Acta Crystallographica Section A
Foundations of Crystallography
ISSN 0108-7673

Received 19 July 2002
Accepted 1 September 2002

The crystallographic fast Fourier transform. I. *p*3 symmetry

Małgorzata Rowicka,* Andrzej Kudlicki and Zbyszek Otwinowski

Department of Biochemistry, UT Southwestern Medical Center at Dallas, 5323 Harry Hines Boulevard, Dallas, TX 75390-9038, USA. Correspondence e-mail: maga@work.swmed.edu

An algorithm for computing the discrete Fourier transform of data with threefold symmetry axes is presented. This algorithm is straightforward and easily implemented. It reduces the computational complexity of such a Fourier transform by a factor of 3. There are no restrictive requirements imposed on the initial data. Explicit formulae and a scheme of computing the Fourier transform are given. The algorithm has been tested and benchmarked against FFT on the unit cell, revealing the expected increase in speed. This is a non-trivial example of a more general approach developed recently by the authors.

1. Introduction

It has been known for at least 30 years that taking advantage of crystallographic symmetries during computation of the Fourier transform would yield much more efficient fast Fourier transform (FFT) routines for crystallographic data processing. This is the first in a series of papers devoted to this problem. Our work has resulted in a set of algorithms for all 230 crystallographic groups. For every one of them, we can achieve maximal symmetry reduction. By maximal symmetry reduction, we mean that calculations are performed with the use of data from an asymmetric unit only. Moreover, at all times only a region of memory corresponding to the asymmetric unit has to be allocated. The present paper deals with the *p*3 symmetry case. The algorithm for *p*3 symmetry is presented here separately because it is one of the most difficult cases. The notions introduced here will be used in our forthcoming papers (Rowicka *et al.*, 2002*a*,*b*). Last but not least, focusing on only one symmetry group allows one to avoid the mathematical formalism necessary to treat the general case.

Our algorithms reduce both execution time and memory usage of Fourier transform calculation. In the case of *p*3 symmetry, this gain would be approximately by a factor of three compared to the usual *p*1 (*i.e.* ignoring underlying symmetries) FFT routine. There were attempts to handle such data by a generalized Rader–Winograd approach (Bricogne, 1996). However, the resulting algorithms were too complicated for practical use. We based our approach on the Cooley–Tukey (Cooley & Tukey, 1965) decomposition, which has the advantage of a simple geometric interpretation, unlike the widely discussed Winograd scheme. In the Cooley–Tukey algorithm, data are divided into subsets consisting of points regularly distributed in space. Regular spacing has an additional implementation advantage of an easy-to-optimize memory-access pattern.

The paper is organized as follows. In §2, we will describe our choice of the computational grid for the *p*3 group. In §3, we choose a non-contiguous asymmetric unit. Next, in §4, we describe symmetry in the reciprocal space and our choice of an asymmetric unit in this space. In §5, we derive a formula that shows that in order to compute a Fourier transform of $9N^2$ *p*3-symmetric points it is enough to compute three Fourier transforms of N^2 points each. In §§6 and 7, we present a detailed description of our algorithm and tests of its implementation.

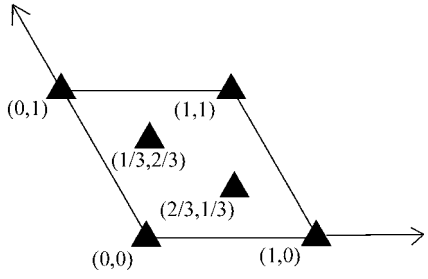

Figure 1
The *p*3 planar symmetry group: unit cell in the real space. Filled triangles depict symmetry axes.

2. Computational grid in the real space

The first step in the calculation of a discrete Fourier transform is to choose a sampling for the data. To allow for the use of the FFT algorithm, the data have to be sampled on a regular grid. The choice of an appropriate computational grid is a crucial step in our scheme. Such a grid has to be carefully adjusted to the underlying crystallographic symmetry. A valid computational grid should be invariant under the crystallographic group action. In other words, every symmetry operator should transform all grid points onto grid points.

The *p*3 unit cell in the crystallographic coordinate system is shown in Fig. 1, with triangles denoting threefold symmetry axes. As shown in this picture, the unit-cell vertices have

coordinates $(0,0)$, $(0,1)$, $(1,0)$ and $(1,1)$, and the threefold symmetry axes within the unit cell are at $(\frac{1}{3},\frac{2}{3})$ and at $(\frac{2}{3},\frac{1}{3})$.

To define our computational grid, we shall introduce a new *grid coordinate system* in which all data points have integer coordinates. In our case, the grid coordinates (x, y) are related to the crystallographic coordinates (ρ, σ) by

$$(x, y) = 3N(\rho, \sigma) - \tfrac{1}{3}(2, 1)$$

and conversely

$$(\rho, \sigma) = (1/3N)(x, y) + (1/9N)(2, 1).$$

The origin of the grid coordinate system is shifted with respect to the origin of the crystallographic coordinate system by vector $(\frac{1}{3},\frac{2}{3})$ in the grid coordinates, as shown in Fig. 2

Let Γ denote the computational grid described above:

$$\Gamma = \{(x, y) : x, y = 0, \ldots, 3N - 1\}.$$

Observe that Γ is invariant under the $p3$ group action, which is a necessary condition for Γ to allow for symmetric calculations.

In our reasoning, we have never required that the symmetry axes have integer coordinates in the grid coordinate system. In fact, they are not in the presented case. Consequently, no data point will lie on a symmetry axis. Thus, every point from Γ is transformed by symmetry operators into a *different* grid point. It is one of the main innovations that allows all grid points to be treated the same way during computations and results in a simple and practical algorithm.

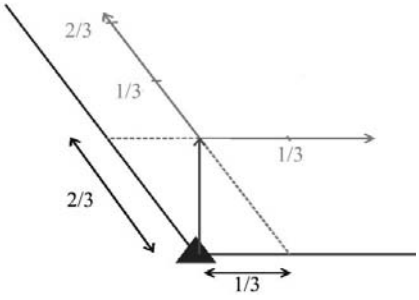

Figure 2
Shift of the origin of the coordinate system, in grid coordinates

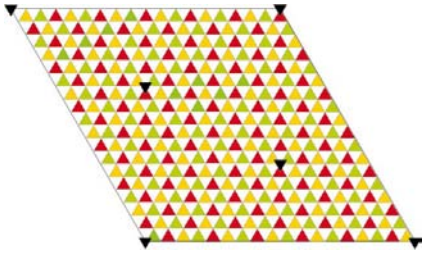

Figure 3
The unit cell in the real space for $N = 6$, corresponding to 9×6^2 grid points. Data points are located at centers of the colored triangles. Red denotes the selected asymmetric unit.

3. Choice of an asymmetric unit in the real space

Following the notation of Bricogne (1996), we denote the period lattice by Λ. In grid coordinates Λ is spanned by the vectors $(0, 3N)$ and $(3N, 0)$. The counterclockwise rotation by 120° around the origin of the crystallographic coordinate system will be denoted by α. Let the action of α be denoted by S_α. The result of action of α on the point with grid coordinates (x, y) is

$$S_\alpha \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -y-1 \\ x-y \end{pmatrix} \bmod \Lambda. \tag{1}$$

The symbol $\bmod \Lambda$ should be understood as follows: $m = n \bmod \Lambda$ if and only if there exist integers p and q such that $m - n = p(3N, 0) + q(0, 3N)$. This means that m and n have the same positions in their respective unit cells.

Since the origin of the coordinate system has been shifted, it no longer coincides with the threefold axis. As a result, the action of α is no longer represented solely by a rotation matrix; now it is a superposition of a rotation by matrix R_α and a translation by vector $\mathbf{t}_\alpha$:

$$R_\alpha = \begin{pmatrix} 0 & -1 \\ 1 & -1 \end{pmatrix} \quad \text{and} \quad \mathbf{t}_\alpha = -\begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Analogously, the action of α^2 (counterclockwise rotation by 240°) can be represented as a superposition of a rotation by matrix R_{α^2} and a translation by vector $\mathbf{t}_{\alpha^2}$:

$$R_{\alpha^2} = \begin{pmatrix} -1 & 1 \\ -1 & 0 \end{pmatrix} \quad \text{and} \quad \mathbf{t}_{\alpha^2} = -\begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

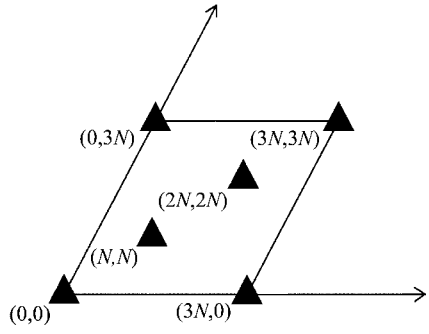

Figure 4
FFT unit cell in the reciprocal space.

Figure 5
The FFT unit cell in the reciprocal space for $N = 6$. The data points lie at the centers of the colored triangles. Red points form the chosen FFT asymmetric unit.

From (1), the coordinates of a point invariant under the action of α should be an integer solution of the pair of equations

$$\begin{cases} -y-1=x \\ x-y=y \end{cases} \bmod 3N.$$

Hence we get an equation for the second coordinate of the invariant point

$$3y=-1 \mod 3N.$$

It is easily seen that this equation has no integer solutions. Consequently, any point invariant under the action of α has a fractional coordinate and as such is not contained in Γ. It can be easily checked that the same holds for the action of α^2. It means that there are no special points on the grid Γ. This is a very desirable property from the computational point of view. In fact, it was one of the reasons we chose such a coordinate system.

Let us divide the grid Γ into three subgrids Γ_0, Γ_1 and Γ_2:

$$\Gamma_0=\{(x,y)\in\Gamma:\ x+y=\ 0 \bmod 3\},$$
$$\Gamma_1=\{(x,y)\in\Gamma:\ x+y=1 \bmod 3\},$$
$$\Gamma_2=\{(x,y)\in\Gamma:\ x+y=2 \bmod 3\}.$$

In other words, the grid Γ_0 consists of points whose sum of coordinates is divisible by 3. From the above definition, it is easily seen that subgrids Γ_0, Γ_1 and Γ_2 are mutually disjoint. Moreover, since every integer equals 0, 1 or 2 modulo 3, it follows that

$$\Gamma=\Gamma_0\cup\Gamma_1\cup\Gamma_2. \tag{2}$$

The subdivision of Γ is depicted in Fig. 3. All points from Γ lie in the centers of colored triangles. Points from Γ_0 are red, points from Γ_1 are green and Γ_2 is yellow. Observe that every point (x,y) from Γ_0 is transformed by the action of α into a point from Γ_2. Analogously, every point (x,y) from Γ_0 is transformed into a point from Γ_1 by the action of α^2. Moreover, every point of Γ_1 and Γ_2 is an image of some point from Γ_0 under the action of α^2 or α, respectively. It follows that

$$\Gamma_2=S_\alpha\Gamma_0 \quad\text{and}\quad \Gamma_1=S_{\alpha^2}\Gamma_0. \tag{3}$$

Substituting (3) into (2), we get

$$\Gamma=\Gamma_0\cup S_\alpha\Gamma_0\cup S_\alpha^2\Gamma_0. \tag{4}$$

We have shown that Γ_0 is a true asymmetric unit because its images fill the unit cell. Moreover, it has no common points with its symmetric images. The reason that we decided to work in a non-standard coordinate system was to make such a convenient choice of the asymmetric unit possible. Moreover, as will become evident in §5, such a choice of an asymmetric unit substantially simplifies Cooley–Tukey FFT calculation.

4. Symmetry in the reciprocal space

Let Γ^* denote a computational grid in the reciprocal space:

$$\Gamma^*=\{(h,k):h,k=0,\ldots,3N-1\}.$$

From the mathemathical point of view, the discrete Fourier transform acts between two identical spaces. The grid Γ in the real space was called the unit cell. Therefore, its counterpart, the grid Γ^*, will be called a *FFT unit cell* in the reciprocal space. Analogously, a minimal set of points at which the Fourier transform must be evaluated will be called a *FFT asymmetric unit* in the reciprocal space.

Let us introduce a shorthand notation

$$e(x)=\exp(-2\pi ix), \tag{5}$$

for real x. With this notation, $e(0)=e(1)=1$. Let $\mathbf{x}\in\Gamma$ denote a vector with coordinates (x,y) and $\mathbf{h}\in\Gamma^*$ denote a vector with coordinates (h,k). Let $\mathbf{h}\cdot\mathbf{x}$ denote the scalar product of vectors $\mathbf{h}$ and $\mathbf{x}$. Let us also denote the Fourier transform of function f by F. In our case, the Fourier transform is given for any $\mathbf{h}\in\Gamma^*$ by the formula

$$F(\mathbf{h})=\sum_{\mathbf{x}\in\Gamma}f(\mathbf{x})e\left(\frac{\mathbf{h}\cdot\mathbf{x}}{3N}\right). \tag{6}$$

The FFT unit cell in the reciprocal space is depicted in Fig. 4 Here, the symmetry axes (black triangles) are not true threefold axes; the symmetry operation is a superposition of rotation and multiplication by phase factors. Precisely, the symmetry operators S^*_α and $S^*_{\alpha^2}$ in the reciprocal space act as follows:

$$S^*_\alpha F(\mathbf{h})=e\left(\frac{\mathbf{h}\cdot\mathbf{t}_\alpha}{3N}\right)F(R^T_\alpha\mathbf{h}) \bmod \Lambda \tag{7}$$

and

$$S^*_{\alpha^2}F(\mathbf{h})=e\left(\frac{\mathbf{h}\cdot\mathbf{t}_{\alpha^2}}{3N}\right)F(R^T_{\alpha^2}\mathbf{h}) \bmod \Lambda. \tag{8}$$

The symbol R^T denotes the transposition of the matrix R. As explained in §3, in the grid coordinate system the symmetry operator S_α contains a translation by the vector $\mathbf{t}_\alpha$. This is why there are phase shifts in (7) and (8).

At a first glance, one might think that a good FFT asymmetric unit in the reciprocal space would consist of three rhombi stretched along the diagonal of the FFT unit cell. However, since

$$F(\mathbf{h})=S^*_\alpha F(\mathbf{h})=S^*_{\alpha^2}F(\mathbf{h}),$$

it follows from (7) and (8) that

$$F(N,N)=e(2/3)F(N,N)$$

and

$$F(2N,2N)=e(1/3)F(2N,2N).$$

Consequently,

$$F(N,N)=F(2N,2N)=0. \tag{9}$$

Therefore, instead of the vertices (N,N) and $(2N,2N)$, we add to the FFT asymmetric unit two extra points at $(0,N)$ and $(N,0)$. The FFT asymmetric unit in the reciprocal space is presented in Fig. 5

5. Computation of the Fourier transform

The definition of the Fourier transform is given by (6). Since Γ consists of three disjoint subsets [equation (2)], it follows that the Fourier transform can be expressed as a sum of three parts:

$$\begin{aligned} F(\mathbf{h}) &= \sum_{\mathbf{x}\in\Gamma} f(\mathbf{x}) e\left(\frac{\mathbf{h}\cdot\mathbf{x}}{3N}\right) \\ &= \sum_{\mathbf{x}\in\Gamma_0} f(\mathbf{x}) e\left(\frac{\mathbf{h}\cdot\mathbf{x}}{3N}\right) + \sum_{\mathbf{x}\in\Gamma_1} f(\mathbf{x}) e\left(\frac{\mathbf{h}\cdot\mathbf{x}}{3N}\right) \\ &\quad + \sum_{\mathbf{x}\in\Gamma_2} f(\mathbf{x}) e\left(\frac{\mathbf{h}\cdot\mathbf{x}}{3N}\right). \end{aligned}$$

Hence, using (3) we get

$$\begin{aligned} F(\mathbf{h}) &= \sum_{\mathbf{x}\in\Gamma_0} f(\mathbf{x}) e\left(\frac{\mathbf{h}\cdot\mathbf{x}}{3N}\right) + \sum_{\mathbf{x}\in\Gamma_0} f(S_{\alpha^2}\mathbf{x}) e\left(\frac{\mathbf{h}\cdot S_{\alpha^2}\mathbf{x}}{3N}\right) \\ &\quad + \sum_{\mathbf{x}\in\Gamma_0} f(S_{\alpha}\mathbf{x}) e\left(\frac{\mathbf{h}\cdot S_{\alpha}\mathbf{x}}{3N}\right). \end{aligned}$$

The asymmetric unit Γ_0 was chosen in such a way that each part of the expression above is a Fourier transform. Let Y denote the Fourier transform of the asymmetric unit:

$$Y(\mathbf{h}) = \sum_{\mathbf{x}\in\Gamma_0} f(\mathbf{x}) e\left(\frac{\mathbf{h}\cdot\mathbf{x}}{3N}\right).$$

Apart from the primitive-cell periodicity, the function Y has an additional centering:

$$Y(h,k) = Y(h+N, k+N).$$

This means that there are only $3N^2$ independent values of Y. We stress that the symmetry of Y is different from that of the final Fourier transform F (described in §4).

Since the symmetry operators act in the real space, then for every function f defined on the real space the following is true:

$$f(\mathbf{x}) = f(S_\alpha\mathbf{x}) = f(S_{\alpha^2}\mathbf{x}).$$

Moreover,

$$\sum_{\mathbf{x}\in\Gamma_0} f(\mathbf{x}) e\left(\frac{\mathbf{h}\cdot S_{\alpha^2}\mathbf{x}}{3N}\right) = e\left(\frac{\mathbf{h}\cdot\mathbf{t}_{\alpha^2}}{3N}\right) Y(R^T_{\alpha^2}\mathbf{h})$$

and

$$\sum_{\mathbf{x}\in\Gamma_0} f(\mathbf{x}) e\left(\frac{\mathbf{h}\cdot S_{\alpha}\mathbf{x}}{3N}\right) = e\left(\frac{\mathbf{h}\cdot\mathbf{t}_{\alpha}}{3N}\right) Y(R^T_{\alpha}\mathbf{h}).$$

For $h, k = 0, \ldots, N-1$, let us define three new functions Z^0, Z^1 and Z^2:

$$\begin{aligned} Z^0(\mathbf{h}) &= Y(\mathbf{h}), \\ Z^1(\mathbf{h}) &= e\left(\frac{\mathbf{h}\cdot\mathbf{t}_{\alpha^2}}{3N}\right) Y(R^T_{\alpha^2}\mathbf{h}), \\ Z^2(\mathbf{h}) &= e\left(\frac{\mathbf{h}\cdot\mathbf{t}_{\alpha}}{3N}\right) Y(R^T_{\alpha}\mathbf{h}). \end{aligned}$$

Now, we can express the final Fourier transform F by Z^0, Z^1 and Z^2:

$$F(\mathbf{h}) = Z^0(\mathbf{h}) + Z^1(\mathbf{h}) + Z^2(\mathbf{h}).$$

Let us assume that $h, k = 0, \ldots, N-1$ and observe that in order to compute the Fourier transform in the FFT asymmetric unit in the reciprocal space it is enough to compute Z^0, Z^1 and Z^2 for such values. The reason is that, knowing these values,[1] we can compute the Fourier transform F in the FFT asymmetric unit in the reciprocal space. To this end, we will use the following formulae for $h, k = 0, \ldots, N-1$:

$$\begin{aligned} F(\mathbf{h}) &= Z^0(\mathbf{h}) + Z^1(\mathbf{h}) + Z^2(\mathbf{h}), \\ F(\mathbf{h}+(N,N)) &= Z^0(\mathbf{h}) + e(1/3)Z^1(\mathbf{h}) + e(2/3)Z^2(\mathbf{h}), \\ F(\mathbf{h}+(2N,2N)) &= Z^0(\mathbf{h}) + e(2/3)Z^1(\mathbf{h}) + e(1/3)Z^2(\mathbf{h}). \end{aligned}$$

We have just achieved a reduction of the problem of finding the simple relationship of the Fourier transform of the unit cell and the Fourier transform of the asymmetric unit.

6. Description of the algorithm

We present the conceptual flow chart for the algorithm in Fig. 6. For better performance, we reorganize the $3N^2$ independent data points in the asymmetric unit into three subsets of N^2 points each. The values of f on these subsets are denoted by a, b and c:

$$\begin{aligned} a(x,y) &= f(3x, 3y), \\ b(x,y) &= f(3x+1, 3y+2), \\ c(x,y) &= f(3x+2, 3y+1), \end{aligned}$$

where $x, y = 0, \ldots, N-1$.

This reorganization is visualized in Fig. 7. In the first step, an external $p1$ FFT routine is used to compute Fourier transforms of a, b and c; the results are stored in A, B and C, respectively. Precisely, for $h, k = 0, \ldots, N-1$,

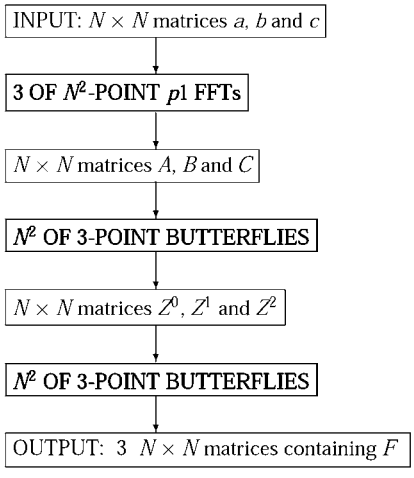

Figure 6
Flow chart for the FFT algorithm for $p3$ symmetric data.

[1] In fact, as explained in §4, we have to know values of Y at two additional points $(0, N)$ and $(N, 0)$.

$$A(h,k)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}a(x,y)e(hx/N)e(ky/N),$$
$$B(h,k)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}b(x,y)e(hx/N)e(ky/N), \qquad (10)$$
$$C(h,k)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}c(x,y)e(hx/N)e(ky/N).$$

Most of the CPU time is spent in this step. Next, a 3-point 'butterfly' operation is performed to calculate the values of Y. For $h,k=0,\ldots,N-1$ and $m:=-h-k$, we compute

$$Y(h,k)=A(h,k)+e((k-m)/3N)B(h,k)+e((h-m)/3N)C(h,k)$$

and store the values of Y in three $N\times N$ arrays Z^0, Z^1 and Z^2:

$$Z^0(h,k)=Y(h,k),$$
$$Z^1(h,k)=e(m/3N)Y(m,h),$$
$$Z^2(h,k)=e(-h/3N)Y(k,m).$$

In the last step, we retrieve the values of the final Fourier transform F from Z^0, Z^1 and Z^2:

$$F(\mathbf{h})=Z^0(\mathbf{h})+Z^1(\mathbf{h})+Z^2(\mathbf{h}),$$
$$F(\mathbf{h}+(N,N))=Z^0(\mathbf{h})+e(1/3)Z^1(\mathbf{h})+e(2/3)Z^2(\mathbf{h}),$$
$$F(\mathbf{h}+2(N,N))=Z^0(\mathbf{h})+e(2/3)Z^1(\mathbf{h})+e(1/3)Z^2(\mathbf{h}).$$

The last two steps, calculating Z^0, Z^1 and Z^2, and using them to compute F, can be performed at one time without using any temporary data storage, save a few register variables. To achieve this, one has to rearrange the loops in the program. Each of the $N\times N$ arrays is divided into three subsets, such that, for a point $\mathbf{h}$, each of the points (h,k), (k,m) and (m,h) lies in a different subset. Here, as before, $m=-h-k$. If N is not divisible by three, one extra point with $h=k=m$ has to be treated separately; if N is a multiple of 3, there are three such points left. An example of such a subdivision is presented in Fig. 8. The loops are performed over only one of the subsets in each of the three $N\times N$ arrays forming Y. This way nine points are processed at a time, and no extra memory needs to be allocated.

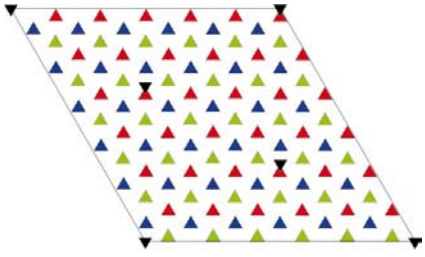

Figure 7
Points in the asymmetric unit in the real space for $N=6$. Decomposition into three regular subgrids (colored red, green and blue here) eases the computation of the $p1$ Fourier transform.

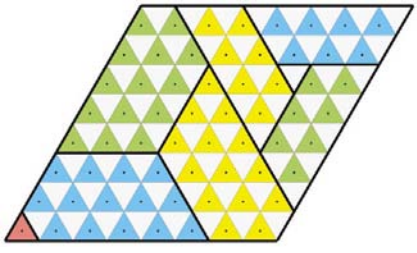

Figure 8
Subdivision of Z^0, Z^1 and Z^2, allowing for a fully in-place FFT calculation, for $N=8$. The loop is executed over the data points marked blue; the red point $(0,0)$ is treated separately.

In §4 [following equation (9)], we observed that the FFT asymmetric unit in the reciprocal space contains also the points $(N,0)$ and $(0,N)$. The Fourier transform F at these points should be computed before overwriting A, B and C, using the following formulae:

$$F(0,N)=(2+e(2/3))\{A(\mathbf{0})+e(2/3)B(\mathbf{0})+e(1/3)C(\mathbf{0})\}$$

and

$$F(N,0)=(1+2e(2/3))\{A(\mathbf{0})+e(1/3)B(\mathbf{0})+e(2/3)C(\mathbf{0})\},$$

where $\mathbf{0}$ denotes the vector $(0,0)$. The presented scheme allows one to compute the Fourier transform F in the FFT asymmetric unit of the reciprocal space. It is done by three $p1$ FFTs on 1/9 of the unit cell each, followed by two passes of butterfly summation. Thus, the computational complexity is reduced three times compared to the usual FFT on the whole unit-cell data.

7. Tests and benchmarks

The speed of computing the Fourier transform of the $p3$ symmetric data using the algorithm described in the present paper was compared against computing $p1$ FFT on the entire unit cell.

The tests were performed as follows. The asymmetric unit, Γ_0, was filled with random numbers. Next, the data were

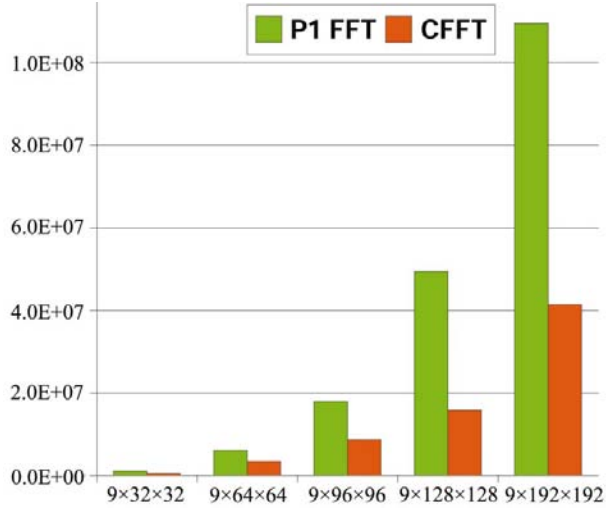

Figure 9
Comparison of speed of the algorithm presented in this paper (CFFT, red) and non-symmetric Fourier transform in the unit cell (P1 FFT, green). Average number of CPU cycles elapsed is plotted *versus* the size of the unit cell.

expanded into Γ using symmetry operators. Then the Fourier transform of the data was calculated by two methods. First, it was computed using the traditional scheme of directly applying $p1$ FFT to the expanded data. Second, we applied the algorithm we invented (§6) to Γ_0. The latter involved doing three $p1$ FFT calculations to compute the A, B and C data arrays at its first stage [see equation (10)]. The numerical values of the final Fourier transform were the same in the two cases to the machine precision. To make the benchmarks fair, we utilized the same $p1$ FFT routine for calculating A, B and C, and for the unit-cell calculation. In both cases, we used functions from the *FFTW* library by Frigo & Johnson (1998). To estimate the gain in speed, we have applied both methods 1000 times for each of several sizes of data arrays, measuring the number of CPU cycles elapsed in each calculation. We did not include in our benchmarks the CPU time spent on expanding the data from the asymmetric unit to the unit cell. The results for asymmetric units containing 3×32^2, 3×64^2, 3×96^2, 3×128^2 and 3×192^2 data points are presented in Fig. 9. The amount of time spent in FFT gets up to three times smaller when our algorithm is used. Three, being the number of symmetry operators in the $p3$ group, is the maximal possible gain due to symmetry reduction.

8. Conclusions

The main idea of the presented approach is to reduce symmetry in such a way that it is enough to calculate $p1$ FFT in the asymmetric unit only and then, in a computationally simpler step, recover the final result. For the central step in the calculation consisting of general $p1$ FFTs, any generic fast Fourier subroutine can be used. Thus, one can profit from the substantial effort made in developing very efficient $p1$ FFT routines (*e.g.* Frigo & Johnson, 1998; Intel, 2001).

In paper II of this series (Rowicka *et al.*, 2002*a*), we will show that schemes similar to the one presented here can be designed for more than 110 crystallographic groups. All these cases share the use of a non-standard computational grid and asymmetric unit. This may bring an issue of compatibility with conventions used in some existing programs. However, conventions used in these programs have no fundamental justification and can be easily adjusted. In paper III (Rowicka *et al.*, 2002*b*), we will present a conceptually more complex solution that allows for a conventional choice of coordinate system (*i.e.* with symmetry axes going through the origin).

The authors thank Keith Henderson and Adam Górecki for valuable discussions. The research was supported by NIH grant No. 53163.

References

Bricogne, G. (1996). *International Tables for Crystallography*, Vol. *B*. Dordrecht: Kluwer Academic Publishers.
Cooley, J. & Tukey, J. (1965). *Math. Comput.* **19**, 297–301.
Frigo, M. & Johnson, S. (1998). Proc. 1998 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, pp. 1381–1384.
Intel (2001). *Intel Math Kernel Library*, http://www.intel.com/software/products/mkl/mkl52/.
Rowicka, M., Kudlicki, A. & Otwinowski, Z. (2002*a*). In preparation.
Rowicka, M., Kudlicki, A. & Otwinowski, Z. (2002*b*). In preparation.