

MCGR

Version 3.3

M A Howe, R L McGreevy and L Pusztai

Changed by P. Zetterström and A.Mellergård

January 2005

1.	Introduction	2
1.1.	Changes from version 3.2 of MCGR.....	2
2.	Principle of MCGR	2
3.	MCGR details.....	3
4.	Use of the program	5
4.1.	The format of the control data file	5
4.2.	The format of experimental data files.....	10
5.	Compiling and running the program.	10

1. Introduction

MCGR is a program for determining total or partial radial distribution functions from one or more total structure factors measured by neutron or X-ray diffraction, by an inverse method. This is basically the Monte Carlo method developed by Alan Soper (and implemented in his program MCGOFR), with a few modifications. The input data, output and control files are formatted in the same way as for RMC [1] programs. Inverse methods have considerable advantages over the conventional direct methods; for instance they avoid truncation errors and allow the estimation of errors in the radial distribution functions.

1.1. Changes from version 3.2 of MCGR

- explicit reading of the (fixed or initial) normalisation factors, see parameter **beta** in section 4.1.
- enhanced standard error approach, new parameter **isig** is described in section 4.1.
- parameter **cycles** now determines how many moves are accepted before saving multiple sets.
- no convolution (even if **convol** is true) performed if $dQ \cdot r_{\max} > 2\pi$ since then it is not well defined

2. Principle of MCGR

The relationship between a total structure factor, $F(Q)$, and a set of partial radial distribution functions, $g_{\alpha\beta}(r)$, for an N component system may generally be written as

$$F(Q) = \sum_{\alpha=1}^N \sum_{\beta=1}^N \gamma_{\alpha\beta}(Q) (A_{\alpha\beta}(Q) - 1) \quad (1)$$

$$A_{\alpha\beta}(Q) - 1 = \int 4\pi r^2 (g_{\alpha\beta}(r) - 1) \frac{\sin Qr}{Qr} dr \quad (2)$$

where $A_{\alpha\beta}(Q)$ are the partial structure factors. $\gamma_{\alpha\beta}$ are coefficients, Q dependent for X-ray diffraction and constant for neutron diffraction. Conventionally $g_{\alpha\beta}(r)$ are determined from $A_{\alpha\beta}(Q)$ by the Fourier transform

$$g_{\alpha\beta}(r) - 1 = \frac{1}{(2\pi)^3 \rho} \int 4\pi Q^2 (A_{\alpha\beta}(Q) - 1) \frac{\sin Qr}{Qr} dQ \quad (3)$$

However if $A_{\alpha\beta}(Q)$ is truncated or contains statistical errors then spurious oscillations are introduced into $g_{\alpha\beta}(r)$. In addition any other errors in $A_{\alpha\beta}$ are redistributed in an unknown fashion in $g_{\alpha\beta}(r)$. Such effects can be particularly problematic when the partial structure factors are obtained by direct separation from a set of total structure factors obtained by either neutron diffraction with isotopic substitution or X-ray diffraction with anomalous scattering. In many such cases the separation matrix is ill-conditioned and errors in the total structure factors are considerably magnified in the partial structure factors.

An alternative approach is to 'generate' possible $g_{\alpha\beta}(r)$ by some method, and then to modify these to fit the data, that is the total structure factor(s). Since $g_{\alpha\beta}(r)$ can be generated over as wide an r range as required there will be no truncation of the Fourier transform. In addition different sets of $g_{\alpha\beta}(r)$ that fit the data can be generated, and from these an average and a standard deviation can be calculated, thus giving some idea of the errors.

In MCGR the $g_{\alpha\beta}(r)$ are generated by a Monte Carlo method. The sets of $g_{\alpha\beta}(r)$ are defined as histograms of n_r points with spacing dr . The basic algorithm is as follows:

1. Initially

$$g_{\alpha\beta}(r) = 0.0 \quad r < r_{\alpha\beta}$$

$$g_{\alpha\beta}(r) = 1.0 \quad r > r_{\alpha\beta} \quad (4)$$

where $r_{\alpha\beta}$ are the closest approach distances of atoms type α and β . These distances may not be well known in which case an underestimate should be used. (It is also possible to define $g_{\alpha\beta}(r)$ to be zero at other r values.)

2. Fourier transform (equation (3)) to obtain the partial structure factors and combine to obtain the calculated (C) total structure factor for this old (o) set of radial distribution functions

$$F_o(Q_i) = \sum_{\alpha} \sum_{\beta} \gamma_{\alpha\beta}(Q_i) (A_{\alpha\beta}(Q_i) - 1) \quad (5)$$

3. Determine the deviation from the experimental (E) data

$$\chi_o^2 = \sum_{i=1}^m (F_o^C(Q_i) - F^E(Q_i))^2 / \sigma(Q_i)^2 \quad (6)$$

where σ is an estimate of the experimental error and m is the number of Q_i points. For multiple structure factors the individual χ^2 values are simply added.

4. Change one point in $r g_{\alpha\beta}(r)$ at random, by a maximum amount $\pm\delta$. Calculate the new (n) total structure factor(s) and the deviation from experiment

$$\chi_n^2 = \sum_{i=1}^m (F_n^C(Q_i) - F^E(Q_i))^2 / \sigma(Q_i)^2 \quad (7)$$

5. If $\chi_n^2 < \chi_o^2$ the move (change in $r g_{\alpha\beta}(r)$) is accepted and the new set of $g_{\alpha\beta}(r)$ becomes the old set. If $\chi_n^2 > \chi_o^2$ the move is accepted with probability $\exp(-(\chi_n^2 - \chi_o^2)/2)$. Otherwise it is rejected.

6. Repeat from step 2.

As this process is iterated χ^2 will initially decrease until it reaches an equilibrium value about which it will fluctuate. The resulting set of $g_{\alpha\beta}(r)$ should then be consistent with the experimental structure factor(s) within the experimental error. Multiple sets can be collected and averaged.

3. MCGR details

MCGR can be used in two different ways.

(a) To fit a set of partial radial distribution functions to a set of total structure factors. In this case the procedure is as described above. The number of total structure factors must be greater than or equal to the number of partial radial distribution functions.

(b) To fit a single total radial distribution function, $G(r)$, to a single total structure factor. $G(r)$ is the direct transform of $F(Q)$, i.e.

$$F(Q) = \rho \int 4\pi r^2 G(r) \frac{\sin Qr}{Qr} dr \quad (8)$$

In this case the low r part of $G(r)$ is constrained to the value $\sum_{\alpha=1}^N \sum_{\beta=1}^N \gamma_{\alpha\beta}$. Note that if $F(Q)$ has been normalised to be 1 at high Q then the sum of coefficients is by definition also 1. It is not strictly possible to do this with X-ray data because of the Q dependence of the coefficients. It can be done approximately by dividing $F(Q)$ by $\sum_{\alpha=1}^N \sum_{\beta=1}^N \gamma_{\alpha\beta}$, but $G(r)$ will not strictly be flat at low r and a constraint should be used with caution.

Constraints can be applied to make $g_{\alpha\beta}(r)$ zero and/or positive between any required r values. For example in a covalently bonded system with clear separation between the first and second peak in $g_{\alpha\beta}(r)$ the

intermediate r points can be set to zero. The coordination between any required r values can be constrained by adding an extra term to χ^2 ,

$$\chi_{coord}^2 = \left(C_{\alpha\beta} - \int_{r_1}^{r_2} 4\pi r^2 c_{\beta} g_{\alpha\beta}(r) dr \right)^2 / \sigma_{coord}^2 \quad (9)$$

where $C_{\alpha\beta}$ is the required coordination of atoms of type β around type α and c_{β} is the concentration of type β . σ_{coord} is the weighting of the coordination constraint relative to the data and smoothing terms. This is used if generating partial $g_{\alpha\beta}(r)$. If a total $G(r)$ is generated then

$$\chi_{coord}^2 = \left(C_{\alpha\beta} - \int_{r_1}^{r_2} 4\pi r^2 c_{\beta} G(r) dr / \gamma_{\alpha\beta} \right)^2 / \sigma_{coord}^2 \quad (10)$$

It is of course only applicable if a single peak can be identified in $G(r)$ as being due to a particular partial $g_{\alpha\beta}$. These constraints mimic those available in RMCA, though they will not have the exactly the same effect since MCGR operates purely on a mathematical function while RMCA operates on a physical model.

Statistical fluctuations in the generated set of $g_{\alpha\beta}(r)$ will not be apparent in the partial structure factors because dr should be chosen to be much smaller than $2\pi/Q_m$. These fluctuations will become smaller when multiple sets of $g_{\alpha\beta}(r)$ are collected and averaged. However there are three options to smooth them during the fitting procedure. In options 1 and 2 this is done by adding the following terms, respectively, to χ^2 .

$$S^2 = \sum_{\alpha,\beta} \sum_{j=2}^{n_r} \left\{ \left(1 + [g_{\alpha\beta}(r_{j+1}) - g_{\alpha\beta}(r_j)]^2 \right) - 1 \right\} \{ w_1 r_j \exp(r_j / w_2) \} \quad r_j > r_{\min} \quad (11)$$

or

$$S^2 = \sum_{\alpha,\beta} \sum_{j=2}^{n_r} \left\{ g_{\alpha\beta}(r_{j+1}) + g_{\alpha\beta}(r_{j-1}) - 2g_{\alpha\beta}(r_j) \right\}^2 \{ w_1 r_j \exp(r_j / w_2) \} \quad r_j > r_{\min} \quad (12)$$

These terms have the effect of decreasing the difference between $g_{\alpha\beta}(r)$ at neighbouring r points, thus minimising statistical fluctuations. However we do not wish to smooth out any natural structure in $g_{\alpha\beta}(r)$. This would be expected to be more significant at small r , so the smoothing term is weighted to be more significant at large r where $g_{\alpha\beta}(r)$ would be expected to tend smoothly to unity. The parameters w_1 and w_2 govern the weight of the smoothing term relative to χ^2 and the increase in smoothing at large r respectively.

In option 3 $g_{\alpha\beta}(r)$ are changed by adding or subtracting a Gaussian function centred around a randomly chosen r point, rather than just changing the single r point. Note that this is not quite the same as defining $g_{\alpha\beta}(r)$ as a sum of Gaussians. Different width Gaussians can be used in different r ranges if required; normally the width would increase with r .

During the MCGR process, the real experimental total structure factors, $F^E(Q)$, can be considered as simple transformations of the structure factors actually measured, $F_m^E(Q)$ since the latter may contain systematic errors:

$$F^E(Q) = \left(F_m^E(Q) - \sum_{i=0}^{n_B} a_i Q^i \right) / b \quad (13)$$

where n_B is the degree of the background polynomial. The expression that is actually minimised in the simulation procedure is then accordingly

$$\chi_n^2 = \sum_{i=1}^m \left(b F^C(Q_i) + \sum_{j=0}^{n_B} a_j Q_j - F^E(Q_i) \right)^2 / \sigma(Q_i)^2 \quad (14)$$

The application of constraints permits the refinement of normalisation factor(s) (b) and sloping backgrounds (a_0, a_1, a_2) in $F_m^E(Q)$. This feature is particularly useful for correcting for incoherent inelastic scattering for hydrogenous materials. Optimisation of any combination of the parameters is possible; it is recommended, however, that the normalisation factor (parameter b) be refined only when the best possible fit without changing it could be achieved. The values of the above parameters, together with the χ^2 value, provide a sensitive test of data quality, particularly in terms of systematic errors.

It is recommended that MCGR be used to obtain total or partial radial distribution functions before modelling with the RMC program. In the first instance this is a valuable check on the quality of the data; if a good fit cannot be obtained with MCGR then there is no point in using RMC. RMC can then be used initially to fit the radial distribution functions obtained from MCGR, before fitting to the structure factors. This saves a considerable amount of time in the modelling.

4. Use of the program

With VMS (VAX/Alpha) or LINUX the program is run interactively by typing

MCGR name

or (assuming you are logged on to the NFL VMS cluster) as a batch job by typing

RMCSUB

and typing name when prompted for the file name and MCGR when prompted for the program name. *name.dat* is the name of the file containing control data and information.

The Windows version is simply started by clicking the program icon, or it may be started from the RMC menu using the WinNFLP program. An Open file window appears where the user can choose the relevant *name.dat* file and then the simulation will start. During the run menus can be accessed to save current data promptly or to exit the program with or without saving. Plotting of the current fitted patterns and the progress of χ^2 can also be turned on/off. A record of the last work directory will be saved to a file mcgr.ini in the directory where the exe file is stored so that easy continuation is possible next time the program is run.

The program will produce an output file *name.out*, an intermediate file (for use if the program is to be restarted) *name.g*, files *name.gsv* if the radial distribution function(s) are being saved and a file *name.log* containing information on the progress of the program. All output files will be in the directory given by the parameter outfile (see table below). It is recommended that version limits are set on the VAX/Alpha to avoid creating large numbers of files.

4.1. The format of the control data file

name.dat has a format very similar to the equivalent file for RMCA, so it is easy to edit from one file to the other. The parameters are described in order below.

title	(character*80) A title to be used in all output.
rerun	(logical) If .true, then the program will continue on an earlier calculation (the output from this earlier calculation must exist). If .false, it will start a new calculation.
rho	(real) The sample density in atoms per cubic Ångström.
partials	(logical) If fitting partial $g_{\alpha\beta}(r)$ then set to .true.. For total $G(r)$ set to .false.
npar	(integer) Number of partial $g(r)$'s to be generated. For an N component system there are $N(N+1)/2$ partial structure factors/radial distribution functions so $N(N+1)/2$ total structure factors are required to determine them. This means that if the number of data sets is less than $N(N+1)/2$ then MCGR should only be used to generate a single total radial distribution function for each total structure factor and partial radial distribution functions cannot be obtained. A

separate run of MCGR is then needed for each data set. Set **npar** to 1 for a total $G(r)$.

nzc	(integer) The number of zero constraints, i.e. regions in which $g_{\alpha\beta}(r)$ is defined to be zero. These must be defined separately for each partial.
izpar, rz1, rz2	(integer, 2*real) This line is repeated nzc times, once for each zero constraint. izpar is the number of the partial to which the constraint applies (they come in the order 11, 12, ... 1n, 22, 23 ... 2n etc) and rz1 , rz2 are the r values between which the constraint is applied. At least one constraint should be applied for each partial, equivalent to the cut-off constraints in earlier versions of MCGR and in RMCA. For example to apply a cut-off constraint of 2.5 Å to partial 1 a constraint 1 0.0 2.5 should be used.
delta	(real) The maximum change per Monte Carlo step in any basis $rg(r)$. If delta is large to start with, e.g. 1, then the fit will initially converge quickly but will then take longer reach equilibrium. If delta is small, e.g. 0.01, then the fit will initially converge more slowly but will converge better when equilibrium is approached. Generally a small value can be used all the time, except when $g(r)$'s are likely to contain very high peaks, e.g. in the case of covalent bonds. $rg(r)$ is modified uniformly rather than $g(r)$ since $A(Q)$ is the transform of the former quantity. A value of 0.05 - 0.1 is generally suitable.
mr, rmax	(integer, real) mr is an integer relating the maximum Q -value to the r -spacing, recommended values of mr are 5 – 7. rmax is the maximum r -value which should be used for the calculation of $g(r)$. For most glasses and liquids 20 – 40 Å is sufficient. For crystals ΔQ is determined by the best instrumental resolution and more than 400 Å may be required. Obviously the cpu time required will be proportional to the number of r -values in $g(r)$. PS! This replaces the parameters nr and dr from earlier versions of MCGR. The relation is dr = $2\pi / (\mathbf{mr} \ Q_{\max})$ and nr = rmax / dr .
save	(logical) Whether to save multiple sets of $g_{\alpha\beta}(r)$ or $G(r)$. This should only be done when equilibrium has been reached.
nsave, cycles	(integer, real) These parameters are only present if save is .true.. nsave is the number of sets of $g(r)$ to save when the fit to the data is considered satisfactory. When χ^2 reaches the target value(s) (parameters chisav , read below) another cycles *nacc moves will be performed and then $g(r)$ will be saved and the program will automatically restart from the initial (uniform) state. The average and standard deviation of the multiple sets of $g(r)$ can be determined using the program GSV.
conv	(logical) Whether to allow converging moves only. If .true. then only moves that result in a decrease in χ^2 will be accepted. Normally set to .false..
npc	(integer) The number of positivity constraints.
ippar, rp1, rp2	(integer, 2*real) This line is repeated npc times, once for each constraint. ippar is the number of the partial to which the constraint applies, defined as for izpar above. rp1 and rp2 are the r values between which the constraint is applied. Normally each partial will be constrained to be positive for all r values and a good fit should be obtainable unless there are

significant systematic errors. However if any atoms have negative scattering lengths then it may be necessary to let some partials be negative in some regions. When generating a total $G(r)$ positivity means that this is constrained to be greater than the low r value given by **-coeff** (see later). Note that if the structure factor has been normalised to 1 at high Q then **coeff** should also be 1.

ncc	(integer) The number of coordination constraints. The following seven parameters are given on a single line which is repeated ncc times, once for each constraint.
icpar, rc1, rc2	(integer, 2*real) icpar is the number of the partial to which the constraint is applied, defined as for izpar above. rc1 and rc2 are the r values between which the coordination is calculated.
cconc, ccoeff	(2*real) cconc is the concentration of the neighbour species, ccoeff is the value of $\gamma_{\alpha\beta}$ corresponding to the particular partial $g_{\alpha\beta}(r)$ giving rise to the peak in $G(r)$ for which coordination is being calculated. ccoeff should be 1 if partial $g_{\alpha\beta}(r)$ are being generated. For a total $G(r)$ ccoeff must be less than or equal to coeff (defined later).
ccoord, csigma	(2*real) ccoord is the required coordination and csigma is the weighting of the constraint. The value of csigma should be approximately the allowed fluctuation of the coordination in equilibrium. It should not be set too small, particularly when generating $g(r)$ initially, or it will be hard to accept any moves. It is recommended that coordination constraints only be applied after a first $g(r)$ has been obtained and it can be checked whether the constraint is compatible with the data.
smooth	(logical) Whether to smooth the basis functions or not.
ns	(integer) The type of smoothing chosen. This is only present if smooth is .true...
w1, w2, r0	(3*real) These parameters are only present if smooth is .true. and ns is 1 or 2. w1 and w2 are weights used to control $g(r)$ smoothing as described in section 1, w1 being the pre-exponential term and w2 the exponential term in equations (8) and (9). If w1 is zero then there is no smoothing and $g(r)$ will be statistically noisy. If it is large $g(r)$ will be very smooth but all real sharp features will be broader and it may not be possible to obtain a good fit to the data. If w2 is large then the smoothing is linear in r while if it is small then $g(r)$ will get smoother at large r , as should naturally occur anyway. w2 should not be set to zero. r0 is the distance r_{\min} in equations (8) and (9).
nchanges	(integer) The number of regions the basis $g(r)$'s are to be divided. This parameter is only present if smooth is .true. and ns is 3.

The following line is repeated once for each region with different Gaussian width parameters, i.e. **nchanges** times.

gwidth, rch	(2*real) gwidth is the full width at half height of the Gaussian. rch is the maximum r value for this particular width. If only one width is used for all r points then rch should be equal to rmax .
resol	(logical) Whether to convolute the calculated structure factors with the

experimental resolution function before fitting to the data. If **resol** is **.true.** Then parameters are given separately for each data set, since different data sets may have different resolutions.

iprint, iplot	(2*integer) Controls printing of information to the screen and in the log file. The number of moves, convergence of χ^2 etc. will be printed following the first accepted move after every iprint generated moves. iprint should be at least 1000 or the file <code>name.log</code> will be very large and simulation speed slowed down. If iplot is 1 then the plotting of the fitting of $F(Q)$ with background, the χ^2 progress, and $G(r)$ in separate windows will be turned on at the start of the run. The plotting will be updated during the run at each iprint event. If iplot is 0 plotting is initially turned off. In the Windows version it can be turned on and controlled during the run using the Plot menu items and the mouse (left-click and drag to select region, right-click to reset).
tlim, tsav	(2*real) Time limits for saving and running. The status of the program (<code>name.g</code> and <code>name.out</code> files) will be saved every tsav minutes. The program will terminate and save after tlim minutes. These need to be appropriate for the computer/ batch queue being used. Typical test runs can be made interactively in a few minutes, with proper runs requiring up to a few hours.
ntot1, ntot2	(2*integer) The numbers of $S(Q)$ and $F(Q)$ data sets respectively. $S(Q)$ are defined as having constant coefficients, while $F(Q)$ have Q dependent coefficients.

The following lines are then repeated for each $S(Q)$, i.e. **ntot1** times.

filesq	(character*80) The name of the data file containing the total structure factor.
nq1, nq2	(2*integer) The indices of the first and last data points in the experimental data file that are to be used for fitting. If nq2 is greater than the total number of data points then the last data point to be used will be the last data point given.
coeff	(real array dimension npar) If generating partial $g_{\alpha\beta}(r)$ then these are the coefficients of the partial structure factors that make up the total structure factor, i.e. $\gamma_{\alpha\beta}$ in equation (1). If generating total $G(r)$ then there is only one coefficient and this is the sum of the coefficients $\gamma_{\alpha\beta}$ which determines the low r value of $G(r)$.
isig, sigma	(integer,real) isig - code for determining the experimental standard errors. If isig = 1 standard errors are read from an extra column in the experimental data file. If isig = 2 standard errors are estimated as the square root of the experimental data. If isig = 3 standard errors are constant = 1; sigma - standard errors obtained as above are scaled by this value to yield the $\sigma(Q_i)$ in equation (6). In practice, with isig = 3, sigma can typically start at 1 % (the units of sigma are then the same as those of the total structure factor so the actual value must be decided as appropriate) and can then be reduced until either a visually satisfactory fit is obtained or convergence stops. It is of course possible to weight different data sets differently if some are considered less accurate.

wav, u, v, w	(4*real) These parameters are only present if resol is .true. . They are the
---------------------	---

experimental wavelength in Å and the standard u,v,w parameters used to describe the resolution of a powder diffractometer in Rietveld refinement.

chisav	(real) This parameter is only present if save is .true. . chisav is the target χ^2 , i.e the χ^2 below which the fit to the data is considered satisfactory.
convol	(logical) Whether to convolute experimental data to compensate for the real space truncation effect, see the RMCA manual.
renorm	(logical) Whether to renormalise. If .true. then the experimental total structure factors will be automatically renormalised. This should only be done once a reasonable fit has been obtained. If the resulting renormalisation is large then either the original data are badly normalised or some of the parameters in name.dat are wrong.
beta	(real) Initial normalisation factor.
backgr	(logical) Whether to refine the background.
nbackgr	(integer) No.of terms in the polynomial background.
bcoeff	(nbackgr*real) Initial coefficients of the polynomial background.
magnetic	(logical) If .true. then a paramagnetic form factor found in the structure factor file, in the same way as for X-ray form factors, will be added to the calculated structure factor before fitting to the data.

The following lines are then repeated for each $F(Q)$, i.e. **ntot2** times.

filesq	(character *80) The name of the data file containing the total structure factor and Q dependent coefficients.
nq1,nq2	(2*integer) The indices of the first and last data points in the experimental data files that are to be used for fitting.
isig, sigma	(integer,real) isig - code for determining the experimental standard errors. If isig = 1 standard errors are read from an extra column in the experimental data file. If isig = 2 standard errors are estimated as the square root of the experimental data. If isig = 3 standard errors are constant = 1; sigma - standard errors obtained as above are scaled by this value to yield the $\sigma(Q_i)$ in equation (6). In practice, with isig = 3, sigma can typically start at 1 % (the units of sigma are then the same as those of the total structure factor so the actual value must be decided as appropriate) and can then be reduced until either a visually satisfactory fit is obtained or convergence stops. It is of course possible to weight different data sets differently if some are considered less accurate.
wav,u,v,w	(4*real) These parameters are only present if resol is .true. . They are the experimental wavelength in Å and the standard u,v,w parameters used to describe the resolution of a powder diffractometer in Rietveld refinement.
chisav	(real) This parameter is only present if save is .true. . chisav is the

target χ^2 , i.e the χ^2 below which the fit to the data is considered satisfactory.

convol	(logical) Whether to convolute experimental data to compensate for the real space truncation effect.
renorm	(logical) Whether to renormalise.
beta	(real) Initial normalisation factor.
offset	(logical) Whether to refine the background.
nbackgr	(integer) no.of terms in the polynomial background.
bcoeff	(nbackgr*real) Initial coefficients of the polynomial background.
compton	(logical) If .true. then Compton scattering, found in the structure factor file in a column following the Q dependent coefficients, will be added to the calculated structure factor before fitting to the data.

The following line should always exist.

outfile	(character*80) Name of the files <u>without extension</u> to which the output from MCGR should be written.
----------------	---

4.2. The format of experimental data files

Input structure factors for neutron data should be in the DATA format as defined for the NDP series of programs. The first line contains the number of points, the second is available for any required descriptive text, and then $(Q, F(Q))$ values follow on subsequent lines. If **magnetic** is .true. then a third column contains a paramagnetic form factor which will be added to the calculated structure factor before the data are fitted.

For X-ray data the coefficients that weight the partial structure factors to produce the total structure factors are Q dependent and are given in columns following the $F(Q)$ values. For a two component system there are three partial structure factors and the coefficients are given in the order 11,12,...1n,22,23,...2n,...nn. A file in this format can be produced from a file in the DATA format using the program XCOEFF. This has the option to define the coefficients in one of three ways, depending on how the total structure factor is normalised. If **compton** is .true. then a final column contains Compton scattering which will be added to the calculated structure factor before the data are fitted.

Note that if more than one set of experimental data is supplied they must all be defined at the same Q points. You only need to use a subset of these points for fitting so it is possible to use data sets that cover different Q ranges provided that they are defined (for instance set to zero) at the Q points where data are not available. Files that satisfy these requirements can be produced using the NDP program REBIN.

The output file has extension .OUT and the same format as .OUT files from RMC programs, that is it is suitable for graphical display using the RMC PLOT program. Alternatively the various functions contained in the file can be written into files in the DATA format using the program EXTRACT. If $g(r)$'s are being saved (**save** = .true.) then files with extension .gsv are produced. Average radial distribution functions and standard deviations can be produced from these using the program GSV.

5. Compiling and running the program.

MCGR is written in FORTRAN 90 and source code exist for Windows/DOS (PC), LINUX (PC) and VMS (VAX/Alpha). The main program is self-contained except for some platform dependent routines for timing, initialisation and input/output. Plotting is done using PG PLOT routines [2]. If MCGR is downloaded as part of the WinNFLP suite of programs, then all necessary set-up would have been done following the

installation procedure in the WinNFLP manual. The WinNFLP suite can be downloaded from <ftp://ftp.studsvik.uu.se/Pub/WinNFLP> or <http://www.studsvik.uu.se>.

Otherwise the source code distribution should include the following files:

```
readme.txt
rmca.pdf
mcgr.pdf
rmca.for
mcgr.for
fileio_ WIN/UNIX/VMS.for
rmc_ WIN/UNIX/VMS.for
RMCLIB.for
grfont.dat
Makefile_LINUX or
MAKEFILE_VMS.COM
[RMC_logicals.ini]
```

Compile the relevant Fortran files and link with appropriate PGPLOT library. Sample makefiles are included with the source code and in appendix II for various compilers.

The executable distribution includes

```
readme.txt
mcgr[.exe]
rmca[.exe]
grfont.dat
RMC_logicals.ini
```

which should be extracted into a single directory, e.g. `c:\nflp\rmc`

For the plotting to work with VMS or LINUX, Xwindows have to run on the machine. Also, for proper plotting, the PGPLOT font and device environment variables should be defined. In Windows the MCGR program will attempt to read the RMC_logicals.ini file to define

```
PGPLOT_FONT = c:\nflp\rmc\grfont.dat
PGPLOT_DEV = /W9
```

assuming you extracted files into `c:\nflp\rmc`. However, if the definitions are not found at run-time then the program will attempt to find the necessary files in the program start-up directory. For LINUX version the corresponding definitions could be done in your *rlogin* file using the `setenv` command, also set the path to include your MCGR directory. For VMS version define a symbol to run MCGR.

```
$ mcgr ::= "$user$disk:[user.directory]mcgr.exe"
```

This and the environment settings could be done in your `login.com` file. Both LINUX and VMS works best with device `/xw`. If PGPLOT_DEV is not defined or set to `?` a list of selectable modes will be displayed at run-time

Appendix I Liquid copper example

In this example the total $G(r)$ for liquid copper is obtained with MCGR from a fit to $F(Q)$. In this example a PC was used.

INPUT: cu_mcgr.dat (parameter file)

```
Liquid copper (test MCGR control file)
.false.          ! rerun
0.0721           ! density
.false.          ! generate partials
1                ! no of partials
1                ! no of zero constr.
1 0. 1.8         ! izpar, rz1, rz2
0.05             ! delta
7 50.           ! mr, rmax
.false.          ! save
.false.          ! converge only
1                ! no of positivity constr.
1 0. 15.         ! ippar, rp1, rp2
0                ! no of coord. constr.
.true.           ! smoothing
3                ! nsmooth
1                ! nchanges
0.3 50.1         ! gau_sig, r_change
.false.          ! resolution
1000 1           ! printing, plotting on
10 5             ! time limits
1 0              ! no of data sets
c:\directory\cusq.dat
1 1000           ! data points to fit
1.              ! constant to subtract
1.0             ! coeff
0.01            ! sigma
.false.         ! convolute
.false.         ! renormalise
.true.          ! offset
1               ! no. of background terms
0.0            ! initial coefficient
.false.        ! magnetic
c:\directory\cusq
```

INPUT: cusq.dat (Neutron diffraction data $F(Q)$)

90	
Q	S (Q)
0.1	0.032
0.2	0.032
0.3	0.032
0.4	0.032
0.5	0.033
0.6	0.033
0.7	0.034
0.8	0.035
0.9	0.037
1.0	0.040
1.1	0.045
1.2	0.050
1.3	0.056
.	.
.	.
.	.
8.4	1.081
8.5	1.074
8.6	1.064
8.7	1.051
8.8	1.037
8.9	1.022
9.0	1.006

OUTPUT: cusq.out (contains $G(r)$, $F^C(Q)$, $F^E(Q)$ and Background(Q)). This can be plotted with the program RMCPLLOT). See figure 1.

OUTPUT: cusq.g (contains $G(r)$). See figure 2.

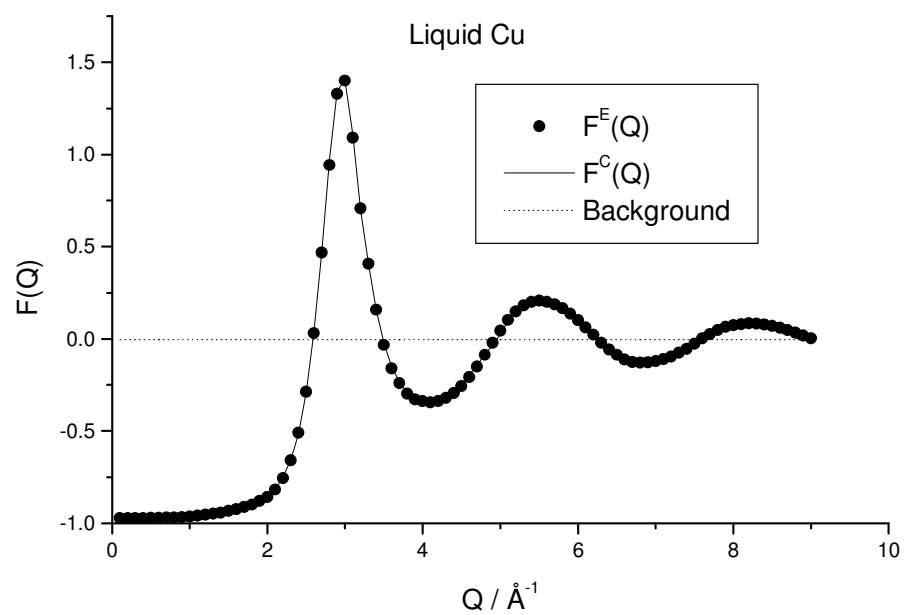


Figure 1 MCGR fit to the structure factor of liquid Copper.

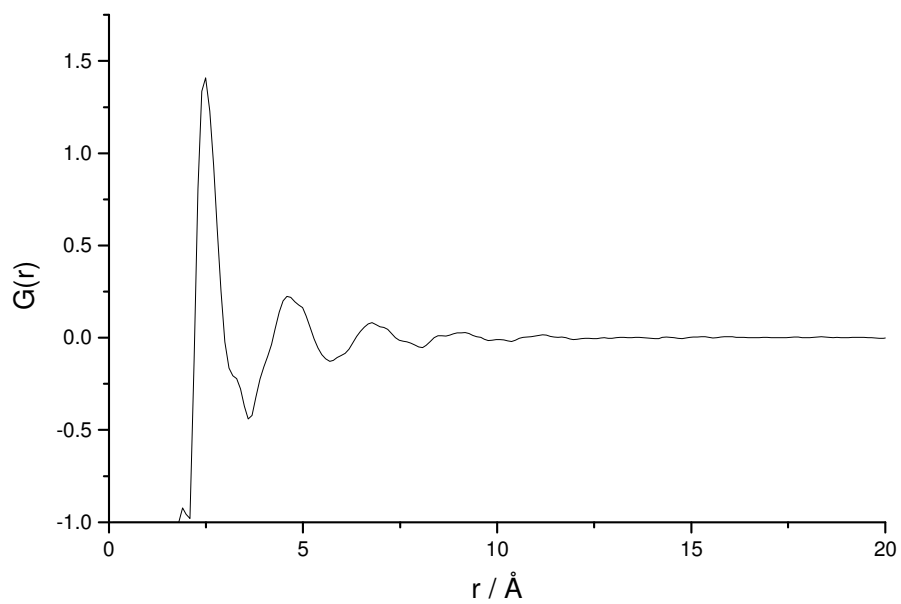


Figure 2 $G(r)$ for liquid Copper produced by MCGR.

Appendix II Sample makefiles

Windows

LINUX (ABSOFT F90 compiler)

```
SYSTEM= $(shell uname -s)

ifeq ($(SYSTEM),Linux)
L1:= -lpgplot -L/usr/pgplot
L2:= -lX11 -L/usr/X11/lib -lm
L3:= -B108 -YEXT_NAMES="LCS" -lm -YCOM_NAMES="LCS"
L4:= -lg2c
L5:= -O -B108 -YEXT_NAMES="LCS" -lm -YCOM_NAMES="LCS"
endif

all: RMCA.o MCGR.o fileio_UNIX.o RMC_UNIX.o RMCLIB.o mcgr rmca

rmca: fileio_UNIX.o RMC_UNIX.o RMCLIB.o RMCA.o
      f90 -o rmca -l U77 fileio_UNIX.o RMC_UNIX.o RMCLIB.o RMCA.o ${L1} ${L2}
      ${L3} ${L4}

RMCA.o: RMCA.for
      f90 -c MCGR.for ${L3}

mcgr: fileio_UNIX.o RMC_UNIX.o RMCLIB.o MCGR.o
      f90 -o mcgr -l U77 fileio_UNIX.o RMC_UNIX.o RMCLIB.o MCGR.o ${L1} ${L2}
      ${L3} ${L4}

MCGR.o: MCGR.for
      f90 -c MCGR.for ${L3}

fileio_UNIX.o: fileio_UNIX.for
      f90 -c fileio_UNIX.for ${L3}

RMC_UNIX.o: RMC_UNIX.for
      f90 -c RMC_UNIX.for ${L3}

RMCLIB.o: RMCLIB.for
      f90 -c RMCLIB.for ${L3}

clean:
      rm -f *.o *.mod
```

VMS

```
$
$ L1="PGPLOT_DIR:GRPSHR.OLB/LIB"
$
$! Determine target(s)
$ target="'pl'"
$ if target.eqs.""
$ then
$   target="all"
$ else
$   goto 'target'
$ endif
$
$ all:
$
```

```

$ fileio_VMS:
$   for fileio_VMS.for
$   library/create fileio_VMS fileio_VMS
$   if target.nes."all" then exit
$
$ RMC_VMS:
$   for RMC_VMS.for
$   library/create RMC_VMS RMC_VMS
$   if target.nes."all" then exit
$
$ RMCLIB:
$   for RMCLIB.for
$   library/create RMCLIB RMCLIB
$   if target.nes."all" then exit
$
$ rmca:
$   for rmca
$   link/nouser/executable=rmca -
$     rmca,RMC_VMS/lib,RMCLIB/lib,'L1'
$   if target.nes."all" then exit
$
$ mcgr:
$   for mcgr
$   link/nouser/executable=mcgr -
$     mcgr,RMC_VMS/lib,RMCLIB/lib,'L1'
$   if target.nes."all" then exit
$
$! Clean only if not "all"
$ exit
$
$ clean:
$   delete/noconfirm *.obj;*,*.olb;*,*.f90$mod;*
$   exit

```

REFERENCES

- [1] RMCA Version 3, R.L. McGreevy, M.A. Howe and J.D.Wicks, (1993), available at <http://www.studsvik.uu.se>
- [2] PGPLOT Copyright © 1983-2001 by the California Institute of Technology. <http://astro.caltech.edu/~tjp/>