Version 3.5, Revised December, 1997

General Locked Rotation Function

Program GLRF

Liang Tong†

INTRODUCTION

Given two homologous molecules, A and B, the rotation ($[\rho]$) that brings them into the same orientation will also bring the inter-atomic vectors of each molecule into overlap. Crystallographically, the set of inter-atomic vectors is represented by the Patterson function. Hence, an ordinary rotation function (M. G. Rossmann & D. M. Blow. *Acta Cryst.* **15**, 24–31, (1962)) can be defined as the overlap of one Patterson function and the rotated version of another Patterson function,

$$RF([C]) = \int_{\Omega} P_A(u) P_B([C]u) du \tag{1}$$

where

$$[C] = [\alpha_B][\rho][\beta_A] \tag{2}$$

and

$$x_B = [C]x_A + d \tag{3}$$

$[\alpha]$ is the deorthogonalization matrix which converts Ångstrom coordinates relative to a Cartesian system to fractional coordinates relative to the crystal unit cell. $[\beta]$, the inverse of $[\alpha]$, is the orthogonalization matrix. The integration volume (Ω) is usually a sphere centered at the Patterson origin. The radius (R) of this sphere can be chosen to exclude most of the vectors between atoms in different molecules in the crystal, which are generally longer than the inter-atomic vectors within one molecule.

The rotation $[\rho]$ is usually represented by a set of polar (ϕ, ψ, κ) or Eulerian ($\theta_1, \theta_2, \theta_3$) angles. In the case of polar angles, ϕ and ψ define the orientation of a rotation axis relative to a Cartesian coordinate system and κ defines the angle of rotation around this axis. In the case of Eulerian angles, the three angles define a sequential set of rotations around the X, Y, Z axes of a Cartesian coordinate system. For both polar and Eulerian angles, the actual meaning of each angle depends on the convention that is used.

The function RF is generally expected to reach its maximum when the rotation $[\rho]$ corresponds to the rotation that brings the two molecules to the same orientation. In practice, a grid search is carried out covering the entire unique region of the rotational space and Eq. (1) is evaluated at each grid point. The grid point that gives the highest value for Eq. (1) is then possibly the correct rotation. The rotation function values at other grid points that are evaluated in such a search give an estimate for the background noise in the calculation.

---

† Please send comments and bug reports to : tong@como.bio.columbia.edu.
Reference: L. Tong & M. G. Rossmann *Acta Cryst.* **A46**, 783–792, (1990).
Reference: L. Tong & M. G. Rossmann *Methods in Enzymology* **276**, 594–611, (1997).

Rotation functions can be classified by the way Eq. (1) is evaluated (for example, the slow and the fast rotation functions) or by whether the two molecules reside in the same crystal (self and cross rotation functions).

I. THE SLOW ROTATION FUNCTION

Eq. (1) can be evaluated in many ways in practice. If both Patterson functions are expressed as their individual Fourier transforms,

$$P_A(u) = \sum_h F_h^2 e^{-2\pi i h u}$$

$$P_B(u) = \sum_p F_p^2 e^{-2\pi i p u}$$

it can be shown (M. G. Rossmann & D. M. Blow. *Acta Cryst.* **15**, 24–31, (1962)) that

$$RF([C]) = \sum_h \sum_p F_h^2 F_p^2 G_{hp} \tag{4}$$

where

$$G_{hp} = \int_\Omega e^{-2\pi i (h+p[C])u} du \tag{5}$$

$$= \frac{3(sin2\pi HR - 2\pi HRcos2\pi HR)}{(2\pi HR)^3} \tag{6}$$

is the G function, which represents the Fourier transform of a sphere of radius R. H is the length of the reciprocal space vector $h + p[C]$. The function assumes the maximum value of 1 when $h + p[C] = 0$ and it quickly approaches and then oscillates around zero as $h + p[C]$ deviates from zero.

Eq. (4) is sometimes referred to as the 'slow' rotation function, as its evaluation is generally time-consuming. Several techniques have been developed to speed up the calculations of the slow rotation function. The rotation function value is proportional to the intensity of the reflections (Eq. (4)). It can then be expected that rotation function values are dominated by strong reflections. Therefore, only the strong reflections of crystal B (p) are used in the calculation. These reflections, also known as the large terms (P. Tollin & M. G. Rossmann. *Acta Cryst.* **21**, 872–876, (1966)), are selected based on the criterion

$$I_p \geq C\times < I_p >$$

where C is the cut-off value and the average intensity $< I_p >$ is generally calculated in shells of equal reciprocal volume. A cut-off value (C) of 1.5 usually selects about 20% of the reflections as large terms.

Given the large terms of crystal B, the summation over the reflections of crystal A (h) can be limited to those which make $h + p[C]$ small, as otherwise the value of the G function will be negligible. For each large term (p) in crystal B, only reflections in crystal A that are within an interpolation box centered on the reciprocal lattice point closest to $-p[C]$ are used in the summation. The size of this interpolation box is generally 3 (reciprocal lattice points) $\times$ 3 $\times$ 3.

The calculation of the G function value based on Eq. (6) at each rotation search grid point is tedious. Tables of the G function values can be set up beforehand, either in reference to the value of H or in reference to the h, k, l index in the interpolation box, to speed up the calculation.

The values of the rotation function vary relatively slowly as a function of the rotation angles. If a reasonably fine search grid is used, it will be unlikely for the neighbors of a grid point to have high rotation function values if the grid point itself has a low rotation function value. Therefore, the evaluation of rotation functions using Eq. (4) can be accelerated by ignoring such grid points in the calculations (L. Tong. *J. Appl. Cryst.* **26**, 748–751, (1993)).

II. THE FAST ROTATION FUNCTION

Even with the improvements described, the calculation of the slow rotation function is still rather time-consuming. The fast rotation function (R. A. Crowther. *The Molecular Replacement Method*, pp 173–178, Ed. M. G. Rossmann. Gordon & Breach, New York) uses a different approach to evaluate Eq. (1). The Patterson function is expanded in terms of the spherical harmonics,

$$S_{lmn}(r,\theta,\phi) = j_l(k_{ln}r)Y_l^m(\theta,\phi)$$

and

$$P_A(r,\theta,\phi) = \sum_{lmn} a_{lmn}S_{lmn} \tag{7}$$

$$P_B(r,\theta,\phi) = \sum_{l'm'n'} b^*_{l'm'n'}S^*_{l'm'n'} \tag{8}$$

where j_l is the normalized spherical Bessel function of order l and k_{ln} is chosen such that $j_l(k_{ln}R) = 0$. Y_l^m is the normalized spherical harmonics function. The expansion coefficient a_{lmn} can be calculated from

$$a_{lmn} = \sum_h F_h^2 T_{lmn}(h) \tag{9}$$

where T_{lmn} is the Fourier transform of S_{lmn}.

With such an expansion for both Patterson functions and utilizing the special properties of the spherical harmonics, Eq. (1) can be simplified to

$$RF(\theta_1,\theta_2,\theta_3) = \sum_m \sum_{m'} f_{mm'}(\theta_2)e^{-im\theta_1}e^{-im'\theta_3} \tag{10}$$

where the rotation is represented by a set of Eulerian angles $(\theta_1,\theta_2,\theta_3)$, and the function $f_{mm'}$ is only dependent on θ_2. Eq. (10) is a two-dimensional Fourier transform over θ_1 and θ_3. Therefore, for any given θ_2, the equation can be evaluated by the fast Fourier transform (FFT) technique, thus greatly accelerating the rotation function calculation.

The calculation of the expansion coefficients a_{lmn}, however, does still require some amount of computing time. The large term approach as developed for the slow rotation function can also be employed here. Test calculations have shown that a much smaller cut-off (C) value should be used (roughly 0.1 to 0.5) to obtain better rotation function results.

III. THE ORDINARY SELF ROTATION FUNCTION

When the two molecules A and B exist in the same crystal, a self rotation function can be used to determine the orientation and the angle of rotation of the local symmetry axis relating the two molecules.

Polar angles are usually preferred for representing the rotation $[\rho]$ in ordinary self rotation functions. For example, if the molecules are related by a two-fold axis, κ can be fixed at 180° and the rotation search can be limited to the ϕ and ψ angles. The search results can then be presented as a stereographic projection.

If the two molecules are related by an improper rotation (*i.e.*, an angle of rotation not divisible to 360), a full three-dimensional rotation search may be necessary. It is usually more advantageous to represent the rotation $[\rho]$ as a set of Eulerian angles in such a case. The unique region of rotational space for different non-cubic space group symmetries is given by S. N. Rao, J.-H. Jih & J. A. Hartsuck. *Acta Cryst.* **A36**, 878–884, (1980). Once the rotation $[\rho]$ is known, a set of polar angles can be extracted from it.

The ordinary rotation function will always have a maximum value at the origin ($\kappa = 0$ for polar angles, $\theta_2 = 0, \theta_1 + \theta_3 = 0$ for Eulerian angles) corresponding to the overlap of the Patterson function onto itself. This value can be set to be a specific constant (for example, 1000) and a scale factor is then applied to all the rotation function values to make the maximum 1000. This will bring the self rotation function to an 'absolute' scale. The rotations corresponding to the crystallographic symmetry elements should all produce rotation function values of 1000.

IV. THE ORDINARY CROSS ROTATION FUNCTION

If the two molecules are in two seperate crystals, the ordinary cross rotation function can be used to determine the relative orientation of the two molecules. The two crystals may have been prepared experimentally, or an artifical crystal can be prepared by placing a model, usually the structure of a homologous protein, into an arbitrary unit cell. In the latter case, the dimensions of the unit cell are usually chosen large enough so that the inter-molecular vectors are longer than the intra-molecular ones to avoid their interference in the rotation function calculations. Cell dimensions that are twice the dimensions of the model should generally suffice for this purpose.

Eulerian angles are generally preferred for the cross rotation function. The unique region of rotational space for non-cubic space group symmetries of the A and B crystals is given by S. N. Rao, J.-H. Jih & J. A. Hartsuck.

Sometimes it is known that a direction in crystal A is aligned with a direction in crystal B (for example, the alignment of the local two-fold axis of a dimer in two different crystal forms). In such a case, a pre-rotation can be used first to align the two directions. Subsequent rotation searches is then carried out in polar angles, varying only the angle κ.

V. THE LOCKED SELF ROTATION FUNCTION

Many macromolecular crystals contain assemblies of the macromolecules obeying a simple (local) point group symmetry. The ordinary self rotation function can be used to determine the orientation of each local symmetry axis individually. However, if the symmetry of the point group is known, orientations of all the symmetry elements of the point group can be determined at the same time, using the so-called locked self rotation function (L. Tong & M. G. Rossmann. *Acta Cryst.* **A46**, 783–792, (1990)). It assumes a point group symmetry for the macromolecular assembly and maintains that symmetry throughout its searches (i.e., the local symmetry is *locked* to the one specified at the outset).

A standard orientation is defined for the local symmetry point group. For example, a 222 point group symmetry can be defined with its three two-fold axes along the X, Y and Z axes of a Cartesian coordinate system. If $[I_n](n = 1, \cdots, N)$ represents the set of local symmetry rotation matrices in the standard orientation, the rotation matrices after a rotation $[E]$ has been applied is given by

$$[\rho_n] = [E][I_n][E]^{-1} \tag{11}$$

An ordinary self rotation function value (R_n) can be calculated corresponding to each of the rotation matrices. The locked rotation function value (R_L) is defined as the average of the individual values,

$$R_L = \frac{1}{N-1} \sum_{n=2}^{N} R_n \tag{12}$$

The locked self rotation function calculations are generally carried out in terms of Eulerian angles. In special cases where a local symmetry axis is known to be aligned with a direction in the crystal, polar angles can be used.

The locked rotation function simplifies the self rotation searches for large macromolecular assemblies (for example, viruses). Moreover, it also leads to a reduction, by a factor of about $\sqrt{N}$, in the background noise level of the rotation function as compared to the ordinary self rotation function.

The slow rotation function can be employed to calculate the locked self rotation function. Alternatively, the fast rotation function can be used to calculate an ordinary rotation function covering the entire Eulerian space. The locked rotation function can then be calculated by interpolating among the computed ordinary rotation function values.

The locked self rotation function can also be applied to cases where the macromolecular assembly does not obey a simple point group. In such cases, however, it is usually difficult to define a standard orientation.

VI. THE LOCKED CROSS ROTATION FUNCTION

The presence of local symmetry can also be utilized in the cross rotation searches if only the monomer is used as the model. For every rotation $[E]$ that brings the search model into the same orientation as that of one of the molecules in the assembly, there should be a set of other rotations that will bring the search model into overlap with other molecules in the assembly. Assuming that $[J_n](n = 1, \cdots, N)$ are the local symmetry rotation matrices in crystal B, which can be determined from either ordinary or locked self rotation function calculations, the set of rotations is given by

$$[C_n] = [\alpha][J_n][E][\beta] \tag{13}$$

Therefore, a locked cross rotation function can be defined as the average of the individual ordinary cross rotation function values evaluated with the rotation matrices $[C_n]$,

$$R_L = \frac{1}{N} \sum_{n} R_n \tag{14}$$

Alternatively, assume that $[I_n](n = 1, \cdots, N)$ are the local symmetry rotation matrices in a standard orientation, $[F]$ is the rotation that brings the standard orientation to that of the assembly in the crystal, and

$[E]$ is the rotation that brings the monomer to the same orientation as one of the monomers in the standard orientation,

$$[\rho_n] = [F][I_n][E] \tag{15}$$

With this definition, it is easier to define the unique region of rotational space for a locked cross rotation funcion.

VII. GENERAL STRATEGIES FOR ROTATION FUNCTION CALCULATIONS

Rotation function calculations usually proceed in two steps. In the first step, a search is carried out covering the unique region of the rotational space to find the sets of angles that maximize the rotation function. Self rotation function calculations are usually limited to sections of constant κ and the slow rotation function can be used, with grid intervals along ϕ and ψ of 2 to 3°. For ordinary cross rotation function calculations, the fast rotation function should be used in this initial search. If the results from the fast rotation function is unsatisfactory, the slow rotation function should be tried (but it will take some more CPU time), as it uses a completely different approach. A relatively coarse search grid (generally 3 or 4° intervals) is needed to reduce the total number of grid points in the search.

Once an estimate for the set of angles is obtained, the rotation search should be repeated using finer grid intervals (0.5 to 1°). The slow rotation function is used for this purpose. The search should be limited to a small region (3 to 5°) around the values for the rotation angles as determined by the initial search. This will give the optimized values for the rotation angles.

Alternatively, a Patterson-correlation refinement can be used to optimize the angular parameters.

VIII. SPECIALIZED TRANSLATION FUNCTIONS

If the local symmetry is a two-fold, the translation element along this axis $d_{//}$ can be determined (M. G. Rossmann, D. M. Blow, M. M. Harding, E. Coller. *Acta Cryst.* **17**, 338–342, (1964)),

$$T_1(\Delta) = \sum_h \sum_p F_h^2 F_p^2 G_{hp} cos 2\pi (h-p)\Delta \tag{16}$$

This calculation does not need the phase information of the reflections.

Once phase information is available, the location of the local symmetry axis in the unit cell can be determined based on the maximal electron-density overlap of the subunits related by the local symmetry (D. M. Blow, M. G. Rossmann, B. A. Jeffrey. *J. Mol. Biol.* **8**, 65–78, (1964); L. Tong. *J. Appl. Cryst.* **26**, 748–751, (1993)).

If s_A and s_B are the centers of the two molecules,

$$s_B = [C]s_A + d \tag{17}$$

The electron density overlap of the two molecules,

$$\int_\Omega \rho(x_A)\rho(x_B)dx_A$$

where the integration volume is a sphere of radius R centered at s_A, can be written as

$$T([C], s_A, s_B) = \sum_h \sum_p F_h e^{i\phi_h} F_p e^{i\phi_p} G_{hp} e^{-2\pi i h s_A} e^{-2\pi i p s_B} \tag{18}$$

Given the rotational relationship between molecules A and B and an estimate for the center of molecule A, the center of molecule B can be determined by a Fourier transform (Eq. 18). In this application of Eq. (18), the two molecules can reside either in the same crystal or in two different crystals.

Once initial estimates for $[C], s_A, s_B$ are available, Eq. (18) can be used to optimize these parameters. A fine search can be carried out around the current values for $[C]$ and s_A (or s_B, only one of which needs to be varied in the search) to obtain values that will maximize the correlation.

If molecules A and B reside in the same crystal, the position of this local symmetry axis can be determined. Assuming s lies on this axis,

$$s = [C]s + d_{//} \tag{19}$$

where $d_{//}$ is the translation element along this local symmetry axis (see Eq. 16). Eq. 18 then becomes

$$T_2(s) = \sum_h \sum_p F_h e^{i\phi_h} F_p e^{i\phi_p} G_{hp} e^{-2\pi i p d_{//}} e^{-2\pi i (h+p) s} \tag{20}$$

Therefore, the center of the local symmetry axis can be determined by a Fourier transform. This function will produce a long, sausage-shaped density corresponding to the position of the local symmetry axis. To obtain more accurate parameters for the position of the local symmetry element, Eq. (18) should be used. After the maximization of the overlap, the rotational parameters of the local symmetry axis is given by $[C]$. The position of the local symmetry axis is given by $(s_A + s_B)/2$, and $d_{//}$ is given by the projection of $s_B - s_A$ along the direction of the local symmetry axis.

DESCRIPTION OF INPUT COMMANDS

Appendix A contains a listing of the new features that are available for the current version of the program. Appendix B contains example inputs.

All input commands to the program are keyword-based and free-formatted. The input parser converts lower case characters to upper case so the program commands are not case dependent (though the program keeps the cases of the file names and the title unchanged). Each input line can contain at most 80 characters. The following characters are recognized as delimiters between words – a space, a comma, a tab, and an equal sign. More delimiters can be implemented by inserting them in the array SPACER in subroutine PARSER (and change the variable NSP at the same time). Comments in the input can be introduced by using the COMMent command or using the special character "!" — in the input parser, all characters in the input line beginning at the "!" are ignored.

Presently, the program uses five logical I/O units, labelled by the variables LIN, LOG, LPRT, LOBS, and LSCRCH, which are initialized to be 5, 6, 3, 1, and 2, respectively. (The default values of most of the input variables are initialized in subroutine INITSS). All input commands are read from unit LIN. The output of the program will be written to unit LPRT. A name can be specified for this print file by using the PRINt command. The reflection data (if any) are read from unit LOBS. This unit is also used internally to read and/or write rotation function map files. The scratch file on unit LSCRCH is used for echoing the input commands at the end of the output file. Program messages are written to unit LOG. They are informational messages, warning error messages (*Warning*), or fatal error messages (*Fatal*). Each message is preceded by the name of the subroutine that produced it.

What follows is a description of the input commands that are currently supported by the program. A new command can be incorporated by inserting the command name in array COMMND, in subroutine INITSS, and by inserting a segment in subroutine INTPRT that defines the input parameters (if any) of the command. The program identifies each command by its first four letters, although more can be input for readability. In the following listing, the four-letter command keyword will be given in **bold face**, the names of input variables associated with the command will be given in UPPER case. The expected length of the character variables are also given (for example, ORDOR*5 means ORDOR is a character variable with 5 letters). Variables whose names begin with the letter 'Q' are logical variables. A 'True' of 'False' input is expected for these variables. The defaults (if any) of the variables are given in square brackets. These default values will be used by the program if the command is omitted from input, or if a value of 0 is input to the command.

I. GENERAL INPUT COMMANDS

COMMent none [none]

This can be used to incorporate comments in the command input file. The entire input line is simply ignored by the program. The other way to introduce comments is through the use of the special character "!" (see above). The difference between the two is that comments incorporated by this command will be echoed at the end of the program print file.

PRINt-File PRTFIL*72 [GLRF.PRT]

This provides a name for the output file of the program (associated with the logical unit LPRT). The LOG unit is meant to be associated with the system output (terminal screen in interactive mode and job log file in batch mode).

STOP none [none]

The program will stop reading from the command input file and start the actual calculation. Otherwise, the program will read until the end of file on unit LIN before initiating the calculation.

TITLe ATITLE*132 [PROGRAM GLRF]

This provides a title for the current run of the program. It will be placed at the beginning of each output page. The program will insert the current date and time at the beginning of ATITLE (characters 2-18), and the program will append a version number identifier at the end of ATITLE (characters 111-130).

II. DEFINITION OF A FEW CONVENTIONS

EULEr-Angle EULER*3 [ZYZ]

This specifies the definition convention of Eulerian angles. Two conventions are currently supported — ZXZ and ZYZ, corresponding to rotation around the Cartesian Z axis (θ_3), then around the X (and Y, respectively) axis (θ_2), and finally around the Z axis (θ_1). ZXZ is the convention described in M. G. Rossmann & D. M. Blow *Acta. Cryst.* **15**, 24, (1962). ZYZ is used in program MERLOT (P. M. D. Fitzgerald, *J. Appl. Cryst.* **21**, 273, (1988)). The angles should be input to the program in the following order — $\theta_1, \theta_2, \theta_3$, and the program outputs the angles in the same order.

WARNING: The matrix used in this program is the transpose of the matrix as printed in Table 1 of the paper by Rossmann & Blow. Therefore, if you are comparing the results from this program with other programs, be sure that the programs are using the matrices the same way! The same goes for the command POLAr (see below).

Consequently, the search limits for angles θ_1 and θ_3 as defined by S. N. Rao, J.H. Jih & J. A. Hartsuck (*Acta Cryst.* **A36**, 878–884, (1980)) should be swapped. The θ_1 limit should be used as the limit on θ_3 in this program. In any case, the user is strongly encouraged to check out the symmetry of the rotation function with the program. A few (1 or 2) large terms and very large intervals (5-10°) should be used to speed up the calculation.

POLAr-Angle POLAR*3 [XZK]

This specifies the polar angle definition convention. Two conventions are currently supported — XYK and XZK. In both cases, ϕ is the angle from the Cartesian X axis. In convention XYK (as defined in Rossmann & Blow), ψ is the angle from the Y axis. In convention XZK (as in MERLOT), ψ is the angle from the Z axis. κ is the rotation around the axis defined by ϕ and ψ. The angles should be input to the program in the following order – ϕ, ψ, κ, and the program outputs the angles in the same order.

The XZK convention should be used for most calculations. The XYK convention can be used for monoclinic space groups in the b-unique setting.

ORTHogonalization ORDOR*5 [AXABZ]

This specifies the orthogonalization convention that should be used by the program. Four conventions are currently supported – BYBCX, CZBCX, AXABZ, CXCAZ, and Y3XYC. In convention BYBCX, the real space b axis coincides with the Cartesian Y axis, and $b \times c$ (or a^*) coincides with the X axis (this is first defined in the paper by Rossmann & Blow). In convention CZBCX, the real space c axis coincides with the Cartesian Z axis, and $b \times c$ (or a^*) coincides with the X axis (this is IPER=1 in MERLOT). In convention AXABZ, the real space a axis coincides with the Cartesian X axis, and $a \times b$ (or c^*) coincides with the Z axis (this is used by the Protein Data Bank, FRODO's helper SAM, PROTEIN, and X-Plor. This is also option NCODE=1 in program ALMN). Convention CXCAZ corresponds to NCODE=3 in ALMN and is suggested by David Borhani. This convention should be used only for monoclinic space groups in the b-unique setting. Convention Y3XYC, courtesy of Dr. Murthy, has been designed for rhombohedral space groups in the rhombohedral setting. It aligns the crystal three-fold axis (direction 111) along the Y axis, and puts the real space c axis in the XY plane (−X, +Y quadrant). Other conventions can be incorporated as well, by inserting the codes in subroutine GTOMDM, which calculates the orthogonalization matrix from the cell parameters. The deorthogonalization matrix is calculated as the inverse of the orthogonalization matrix.

The AXABZ convention should be used for most calculations. The BYBCX convention was designed for monoclinic space groups in the b-unique setting to align the crystal b axis along the Cartesian Y axis. The AXABZ convention will maintain this alignment and should therefore be used for monoclinic space groups as well.

III. SPECIFICATION OF LOCAL SYMMETRY

LOCSymmetry (RLCAXS(i), i=1, 3), MLCAXS, TLCAXS*1 [none]

This defines the local symmetry elements in a standard orientation. Only the unique symmetry elements need to be specified for this purpose. For example, for 222 symmetry, only two orthogonal two-folds need to be given. For icosahedral symmetry, only two symmetry axes (one two-fold and one non-orthogonal five-fold, one three-fold and one non-orthogonal five-fold, or two five-folds, etc) need to be given. The rest of the symmetry operators can be automatically generated by the program (see command LOCExpand). The symmetry operator can be specified as a set of Eulerian angles (TLCAXS='E'), or a set of polar angles (TLCAXS='P') (the interpretation of the angles depends on the convention used, see II), or the Cartesian coordinates of the end point of a vector from the origin (TLCAXS='V'), or as direction cosines (TLCAXS='D'). The default for TLCAXS is Polar. The unique rotation around the axis is specified by MLCAXS, either as an angle or as a divisor of 360. If the axis is specified as polar angles, the information from the polar angle supercedes the value specified for MLCAXS, and a warning will be issued if the information from the two sources do not agree. The maximum number of local symmetry axis that can be specified by this command is given by the MAXLAX parameter.

The program converts the input to a rotation matrix immediately, so the angle convention used to define the local symmetry can be different from that used later in the search.

If no local symmetry elements are defined by the input, the program will carry out ordinary rotation function calculations.

LOCExpand QLCXPD [T]

If QLCXPD is true, the program will generate all the local symmetry operators by pairwise multiplication of all the known operators until no new symmetry element is generated by the process. For this algorithm to work, the positions of the starting local symmetry axes must be specified with sufficient accuracy. If QLCXPD is false, the program will not carry out the above expansion. This is useful in cases where improper rotation axes are being dealt with (for example, a local two-fold with a rotation of 175°).

A set of polar angles (in the convention specified by POLAR) will be extracted from the rotation matrices and the local symmetry operators are sorted on these angles before they are output. The maximum number of local symmetry operators is given by the MAXLOC parameter.

LOCInterpolate QLCINT [F]

This is useful only if the slow rotation function is used to calculate the locked rotation function. If QLCINT is true, an ordinary rotation function will be calculated with the slow rotation function, covering the entire rotational space (0-360° in θ_1 and θ_3 with a grid interval of 2.8125°, 0-180° in θ_2 with a grid interval of 3.0°). The locked rotation function is then calculated by interpolating among the ordinary rotation function values. (Note that this is the mode of operation if the fast rotation function is used.) If QLCINT is false, the locked rotation function will be calculated directly, with the slow rotation function. For initial searches, QLCINT should be set to true (or the fast rotation function should be used). For fine searches, QLCINT should be set to false (also see command PKFIt).

LOCOrient (ALCORI(i), i=1, 3), TLCORI*1 [0.0, 0.0, 0.0, E]

The rotation that should be applied to the standard orientation (as defined by LOCSymmetry and LOCUpdate commands) to bring it to that in the crystal is defined with this command. This command is useful only for the locked cross rotation function. The rotation defines the matrix $[F]$. The angles can be either Eulerian (TLCORI='E') or polar (TLCORI='P'). The default for TLCORI is Eulerian. The convention of the angles can be specified by POLAr or EULEr. The input is immediately converted to a rotation matrix, so the angle convention definition can be temporary.

LOCUpdate (ALCUPD(i), i=1, 3), TLCUPD*1 [0.0, 0.0, 0.0, E]

The orientation of the local symmetry operators as defined by the LOCSymmetry commands can be changed with this command. The updating angles can be either Eulerian (TLCUPD='E') or polar (TLCUPD='P'). The default for TLCUPD is Eulerian. The convention of the angles can be specified by POLAr or EULEr. The input is immediately converted to a rotation matrix, so the angle convention definition can be temporary.

IV. COMMANDS THAT DEFINE THE FIRST (A) CRYSTAL

The program can deal with the reflection data of three crystals, named A, B and C. It is the A crystal that is rotated in real space to match the B crystal. This means that, in the case of cross rotation search with a model, the calculated structure factors based on the model should be input as the A crystal, and the resulting rotation matrix should be applied directly to the search model. In the case of self rotation search, the A crystal information is copied into the B crystal.

ACELl-Parameters (CELL(i, 1), i=1, 6) [1.0, 1.0, 1.0, 90.0, 90.0, 90.0]

This inputs the unit cell parameters of the A crystal. The parameters can be either in real or reciprocal space. The angles can be either in degrees or in cosines.

ASYMmetry [none]

This inputs the space group symmetry operators of the A crystal, in the same format as the International Table. Positions related by centering should not be input. Positions related by inversion or mirror planes are not supported by the program. If no symmetry card is given, the program defaults to $P1$. The identity operation is assumed and need not be specified.

The space group symmetry can also be specified by giving the symbol of the space group (for example P212121) or the number of the space group (for example 19). For monoclinic space groups, only the b-unique setting is supported. For rhombohedral space groups, only the hexagonal setting is supported. If you want the other settings, input the symmetry operators explicitly.

AOBS-File OBSFIL*72 [FOBS.DAT]

This specifies the name of the formatted file holding the reflection data of crystal A. The file should contain h, k, l, F, and optionally σ_F for each reflection. The format of the file is specified by the command AFORmat. The file should be terminated by the end of file, not by an end-of-file record. The maximum number of reflections that can be saved is given by the MAXREF parameter.

AFORmat OBSFMT*72 [(3i4, 2f8.2)]

This specifies the format of the reflection file, for the following input list – h, k, l, F, σ_F. The parentheses in the format are optional. Note that a format must be specified for σ_F even if the file does not contain σ_F values.

ACUT-off SIGCUT, FCUTLO, FCUTHI [1.0, 1.0, 0.0]

This specifies the cut-off criteria for screening the reflection data of the A crystal. Reflections with F(input) $<$ SIGCUT $\times$ σ_F(input), or with F(input) $<$ FCUTLO, or with F(input) $>$ FCUTHI will be rejected from the calculation. If FCUTHI is 0.0, no high cut-off will be carried out. F(input) and σ_F(input) are the values as read from the input reflection file. The program will print out how many reflections are removed by each of the above criteria, so reasonable inputs can be selected accordingly.

APOWer POWER [2.0]

The intensity of reflections in the A crystal will be calculated as F(input)**POWER. Therefore, if intensity data is being input, set POWER to 1.0.

NSHEll NSHELL [8]

The reflection data will be divided into NSHELL shells of equal reciprocal volume, and an average intensity will be calculated for each shell. This will be useful for origin removal (see command ORIGin) and for selection of large terms (see command CUTOff). The maximum number of shells is specified by the parameter MAXSHL.

ORIGin-Removal QORIGN [T]

The origin of the A crystal Patterson map will be removed if QORIGN is true. In that case, an average intensity, calculated in shells of resolution, will be subtracted from each reflection.

V. COMMANDS THAT DEFINE THE SECOND (B) CRYSTAL

First, the commands that parallel those for the A crystal –

BCELl-Parameters (CELL(i, 2), i=1, 6)
BSYMmetry
BOBS-File (OBSFIB*72)

BFORmat (OBSFMB*72)
BPOWer (BPOWER)
BCUT-off (BSGCUT, BCUTLO, BCUTHI).

These input variables default to the values for the A crystal.

CUTOff CUTOFF [1.5]

This defines the selection criterion for the strongest reflections (large terms) from the B crystal data. Reflections with I > CUTOFF × I(average) will be saved as the large terms, where I(average) is calculated in NSHELL shells of equal reciprocal volume. A CUTOFF value of 1.5 usually results in about 20% of the data being saved as large terms, which is generally sufficient to obtain accurate slow rotation function values. The maximum number of large terms that can be saved (after expansion to a hemisphere by crystallographic symmetry) is specified by the parameter MAXBIG.

VI. COMMANDS THAT DEFINE THE THIRD (C) CRYSTAL

A third crystal, C crystal, can be input to the program to carry out two cross rotation searches simultaneously. This will be useful, for example, if a sample has been crystallized in two different forms. Knowledge of the orientational relationship between molecules in the B crystal and the C crystal is required. This can be determined by a cross rotation search between the B and the C crystals. For a detailed discussion, see Appendix C.

The C crystal information is input to the program through the command CXTAl –

CXTAl OPTION*4 []

where OPTION is a four-letter keyword that specifies the function of the command. A list of the currently supported OPTIONs follows.

BTOC-relationship — (B2CANG(i), i=1, 3), B2CCON*1
This specifies the rotational relationship between molecules in the B and the C crystals. A molecule in crystal B must be rotated by B2CANG to match the orientation of a molecule in crystal C. The angles B2CANG are interpreted as Eulerian if B2CCON is 'E' or polar if B2CCON is 'P'. The angle convention is defined by EULER or POLAR. The maximum number of relationships that can be specified is given by the MAXLAX parameter.

CELL-parameters — (CELL(i, 3), i=1, 6)
This specifies the cell dimension of the C crystal.

FOBS-file — OBSFIC*72
This specifies the name of the C crystal reflection file.

FORMat — OBSFMC*72
This specifies the format of the C crystal reflection file. The default is (3i4, 2f8.2) for the input list h, k, l, F, and σ_F.

POWEr — CPOWER

This specifies that the input structure factors for the C crystal should be raised to the power of CPOWER to calculate the intensities of the reflections.

REJEction-criteria — CSGCUT, CCUTLO, CCUTHI

This specifies the screening criteria for the C crystal data.

SYMMetry —

This defines the space group symmetry of the C crystal.

The program will read in the reflections and save only the large terms, selected based on the CUTOFF (see command CUTOff).

VII. SPECIFICATION OF SEARCH PARAMETERS

BOXSize (IBXHKL(i), i=1, 3) [3, 3, 3]

This specifies the interpolation box size around each rotated reflection. The size of the box should be big enough to cover the maximum of the G function. At the same time, it should not be too big because the calculation time is proportional to the volume of the box. For most cases, a $3 \times 3 \times 3$ box should be large enough. A bigger box may be used in the case of centered lattices due to the systematic absences. The maximum dimension of the box is specified by the MAXBOX parameter.

CROSs-search QCROSS [F]

This specifies that the program should perform cross rotation calculations. QSELF (see command SELF-search) and QCROSS are mutually exclusive. Specifying QSELF to be true will disable QCROSS automatically. Due to its default, QCROSS will need to be set to true for cross rotation searches. It also must be true if cross rotation calculations need to be performed by the commands LOCRotate, and MATRix. If QCROSS is false, the program will carry out self-search calculations.

GEVAluation IGEVAL [2]

This specifies how the G function values should be calculated. Currently supported options are – IGEVAL=1, which means the G values will be obtained from a table look-up based on the value of r^2; IGEVAL=2, which means the G values will be sampled in the interpolation box at an interval of DHKL in h, k, and l (DHKL is a program parameter, currently set to be 0.1); IGEVAL=3, which means the nearest integer of the rotated index will be used in the calculation (fastest way of getting the G value, but with the most error. Recommended only for checking the symmetry of the rotation function); and IGEVAL=4, which means the G values will be calculated based on the formula from the distance (r). See the paper by Rossmann & Blow for detailed description of the G function.

MAPFile MAPFIL*72 [none]

If a file name is specified by this command, the program will dump the rotation function values as a formatted file. Default is that the map values will not be dumped. This file may be useful for later processing with other programs (see the command PEAK).

OANGle OANGLE*1, OTYPE*3 [P, XZK]

The rotation function peaks will also be converted to Eulerian (OANGLE='E') or polar (OANGLE='P') angles for output. OTYPE specifies the convention that should be used with the Eulerian or polar angle (see II). Therefore, two sets of angles are output corresponding to each peak in the rotation function map – the first set is defined by the SANGle command, the second defined by this OANGle command.

PEAK-Cutoff PKCUT, NLIST, MAPFIL*72 [3.0, 40]

The program will perform a peak search in the resulting rotation function map. In this peak search, grid points whose values are less than PKCUT $\times$ σ (where σ is the standard deviation of the map values) over the map average will be skipped. NLIST peaks will be printed at the end of the peak search. The maximum number of peaks that can be saved is given by the MAXPKS parameter.

If a file name has been specified for MAPFIL, the program will read the map values from that file and perform a peak search. In this case, no self or cross rotation search will be carried out.

PKFIt NPKFIT, PKFCUT [5, 1.5]

The program can automatically carry out fine searches, with the slow rotation function, for the top NPKFIT peaks in the ordinary or locked rotation function. Specifying this command will also set the logical QPKFIT to true, which defaults to false. If a negative number is input for NPKFIT, QPKFIT will be set to false. PKFCUT is the large-term cut-off that will be used by the slow rotation function. The program will first search in 1° intervals to locate the peak. Then the grid interval will be lowered to 0.5°. The peak positions obtained from these fine searches can be used directly by the P1PC and GLTF commands.

PRERotation (AUVW(i), i=1, 3), (BUVW(i), i=1, 3) [0]

This command performs a "pre-rotation" which aligns a direction in the A crystal, specified by AUVW, with a direction in the B crystal, specified by BUVW. This command can be used only if a cross-rotation search is being performed. The direction can be specified either as a set of fractional coordinates, in which case the vector from the origin to this point will define the direction, or as a set of polar angles – ϕ, ψ, and the third field should be 'P'olar or a number less than -999.0. The polar angle convention is taken from the POLAR variable and the orthogonalization convention is taken from ORDOR. The subsequent search will be carried out in polar angles. Only κ will be allowed to change, ϕ and ψ are determined by BUVW.

RADIus RADIUS [30.0]

This specifies the radius of integration, in Å.

RCUToff RCUTLO, RCUTHI [0, 20]

RCUTLO has been implemented mainly as a way of saving computation time.

In the ordinary rotation search calculated by the slow rotation function, if the rotation function value at a search grid point is less than RCUTLO (default is 25 for self-search and 500 for cross-search), all the neighbors of this grid point is flagged and will be removed from further calculations. These grid points are output in the map file (see command MAPFile) with a value of -999. If there are more than 2500 grid points in the search, the program will determine RCUTLO automatically and carry out a coarse search through the grid points first and remove those grid points that are unlikely to give high rotation function values.

In the locked rotation search, if the rotation function value at a search grid point for one of the local symmetry axis is less than RCUTLO (default to be -100 for self-search and 0 for cross-search), this grid point will be flagged and the program will skip to the next grid point. If the fast rotation function is used, the program will determine RCUTLO automatically.

RCUTHI has been implemented to deal with the overlap of local symmetry axes with the crystallographic axes in the locked self rotation function searches. In the locked self rotation search, if a local rotation axis is within RCUTHI degrees of a crystallographic one, the local axis will not be used in the locked self rotation function.

RESOlution-limits DMAX, DMIN [10.0, 3.5]

This specifies the resolution limits for reflection data that will be used in the search. This applies to both the A and the B crystal. The maximum Miller indices are given by the parameters MAXH, MAXK, and MAXL, respectively. The program will abort if either of them is exceeded.

SANGle SANGLE*1 [E]

This specifies whether the search will be carried out in Eulerian or polar angles. For ordinary self rotation searches, polar angles are preferred (for example, section of $\kappa = 180°$ to look for two-fold axes). For cross rotation searches and for locked rotation searches, Eulerian angles is the better choice because the symmetry in the rotation function can be recognized more easily and because it is easier to evenly sample the rotation space with Eulerian angles.

SECTion SECSHN*3 []

The sectioning of the map in the output map file (see command MAPFile) is defined by this command. The first field specifies the map sections, the second field specifies which angle will be output across the page, and the third field specifies which angle will be output down the page. For example, specifying '321' would mean θ_3 (κ) sections, θ_2 (ψ) across, and θ_1 (ϕ) down the page. The second and/or the third field can be omitted and the program will provide reasonable defaults. If the command is omitted, the program will choose the angle with the smallest number of grid points as the section direction. The angle with the largest number of grid points will be output across the page.

SELF-search QSELF [T]

This specifies that the program should perform self rotation calculations. In this case, the information for crystal A will be copied to get the B crystal.

SLIMits IAXIS, (SLIMTS(i, IAXIS), i=1, 3) [none]

This specifies the angle limits for the current search. Three separate SLIMits commands need to be given to define the limits in all three angles. IAXIS can assume values of 1, 2, and 3, corresponding to θ_1 (ϕ), θ_2 (ψ), and θ_3 (κ) in the case of Eulerian (polar) angles. The angle limits are specified as — beginning value, ending value, and increment.

If ordinary self- or cross-rotation function calculations are being performed in Eulerian angles, the program can automatically define the angle limits to cover the entire unique region of rotational space. To activate this feature, specify a number that is greater than 180 for the beginning or ending value of θ_2. Only the hexagonal setting is supported for rhombohedral space groups. If the increment value for θ_2 is input as 0, the program will also assign reasonable grid increment values along the three angles. Therefore, if an ordinary self- or cross-rotation function calculation in Eulerian angles covering the entire unique rotational space is desired, the search limits can be simply defined as 'SLIMits 2 270 270 0'.

For ordinary self rotation function in polar angles, specifying 'SLIMits 270 270 0' will define the search limits as 0 - 180° in ϕ and ψ in 3° intervals, and 60 - 180° in κ in 30° intervals. It is best that this option be coupled with the fast rotation function.

The program can also try to assign reasonable limits if a locked self or cross rotation function is being calculated. For locked self rotation function, the search region assigned depends on the symmetry of the A crystal. For locked cross rotation function, the entire rotational space will be assigned. The unique rotational space for a locked self and cross rotation function also depends on the symmetry of the non-crystallographic point group. It is suggested that the user first carry out a locked rotation search, with the fast rotation function, covering the entire rotational space to identify the symmetry in the locked rotation function.

For self-rotation searches, the map values are scaled such that matching the Patterson map onto itself without any rotation will give a height of 1000. This puts the rotation map values on an "absolute" scale so that results from different runs of the program can be compared. This also means that all crystallographic symmetry operators should give rotation function values of 1000. For cross-rotation searches, a rough scale factor is estimated before the search is carried out. After the search, the maximum value of the map is then re-scaled to 1000. In both cases, the maximum, minimum, average, and the standard deviation from average of the map values will be reported by the program.

XCORrelate QXCORR, NXCORR [F, 5]

This specifies whether the program will try to correlate the ordinary cross rotation function results with the self rotation function. The program will expand the cross RF peaks (or a set of angles given by the ANGLe command) by the crystallographic symmetry (similar to SYMShow command) and identify all the self rotation angles relating the various cross rotation angles. If reflection data have been given, the program will also calculate the self RF value for these angles, with the slow RF. The large-term cutoff for this is set at 1.5, and the origin will be removed.

VIII. THE FAST ROTATION FUNCTION

FAST-rotation-function QFRF [F]

Calculations with the Crowther fast rotation function (R. A. Crowther *The Molecular Replacement Method*, edited by M. G. Rossmann, 174–178, (1972)) are supported by this program. The maximum Bessel function order is 60. This allows the use of a larger radius of integration in the calculation (maximum radius allowed is roughly given by $10.8 \times D$, where D is the high resolution limit of the reflection data used in the calculation). By default, the program will use the 'slow' rotation function.

The ordinary self- or cross-rotation functions can be calculated using this technique, with either Euler or polar angles (see command SANGle). Internally, the program calculate the fast rotation function with Euler angles in convention ZYZ. If other Euler conventions or if polar angles are used for the search, the program will interpolate among the results from the fast rotation function.

Only the "large" terms (see command CUTOff) of BOTH the A and the B crystals are used. The recommended value for CUTOFF however is around 0.25.

If QORIGN (see command ORIGin) is set to true, the Patterson origins of BOTH the A and the B crystals are removed. The user is encouraged to try both possibilities to see which give the better results.

The entire rotational space (0–360°) is covered in θ_1 and θ_3 by the fast rotation function, with a grid interval of 2.8° (or more accurately $360/128 = 2.8125°$, where the number 128 is chosen to simplify the FFT calculation). The command SLIMts should be used to define the unique regions of the rotation function in θ_1 and θ_3. The search range and grid interval along θ_2 should be defined with the SLIMts command as well.

The fast rotation function as implemented here should work for most space groups, including the cubic ones (point groups 23 and 432). In the latter cases, the unique region of the rotation space can be defined based on point groups 222 and 422 and be aware then that the peaks listed by the program will contain crystallographically-related rotations (use the command SYMShow to sort out the peaks).

The fast rotation function can also be used, and is preferred, for locked rotation function calculations. The program will do the necessary interpolation to obtain values for the locked rotation function on the grid of the fast rotation function.

As with the slow rotation function, the program will carry out a peak search through the rotation function map. Similarly, contour maps can also be created based on the fast rotation function results. The fast rotation function results can also be written out to a file (see command MAPFile).

Once an initial peak position has been identified with the fast rotation function, finer searches around that position can be carried out with the slow rotation function.

NORMalize QNORME, EEMAX [F, 8.0]

Normalized structure factors (E values) can be used in the calculation of the fast rotation function. The reflection intensities are normalized by dividing with the average intensity at the resolution shell. The E values are allowed a maximum value of EEMAX. Default is that the intensities, rather than the E values, will be used.

IX. PATTERSON CORRELATION REFINEMENT

COORdinate OPTION*2 []

This defines the atomic model for the PC refinement. OPTION is a two-letter keyword that define the function of the command. Supported options are –

BOverall – BOVRAL.

This defines the overall B value to replace any that is on the input file.

BZero – BZERO.

This defines a B value that will be applied to any input atom with a B of zero.

INput – CORINP*72.

This defines the name of the file that contains the atomic coordinates in PDB format. The coordinates are expected to be in the same orientation as that used to calculate the structure factors supplied with the AOBS command. The program will automatically translate the model to place its center at the origin.

OUtput – COROUT*72.

The atomic coordinates after PC refinement can be written out. The default file name is rigid.pdb. The coordindates for all the rotation function peaks that are used in the PC refinement are written to this file. There are comment lines at the beginning of each segment of coordinate entries defining which rotation function peak they correspond to.

GROUP IGROUP, RES1*6, RES2*6 []

The first step of the PC refinement is always an overall refinement, using the entire molecule as a rigid body. This command can be used to define smaller groups that will be used in the second step of the PC refinement. IGROUP defines the number of the group, and RES1 and RES2 defines the beginning and ending residue number for the group. The command can be repeated to specify different groups or groups with more than one residue ranges. Residues that are not selected by any of the groups will not be changed by the refinement. Chain names are supported and should be separated from the residue name with /, |, or :. The question mark ? can be used as a wild card and matches to any character. For example, GROUp 1 A:???? A:????, GROUp 2 B:???? B:????. will define two groups for chains A and B.

P1PC-refinement NPCPKS, NPCCYC, PCCUT [5, 10, 1.0]

The program will carry out a Patterson-correlation refinement for the first NPCPKS peaks in the ordinary cross rotation function. NPCCYC specifies the number of cycles for each refinement step. PCCUT specifies a large-term cutoff that should be used with the PC-refinement. Default is that the PC-refinement will not be done.

X. LOCKED TRANSLATION FUNCTION.

GLTFunction OPTION*4 [ ]

This activates the locked translation function. For this function to work, a locked cross rotation search (either with slow or fast rotation function) must have been carried out first. The peaks in this search can then be used for the locked translation function (see option RFPEak below). The atomic coordinates, corresponding to the A crystal reflection data in the search, should be input with the COORdinate command (see the section on P1PC refinement above).

The program will do a peak search through the translation function and lists the top 10 peaks in the map. The maximum value of the function is scaled to 1000. The peak cut-off for this search is set at 3σ or 400, whichever is smaller. If packing check is not used during the calculation (see option PACK below), the position with a height of 1000 may not be present in the peak listing as it may have packing problems. The program will output the atomic model for the entire assembly corresponding to the top peak in the locked translation function to a file. A name can be specified for this file with the COORdinate OUtput command (see above). If more than one rotation peaks are used, the file will contain separate assemblies, each identified with the rotation function peak number. This model for the assembly should then be input to an ordinary translation function program (for example, TF) to find the center of the assembly in the unit cell.

OPTION is a four-letter keyword that define the function of the command. Currently supported options are (in alphabetical order) –

CUTOff – GLTFCT.

The large-term cutoff that should be used for the locked translation function. Default is 1.5.

DIREct-summation – QDRECT.

This specifies whether the locked translation function should be evaluated by direct summation. Default for QDRECT is false, meaning the function will be evaluated by FFT. If the local symmetry is high (point group 432 or higher), the program will force direct summation.

MAPFile – LTFMAP*72.

This specifies a name for a file where the locked translation function map will be dumped. Not implemented yet.

OUTPut – NTFOUT, NTFOUP.

This specifies whether the program will output all the atoms (NTFOUT=1) or only the Ca atoms (NTFOUT=2) for the locked TF solutions. The default for NTFOUT is 1. NTFOUP specifies the number of solutions that should be written out for each orientation. Default for NTFOUP is one, meaning only the top solution will be written out.

OVERlap – NOVLP, DCUTTF.

This specifies the maximum number of close Ca - Ca (or P - P for nucleic acids) contacts that are allowed in the packing check. The distance cut-off for this check is given by DCUTTF. The default for NOVLP is 40. The default for DCUTTF is 3Å, which should be appropriate for protein structures. For nucleic acid structures, try using DCUTTF of 5Å. The program will try to save only 200 Ca atoms, which may lead to the selection of every second, or third Ca atoms only. In such cases the value of NOVLP will be changed automatically by the program.

PACKing-check – QPACK.

This specifies whether packing of the monomers in the assembly should be checked during the calculation. Default for QPACK is false. This is coupled with the default for QDRECT of false. If direct summation is used, QPACK should be set to true. If this packing check is not done during the calculation, it will be performed during the peak search. The packing check during the peak search also examines the steric contacts among the assemblies that are related by unit cell translations. If the number of contacts for either packing check is greater than a pre-set number (see option OVERlap), the grid point will be ignored. The number of contacts found within the assembly is listed under 'Overlap'. The number of contacts found among the assemblies is given under 'Cell' in the peak listing.

RADIus – RADMAX, RADMIN.

This defines the maximum and the minimum lengths of the translation vector. Grid points in the search region (see option SLIMts below) that are more than RADMAX angstrom away from the origin will be ignored. The default for RADMAX is 30Å. Grid points in the search region that are less than RADMIN angstrom away from the origin will be ignored. The default for RADMIN is 5Å.

RFPEak – NTFPKS.

The top NTFPKS peaks in the locked cross rotation function will be used for the locked translation function. The program will do a separate calculation for each rotation peak. Alternatively, if NTFPKS has been specified as a number between -1 and 1, the program will select all the locked cross RF grid points with heights greater than NTFPKS times the maximum value of the RF for the locked TF calculation. This is similar to the combined molecular replacement option in the TF program of this package.

SLIMits – I, BEGIN, END, INCREMENT.

This defines the search region for the translation vector. The integer I can take the value 1, 2, or 3, corresponding to the Cartesian coordinates X, Y, or Z. The values should be given in angstrom units.

XI. MISCELLANEOUS COMMANDS

ANGLe X, Y, Z, TANGLE*1 []

This defines an angle to the program. Currently this information is used only by the XCORrelate command. See command LOCRotate for more information.

LOCAsu (RLCASU(i), i=1, 3) [10.0, 20.0, 30.0]

This command instructs the program to explore the symmetry of the locked self rotation function (QSELF should be true). The program first updates the local symmetry matrices based on the angles specified by RLCASU (similar to LOCUpdate), it then carries out a grid search, using parameters defined by the SLIMits commands and the SANGle command, to find other sets of angles that are related to RLCASU by the symmetry. Internally, the command LOCAsu assigns a true value to the logical QLCASU, which defaults to false. Therefore, LOCAsu command must be supplied, even though you want the default angles for RLCASU (i.e., you can input a line that says “LOCA” only).

Symmetry in the locked cross rotation function can be determined analytically and is not supported as yet.

LOCFit ANGINC, ZANGLE [0.5, 2.5]

This command performs an ordinary self rotation search for each of the local symmetry operators at a given orientation. The orientation can be derived by updating the standard orientation with the LOCUpdate command. The program searches ZANGLE degrees to either side of the known ϕ and ψ positions, with ANGINC as the interval between grid points. Internally, the commands sets the logical variable QLCFIT to true, which defaults to false.

LOCRotate (ALCSHO(i), i=1, 3), TLCSHO*1 [0]

If QSELF is true, this command takes a set of angles and updates the local symmetry operation matrices based on the rotation matrix of the input angles. The angles can be either Eulerian (TLCSHO=‘E’) or polar (TLCSHO=‘P’). The default for TLCSHO is Eulerian. The convention of the angles can be specified by using the EULEr or the POLAr commands (see II). If reflection data for crystal A have also been supplied, the program will obtain the rotation function value at each of the updated positions. The program also produces a stereographic representation of the set of updated local symmetry rotation axis. A set of polar angles are extracted from each of the updated matrices using the convention specified by the POLAr command. The top half of the sphere (z > 0 in the case of XYK polar angles, so $\pi < \phi < 2\pi$, and y > 0 in the case of XZK polar angles) is projected. The PostScript file is named STEREO.PS.

If QCROSS is true, the updated matrix is calculated as $[F][I_i][E]$, where $[E]$ is the matrix based on the input angles. A set of angles (polar if SANGLE='P', Eulerian if SANGLE='E') is extracted from the matrix. If reflection data has also been input for the A and the B crystals, the program will calculate the rotation function value corresponding to each of the updated matrices.

The maximum number of angles that can be input by this command is specified by the parameter MAXSHO.

MATRix (AMTSHO(i), i=1, 3), TMTSHO*1 [0]

This commands takes a set of angles and outputs the corresponding rotation matrix. The variables AMTSHO and TMTSHO are similar to those for command LOCRotate. The program calculates the rotation matrix immediately after this command is specified. Therefore, a temporary angle convention definition can be specified. If a pre-rotation has been applied (through the PRERotate command), the program will concatenate the two rotation matrices and extract a set of angles based on the new matrix. The angle type is specified by SANGle, and the angle convention by EULEr or POLAr. The program then applies the orthogonalization and deorthogonalization matrices. Therefore, the correct cell dimensions should be input to the program. If reflection data has been input, the program will output the contribution of the top 50 large terms to the final rotation function value. In this case, IGEVAL (see command GEVAluate) is set to 2. The maximum number of angles that can be input by this command is specified by the parameter MAXSHO.

SYMShow (ASYMSH(i), i=1, 3), TSYMSH*1 [0]

This commands takes a set of angles and outputs all the angles that are related to the input set by the symmetry of the ordinary rotation function. The variables ASYMSH and TSYMSH are similar to those for command LOCRotate. The program calculates the rotation matrix immediately after this command is specified. Therefore, a temporary angle convention definition can be specified. Calculations with both self and cross rotation functions are possible. The program will output the angles as Eulerian if SANGle is 'E' or as polar if SANGle is 'P'. The angle convention is given by EULEr or POLAr. Note that the calculations in this command ignore the local symmetry information. Use the command LOCAsu to explore the symmetry of the locked rotation function. The maximum number of angles that can be input by this command is specified by the parameter MAXSHO.

XII. ELECTRON DENSITY ROTATION TRANSLATION FUNCTION

RSRF OPTION*1, X, Y, Z, CODE*1 [none]

This command specifies parameters for the real space (electron density) rotation function (see Appendix D for an introduction). This can be used to maximize the overlap of the electron density of two molecules related by local symmetry. When OPTION='R', X, Y, and Z will be interpreted as a set of rotation angles relating the molecule in the A crystal to that in the B crystal. The rotation angle type (Eulerian or polar) will be taken from SANGLE (command SANGle) and the angle convention will be taken from EULER or POLAR. When OPTION='A', X, Y, and Z will be interpreted as the center of the molecule in the A crystal, in fractional coordinates. When OPTION='B', X, Y, and Z will be taken as the center of the molecule in the B crystal. Therefore, three separate RSRF commands will be needed to define the relationship between the molecule in the A crystal and that in the B crystal. Set the CODE to 'S' if a search is to be carried out for the corresponding OPTION. Note that only one OPTION can have the code 'S' at one time.

The two molecules can reside in the same crystal (self-rotation calculation) or different crystals (cross-rotation calculation). The program will read in phase angles for each reflection. The search limits will need to be defined with the SLIMts commands. Note that in this case the limits defined are taken as the offsets from those values defined by the RSRF commands. The G function evaluation option (GEVAluate option) will be set to 2.

TFUNction ITFUNC, (TFANGL(i), i=1, 6) [0]

This command calls the translation function routines. The program has an internal logical variable called QTFUNC, which defaults to false. Specification of this command sets QTFUNC to true. Currently, three types of translation function are supported.

The first, ITFUNC=1, searches for the translation element along the local symmetry axis, whose direction is given by the angles TFANGL(1), TFANGL(2), TFANGL(3). (The angle type is specified by the SANGLE variable and the angle convention is given by the POLAR or EULER variable). This function only works for local two-fold axes (M. G. Rossmann, D. M. Blow, M. M. Harding, E. Coller, *Acta Cryst.* **17**, 338, (1964)).

The second, ITFUNC=2, searches for the location of the specified local symmetry axis in the unit cell (see Appendix E). In this case, a set of phase angles must be present in the reflection file. Similar to the case for ITFUNC=1, TFANGL(1), TFANGL(2), TFANGL(3) defines the orientation of the local symmetry axis as a set of polar or Eulerian angles. TFANGL(4) defines the translational element (in Å) along this local symmetry axis. The program will break it down to three translation elements along the unit cell edges. Be sure to confirm that the signs of those elements are correct. If not, then the sign of TFANGL(4) may need to be reversed.

The third, ITFUNC=3, searches for the position that is related to an input point by the local symmetry axis (see Appendix F). This also requires a set of phases to be input to the program. TFANGL(1), TFANGL(2), TFANGL(3) defines the orientation of the local symmetry axis. TFANGL(4), TFANGL(5), TFANGL(6) defines the input position, in fractional deorthogonalized units.

In all three cases, the program produces a formatted file called TFUNi.DAT, where i=1, 2, or 3, which contains h, k, l, F, and ϕ values for a set of reflections. The maximum coefficient (F) has been scaled to a value of 1400. These reflections can then be input to any Fourier summation program to obtain the translation function map (in space group $P1$). The maximum number of reflections that can be saved by the program is given by the parameter MAXREF.

The program will also try to carry out the Fourier transform directly. A warning will be issued if array limits are exceeded and the program can not perform the transform. A peak search is done through the transform map. The map can also be contoured (command CNTFile). In this case, the desired sectioning of the contour should be specified (command SECTion). The contour level (command CNTLevel) can be set at 500, 1000, 50, as the program will scale the maximum of the transform to 1000. The contour sections (command CNTSection) can be left as default, which will lead to the contouring of 5 sections containing the top peaks. The map can be written to a file (command MAPFile), which can be used for re-contouring the map later (command PEAK).

XIII. CONTOUR PLOTS

CNTFile CNTFIL*72 [none]

If a file name has been specified by this command, the program will produce contour plots for the rotation function map. Default is that the contour plot will not be created.

If an one-dimensional search has been carried out (*i.e.*, only one angle was allowed to vary), the rotation function values will be plotted as a function of the varying angle. In this case, the CNTLevel and CNTSection commands are ignored by the program.

If polar angles were used in the search, the contour plot will be done stereographically in sections of κ (with projection down Z if POLAR is XYK and down Y if POLAR is XZK). To make this projection consistent with the convention used in other plots (see command LOCRotate), the ψ angle needs to be kept between 0 and 180°; the ϕ angle should lie between 0 and 180° if convention XZK is used, and between 180 and 360° if convention XYK is used (see command SLIMits). Normally, stereographic projection will plot out the bottom half ($y < 0$) of the sphere for XZK polar angles and the top half of the sphere ($z > 0$) for XYK polar angles. If a self rotation function is calculated in polar angles, the three coordinate axes in the stereographic contour plot will be labelled with the appropriate unit cell axes (a, $a*$, or $a//$, which means the direction is roughly along the unit cell a axis).

If Eulerian angles were used in the search, the sectioning of the contour plot will be taken from the SECSHN parameter (see command SECTion).

The contouring algorithm assumes continuity in the rotation function map values. This may not be true if a high RCUTLO value (see command RCUToff) has been selected during the calculation. In these cases, there might be some strange contour lines in the plot. To alleviate this problem, the RCUTLO value will need to be reduced.

The maximum number of points in each contour trace is given by the parameter MAXCVS. If the program is stopped because this limit is exceeded, either recompile the program with a larger MAXCVS or raise the starting contour level.

CNTLevel (CNTLVL(i), i=1, 3), LINTYP [none]

This command specifies the beginning, ending, and increment for the contour level. LINTYP specifies the line type to be used for the contours in this level. This command can be repeated to obtain different contour levels. The contour levels can be specified as absolute values (with the maximum of the map scaled to 999) or in terms of the σ value of the map. In the latter case, the beginning and ending contour levels must be specified as less than 10 and the increment must be less than 2. For example, specifying 'CNTLevel 1 5 0.5' means the contouring should begin at 1σ above the map average and end at 5σ above the map average, with increment of 0.5σ. The maximum number of contour levels that can be specified is given by the MAXLVL parameter.

Currently 7 different line types are supported. LINTYP=1 is a solid line 0.1 point wide (there are 72 points to an inch). LINTYP=2 is a solid line 0.5 point wide. LINTYP=3 is a solid line 1 point wide. LINTYP=4 is a dashed line (2 on 2 off) 0.1 point wide. LINTYP=5 is a dashed line 0.5 point wide. LINTYP=6 is a dashed line (4 on 1 off) 0.1 point wide. LINTYP=7 is a dashed line 0.5 point wide.

Since it is difficult to predict the optimal contour levels beforehand, it is suggested that the rotation function values be written out with the MAPFile command. Then contour level(s) can be chosen and the rotation function map values can be read in with the command PEAK.

CNTSection ICNTSB, ICNTSE [0, 0]

This specifies the section range for which the contour plot should be created. The numbering of the sections starts with the first section being number 1. The default is that all the sections will be plotted if there are 5 or less sections in the map. If there are more than 5 sections, the program will plot a maximum of 5 sections containing the top peaks in the map. If no peaks are found in the map, the first 5 sections of the map will be plotted.

XIV. STORAGE LIMITS

The storage limits for different parameters are defined in a PARAMETER statement in the Fortran INCLUDE file GLRF.CMN. The program will need to be recompiled if any of these limits is exceeded. Especially, the parameters MAXH, MAXK, and MAXL should be adjusted to best match the cell parameters under study. The following table lists a set of reasonable values for the parameters. The actual values used are reported at the end of the output file.

Parameter	Limit
MAXBIG	80000
MAXBOX	5
MAXGRD	1000000
MAXH	75
MAXK	75
MAXL	75
MAXLAX	10
MAXLOC	100
MAXPKS	1000
MAXREF	100000
MAXSHL	10
MAXSHO	10
MAXATM	12000
MAXLVL	5
MAXLIN	7

Appendix A.

New Features

New Features of Version 3.4 — June, 1996

- The locked TF can now use grid points in the locked cross RF that are above a specified threshold. Also, an option to output only the Ca atoms of the locked TF solutions is implemented (GLTF OUTPut).
- The fast RF can now be used for any Euler angle conventions or even polar angles. The program will do interpolation if necessary.
- Bug fix in ORFLRF for the interpolation.

New Features of Version 3.3 — April, 1996

- Calculation of fast rotation functions with normalized structure factors is supported. New command NORMalize.
- Calculation of the self rotation function in polar angles with the fast RF, by interpolation.
- Correlation of ordinary cross RF results with the self RF. New command XCORrelate.
- Bug fixes and documentation updates.

New Features of Version 3.2 — September, 1995

- A general locked translation function is implemented. A new command GLTF is introduced.
- The program can automatically carry out fine searches, with the slow rotation function, for the top peaks of the RF. A new command PKFIt is implemented. These fine search results are used by P1PC and GLTF commands.
- An option of calculating the locked RF by interpolating among ordinary slow RF values is implemented. A new command LOCInterpolate is introduced.
- The program will select both Ca and P atoms for packing check in the locked translation function.
- Modified the coordinate I/O routines to fully support the PDB format.
- Implemented a new orthogonalization convention, CXCAZ.
- Various bug fixes.
- Updated the documentation.

New Features of Version 3.1 — June, 1995

- The Patterson correlation refinement of ordinary cross RF peaks is now possible. The correlation that is maximized is different from that used in X-Plor. New commands COORdinate, GROUp, and P1PC.
- The Crowther fast rotation function is now fully implemented. It can be used to calculate locked as well as ordinary self or cross rotation functions.

- The unique rotational space for Eulerian searches can be assigned automatically for ordinary self and cross RF (a special feature of the command SLIMits). The program can also try to assign the limits for the locked RF.
- A new algorithm for the calculation of the slow RF is implemented, which will determine RCUTLO (command RCUToff) automatically and can lead to a three-fold speed-up. The implementation of RCUTHI (command RCUToff) has been changed.
- A new definition of the locked cross RF is used, which should simplify the definition of the unique rotational space.
- FFT routines have been incorporated to perform the transform from calculations done with the command TFUNction directly.
- The angles in the peak listing in the print file are now labeled. The three coordinate axes in the stereographic projection are labeled with the approapriate unit cell axis.
- Bug fixes.
- Updated the documentation.

Appendix B.

Example Inputs

Example input files to this program are provided together with the program distribution. This appendix shows examples for blocks of related inputs. The required portion of each command is given in upper case.

General Set-up
TITLe RF calculation
COMMent anything goes here
PRINt glrf.prt
EULEr zxz
POLAr xzk
ORTHogonalization axabz

Define Crystal A
ACELl 90 90 90 90 90 90
ASYMmetry p212121
AOBS-file fobs.dat
AFORmat 3i4, 2f8.2
ACUToff 1 1 0
APOWER 2
NSHEll 8
ORIGin true

Slow Self Rotation Function
SELF true
CROSs false
FAST false
CUTOff 1.5
BOXSize 3 3 3
GEVAluation 2
RADIus 20
RESOlution 10 3.5
SANGle polar
OANGle euler zxz
SLIMits 1 0 180 3
SLIMits 2 0 180 3
SLIMits 3 180 180 2
MAPFile srf.map

Fast Cross Rotation Function
SELF false
CROSs true
FAST true
RADIus 30
RESOlution 10 3.5
CUTOff 0.2
SANGle euler
OANGle euler zxz
SLIMits 270 270 0
MAPFile xrf.map

Peak Search and Fitting
PEAK 3 50
PKFIt 10 1.5

Contour Plots
CNTFile glrf.ps
CNTLevel 500 1000 50
SECTion 123

Non-crystallographic Symmetry
LOCSymmetry 1 0 0 2 vector
LOCSymmetry 1 1 1 3 vector
LOCExpand true

Real Space Rotation Function
RSRF rotation 20 30 40 search
RSRF a-center 0.1 0.2 0.3
RSRF b-center 0.4 0.6 0.5

Read in Old Map File
PEAK 3 30 glrf.map
CNTFile glrf.ps
CNTLevel 500 1000 50
SECTion 123

Fast Locked Cross RF
LOCSymmetry 1 0 0 2 vector
LOCSymmetry 0 1 0 2 vector
LOCExpand true
LOCOrient 10 20 30 euler
SELF false
CROSs true
FAST true
RADIus 30
RESOlution 10 4
CUTOff 0.2
SANGle euler
OANGle euler zyz
SLIMits 1 0 180 3
SLIMits 2 0 90 3
SLIMits 3 0 360 3

Patterson Correlation Refinement
COORdinate input model.pdb
COORdinate boverall 30
COORdinate output rigid.pdb
GROUp 1 1 60
GROUp 1 181 250
GROUp 2 61 180
P1PC 10 5 1.5

Locked Translation Function
COORdinate input model.pdb
COORdinate boverall 30
COORdinate output gltf.pdb
GLTF cutoff 1.5
GLTF direct false
GLTF overlap 10 3
GLTF packing false
GLTF radius 30
GLTF rfpeak 10
GLTF SLIMits 1 -30 30 1
GLTF SLIMits 2 -30 30 1
GLTF SLIMits 3 -30 30 1

Appendix C.

The Cross-Locked Rotation Function

In many cases a protein sample has been crystallized into two (or more) different space groups. The orientational relationship between a molecule in the first crystal form and one in the second can be determined by a cross-rotation function between the two crystal forms. Thus, we have the following relationship,

$$y_C = [\alpha_C][R_{BC}][\beta_B]y_B,$$

where the two crystal forms are called B and C.

Now a cross rotation function can be calculated between a search model and crystal forms B and C simultaneously. The two individual rotation functions are,

$$y_B = [\alpha_B][R_{AB}][\beta_A]y_A$$

$$y_C = [\alpha_C][R_{AC}][\beta_A]y_A$$

Noting the relationship between the B and the C crystals, we should have

$$\begin{aligned} y_C &= [\alpha_C][R_{BC}][\beta_B]y_B \\ &= [\alpha_C][R_{BC}][\beta_B][\alpha_B][R_{AB}][\beta_A]y_A \\ &= [\alpha_C][R_{BC}][R_{AB}][\beta_A]y_A \end{aligned}$$

Therefore,

$$[R_{AC}] = [R_{BC}][R_{AB}]$$

The cross-locked rotation function (or triple rotation function) can be defined as the product of the rotation function values corresponding to matrix $[R_{AB}]$ between A and B crystal and matrix $[R_{BC}][R_{AB}]$ between A and C crystal.

The above derivation can be easily generalized to cases where there are more than one rotational relationships between crystals B and C, and to cases where there are more than two crystal forms (quadruple rotation function ...).

Appendix D.

The Real Space Rotation Function

This function is defined as the product of the electron density for two copies of the same molecule. It is dependent on both the rotational and the translational relationship between the two copies (and in this sense the function is more than just a rotation function).

The relationship between the two copies of the molecule can be written as,

$$x_B = [C]x_A + d.$$

If S_A and S_B are the centers of the molecule in crystals A and B, respectively, then

$$S_B = [C]S_A + d.$$

The correlation between the two molecular copies should be a function of $[C], S_A$ and S_B,

$$\begin{aligned}
R([C], S_A, S_B) &= \int_{\substack{\Omega \\ (|x-S_A| \leq R)}} \rho(y)\rho(x)dx \\
&= \int_\Omega \sum_p F_p e^{-2\pi ipy} \sum_h F_h e^{-2\pi ihx} dx \\
&= \sum_p \sum_h F_p F_h \int_\Omega e^{-2\pi ip([C]x+d)} e^{-2\pi ihx} dx \\
&= \sum_p \sum_h F_p F_h e^{-2\pi ipd} \int_\Omega e^{-2\pi i(h+p[C])x} dx \\
&= \sum_p \sum_h F_p F_h e^{-2\pi ipd} G_{hp} e^{-2\pi i(h+p[C])S_A} \\
&= \sum_p F_p e^{-2\pi ip([C]S_A+d)} \sum_h F_h G_{hp} e^{-2\pi ihS_A} \\
&= \sum_p F_p e^{-2\pi ipS_B} \sum_h F_h G_{hp} e^{-2\pi ihS_A} \\
&= \sum_p F_p e^{-2\pi ipS_B} A_p e^{iB_p} \\
&= \sum_{\substack{p \\ (l \geq 0)}} \left(F_p A_p e^{i\phi_p} e^{-2\pi ipS_B} e^{iB_p} + F_p A_p e^{-i\phi_p} e^{2\pi ipS_B} e^{-iB_p} \right) \\
&= 2 \sum_{\substack{p \\ (l \geq 0)}} F_p A_p cos(\phi_p + B_p - 2\pi pS_B)
\end{aligned}$$

Appendix E.

The Location of a Local Symmetry Axis

Given the orientation of a symmetry axis and the translational elements along the axis, the location of the symmetry axis in the unit cell can be determined by correlating the electron density of the two molecules that are related by the symmetry axis.

The local symmetry axis can be written as

$$x' = [C]x + d_{\|} + d_{\perp},$$

where $d_{\|}$ is the translation element along the direction of the symmetry axis and $d_{\perp}$ the translation element in the plane perpendicular to the symmetry axis. If s is a point lying on the symmetry axis, it follows that

$$s' = s + d_{\|},$$

and

$$s = [C]s + d_{\perp}.$$

The translation function is defined as a function of s,

$$\begin{aligned}
T(s) &= \int_{\substack{\Omega \\ (|x-s|\le R)}} \rho(x)\rho(x')dx \\
&= \int_{\Omega} \sum_h F_h e^{-2\pi i h x} \sum_p F_p e^{-2\pi i p x'} dx \\
&= \sum_h \sum_p F_h F_p \int_{\Omega} e^{-2\pi i h x} e^{-2\pi i p([C]x + d_{\|} + d_{\perp})} dx \\
&= \sum_h \sum_p F_h F_p e^{-2\pi i p(d_{\|} + d_{\perp})} \int_{\Omega} e^{-2\pi i (h + p[C]) x} dx \\
&= \sum_h \sum_p F_h F_p e^{-2\pi i p(d_{\|} + d_{\perp})} G_{hp} e^{-2\pi i (h + p[C]) s} \\
&= \sum_h \sum_p F_h F_p G_{hp} e^{-2\pi i p d_{\|}} e^{-2\pi i p([C]s + d_{\perp})} e^{-2\pi i h s} \\
&= \sum_h \sum_p F_h F_p G_{hp} e^{-2\pi i p d_{\|}} e^{-2\pi i (h+p) s}
\end{aligned}$$

Appendix F.

Positions Related by Local Symmetry

Given the orientation of a symmetry axis and an input position belonging to one of the molecules related by the symmetry axis, other positions related to the input position by the local symmetry axis can be calculated.

The local symmetry axis can be written as

$$x' = [C]x + d.$$

If s is the input position, a translation function can be defined in terms of its symmetry-related position $s' = [C]s + d$,

$$\begin{aligned}
T(s') &= \int_{\substack{\Omega \\ (|x-s| \leq R)}} \rho(x)\rho(x')dx \\
&= \int_{\Omega} \sum_h F_h e^{-2\pi i h x} \sum_p F_p e^{-2\pi i p x'} dx \\
&= \sum_h \sum_p F_h F_p \int_{\Omega} e^{-2\pi i h x} e^{-2\pi i p([C]x+d)} dx \\
&= \sum_h \sum_p F_h F_p e^{-2\pi i p d} \int_{\Omega} e^{-2\pi i (h+p[C])x} dx \\
&= \sum_h \sum_p F_h F_p e^{-2\pi i p d} G_{hp} e^{-2\pi i (h+p[C])s} \\
&= \sum_h \sum_p F_h F_p G_{hp} e^{-2\pi i h s} e^{-2\pi i p([C]s+d)} \\
&= \sum_h \sum_p F_h F_p G_{hp} e^{-2\pi i h s} e^{-2\pi i p s'}
\end{aligned}$$

Appendix G.

Special Commands for Purdue Files

I. Reflection Data.

A special command has been set up in the program to read the reflection data from the FIT tapes.

AFIT-File	AFITFL*8, QAREJ3, IACOLN	[AFITFILE, T, 1]

Routines from the standard X200LIB library are used to perform I/O's on FIT tapes. This command and the AOBS command are mutually exclusive. Specifying AOBS will disable input from FIT file. QAREJ3 specifies whether reflections with bit 3 set should be rejected from the calculation. IACOLN specifies the column number in the FIT tape from which the Fobs values should be taken.

Similarly, we have the command to read the B crystal reflection data from a FIT tape -

BFIT-File	BFITFL*8, QBREJ3, IBCOLN	[BFITFILE, T, 1]

II. Plotting the Output from the GRFN Program.

A special command, GRFN, has been set up to read in the RHFILE output file from GRFN.

GRFN-input	GRFNFL*8, NREFO, NBIGO	[none]

GRFNFL defines the name of the file. NREFO and NBIGO, optional, specify the number of reflections and the number of large terms that were used in the search, respectively. The search parameters can be specified by using the corresponding commands from this (GLRF) program. Specifically, the search region has to be defined with the SLIMts command, and the search angle type has to be specified with the SANGle command. Because of the way the data is stored, the search intervals can not be changed. Once the map values are read in, the maximum will be scaled to 999 (note that self rotation function values may no longer be on 'absolute' scale). A peak search (command PEAK) will be carried out. A map file will be written if command MAPFile has been given. A stereographic projection (file PSPLOT) will also be created if SANGLE is polar. Contour plots can also be generated by using the commands CNTFile and CNTLevel.

Appendix H.

Handling of Logicals in Subroutine CHEKIN

	QSELF	QCROSS	Reflection Data
Self Search	T	F	A
Cross Search	F	T	AB
LOCAsu	T	F	none
	F	T	none
LOCFit	T	F	A
LOCRotate	T	F	none, or A
	F	T	none, or AB
MATRix	T	F	none, or A
	F	T	none, or AB
PRERot	F	T	none, or AB
TFUNction	T	F	A