HALL, C. R. & HIRSCH, P. B. (1965). *Proc. R. Soc. London Ser. A*, **286**, 158-177.
HOWIE, A. (1963). *Proc. R. Soc. London Ser. A*, **271**, 268-287.
HUMPHREYS, C. J. (1979). *Rep. Prog. Phys.* **42**, 1825-1887.
JAMES, R. W. (1948). *The Optical Principles of the Diffraction of X-rays*, p. 259. London: Bell.
MASLEN, V. W. & ROSSOUW, C. J. (1984). *Philos. Mag.* **49**, 735-742.
REZ, P., HUMPHREYS, C. J. & WHELAN, M. J. (1977). *Philos. Mag.* **35**, 81-96.
ROSSOUW, C. J. & MASLEN, V. W. (1984). *Philos. Mag.* **49**, 743-757.
SAMARA, G. A. & PEERCY, P. S. (1973). *Phys. Rev. B*, **7**, 1131-1148.
SMITH, B. T., BOYLE, J. M., DONGARRA, J. J., GARBOW, B. S., IKEBE, Y., KLEMA, V. C. & MOLER, C. B. (1976). *Matrix Eigensystem Routines—EISPACK Guide. Lecture Notes in Computer Science*, Vol. 6, edited by G. GOOS & J. HARTMANIS, pp. 19-33. Berlin: Springer-Verlag.
YOSHIOKA, H. (1957). *J. Phys. Soc. Jpn*, **12**, 618-628.

Acta Cryst. (1985). **A41**, 327-333

Program Construction for Macromolecule Atomic Model Refinement Based on the Fast Fourier Transform and Fast Differentiation Algorithms

BY V. YU. LUNIN AND A. G. URZHUMTSEV

Research Computer Center, USSR Academy of Sciences, 142292 *Pushchino, Moscow Region, USSR*

(*Received* 9 *November* 1983; *accepted* 13 *December* 1984)

Abstract

Structure refinement may be considered as a minimization of a function $R(\chi)$ of a large number of refineable parameters. A new type of function incorporating phase probability distribution is proposed. The minimization of the function utilizing gradient methods requires the computation of gradient ∇R, as well as the product of the gradient and the matrix of second derivatives with some direction. The algorithm of Kim, Nesterov & Cherkassky [*Dokl. Akad. Nauk SSSR* (1984), **275**, 1306-1309] adapted to macromolecular structure refinement takes about four times longer for the computation of these values compared to the computation of the value of the minimized function. The matrix of second derivatives is used without any approximation.

Introduction

The refinement of a structure implies that there is a model with parameters to be changed until they most closely fit X-ray scattering data, stereochemical restraints, energy minimum conditions *etc.* The refinement proper should be distinguished from the elaboration of its instrumental part, that is computer programs. And if for the former the most important are the researcher's experience and intuition, the latter puts more emphasis on the 'computer' problems such as efficiency of the algorithms, user's convenience *etc.* Different approaches to refinement of macromolecular structures have computational similarities, so that it becomes possible to solve most general problems of developing the corresponding programs.

The large number of refineable parameters is an essential feature of macromolecular structure refinement. This involves considerable computational difficulties, therefore routinely applicable algorithms need computation increasing linearly with the size of the refineable object. The Cooley-Tukey (1965) algorithm based on the fast Fourier transform and the fast differentiation algorithm developed by Kim, Nesterov & Cherkassky (1984) allow a general algorithm for model refinement whose computation per cycle increases almost linearly with the size of molecule. In § 2 we consider the algorithm constructed by Kim *et al.*, in which the n components of the gradient of a function $f(x_1, \dots, x_n)$ require much the same computation as the single function. Note that an algorithm of this type for some particular criterion used in refinement of atomic models was earlier proposed by Agarwal (1978) and later improved by Lifshitz (Agarwal, 1981). In § 3 we show how to develop similar algorithms for every criterion and refineable parameter. It should be emphasized that these rapid algorithms compute the accurate product of a full second-derivative matrix and a direction without any approximation. Application of the routine based on these algorithms will be considered elsewhere.

1. Problem of the atomic model refinement

1.1. *Atomic models*

In this paper we consider only the models where the distribution of electron density can be a sum of the contributions of individual atoms

$$\rho(\mathbf{r}) = \sum \rho_j(\mathbf{r}, \mathbf{q}_j). \qquad (1)$$

Here $\mathbf{q}_j = \{q_{j\alpha}\}$ are refineable parameters determining the contribution of an atom to the electron density and $\rho_j(\mathbf{r}, \mathbf{q}_j)$ are the known functions describing this contribution. For instance, it is common to determine the contribution of a single atom by four parameters: isotropic thermal motion parameter B_j and atomic coordinates $\mathbf{r}_j = (x_j, y_j, z_j)$; here $\mathbf{q}_j = (B_j, x_j, y_j, z_j)$. The function $\rho_j(\mathbf{r}, \mathbf{q})$ is usually described by a sum of Gaussians.

We write $\mathbf{q} = \{\mathbf{q}_j\}$ for the parameters of all the atoms contributing to $\rho(\mathbf{r}) = \rho(\mathbf{r}, \mathbf{q})$ and call them the atomic parameters.

Sometimes, to impose rigorous stereochemical restraints on the atomic parameters $\mathbf{q}$, one describes a model by generalized parameters $\chi = (\chi_1, \ldots, \chi_n)$. These are, for example, dihedral angles θ_j (Diamond, 1971) or parameters of some 'rigid' molecular fragments (Sussman, Holbrook, Church & Kim, 1977; Sussman, 1981). In this paper the generalized parameters are only used to restore all the atomic parameters so that we can apply (1) to calculate the electron density in the cell. The transform of the generalized parameters into the atomic ones is assumed to be done by functions

$$\mathbf{q}_j = \mathbf{q}_j(\chi). \tag{2}$$

1.2. *Refinement criteria*

Qualitative criteria may be expressed by different characteristics of a refined model such as generalized parameters χ, atomic parameters $\mathbf{q}$, electron density $\rho(\mathbf{r})$ and structure factors

$$\mathbf{f}(\mathbf{s}) = f_s \exp(i\varphi_s) = f_s^R + i_s^I = \int_V \rho(\mathbf{r}) \exp[2\pi i(\mathbf{s},\mathbf{r})] \mathrm{d}V_r. \tag{3}$$

A few examples will illustrate most usable criteria of different types.

Reciprocal-space refinement (*Agarwal*, 1978).

$$R_X[\mathbf{f}(\chi)] = \sum_s w_s[f_s(\chi) - f_s^{\mathrm{obs}}]^2 \Rightarrow \min, \tag{4}$$

where f_s^{obs} are the experimental structure-factor modules and w_s are the preset weights.

Real-space refinement (*Diamond*, 1971).

$$R_\rho[\rho(\chi)] = \sum_{\mathbf{r}} [\rho(\mathbf{r}, \chi) - \rho^{\mathrm{obs}}(\mathbf{r})]^2 \Rightarrow \min, \tag{5}$$

where $\rho^{\mathrm{obs}}(\mathbf{r})$ is an electron density distribution that is treated as 'observed'.

Energy refinement (*Warme & Scheraga*, 1974).

$$R_E[\mathbf{q}(\chi)] \Rightarrow \min, \tag{6}$$

where R_E can be interpreted either as the conformation energy or as a penalty function for the violated stereochemical standards. Generally, one uses summarized criteria, such as, for example,

$$\alpha R_X[\mathbf{f}(\chi)] + \beta R_E[\mathbf{q}(\chi)] \Rightarrow \min \tag{7}$$

(Jack & Levitt, 1978; Hendrickson & Konnert, 1980).

Usually, a criterion requires an amount of computation depending upon which characteristics of the model it is expressed by. The most difficult is to calculate a criterion expressed by structure factors $\mathbf{f}(\mathbf{s})$, because these need themselves more computation than the values of the electron density distribution $\rho(\mathbf{r})$. It is this criterion that we shall consider in § 3. To adapt the algorithms to other criteria, one only needs to exclude several steps.

1.3. *Inclusion of the phase information into the criterion*

The use of the information on structure-factor phases obtained in some way can enhance the efficiency of refinement. In the early stages of protein X-ray structure analysis the phase information is essentially based on additional experiments with heavy-atom derivatives. Inclusion of the information in the refinement means that additional experimental data are used. In the next stages the phase information may accumulate some supplementary knowledge about the structure (atomicity, electron density non-negativity, non-crystallographic symmetry *etc.*).

Normally, we do not know a single value of the phase but rather are restricted to a phase probability distribution $P_\varphi(\varphi, \mathbf{s})$ for the values of the phase φ_s. This may be represented in the general form

$$P_\varphi(\varphi, \mathbf{s}) \simeq \exp\{A_s \cos\varphi + B_s \sin\varphi + C_s \cos 2\varphi + D_s \sin 2\varphi\} \tag{8}$$

(Hendrickson & Lattman, 1970), here A_s, B_s, C_s and D_s are constants determining this distribution. Assuming the experimental estimate of the intensities $I_s = f_s^2$ gives a normally distributed error with a zero mean and dispersion $\sigma_I^2(\mathbf{s})$, we obtain a probability distribution for f_s with the density

$$P_f(f, \mathbf{s}) \simeq \exp\{-[f_s^2 - (f_s^{\mathrm{obs}})^2]^2 / 2\sigma_{I(s)}^2\}.$$

If now f_s and φ_s are supposed to be mutually independent, the joint probability distribution will be

$$\prod_s P_f(f_s, \mathbf{s}) P_\varphi(\varphi_s, \mathbf{s}) \simeq \exp\Big\{-\sum_s [1/2\sigma_I^2(\mathbf{s})][f_s^2 - (f_s^{\mathrm{obs}})^2]^2 + [A_s \cos\varphi_s + B_s \sin\varphi_s + C_s \cos 2\varphi_s + D_s \sin 2\varphi_s]\Big\}, \tag{9}$$

i.e. the most probable is the model where the following value is minimal

$$\begin{aligned} R_X[\mathbf{f}(\chi)] &= \sum_{\mathbf{s}} \{[1/2\sigma_I^2(\mathbf{s})][f_{\mathbf{s}}^2(\chi)-(f_{\mathbf{s}}^{\text{obs}})^2]^2 \\ &\quad -[A_{\mathbf{s}}\cos\varphi_{\mathbf{s}}(\chi)+B_{\mathbf{s}}\sin\varphi_{\mathbf{s}}(\chi) \\ &\quad +C_{\mathbf{s}}\cos 2\varphi_{\mathbf{s}}(\chi)+D_{\mathbf{s}}\sin 2\varphi_{\mathbf{s}}(\chi)]\}. \quad (10)\end{aligned}$$

The use of this function in the course of refinement is advantageous in many respects. Firstly, we take into consideration the additional phase information. Secondly, we can change $\varphi_{\mathbf{s}}(\chi)$ based on the reliability of this information. In the course of refinement we may use, in particular, structure factors with undetermined phases by setting $A_{\mathbf{s}}=B_{\mathbf{s}}=C_{\mathbf{s}}=D_{\mathbf{s}}=0$. Thirdly, using (10), we can take account of the multimodality of the probability distributions for the phase $\varphi_{\mathbf{s}}$, which is impossible with the criterion

$$\sum_{\mathbf{s}} w_{\mathbf{s}}(\varphi_{\mathbf{s}}-\varphi_{\mathbf{s}}^{\text{obs}})^2 \quad (11)$$

(Rees & Lewis, 1983) where the phases are associated with a single chosen value of $\varphi_{\mathbf{s}}^{\text{obs}}$. As will be seen from further discussion, the computation by (10) is not more complicated than with (4).

2. Fast differentiation algorithm

2.1. *Gradient computation*

In minimizing a complex function of a large number of variables, the gradient seems much more difficult to compute than the function $f(\mathbf{x})$ itself. For instance, in the well known method of numerical differentiation

$$\begin{aligned} \partial f/\partial x_j &\simeq [f(\mathbf{x}+\mathbf{h}_j)-f(\mathbf{x})]/h, \\ \mathbf{h}_j &= (0,\dots,0,h_j,0,\dots,0), \quad (12)\end{aligned}$$

the gradient requires $2n$ computations of the function. On thorough examination, one can find simpler ways of computing this (see, for example, Agarwal, 1978). But this approach based mainly on intuition and 'luck' is not the best. Thus, Agarwal's algorithm has been greatly improved by Lifshitz (Agarwal, 1981).

Kim *et al.* (1984) have recently shown that for any $f(\mathbf{x})$ the gradient can be computed in much the same time as the function. In particular, two points here are noteworthy. Firstly, this algorithm deals only with the exact computation of ∇f, without any approximation. Secondly, it gives detailed indications of how to obtain 'fast' routines for the computation of ∇f, making this process almost 'self-running'.

The algorithm of Kim *et al.* computes the gradient in not more time than $CT(f)$, where $T(f)$ is the time for calculation of $f(\mathbf{x})$ and C is a rather small constant (of order 2–4) *independent of n*. We do not specify the value of C, mention only that if this algorithm is applied to refinement of macromolecular structures the computation of all the values of $f(\mathbf{x})$, $\partial f/\partial\mathbf{e}$, ∇f and $\nabla^2 f\mathbf{e}$ will take time of the order of $4T(f)$, *i.e.* the 'mean time' is almost equal to that needed for the calculation of $f(\mathbf{x})$.

A convenient form of the fast differentiation algorithm will be shown below.

The main idea of the fast gradient computation for a function $f(\mathbf{x})$ may be formulated as follows. We calculate $f(\mathbf{x})$ as a chain of transforms of the variables

$$\mathbf{x}\rightarrow\mathbf{y}^1(\mathbf{x})\rightarrow\mathbf{y}^2(\mathbf{y}^1)\rightarrow\dots\rightarrow\mathbf{y}^N(\mathbf{y}^{N-1})\rightarrow F(\mathbf{y}^N)\equiv f(\mathbf{x}) \quad (13)$$

so that the gradient is computed as a chain of consecutive computations of the gradient of a complex function $F\{\mathbf{y}^N[\mathbf{y}^{N-1}(\dots\{\mathbf{y}^2[\mathbf{y}^1(\mathbf{x})]\}\dots)]\}$ with respect to the variables $\mathbf{y}^N,\mathbf{y}^{N-1},\dots,\mathbf{x}$:

$$\nabla_{\mathbf{y}^N}F\rightarrow\nabla_{\mathbf{y}^{N-1}}F\rightarrow\dots\rightarrow\nabla_{\mathbf{y}^1}F\rightarrow\nabla_{\mathbf{x}}F. \quad (14)$$

As will be shown later, every transform $\nabla_{\mathbf{y}^k}F\rightarrow\nabla_{\mathbf{y}^{k-1}}F$ takes as much time as the transform $\mathbf{y}^{k-1}\rightarrow\mathbf{y}^k$, so that the entire chain (14) may be computed in the same time as the chain (13). Let us detail this.

Consider first a case where the function $f(\mathbf{x})$ is calculated as a chain of two superpositions:

$$\mathbf{x}\rightarrow\mathbf{y}(\mathbf{x})\rightarrow F(\mathbf{y})\equiv f(\mathbf{x}), \quad (15)$$

such that $f(\mathbf{x})=F[\mathbf{y}(\mathbf{x})]$ where $\mathbf{x}=(x_1,\dots,x_n)$, $\mathbf{y}(\mathbf{x})=[y_1(\mathbf{x}),\dots,y_m(\mathbf{x})]$ and $y_1(\mathbf{x}),\dots,y_m(\mathbf{x})$ are preset functions. Then

$$\partial f/\partial x_j=\sum_{i=1}^{n}(\partial F/\partial y_i)(\partial y_i/\partial x_j) \quad (16)$$

so that the transform $\nabla_{\mathbf{y}}F\rightarrow\nabla_{\mathbf{x}}F=\nabla_{\mathbf{x}}f$ may be given by

$$\nabla_{\mathbf{x}}f=(\partial\mathbf{y}/\partial\mathbf{x})^T\nabla_{\mathbf{y}}F. \quad (17)$$

Here $\nabla_{\mathbf{y}}F=(\partial F/\partial y_1,\dots,\partial F/\partial y_m)^T$ is the gradient of F with respect to the variables $\mathbf{y}=(y_1,\dots,y_m)$, $\partial\mathbf{y}/\partial\mathbf{x}$ is the Jacobian matrix

$$\frac{\partial\mathbf{y}}{\partial\mathbf{x}}=\begin{pmatrix}\partial y_1/\partial x_1 & \cdots & \partial y_1/\partial x_n\\ \vdots & \cdots & \vdots\\ \partial y_m/\partial x_1 & \cdots & \partial y_m/\partial x_n\end{pmatrix} \quad (18)$$

and T means matrix transposition.

Analogously, in the case of a greater number of superpositions [see (13)], the transform $\nabla_{\mathbf{y}^k}F\rightarrow\nabla_{\mathbf{y}^{k-1}}F$ in the chain (14) can be obtained by

$$\nabla_{\mathbf{y}^{k-1}}F=(\partial\mathbf{y}^k/\partial\mathbf{y}^{k-1})^T\nabla_{\mathbf{y}^k}F, \quad (19)$$

where $\nabla_{\mathbf{y}^k}F=\nabla_{\mathbf{y}^k}F(\mathbf{y}^N\{\mathbf{y}^{N-1}[\dots\mathbf{y}^{k+1}(\mathbf{y}^k)\dots]\})$.

Thus, the computation of an arbitrary function can be reduced to a sequence of elementary operations (addition, subtraction, multiplication, division) such that

$$\begin{aligned} r_1 &= x_1, \\ &\vdots \\ r_n &= x_n, \\ r_{n+1} &= f_1(r_1, \ldots, r_n), \\ r_{n+2} &= f_2(r_1, \ldots, r_n, r_{n+1}), \\ &\vdots \\ r_{n+N} &= f_N(r_1, \ldots, r_{n+N-1}), \end{aligned} \tag{20}$$

which gives $f(\mathbf{x}) = r_{n+N}$. Here $r_1, \ldots, r_{n+N-1}$ are the results of intermediate computations and $f_j(r_1, \ldots, r_{n+j-1})$ are the corresponding elementary operations, so that every f_j depends on not more than two arguments. The sequence (20) can be written as the chain (13), if we introduce

$$\begin{aligned} \mathbf{y}^1(\mathbf{x}) &= [y_1^1(\mathbf{x}), \ldots, y_{n+1}^1(\mathbf{x})] \\ &\equiv [x_1, \ldots, x_n, f_1(x_1, \ldots, x_n)] \\ &\vdots \quad \ldots \\ \mathbf{y}^k(\mathbf{y}^{k-1}) &= [y_1^k(\mathbf{y}^{k-1}), \ldots, y_{n+k}^k(\mathbf{y}^{k-1})] \\ &\equiv [y_1^{k-1}, \ldots, y_{n+k-1}^{k-1}, f_k(y_1^{k-1}, \ldots, y_{n+k-1}^{k-1})]. \end{aligned} \tag{21}$$

The Jacobian matrix corresponding to the transform $\nabla_{\mathbf{y}^k} F \rightarrow \nabla_{\mathbf{y}^{k-1}} F$ in the chain (14) then assumes the form

$$\frac{\partial \mathbf{y}^k}{\partial \mathbf{y}^{k-1}} = \begin{pmatrix} 1 & & & 0 \\ 0 & \ddots & & \\ & & & 1 \\ \alpha_1^k & \alpha_2^k & \ldots & \alpha_{n+k-1}^k \end{pmatrix}, \tag{22}$$

where

$$\alpha_j^k = \partial f_k(r_1, \ldots, r_{n+k-1})/\partial r_j;$$

$k = 1, \ldots, N$ and $j = 1, \ldots, n+k-1$, so that there are only two non-zero elements in the last row of matrix (22). To compute the product of the matrix $(\partial \mathbf{y}^k/\partial \mathbf{y}^{k-1})^T$ and the vector $\nabla_{\mathbf{y}^k} F$, we thus need two multiplications and two additions, because all the $\nabla_{\mathbf{y}^k} F$ components except two will pass unchanged to $\nabla_{\mathbf{y}^{k-1}} F$. Hence, the computation of the entire chain of N transforms in (14) can be evaluated by $2N$ multiplications, whereas, as can be seen from (20), the computation of $f(\mathbf{x})$ requires N operations, only half as much as the gradient computation.

2.2. *Computation of* $\nabla^2 f\mathbf{e}$ *and a derivative in direction*

Several methods of minimization of $f(\mathbf{x})$ require computation of $\nabla^2 f\mathbf{e}$, where $\nabla^2 f$ is the matrix of second derivatives and $\mathbf{e}$ is the vector. This problem appears to be much more complicated than the gradient computation. In a number of protein structure refinements (Agarwal, 1981; Dodson, 1981; Hendrickson & Konnert, 1980), the vector $\nabla^2 f\mathbf{e}$ has been computed by approximating the matrix $\nabla^2 f$. It should however be emphasized that

$$\nabla^2 f\mathbf{e} = \nabla\left(\sum_{i=1}^{n} e_i(\partial f/\partial x_i)\right), \tag{23}$$

i.e. the product $\nabla^2 f$ is just the gradient of an auxiliary function $\sum e_i(\partial f/\partial x_i)$, a derivative of $f(\mathbf{x})$ in the $\mathbf{e}$ direction. Therefore, as shown above, the vector $\nabla^2 f\mathbf{e}$ requires as much computation as

$$\partial f/\partial \mathbf{e} = \sum_{i=1}^{n} e_i(\partial f/\partial x_i).$$

Now we show that the function $f(\mathbf{x})$ has an equivalent computational cost.

Note that

$$\begin{aligned} \sum_{i=1}^{n} e_i(\partial f[\mathbf{y}(\mathbf{x})]/\partial x_i) &= \sum_{i=1}^{n} e_i \sum_{j=1}^{m} (\partial f/\partial y_j)(\partial y_j/\partial x_i) \\ &= \sum_{j=1}^{m}\left(\sum_{i=1}^{n} e_i(\partial y_j/\partial x_i)\right)\partial f/\partial y_j, \end{aligned} \tag{24}$$

i.e. the derivative of a complex function $f(\mathbf{x}) = F[\mathbf{y}(\mathbf{x})]$ in the $\mathbf{e}$ direction can be computed as a derivative of the function $F(\mathbf{y})$ in a new $\mathbf{e}'$ direction given by

$$\mathbf{e}' = (\partial \mathbf{y}/\partial \mathbf{x})\mathbf{e}, \tag{25}$$

where $\partial \mathbf{y}/\partial \mathbf{x}$ is the Jacobian matrix (18). This implies that if the algorithm for the computation of $f(\mathbf{x})$ is represented by the chain (13), the derivative in the $\mathbf{e}$ direction can be obtained by differentiating a function $F(\mathbf{y}^N)$ in a new $\mathbf{e}^N$ direction resulting from the chain of transforms

$$\mathbf{e}^1 \rightarrow \mathbf{e}^2 \rightarrow \ldots \rightarrow \mathbf{e}^N, \tag{26}$$

where

$$\mathbf{e}^k = (\partial \mathbf{y}^k/\partial \mathbf{y}^{k-1})\mathbf{e}^{k-1}. \tag{27}$$

Thus, if the computational algorithm is expanded in elementary operations by (20), the matrices $\partial \mathbf{y}^k/\partial \mathbf{y}^{k-1}$ will take the form (22) with two non-zero elements in the last row. Hence, in this application the transform (27) will require two additions and two multiplications so that the computational cost of the chain (26) is proportional to that of $f(\mathbf{x})$ with a small proportionality constant.

Note also that sometimes it is more convenient that the vector $\nabla^2 f\mathbf{e}$ should be defined as the derivative of a vector function $\nabla f(\mathbf{x})$ in the $\mathbf{e}$ direction:

$$\nabla^2 f\mathbf{e} = \sum_{i=1}^{n} e_i(\partial \nabla f/\partial x_i). \tag{28}$$

3. Application of the differentiation algorithm to refinement of macromolecular structures

3.1. *Fast computation of the function $R(\chi)$*

We shall here apply the results obtained in § 2 to the minimization of criteria of the type

$$R(\mathbf{f})=\sum_{\mathbf{s}\in S} a(f_s^R, f_s^I; \mathbf{s}), \tag{29}$$

where S is a given reflexion set; $a(u, v; \mathbf{s})$ are preset functions and f_s^R, f_s^I are the real and imaginary parts of structure factors (3). To be more precise, we shall show how to compute ∇R and $\nabla^2 R\boldsymbol{\omega}$ in much the same time as $R(\mathbf{f})$.

The Hermitian symmetry of structure factors of the real function $\rho(\mathbf{r})$ is

$$\mathbf{f}(-\mathbf{s})=[\mathbf{f}(\mathbf{s})]^* \tag{30}$$

(* is a complex conjugation), which implies that the criteria $a(u, v; \mathbf{s})$ should also be symmetric so that

$$a(u,-v;\mathbf{s})=a(u,v;\mathbf{s}). \tag{31}$$

This involves the corresponding symmetry of the functions

$$\begin{aligned} &a_1(u,v;\mathbf{s})=\partial a(u,v;\mathbf{s})/\partial u \\ \text{and}\quad &a_2(u,v;\mathbf{s})=\partial a(u,v;\mathbf{s})/\partial v \end{aligned} \tag{32}$$

such that

$$\begin{aligned} a_1(u,-v;-\mathbf{s})&=a_1(u,v;\mathbf{s}),\\ a_2(u,-v;-\mathbf{s})&=-a_2(u,v;\mathbf{s}). \end{aligned} \tag{33}$$

We also assume that the set of reflexions S summarized in (29) is symmetric about the origin.

As has already been shown, the construction of the fast algorithm for the gradient computation may be reduced to the fast computation of $R[\mathbf{f}(\chi)]$. The necessary structure factors may be calculated in two ways. The first is to calculate $\mathbf{f}(\mathbf{s})$ by formulae derived from (3) with analytical integration (see, for example, Hendrickson & Konnert, 1980). The second requires that only the values of the function $\rho(\mathbf{r})$ at points of the U grid of the unit cell should be calculated analytically. The integral computation is then replaced by the computation of an appropriate integral sum:

$$\mathbf{f}(\mathbf{s})\simeq(V/n_xn_yn_z)\sum_{\mathbf{r}\in U}\rho(\mathbf{r})\exp[2\pi i(\mathbf{s},\mathbf{r})], \tag{34}$$

where the fast Fourier transform algorithm can be used (Ten Eyck, 1973, 1977). Here n_x, n_y and n_z are the numbers of U-grid divisions. Since the last equality is approximated, formally speaking, different criteria can be obtained from (29) according to how we calculate $\mathbf{f}(\mathbf{s})$. But with appropriately chosen grid spacing and atomic radii, the difference between the values calculated by (3) and (34) can be remarkably reduced compared with the measurement error of f_s^{obs}. So, in practice we have no grounds to prefer the first way of computing $\mathbf{f}(\mathbf{s})$ to the second. At the same time, the second method of structure factor calculation leads, for large molecules, to considerable savings in computational time (Ten Eyck, 1977).

3.2. *Gradient computation*

In this section we shall show how the general fast differentiation algorithm described in § 2.1 may be realized to calculate a gradient $\nabla_\chi R(\chi)$. As follows from 3.1, the function R is computed as a chain of superpositions:

$$\chi$$

generalized parameters;

$$\mathbf{q}=\mathbf{q}(\chi) \tag{35}$$

parameters of the atoms contributing to the electron density of the unit;

$$\rho_{\mathbf{r}}=\rho(\mathbf{r},\mathbf{q})=\sum_j \rho_j(\mathbf{r},\mathbf{q}_j) \tag{36}$$

electron densities at grid points $\mathbf{r}\in U$;

$$f_s^R+if_s^I=C\sum_{\mathbf{r}\in U}\rho_{\mathbf{r}}\exp[2\pi i(\mathbf{s},\mathbf{r})] \tag{37}$$

structure factors, $\mathbf{s}\in S$;

$$R=\sum_{\mathbf{s}\in S}a(f_s^R,f_s^I,\mathbf{s}). \tag{38}$$

According to the general scheme, we begin with the computation of a gradient $(\mathbf{F}^R,\mathbf{F}^I)=\{F_s^R, F_s^I\}_{\mathbf{s}}$ of R with respect to f_s^R, f_s^I:

$$\begin{aligned} F_s^R&=\partial R/\partial f_s^R=a_1(f_s^R,f_s^I,\mathbf{s}),\\ F_s^I&=\partial R/\partial f_s^I=a_2(f_s^R,f_s^I;\mathbf{s}), \end{aligned} \tag{39}$$

where a_1 and a_2 are defined by (32).

Then, applying (37), we may compute the components $P_{\mathbf{r}}$ of the gradient of R with respect to $\rho_{\mathbf{r}}$:

$$\begin{aligned} P_{\mathbf{r}}=\partial R/\partial\rho_{\mathbf{r}}&=\sum_{\mathbf{s}\in S}\{F_s^R[\partial f_s^R(\boldsymbol{\rho})/\partial\rho_{\mathbf{r}}]+F_s^I[\partial f_s^I(\boldsymbol{\rho})/\partial\rho_{\mathbf{r}}]\}\\ &=C\sum_{\mathbf{s}\in S}[F_s^R\cos 2\pi(\mathbf{s},\mathbf{r})+F_s^I\sin 2\pi(\mathbf{s},\mathbf{r})]\\ &=C\sum_{\mathbf{s}\in S}(F_s^R+iF_s^I)\exp[-2\pi i(\mathbf{s},\mathbf{r})], \end{aligned} \tag{40}$$

where S symmetry and equations $F_{-s}^R=F_s^R$, $F_{-s}^I=-F_s^I$ following from (33) have been used to derive the last equation. As in (37), the algorithm of the fast Fourier transform can be applied to calculate $P_{\mathbf{r}}$.

The transform into a gradient $\mathbf{Q}=\{Q_{j\alpha}\}$ with respect to $\mathbf{q}_j$ gives by (36) the following equality:

$$\begin{aligned} Q_{j\alpha}=\partial R/\partial q_{j\alpha}&=\sum_{\mathbf{r}\in U}P_{\mathbf{r}}(\partial\rho_{\mathbf{r}}/\partial q_{j\alpha})\\ &=\sum_{\mathbf{r}\in U}P_{\mathbf{r}}[\partial\rho_j(\mathbf{r},\mathbf{q}_j)/\partial q_{j\alpha}]. \end{aligned} \tag{41}$$

Note that in the last equality, as in (36), the summation is only carried out over points $\mathbf{r}\in U$, where

$\rho_j(\mathbf{r}, \mathbf{q}_j) \neq 0$, *i.e.* the amount of computation is proportional to the number of atoms.

Finally, we can obtain a gradient $\mathbf{X} = \{X_{k\beta}\}$ using the generalized parameters

$$X_{k\beta} = \partial R / \partial \chi_{k\beta} = \sum_{j,\alpha} Q_{j\alpha}[\partial q_{j\alpha}(\boldsymbol{\chi})/\partial \chi_{k\beta}]. \tag{42}$$

We do not specify the last transform, because the dependences $\mathbf{q}(\boldsymbol{\chi})$ are not detailed. We mention only that, as is shown in § 2.1, the last transform requires computation proportional to that of the transform (35). It is also worthwhile noting that the crystallographic symmetry allows more rapid transforms (37) and (40).

If the criterion R assumes the form (4), the computational procedure is identical to Agarwal's algorithm (Agarwal, 1978, 1981). In this connection we should make two essential remarks. Firstly, in contrast to Diamond (1971) and Agarwal (1978), we have no need to divide artificially the refineable atomic parameters into different groups. Secondly, the transforms (35)-(42) yield the accurate value of the gradient of R where $\mathbf{f}(\mathbf{s}, \mathbf{q})$ are derived from (34). Some assumptions have been made to choose the function, but once $R(\boldsymbol{\chi})$ is selected, we have no need for further assumptions to compute the gradient. It follows in particular that the U grid and the atomic radii are the same in (41) and (36), which contradicts Agarwal's considerations.

3.3. *Calculation of a product* $\nabla^2_{\chi\chi} R \boldsymbol{\omega}$.

Let $\boldsymbol{\omega} = \{\omega_{k\beta}\}$ be the prescribed direction in space of the generalized parameters $\boldsymbol{\chi} = \{\chi_{k\beta}\}$. The series of transforms obtained

$$\begin{aligned}&\{\chi_{k\beta}\} \to \{q_{j\alpha}\} \to \{\rho_{\mathbf{r}}\} \to \{f^R_{\mathbf{s}}, f^I_{\mathbf{s}}\}\\ &\quad \to \{F^R_{\mathbf{s}}, F^I_{\mathbf{s}}\} \to \{P_{\mathbf{r}}\} \to \{Q_{j\alpha}\} \to \{X_{k\beta}\}\end{aligned} \tag{43}$$

may be considered as the chain of superpositions (13) in the computation of a set of functions $X_{k\beta}(\boldsymbol{\chi}) = \partial R(\boldsymbol{\chi})/\partial \chi_{k\beta}$. As the values

$$\begin{aligned}\Omega_{k\beta} &= \sum_{l,\delta} \omega_{l\delta}[\partial^2 R(\boldsymbol{\chi})/\partial \chi_{k\beta}\, \partial \chi_{l\delta}]\\ &= \sum_{l,\delta} \omega_{l\delta}[\partial X_{k\beta}(\boldsymbol{\chi})/\partial \chi_{l\delta}]\end{aligned} \tag{44}$$

are the derivatives of the functions $X_{k\beta}(\boldsymbol{\chi})$ in the $\boldsymbol{\omega}$ direction, then, according to § 2.2, we have to construct the chain of directions (26) in the new variables on the basis of (43).

We start from the transform in the direction $\mathbf{l} = \{l_{j\alpha}\}$ for the variables $\mathbf{q} = \{q_{j\alpha}\}$:

$$l_{j\alpha} = \sum_{k,\beta} \omega_{k\beta}[\partial q_{j\alpha}(\boldsymbol{\chi})/\partial \chi_{k\beta}]. \tag{45}$$

Further, we construct the direction $\boldsymbol{\tau} = \{\tau_{\mathbf{r}}\}$ in the variables $\boldsymbol{\rho} = \{\rho_{\mathbf{r}}\}$:

$$\tau_{\mathbf{r}} = \sum_{j,\alpha} l_{j\alpha}[\partial \rho_{\mathbf{r}}(\mathbf{q})/\partial q_{j\alpha}] = \sum_j \partial \rho_j(\mathbf{r}, \mathbf{q}_j)/\partial \mathbf{l}_j, \tag{46}$$

i.e. the function $\{\tau_{\mathbf{r}}\}_{\mathbf{r} \in U}$ is a 'modified' electron density.

The transform into the direction $(\mathbf{g}^R, \mathbf{g}^I) = \{g^R_{\mathbf{s}}, g^I_{\mathbf{s}}\}$ in the variables $(\mathbf{f}^R, \mathbf{f}^I) = \{f^R_{\mathbf{s}}, f^I_{\mathbf{s}}\}$ is given by

$$g^R_{\mathbf{s}} = \sum_{\mathbf{r}} \tau_{\mathbf{r}}[\partial f^R_{\mathbf{s}}(\boldsymbol{\rho})/\partial \rho_{\mathbf{r}}] = C \sum_{\mathbf{r}} \tau_{\mathbf{r}} \cos 2\pi(\mathbf{s}, \mathbf{r}),$$

$$g^I_{\mathbf{s}} = \sum_{\mathbf{r}} \tau_{\mathbf{r}}[\partial f^I_{\mathbf{s}}(\boldsymbol{\rho})/\partial \rho_{\mathbf{r}}] = C \sum_{\mathbf{r}} \tau_{\mathbf{r}} \sin 2\pi(\mathbf{s}, \mathbf{r}),$$

so that

$$g^R_{\mathbf{s}} + i g^I_{\mathbf{s}} = C \sum_{\mathbf{r} \in U} \tau_{\mathbf{r}} \exp[2\pi i(\mathbf{s}, \mathbf{r})] \tag{47}$$

are the structure factors of the modified density $\{\tau_{\mathbf{r}}\}_{\mathbf{r}}$.

Now we should pass to the direction $(\mathbf{G}^R, \mathbf{G}^I) = \{G^R_{\mathbf{s}}, G^I_{\mathbf{s}}\}$ in the variables $(\mathbf{F}^R, \mathbf{F}^I) = \{F^R_{\mathbf{s}}, F^I_{\mathbf{s}}\}$:

$$\begin{aligned}G^R_{\mathbf{s}} &= \sum_{\mathbf{t} \in S} \{g^R_{\mathbf{t}}[\partial F^R_{\mathbf{t}}(\mathbf{f}^R, \mathbf{f}^I)/\partial f^R_{\mathbf{s}}]\\ &\quad + g^I_{\mathbf{t}}[\partial F^R_{\mathbf{t}}(\mathbf{f}^R, \mathbf{f}^I)/\partial f^I_{\mathbf{s}}]\}\\ &= g^R_{\mathbf{s}} a_{11}(f^R_{\mathbf{s}}, f^I_{\mathbf{s}}; \mathbf{s}) + g^I_{\mathbf{s}} a_{12}(f^R_{\mathbf{s}}, f^I_{\mathbf{s}}; \mathbf{s}),\\ G^I_{\mathbf{s}} &= \sum_{\mathbf{t} \in S} \{g^R_{\mathbf{t}}[\partial F^I_{\mathbf{t}}(\mathbf{f}^R, \mathbf{f}^I)/\partial f^R_{\mathbf{s}}]\\ &\quad + g^I_{\mathbf{t}}[\partial F^I_{\mathbf{t}}(\mathbf{f}^R, \mathbf{f}^I)/\partial f^I_{\mathbf{s}}]\}\\ &= g^R_{\mathbf{s}} a_{12}(f^R_{\mathbf{s}}, f^I_{\mathbf{s}}; \mathbf{s}) + g^I_{\mathbf{s}} a_{22}(f^R_{\mathbf{s}}, f^I_{\mathbf{s}}; \mathbf{s}),\end{aligned} \tag{48}$$

where equations (39) are used and

$$a_{11}(u, v; \mathbf{s}) = \partial^2 a(u, v; \mathbf{s})/\partial u^2,$$

$$a_{12}(u, v; \mathbf{s}) = \partial^2 a(u, v; \mathbf{s})/\partial u\, \partial v,$$

$$a_{22}(u, v; \mathbf{s}) = \partial^2 a(u, v; \mathbf{s})/\partial v^2.$$

Note that on account of the general theory of § 2.2, the transform

$$\mathbf{f}^R, \mathbf{f}^I, \mathbf{g}^R, \mathbf{g}^I \to \mathbf{F}^R, \mathbf{F}^I, \mathbf{G}^R, \mathbf{G}^I$$

requires as much computation as that of $(\mathbf{f}^R, \mathbf{f}^I)$ into $R(\mathbf{f}^R, \mathbf{f}^I)$ even for an arbitrary function R, which is not necessarily of the form (38).

Applying (40), we obtain the direction $\mathbf{T} = \{T_{\mathbf{r}}\}$ in the variables $\mathbf{P} = \{P_{\mathbf{r}}\}$ expressed as

$$\begin{aligned}T_{\mathbf{r}} &= \sum_{\mathbf{s}} \{G^R_{\mathbf{s}}[\partial P_{\mathbf{r}}(\mathbf{F}^R, \mathbf{F}^I)/\partial F^R_{\mathbf{s}}]\\ &\quad + G^I_{\mathbf{s}}[\partial P_{\mathbf{r}}(\mathbf{F}^R, \mathbf{F}^I)/\partial F^I_{\mathbf{s}}]\}\\ &= C \sum_{\mathbf{s}} (G^R_{\mathbf{s}} + i G^I_{\mathbf{s}}) \exp[-2\pi i(\mathbf{s}, \mathbf{r})].\end{aligned} \tag{49}$$

To pass from the direction $(\mathbf{T}, \mathbf{l})$ in the variables $(\mathbf{P}, \mathbf{q})$ to the direction $\mathbf{L}$ in the variables $\mathbf{Q}$, we should take into consideration that in (41) the values $Q_{j\alpha}$ depend not only on the variables $\{P_{\mathbf{r}}\}$ but also on the variables $\mathbf{q}_j$, so that this transform may be given by

$$\begin{aligned}L_{j\alpha} &= \sum_{\mathbf{r}} T_{\mathbf{r}}[\partial Q_{j\alpha}(\mathbf{P}, \mathbf{q})/\partial P_{\mathbf{r}}]\\ &\quad + \sum_{k,\beta} l_{k\beta}[\partial Q_{j\alpha}(\mathbf{P}, \mathbf{q})/\partial q_{k\beta}]\\ &= \sum_{\mathbf{r}} \{T_{\mathbf{r}}[\partial \rho_j(\mathbf{r}, \mathbf{q}_j)/\partial q_{j\alpha}]\\ &\quad + P_{\mathbf{r}} \sum_{\beta} l_{j\beta}[\partial^2 \rho_j(\mathbf{r}, \mathbf{q}_j)/\partial q_{j\alpha}\, \partial q_{j\beta}]\}.\end{aligned} \tag{50}$$

Note that, as follows from § 2, the values $\{\partial\rho_j/\partial q_{j\alpha}\}_\alpha$ and $\{\sum_\beta l_{j\beta}(\partial^2\rho_j/\partial q_{j\alpha}\,\partial q_{j\beta})\}_\alpha$ can be computed as fast as the values $\rho_j(\mathbf{r},\mathbf{q}_j)$.

Similarly, passing from the direction $(\mathbf{L},\boldsymbol{\omega})$ in the variables $(\mathbf{Q},\boldsymbol{\chi})$ to the direction $\boldsymbol{\Omega}$ in the variables $\mathbf{X}$, we have by (42) the following equation:

$$\Omega_{k\beta}=\sum_{j,\alpha}\{L_{j\alpha}[\partial q_{j\alpha}(\boldsymbol{\chi})/\partial\chi_{k\beta}] + Q_{j\alpha}\sum_{i,\gamma}\omega_{i\gamma}[\partial^2 q_{j\alpha}(\boldsymbol{\chi})/\partial\chi_{k\beta}\,\partial\chi_{i\gamma}]\}. \quad (51)$$

It should be emphasized that, in contrast to Agarwal (1981), Dodson (1981) and Hendrickson & Konnert (1980), who have made approximations for the matrix $H=\nabla^2_{qq}R(\mathbf{q})$, we obtain by (43)-(51) an absolutely accurate product $\nabla^2_{\chi\chi}R\boldsymbol{\omega}$ without further assumptions for the elements of the matrix $\nabla^2 R$.

3.4. *Résumé*

Thus, we have shown that for any method of describing an atomic model by generalized parameters and for every minimized function $R(\boldsymbol{\chi})$ an algorithm may be obtained that allows $R(\boldsymbol{\chi})$ and the derivative in the direction $\partial R(\boldsymbol{\chi})/\partial\boldsymbol{\omega}$ as well as all the components of the vectors $\nabla_\chi R$ and $\nabla^2_{\chi\chi}R\boldsymbol{\omega}$ to be computed in four times the time needed to calculate the value of $R(\boldsymbol{\chi})$. Given the model and refinement criteria, we must only specify the transforms

$$\boldsymbol{\chi}\rightarrow\mathbf{q}(\boldsymbol{\chi}) \quad \text{and} \quad \{f^R_s, f^I_s\}\rightarrow R.$$

It should be noted that the criteria expressed by atomic parameters (criterion R_S) can be estimated in a similar way, but the procedure in this case is greatly simplified: once the values of $\mathbf{q}$ have been determined, the criterion can be calculated without the transform $\mathbf{q}\rightarrow\boldsymbol{\rho}\rightarrow(\mathbf{f}^R,\mathbf{f}^I)$.

The authors are grateful to K. V. Kim and Yu. A. Nesterov for the fast differentiation algorithm, which made a principal impact on the present paper. The authors also thank V. V. Borisov, A. A. Vagin and A. I. Ryskin for valuable discussions and O. M. Liginchenko for her great help in preparing the manuscript.

References

AGARWAL, R. C. (1978). *Acta Cryst.* **A34**, 791-809.
AGARWAL, R. C. (1981). In *Refinement of Protein Structures*, pp. 24-28. SRC Daresbury Laboratory, Warrington, England.
COOLEY, J. W. & TUKEY, J. W. (1965). *Math. Comput.* **19**, 297-301.
DIAMOND, R. (1971). *Acta Cryst.* **A27**, 436-452.
DODSON, E. J. (1981). In *Refinement of Protein Structures*, pp. 29-39. SRC Daresbury Laboratory, Warrington, England.
HENDRICKSON, W. A. & KONNERT, J. H. (1980). In *Biomolecular Structure, Function, Conformation and Evolution*, edited by R. S. SRINIVASAN, Vol. 1, pp. 43-57. Oxford: Pergamon.
HENDRICKSON, W. A. & LATTMAN, E. E. (1970). *Acta Cryst.* **B26**, 136-143.
JACK, A. & LEVITT, M. (1978). *Acta Cryst.* **A34**, 931-935.
KIM, K. V., NESTEROV, YU. E. & CHERKASSKY, B. V. (1984). *Dokl. Akad. Nauk SSSR*, **275**, 1306-1309.
REES, D. C. & LEWIS, M. (1983). *Acta Cryst.* **A39**, 94-97.
SUSSMAN, J. L. (1981). In *Refinement of Protein Structures*, pp. 13-23. SRC Daresbury Laboratory, Warrington, England.
SUSSMAN, J. L., HOLBROOK, S. R., CHURCH, G. M. & KIM, S. H. (1977). *Acta Cryst.* **A33**, 800-804.
TEN EYCK, L. F. (1973). *Acta Cryst.* **A29**, 183-191.
TEN EYCK, L. F. (1977). *Acta Cryst.* **A33**, 486-492.
WARME, P. & SCHERAGA, H. (1974). *Biochemistry*, **13**, 357-367.

Acta Cryst. (1985). A**41**, 333-340

Distribution Fitting Methods for Centrosymmetric Structures*

BY J. HAŠEK

Institute of Macromolecular Chemistry, 162 06 Praha 6, Czechoslovakia

AND H. SCHENK, C. TH. KIERS AND J. D. SCHAGEN

Laboratory for Crystallography, University of Amsterdam, Nieuwe Achtergracht 166, The Netherlands

(*Received* 13 *July* 1984; *accepted* 19 *December* 1984)

Abstract

Tests of the distribution fitting methods for centrosymmetric structures show that these methods can be used successfully for the search of a correct solution in direct methods. To get good resolving power, different types of seminvariants ($\sum_1$, triplets, quartets) should be used, as is done in other methods.

* This research was carried out when the first author was a visiting scientist at the University of Amsterdam.

1. Introduction

The power of direct methods for solving the phase problem is dependent on the information about the structure that is contained in the structure invariants