. If a move is rejected, the old configuration is maintained.
(viii) Repeat from step (v).

As a result of the above procedure, χ^2 will decrease until it reaches an 'equilibrium value'. Any final particle configuration will be consistent, within errors, with the experimental data modelled, and any discrepancies between different final configurations cannot be discriminated on the grounds of the data in input.

Note that pioneering applications of RMC-like procedures also existed [22]; their main drawback was that they did not accept moves that increased χ^2. This feature, combined with inadequate computer power available at that time, prevented the widespread recognition of the enormous power of such modelling techniques.

RMC is dubbed as an *inverse method*, because it only involves computations from model space to data space, i.e. it never processes data. RMC works by repeatedly sampling the

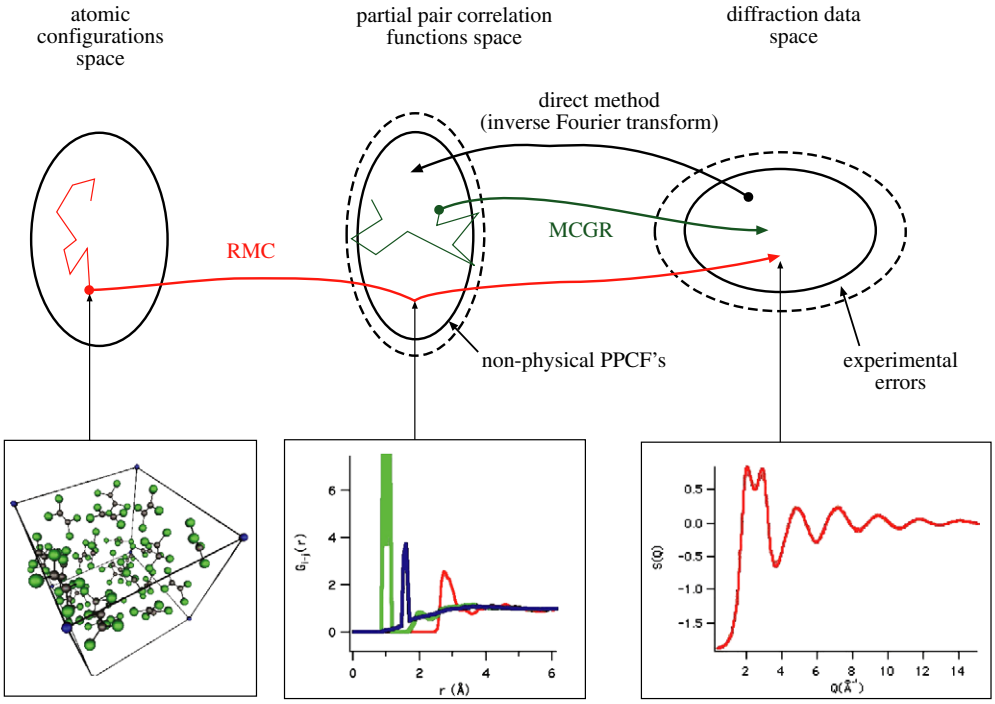

Figure 1. The structural modelling problem: *direct* methods of inversion operate a Fourier transform of $S(Q)$ diffraction data to get the PPCFs $g_{\alpha\beta}$, and thus face problems regarding limited Q-range information, underdetermination and instability. MCGR avoids these problems by repeatedly sampling the space of PPCFs to find a solution. RMC goes one step further and uses the space of atomic configurations as model space. Thus it ensures that the solutions are physically correct (within the constraints applied).

configuration space in order to find a model that is compatible with the input data (see figure 1). This is a Metropolis-type algorithm, i.e. a biased walk in the parameter space driven by the agreement between calculated and real data. Different aspects of the algorithm itself have been extensively described in the literature (see e.g. [23–26] and references therein). Purely practical details encountered when trying to write a working RMC programme are on the other hand seldom addressed.

3. From configuration to (radially averaged) pair correlation functions

A configuration is implemented as the list of coordinates of N atoms put in a virtual box with periodical boundaries. This means that each atom can be seen at the centre of an $L \times L \times L$ cubic box, and thus interatomic distances up to $\sqrt{3}l$ can be obtained, with $l = L/2$.

The PPCF $g_{\alpha\beta}(r)$ is interpreted as the ratio of probabilities

$$g_{\alpha\beta}(r) = \frac{P_{M\alpha\beta}(r, \mathrm{d}r)}{P_{H\alpha\beta}(r, \mathrm{d}r)} \tag{3}$$

where $P_{M\alpha\beta}(r, \mathrm{d}r)$ is the probability of finding one atom of type β at a distance between r and $r + \mathrm{d}r$ of one atom of type α in the material; and $P_{H\alpha\beta}(r, \mathrm{d}r)$ is the probability of finding one atom of type β at a distance between r and $r + \mathrm{d}r$ of one atom of type α in an *ideally*

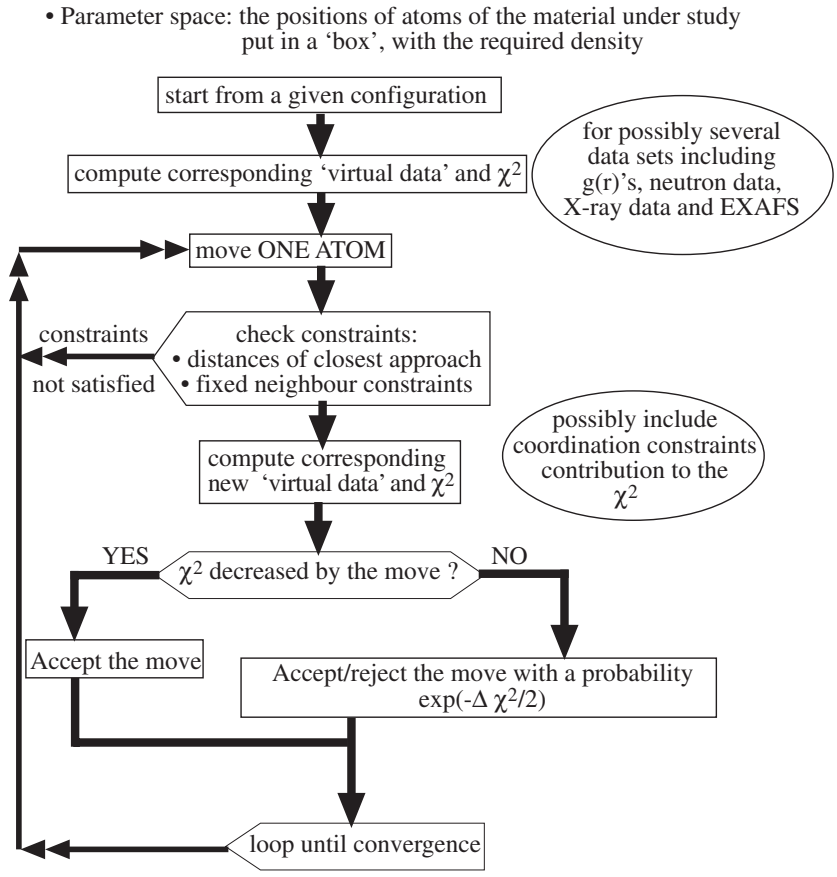

Figure 2. Flowchart of the 'standard' RMC algorithm.

homogeneous material. With this definition, the expected number of atoms of type β at a distance between r and $r + \mathrm{d}r$ from one atom of type α is given by

$$n_{\alpha\beta}(r) = 4\pi r^2 g_{\alpha\beta}(r)\,\mathrm{d}r\,\rho_\beta \tag{4}$$

where ρ_β is the number density of β atoms in the material.

In RMC, the probabilities appearing in the definition (equation (3)) are *estimated* from the counts and binning of distances in histograms. We must note that in general, within a well defined context (summarized by the information I), usual distributions provide the probability of getting n counts given a probability parameter p, say $\Pr(n|p, I)$. What we need here is to go the other way round: we have (histogram) counts, and we want a *value* for p. This can be done using Bayes' theorem [27], and by assuming an *a priori* distribution for p, one gets *a posteriori* distribution for p, $\Pr(p|n, I)$. And finally, one has to *choose* a value for p from that posterior distribution function. In extreme cases (probabilities close to 0 or 1), the assumptions made during this process might play a non-negligible role. In 'standard' cases, however, the usual choice of the 'maximum likelihood' value for p, i.e. the value that maximizes $\Pr(n|p, I)$ for the known n, is appropriate.

In the present case, this routine choice is adopted: given a distance interval $[r, r + \Delta r[$, the probability p of finding one interatomic distance in that range[2] is estimated by the ratio

$$P_M(r, \mathrm{d}r) = \frac{\text{number } n \text{ of distances in the range } [r, r + \Delta r[}{\text{all } N \text{ distances available}}. \tag{5}$$

This value maximizes the likelihood when the binomial distribution is considered for $\Pr(n|p, N, I)$. We must distinguish between different atom type pairs (α, β) to compute the different partials, and therefore we have for each histogram bin, and for distinct and identical atom types:

$$p_{\alpha\beta} = \frac{n_{\alpha\beta}}{N_\alpha N_\beta}; \qquad p_{\alpha\alpha} = \frac{2n_{\alpha\alpha}}{N_\alpha(N_\alpha - 1)} \tag{6}$$

where N_α and N_β are the number of atoms of type α and β in the configuration, and $n_{\alpha\beta}$ (resp. $n_{\alpha\alpha}$) are the counts number in the histogram for different (resp. identical) atom types.

The second probability (for the ideally homogeneous material) in equation (3) reduces to the ratio of volumes:

$$\frac{\text{volume of the spherical shell } [r_i, r_{i+1}[}{\text{total volume available}}. \tag{7}$$

Note that the volume of the spherical shell $[r_i, r_{i+1}[$ has to be modified if the radius goes beyond the dimension l of the box (see the RMC++ manual[3] for details). In the 'standard use' of RMC, distances larger than the box size l (i.e. ∼48% of all distances) are dropped, and the normalizing factor (equation (7)) for the spherical shell $[r_i, r_{i+1}[$ reads

$$\frac{4/3\pi}{8l^3}[r^3]_{r_i}^{r_{i+1}}. \tag{8}$$

The important point to keep in mind in this process is that the g_i values picked are estimates of the PPCFs, and that the uncertainty over the chosen values depends on the number of distances used to compute the estimates. Roughly speaking, the relative uncertainty on g_i can be estimated as $\delta g_i/g_i \sim 1/\sqrt{n_i}$. This point must be considered when choosing the histogram bin size: too small a bin size would yield large statistical uncertainty and 'noisy' calculated $g(r)$s. As usual, there is a tradeoff between the statistical precision over the estimation of g_i (which is the integral of $g_{\alpha\beta}(r)$ over the ith bin), and the resolution in r defined by the bin width. Note that the number of atoms in a given bin at low r range is roughly proportional to the number of atoms in the box, so that the eventual precision required puts a lower limit on the system size. Depending on the level of detail demanded for the PPCF, it is possible that this requirement supersedes the condition imposed by the computation of the integral in equation (1) (see the condition in equation (11) below).

4. From $g(r)$ to $S(Q)$: the sine Fourier integral

Once the PPCFs $g_{\alpha\beta}$ are chosen, the PSFs $S_{\alpha\beta}$ must be calculated via the sine Fourier integral (equation (1)) approximated by

$$S_{\alpha\beta}(Q) - 1 = \frac{4\pi\rho}{Q}\int_0^{r_{\max}} r[g_{\alpha\beta}(r) - 1]\sin(Qr)\,\mathrm{d}r. \tag{9}$$

This approximation is valid only if the box is large enough so that there is no structure beyond $r_{\max}$, i.e. $g_{\alpha\beta}(r) = 1, r \geqslant r_{\max}$. It must be noted that the *lowest* Q value for which the PSFs

[2] We use a half-closed—rather than closed—interval to stick to the non-overlapping intervals defined in the actual code implementation. The physical sense and mathematical calculations are not affected.

[3] Software and documentation available from www.szfki.hu/~nphys/.

must be computed also imposes a lower limit on the system size. Indeed we require that a full period (2π) of the sine function is included in the integral, i.e. $Q_{\min} r_{\max} \geqslant 2\pi$.

There are several possibilities to approximate equation (9) numerically, but since we start from a discretized version of $g(r)$ obtained from the histograms of distances, the so-called rectangular method is the most appropriate, i.e.

$$S_{\alpha\beta}(Q) - 1 = \frac{4\pi\rho}{Q} \sum_{i=0}^{i_{\max}} (g_{\alpha\beta,i} - 1) \int_{r_i}^{r_{i+1}} r \sin(Qr)\, \mathrm{d}r \tag{10}$$

where a constant value $g_{\alpha\beta,i}$ for g on each histogram range $[r_1, r_{i+1}[$ is assumed. For this process to be valid, the discretization step must be kept small with respect to the variations of the integrand. In this case, the fastest variations are assumed to take place for the highest Q value $Q_{\max}$ in the sine part of the integrand. One has to set a lower limit to the number η of discretization points needed over one full period at $Q_{\max}$, so that the Δr bin width for r must satisfy

$$\Delta r \leqslant \frac{2\pi}{\eta Q_{\max}}. \tag{11}$$

A value of η not less than 5 is recommended.

Each integral in the RHS of 10 can be computed analytically. The PSFs must be obtained for the Q_j values where there are experimental data for comparison. Matrix elements U_{ji} can be defined as

$$U_{ji} = \frac{4\pi}{Q_j} \left[\frac{\sin(Q_j r)}{Q_j^2} - \frac{r \cos(Q_j r)}{Q_j} \right]_{r_i}^{r_{i+1}}. \tag{12}$$

The sine Fourier integral (equation (1)) is implemented as the application of the matrix U_{ji} to the 'vector' $\{g_i\}$:

$$S_{\alpha\beta}(Q_j) - 1 = \rho \sum_i U_{ji} [g_{\alpha\beta,i} - 1]. \tag{13}$$

The matrix U_{ij} is initialized at the start of the programme (there is one matrix for each data set). Finally, the total static structure factor (directly comparable with the measured data) is computed by the weighted sum in equation (2).

Fast Fourier transform can be considered as an alternative to the process described above, but it has its own requirements (e.g. regular binning) which makes it less flexible than the raw Fourier integral approximation. Besides, the Fourier transform is not the most time-consuming stage in the algorithm.

5. RMC++ implementation and optimization

A basic RMC flowchart appears in figure 2. The decomposition of the algorithm into elementary tasks, involving defined pieces of data (histograms, partials, etc) is straightforward.

The chosen language for the implementation was C++. As an object-oriented language, it lends itself naturally to this brick-by-brick decomposition and thus allows maximum clarity, flexibility and upgradability. Another valuable feature during code development is that code elements can be tested independently. Table 1 lists the main code pieces implemented as C++ classes in RMC++.

Elementary techniques have been used to optimize the code for computing speed. The most time-consuming task in the algorithm is the calculation of interatomic distances (more precisely, the computation of the distances to the moved atom). Optimization therefore demands that these distances be computed only when necessary, and be computed quickly.

Table 1. The main C++ classes in the RMC++ implementation.

Experimental data sets
Configuration
Run parameters
Coordination constraints
Average coordination constraints
Fixed neighbours constraints
Histograms set
Sine Fourier transform matrix
Move
Pair correlation functions
Calculated partial structure factors
Calculated data
χ^2
Run history

The first requirement applies to the set of constraints which discard the move when they are not satisfied, i.e. distances of closest approach, and FNCs (that mimic intermolecular links). Obviously, the distances to the very few known neighbours of the moved atom must be checked first, because they are the ones which are the most likely not to satisfy the constraints. This is easily implemented for FNCs. A more general definition of a 'neighbourhood' of each atom would be required to really perform the same 'priority distance checks' for all atoms, for example to each atom could be attached the list of the 100 closest atoms[4]. Such a possible development, which would imply extra memory cost, remains to be tried.

Computing distances from the moved atom to all the others has been optimized by carefully implementing the arrays of positions for the atoms so that the transfer of data blocks from the RAM to the cache memory is kept to a minimum.

6. Comparison to the existing RMCA distribution

The resulting code (RMC++) has been put on the testbench with different systems (e.g. H_2O, CCl_4, Cl_2) using the existing `rmca` and `rmc_fi` programmes as references. The comparison was made straightforward by the use of the same input formats for both codes.

As far as performance is concerned, RMC++ is about three times as fast as `rmca` (see figure 3). Small additions improve the user's comfort: for example, the evolution of the values of the χ^2 components as well as the different acceptance ratios is recorded.

We must note that since the algorithms used in both implementations are almost identical, similar features such as rounding error effects occur in both implementations. It is important to notice that small variations in the definition of the histograms (e.g. a shift of all bins by a small distance δr), or of the PPCFs (by using an alternative normalization for the histograms) yield small variations to the calculated data. These changes, in turn, yield large variations of the χ^2 for the *same* configuration. Presumably, a specific choice for these arbitrary parameters does not really affect the result of an RMC simulation, but it points out the lack of a proper tool to compare configurations. This feature should also set limitations on the level of detail that can be accepted as reliable for discussions. At present, however, there is no tool to assess the uncertainty of RMC results.

[4] It is even possible that *only* the new distances to such neighbouring atoms must be updated at each step, because presumably they are the ones which are most likely to be responsible for the evolution of the PPCFs owing to the move.

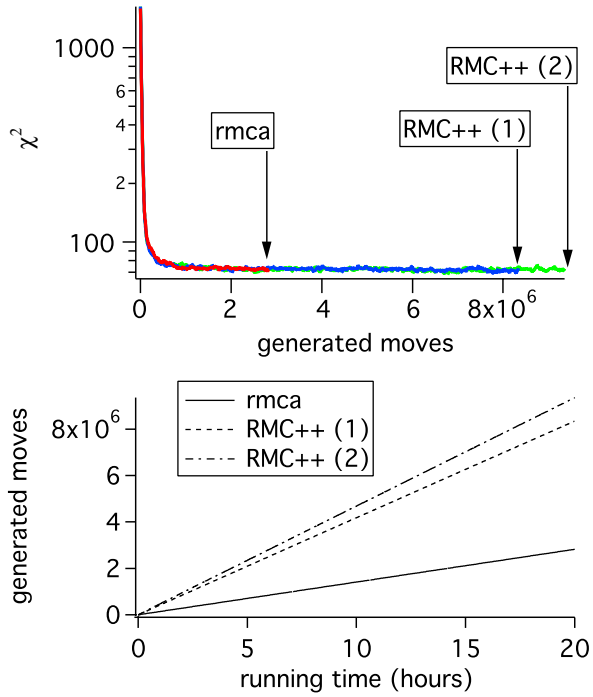

Figure 3. Comparison between `rmca` and RMC++ results. These runs are based on CCl_4 (10240 atoms system, one neutron diffraction data set, FNC applied). The χ^2 curve indicates that both programmes behave identically. Arrows point to the total number of generated moves during the 20 h runs. The two different RMC++ runs refer to different compilers used for producing the executable. It can be seen that for this system RMC++ is more than three times faster than RMCA.

7. The molecular move option

RMC++ was designed mostly for molecular systems; this is why relevant features are discussed here in more detail.

In 'standard' RMC, one step in the Metropolis random walk consists of the move of one single atom. There are cases where the molecular geometry is an important part of *a priori* knowledge that one wants to introduce in the structure modelling process. For a system composed of rigid molecules, moving atoms one by one is likely to destroy the molecular geometry, and the only alternative is to move the whole molecule or at least a whole rigid group of atoms simultaneously. This possibility has been implemented, in combination with the FNCs, which define the molecules.

To illustrate the flexibility of RMC++, the only corresponding significant change required in the programme is the redefinition of the implementation of the 'move' process. Obviously, any such move must be purpose-made for a given system, so that some code writing is necessary. The moves can include rotation around a preferential or random rotation axis, translations in preferential or random directions, constraints on the flatness or the shape of the molecule, as well as intramolecular atomic moves. The RMC++ website provides several examples of the corresponding code parts for H_2O, CCl_4, C_2Cl_4, CS_2 and H_2O.

The 'molecular' option has been tested for these systems. Obviously, it really makes sense for nearly rigid molecules. For CCl_4 for example (or for H_2O), there is no significant change

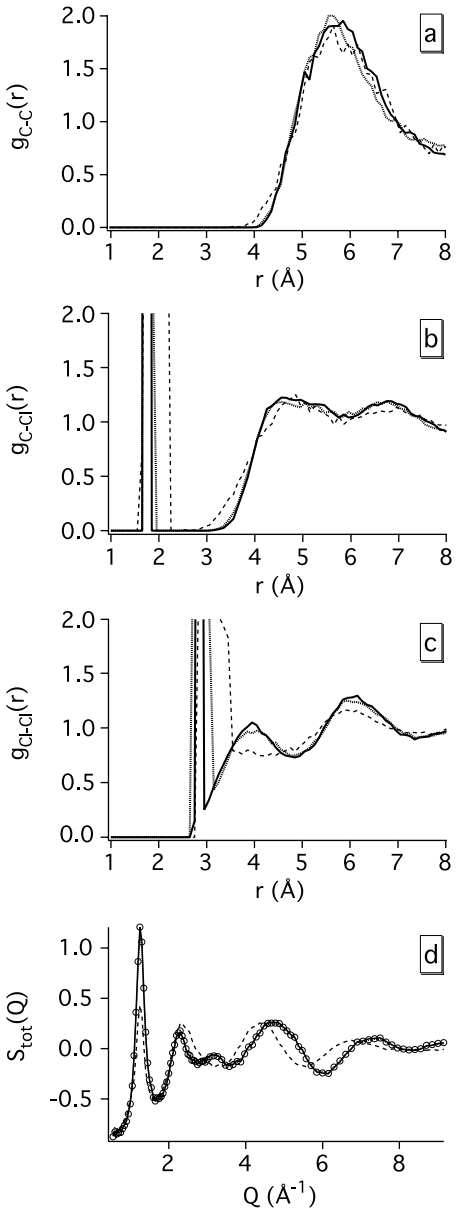

Figure 4. RMC results for CCl_4 modelled from neutron data [14]: (a) C–C PPCF, (b) C–Cl PPCF, (c) Cl–Cl PPCF, (d) calculated versus experimental data: thick continuous curve, 'molecular' RMC run; thick dotted curve, 'standard' RMC; thin dashed curve, 'hard-sphere' (no data used) simulations; circles, experimental data. For such a system whose geometry can be well defined using FNCs, there is almost no difference between the 'molecular' and the 'standard' RMC results. Differences in intermolecular peak widths are due to the use of different FNCs. Notice that the only visible information brought by the data appears on the Cl–Cl partial.

in the resulting configurations (see figure 4). Tests were also conducted for the flat system C_2Cl_4 (see figure 5) for which there is little space to move whole molecules at the considered density. Consequently, the configuration evolves quite slowly. Perhaps more importantly, the

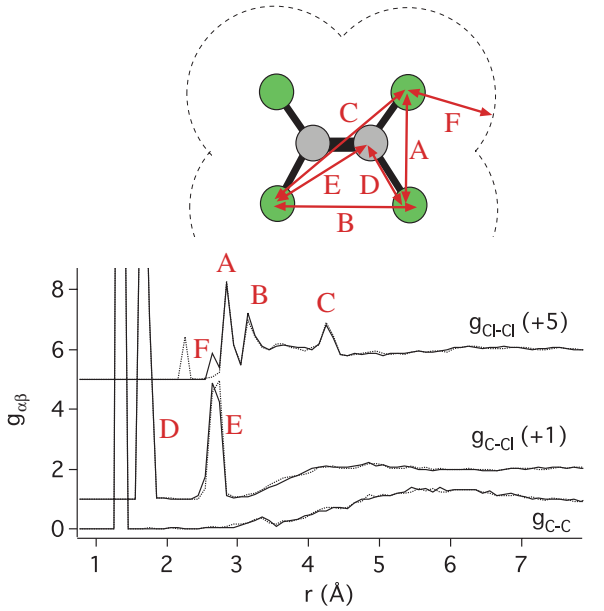

Figure 5. Using the 'molecular' version of RMC++ to model C_2Cl_4 from neutron data [14]. In addition to the intramolecular distances peaks (A to E), the Cl–Cl PPCF shows an intermolecular peak (F) at the distance of closest approach used. Two runs were performed with cutoffs set to 2.2 Å (dotted curve) and 2.6 Å (continuous curve). The corresponding peaks indicate that Cl atoms get artificially as close as allowed by the RMC constraints.

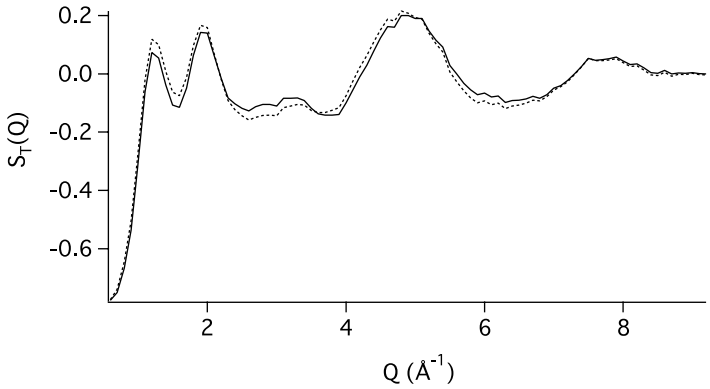

Figure 6. Experimental data (dotted curve) versus calculated data (continuous curve) resulting from an RMC simulation using the 'molecular' move. The quality of agreement with experiment is poor by usual RMC standards. One has to keep in mind, however, the balance between algorithmic constraints such as FNCs or the moving scheme, and the goodness of fit is an adjustable parameter of RMC.

fit to the measured data is not entirely satisfying by usual RMC standards (see figure 6). The Cl–Cl PPCFs shows one artificial intermolecular peak indicating that molecules get as close as the distances of closest approach allow (2.2 or 2.6 Å in the present example). This constraint gives the molecule the shape of four large balls placed at the corners of a rectangle, and creates a 'well' between the neighbouring Cl atoms. The algorithm finds difficulties in moving Cl

atoms trapped in this well, and the corresponding peak in the Cl–Cl PPCF is a quite stable feature, although it is hardly physically acceptable. Further studies are necessary in order to assess the real usefulness of this option, and to find the adequate rules of thumb to conduct the simulations.

Once again this illustrates that RMC cannot be used as a 'black box' and that results must always be scrutinized on physics and chemistry grounds. The introduction of the 'custom' move option in the RMC algorithm requires some initial tests in order to empirically find the good run parameters, with the usual tradeoff between move amplitude, distances of closest approach and the ability of the configuration to evolve.

8. Conclusion

The reverse Monte Carlo structural modelling technique, as applicable for the study of the microscopic structure of disordered materials (liquids and amorphous systems), has been introduced.

We have developed a complete self-consistent C++ implementation of the RMC algorithm. The code has been optimized for speed of execution and extensively tested. Compared to the existing `rmca` distribution (with which it is compatible, using the same input and output format), RMC++ appears to be about three times faster. It also includes additional new options, such as the possibility to move whole molecules simultaneously. The programme is fully documented, and is intended to serve as a backbone for possible future developments (e.g. application to parallel computing, adaptation to peculiar systems, inclusion of dynamics,. . .). The source code, the executables, and the documentation, as well as examples, are freely available to the community via the web server of the SzFKI (see footnote 3).

Acknowledgments

This work was supported by the EU 'Centre of Excellence' grant EU-ICAI-CT-2000-70029. LP was also funded by the 'Hungarian Basic Research Fund' (OTKA), through grants T32308 and T42495. Orsolya Gereben is acknowledged for useful additions to the code and for providing some simulation results.

References

[1] McGreevy R L and Pusztai L 1988 *Mol. Simul.* **1** 359
[2] McGreevy R L 2001 *J. Phys.: Condens. Matter* **13** R877
[3] McGreevy R L, Zetterström P and Ebbsjö I 2002 private communication
[4] Pusztai L and Sváb E 1993 *J. Phys.: Condens. Matter* **5** 8815
[5] Pusztai L and Sváb E 1993 *J. Non-Cryst. Solids* **156–158** 973
[6] Pusztai L and Gereben O 1994 *Mater. Sci. Eng.* A **179/180** 433
[7] Duine P A, Sietsma J, Thijsse B J and Pusztai L 1994 *Phys. Rev.* B **50** 13240
[8] Machado K D, de Lima J C, de Campos C E M, Grandi T A and Triches D M 2002 *Phys. Rev.* B **66** 094205
[9] Jóvári P and Pusztai L 2001 *Phys. Rev.* B **64** 014205
[10] Jóvári P, Delaplane R G and Pusztai L 2003 *Phys. Rev.* B **67** 172201
[11] Machado K D, Jóvári P, de Lima J C, Campos C E M and Grandi T A 2004 *J. Phys.: Condens. Matter* **16** 581
[12] Kaban I, Halm Th, Hoyer W, Jóvári P and Neuefeind J 2003 *J. Non-Cryst. Solids* **326/327** 120
[13] Jóvári P 1999 *Mol. Phys.* **97** 1149
[14] Jóvári P, Mészáros G, Pusztai L and Sváb E 2001 *J. Chem. Phys.* **114** 8082
[15] Pusztai L 1999 *Phys. Rev.* B **60** 11851
[16] Tóth G and Pusztai L 1992 *J. Phys. Chem.* **96** 7150
[17] Tóth G 1997 *Chem. Phys. Lett.* **269** 413

[18] Tóth G 1999 *Langmuir* **15** 6718
[19] Tóth G 2000 *Physica* B **276–278** 404
[20] Pusztai L and McGreevy R L 1997 *NFL Studsvik Annual Report for 1996* OTH:21
[21] Pusztai L and McGreevy R L 1999 *J. Neutron Res.* **8** 17
[22] Kaplow R, Rowe T A and Averbach B L 1968 *Phys. Rev.* **168** 1068
[23] McGreevy R L 1997 *RMCA Manual,* available on www.studsvik.uu.se.
[24] McGreevy R L 2003 *J. Physique Coll.* IV **111** 347
[25] Tóth G and Baranyai A 1999 *Mol. Phys.* **97** 339
[26] Tóth G and Baranyai A 2005 *J. Phys.: Condens. Matter* **17** S159
[27] Sivia D 1996 *Data Analysis: a Bayesian Tutorial* (Oxford: Oxford University Press)