The PROTEIN system: Real-Space Techniques using Patterson and Fourier maps

W. Steigemann
Computer Center, Max-Planck-Institut für Biochemie,
D-82152 Martinsried, Germany
steigemann@biochem.mpg.de
http://www.biochem.mpg.de/PROTEIN

Abstract

Real-space techniques have been an important issue in the PROTEIN program system from its very beginning. In the field of molecular replacement the "real-space search methods" serve the correlation of maps for the exploration of non-crystallographic symmetry (Patterson self-rotation), positioning of known molecular models in unknown structures (cross-rotation, translation) both in Patterson and Fourier maps, location and refinement of local axes in a Fourier map, calculation of "mean Fourier maps", and rotated density maps.

The widely used density modification techniques aiming at the phase improvement and hence interpretability of density maps have required further extension of the software system. Following the principle of flexibility in PROTEIN, this has been achieved by implementing modules with new basic functionality (e.g. simple map algebra incl. rotation, unit-cell generation from an asymmetric unit, etc.). These additional tools accessible to the user through commands may be combined with the existing elements to perform powerful crystallographic tasks on the level of the command language (e.g. superposition of maps, mask calculation by convolution techniques, solvent flattening, molecular averaging).

After a general introduction into the PROTEIN program system, the paper focuses on discussions of its capabilities both for molecular replacement and density modification in real space. Structure analyses from the literature serve as examples.

1 Introduction

The advances both in computer technology and in the methodology of macromolecular crystallography have led to a considerable increase in the speed of structure solution. This development, in turn, has put emphasis on the importance of self-consistent crystallographic software systems for effective working conditions. Two classes may be differentiated:

1. Collection of separate programs exchanging data by standardized formats (e.g. the CCP4 package [1]). The programs may come from different, independent authors and are, therefore, easier to maintain, but less easy to use.
2. Integrated program systems (e.g. XTAL system [2,3], X-PLOR [4], XtalView [5]) require more effort to maintain, but are easier to use. Often they evolve from different primary focuses.

The PROTEIN program system belongs to the second category. The original design goals from its first release [6] today prove to be as important as in the past and have been acknowledged by a wide acceptance within the crystallographic community (over 200 installations): (1) easy and flexible usage, (2) applicability to differently sized problems, (3) wide and growing functionality, (4) efficiency, (5) robustness due to extensive testing, (6) comfortable cooperation with other crystallographic programs and (7) availability on major computer hardware platforms. These goals have been met by:

- Implementation of program modules (building blocks) for performing both simple (basic) and complex tasks
- Use of a command (script) language for parameter specification and activation of the individual tasks
- Chaining of calls of program modules on the user command level which defines the crystallographic task
- Exchange of data between the tasks by use of files and, more recently, memory (file-oriented tasks versus memory-oriented tasks)
- Dynamic use of memory (i.e. independence from problem size)

Further extensions in versions 3 and 4 of PROTEIN [7,8] have been performed along these lines.

1.1 Functionality of PROTEIN

The major purpose of the program system has been and still is the *initial* solution of macromolecular crystal structures by X-ray techniques. As such, the important steps of model building and refinement intentionally have been left out. While interactive computer graphics is absolutely mandatory for the former (e.g. FRODO [9], O [10], MAIN [11], XtalView [5]), quite a number of different powerful comparable methods [4, 12, 13] exist for the latter. Not imposing a preference for a particular approach introduces even higher flexibility for the end user. For suitable and convenient interaction with a variety of programs, features are available in the PROTEIN system which allow the exchange of information on the data level. The most important functions of PROTEIN are:

- Generation and expansion of reflection data files (loading of experimental data with e.s.d.'s)
- Scaling of reflection data (crystals and/or "frames")
- Scaling of derivative mean values
- Averaging of reflection data (incl. summation of partially recorded reflections from pairs of adjacent frames) and data screening with statistics describing their quality
- M.I.R. (multiple isomorphous replacement) with heavy atom parameter refinement
- Encoding of various phase information in Hendrickson-Lattman coefficients (M.I.R., Sim weighting, etc.)
- Combination of independent phase information at various instances (e.g. M.I.R. phasing, structure factor setup, export of data, statistical functions)
- Very general setup of Fourier coefficients (including phase combination)
- Fast calculation of structure factors from atomic coordinates
- Calculation of maps and difference maps (Patterson, Fourier, difference maps) with normal and FFT methods [14]
- Listing, contouring (also stereo), peak searching of maps in all directions of the crystal axes
- Real-space search methods (self- and cross-rotation, translation function, rotation and averaging of maps, vector verification, etc.)
- Density modification techniques in real and reciprocal space (e.g. solvent flattening, non-crystallographic symmetry averaging, convolution techniques)
- Statistical supplement (e.g. figure of merit, significance of anomalous dispersion data, R-factor, etc.)
- Interconversion facilities for exchange of data with other programs and/or systems

1.2 Command language

In order to simplify the command input and reduce the possibility for input errors, use is made of a hierarchically structured command (script) language. The sequence of *control words* on the basic input level (*commands*) (e.g. `LOAD`), each of which corresponds to a particular task, determines the sequence of actions to be taken. In this way it defines the crystallographic task to be executed. Input parameters to the control words are *numbers* (e.g. `100`, `43.1`), *strings* (enclosed within quotes, e.g. `'-X, -Y, Z+1/2'`), *single letters* (e.g. `A`) or sequences of control words on the next higher *input level* (*subcommands*, enclosed within parentheses, e.g. `four (grid 100 100 100 scale 0.1)`). These themselves may carry further arguments or sequences of control words, with possibly further nesting. Organization of the command input in free format, support of default values for non-specified arguments, and mostly free order of subcommands on higher input levels ease usage and make the command input self-descriptive.

Often used command sequences may be deposited in separate files and executed as procedures by a simple reference to their file names (e.g. `@'sp19.cmd'`). In this way control may be passed from one command file to another, up to depth of five, control being returned to the previous one upon exhaustion of the current one. The support of environment variables defined on the operating system level enhances the flexibility of the command structures (e.g. `@'"MPIPROT"/exe/sp19.cmd'`).

Although command sequences are typically set up with an ASCII editor prior to execution, the keyboard itself may serve as the primary input level, as well. In such cases the availability of the procedure mechanism is of particular importance. Despite this possibility, my personal preference for interactive execution is the separate setup of a command file, followed by its execution, since a truly interactive user interface with prompts, menus, etc. and recording into a command file (i.e. an intelligent PROTEIN command editor) is not available. It is appreciated that there is some need for such an interface, from which novices would benefit the most. A WWW interface is being considered as a general graphical user interface (GUI).

The above is summarized in the following example:

```
load (cell 64.9 78.32 38.79
      @'"MPIPROT"/exe/sp19.cmd' !P21221
      format '(3F4.0,2X,2F10.2)' ! CCP4
             (h k l fobs sigma)  )
```

2 Molecular replacement in real-space

Molecular replacement (a term introduced by Rossmann [15]) typically is used for two (in principle independent) purposes: (1) determination of non-crystallographic symmetry (NCS) elements within the asymmetric unit and inclusion of this information in phase determination (e.g. NCS averaging); (2) determination of initial phases for an unknown structure by positioning a molecule (or fragment of it) in the cell of the unknown. In the following I shall concentrate on the latter.

For computational reasons this principally 6-dimensional problem (three rotation angles, three components of the translational vector) is, in general, separated into two 3-dimensional problems. A rotation function is used to determine the angular orientation, and a translation function to determine the position of the correctly oriented molecule. Since phase information is not available, these functions have to be based on the correlation of the Patterson functions of the known and unknown structures, or parts therefrom. Two principally interconvertible approaches are known: one operating in reciprocal space (first introduced by Rossmann and Blow [16]), the other in real space (first mentioned by Hoppe [17]). For this purpose, it is convenient to think of the Patterson function as a superposition of all interatomic vectors within the unit cell, composed of the self-vector sets (between atoms of the same molecule) and cross-vector sets (between atoms of different molecules). Since the self-vector set of a rotated molecule depends merely on the rotation of the unoriented self-vector set (i.e. it is translation independent), any rotational search needs to consider the self-vectors only, whereas any translational search has to involve the cross-vectors of properly oriented molecules.

A number of crystallographic software packages is available for molecular replacement: the *MERLOT* package [18] comprising a consistent collection of several prominent algorithms not mentioned here explicitly, and the *CCP4* package [1], which also includes the very effectively operating fast rotation function *AMoRe* (Navaza [19]). Other recent approaches are the Patterson correlation refinement (Brünger [20]), available in *X-PLOR* [4]), and the direct rotation function (DeLano [21]).

The *real-space search* routines available within PROTEIN are a generalization of Huber's [22] approach. Two 3-dimensional maps (*search map* and *target map*) are correlated by relative orientations or translations in real space. Depending on the maps and operations chosen, the following applications are possible:

- Self-rotation of crystal Patterson maps (both *search* and *target* maps) for the determination of non-crystallographic symmetries
- Rotation of a model self-vector set (*search* map) within the crystal Patterson map (*target* map) to find the orientation of the model in the crystal cell
- Translation of a model cross-vector set (*search* map) on Harker sections of the crystal Patterson map (*target* map) to find the position of the model in the crystal cell
- Rotation of a disc of unity density around the origin of the crystal Patterson map to find "local" Harker sections (i.e. planes of high density)
- Translation of a finite disc of unity density on Harker planes to find a coarse center-of-gravity difference vector between symmetrical molecules
- Location and refinement of the position of local axes in a Fourier map
- Mean Fourier: calculation of the average of rotated and unrotated Fourier sections. Simple rotation of density
- "Vector verification" for the interpretation of difference Patterson maps

The *target map* is represented on a full 3-dimensional grid, whereas the *search map* typically consists of a selected set of coordinates (peaks) from a full map for computational reasons. The "correlation" function is represented as a *correlation map*, which may be defined in several manners:

1. *Product* function $C = \sum\{P_t(R\cdot x)P_s(x)\}$, where P_t represents the target map; R is the product of the orthogonalization, transformation (rotation or translation) and deorthogonalization matrices; and the various x are the coordinates of the search model P_s ("peaks" = grid points above a threshold). High values indicate a high degree of overlap.
2. *Difference* function $C = [\sum\{P_t(R\cdot x)-P_s(x)\}^2]^{½}$, which may be used in two forms: C_1 uses all "peaks", while C_2 uses only those with $P_t < P_s$, since at a correct placement $P_t \geq P_s$. In this case, low values indicate a high degree of overlap.

The product function has the advantage that the search and target maps may be on arbitrary scales. The difference functions are more sensitive but require correct scaling which is often difficult to achieve.

It is obvious that the approach to the Patterson search problem in real space offers the possibility of modifying the search model without great difficulty in intuitively reasonable ways. This first has been pointed out by Nordman [23].

In the following, the individual steps required within PROTEIN to perform such calculations shall be demonstrated using data from structure analyses (Ribonuclease Sa with two molecules in the asymmetric

unit, by Sevcik et al. [24], and Cytochrome *c'* with a four helix bundle, by Baker et al. [25]). These also served as examples in an EU-sponsored training workshop on Molecular Replacement, organized by E.J. Dodson in Spring 1996.

2.1 Rotational search

The calculation of the *target map* involves two major steps: (1) the generation of a reflection file (which includes the loading of experimental data and, formally, crystal and derivative scaling, although multiple reflection data are missing); (2) the calculation of the Patterson map (setup of Patterson coefficients, Fourier summation). If required, the effect of the origin peak can be removed by subtracting its contribution. These two steps are demonstrated in Fig. 1 and 2.

```
work list i label 'RNA SA'
files (A 21 (status'scratch') B 22
      (stat 'scratch') C 24 (stat 'scratch')
      input 30 (name'rnas25.ref' stat'old'))
! == Load data from unit 30 to file B
load (cell 64.90 78.32 38.79 symtries'X,Y,Z'
     '-X+1/2,-Y,Z+1/2'  'X+1/2,-Y+1/2,-Z'
     '-X,Y+1/2,-Z+1/2' spacegroup 'P212121'
     format '(3F4.0,2X,2F10.2)' (h k l fobs
      sigma) assign d 'NATI' assign s 'MEAN')
! == Relative scaling of sources of native
!    and calculation of mean values.
files (interch a b) scale 'NATI' mean
!
! == Now determine derivative scale factors
files (interch a b b 12 (name 'rnasa.fla'
        stat'new')) scale (noref b) mean
```

Figure 1: Reflection file with experimental data

```
work list i label 'RNA SA'
files (A 12 (name 'rnasa.fla' shared)
   C 22 (stat 'scratch') D 21 (stat'scratch')
   E 13 (status 'scratch'))
! == Setup of Patterson coefficients
resol 2.5 SF (ampl S (fo 'NATI')
              comment 'Map: Fo(NATI)**2')
! == Fourier summation in ASU of Patterson
four (grid 0.6 layout 0.5 0.5 0.5 scf 0.1)
! == Convert map from Y- to Z-sectioning for
!    real space search routines
files (f 14 (name 'exp_patt.fle'))
copy e f (dire z)
```

Figure 2: Calculation of crystal Patterson map

Similarly, the calculation of the *search map* requires the generation of a reflection file (this time with "dummy" data, since experimental data are missing). A large P_1 unit cell is chosen to avoid overlap of the model self-vector set with cross-vectors (Fig. 3). This time, structure factors have to be calculated from the known molecular model (external AWK-scripts proved to be most useful for selecting coordinates from PDB files), which are used subsequently for the calculation of the model Patterson and its reduction to the set of peaks used in the rotational search (Fig. 4). In order to exclude intermolecular vectors in the final correlation (restriction of integration volume), a proper range of radii has to be defined for this selection.

```
work list i label 'RNA SA model'
files (A 21 (status'scratch') B 22
      (stat 'scratch') C 24 (stat 'scratch')
      input 30 (name 'dummy.ref' stat'old')
      punch 31 (status 'scratch'))
! == Generate file with one dummy reflection
load (cell 100. 100. 100. symtries 'X,Y,Z'
   format '(3F4.0,2X,2F10.2)' (h k l fo sig)
   assign d 'NATI' assign s 'MEAN')
!
! == Find missing reflections up to the given
!    resolution limit
resol 2.5 (shell 20)
check b (compl (punch '(3i4,2h1.,a4)'))
!
! == Add them to reflection file and perform
!    dummy scaling and averaging
files (interch a b input 31 (rewind))
load (format '(3F4.0,F2.1)' (h k l fobs)
      assign d 'NATI' assign s 'MEAN')
files (interch a b) scale 'NATI' mean
files (interch a b b 23 (name '1sar_p1.fla'))
scale mean
```

Figure 3: Making reflection file of search model

```
work list i label 'RNA SA model' resol 2.5
files (A 11 (name '"fileA"' shared)
       B 21 (name '"fileB"') C 22 (status
 'scratch') input 18 (name '"model"' shared))
!
! == Calculate model structure factors
fc (spacegroup 'P1' bov 15
    atomt 1 'C'
     (formf 1.455 1.462 3.775 22.49 0.7241)
    atomt 2 'N'
     (formf 1.459 2.001 4.471 17.02 1.023)
    atomt 3 '0'
     (formf 2.113 2.867 4.637 14.75 1.211)
    atoms (entry 1 9999) )
!
! == calculation of a model Patterson map
files (interch A B D=25 (status 'scratch')
  F 26 (stat 'scratch') E 13 (name '"mpatt"')
  h 14 (name '"model_patt_peak"') )
resol 8.0 2.5
SF (ampl S (fc 'NATI')
    comm 'model Patterson P1 RNA SA ...')
fourier (fft grid 100 100 100  scf 0.1)
!
! == complete unit cell and sectioning in Z
copy E F (direction z)
files (interch e f )
copy e f (section 0 100 0 100 0 100 G)
!
! == pick density values for cross rotation
!    within a sphere of 20 Angstroms
search (patt F (sym 'P1')  select H
       (thr 400  rad 4 20 centre 0 0 0 g) )
```

Figure 4: Calculation of structure factors and model Patterson map

After these initial steps, all data are available in the form required to calculate a rotation function in real space. The peak set selected from the search map is rotated against the target map, whose values at the target positions are computed by interpolation. The values of the above described "correlation" function (e.g. product function) at the various angular positions are stored on a 3-D grid in the correlation map (Fig. 5). Depending on the kind of rotational search (self- or cross-rotation) different definitions of angular systems may be appropriate (Eulerian or spherical polar angles).

```
work 10000K  list I  label 'RNA SA'
files  ( E 11 (name '"exp_patt"')
         G 14 (name '"model_patt_peak"')
         F=26 (name '"crot"' ) )
! == Do a rotation search in a 5 deg raster
search (patt E (sym 'PMMM') !target map
      peaks 2100 G (Thr 500 grid 100 100 100
       cell 100 100 100 90 90 90) !search map
      rot 2 F P  'Cross rotation R&B angles'
       (angles 0 180 5   0 360 5   0 180 5) )
```

Figure 5: Calculation of rotation function

Finally, the correlation map has to be analyzed which is done by the standard modules of contouring a map and/or searching for peaks (or holes) in it. This is demonstrated in Figs. 6 and 7.

```
work 10000K  list I  label 'RNA SA'
title 'Analysis of cross rotation RNA Sa'
files  ( E 12 (name '"crot"')
         F 13 (status 'scratch')  )
! == Rescale rotation function map:
!             average=0; standard deviation=1
copy e f  (add A -1.0 mult S 1.0)
! == Search for peaks in rotation function
pekpik F (level 2.0)    check  F (density)
! == Contour section containing correct peak
cont f (layer 170  level 2.5 0.5 10.0 d 92
                   level 0.5 0.5 2.0 d 91
                   level 0 0.5 0 d 93)
```

Figure 6: Analysis of the correlation map

Further analysis (e.g. checking for plausibility and symmetry considerations) may be most easily done by supplementary tools provided for use and modification on the operating system level.

2.2 Translational search

In a translational search, the positions of two (symmetry related) correctly oriented molecules are varied and the corresponding cross-vectors (*search map*) are correlated with the Patterson of the unknown structure (*target map*). Again, equivalent formulations exist for real-space and reciprocal-space approaches. In the latter case, quite a variety of different functions (termed translation functions) have been developed. All of these (for a review see Fitzgerald [26]) have in common with the original one of Crowther and Blow [27] that they are evaluated by means of a Fourier transform.

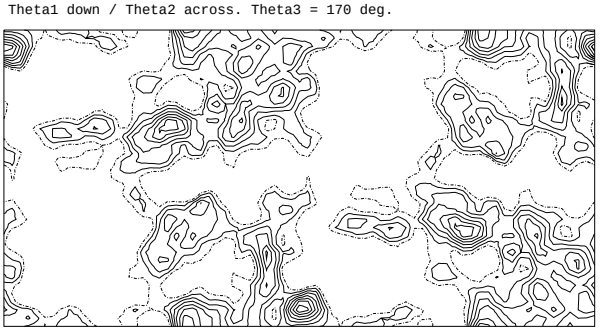

Figure 7: Contour plot of the correlation map. The backbone of molecule A of Ribonuclease Sa has been used as a search model in the crystal Patterson. The symmetry of the rotation function $180\text{-}\theta_1$, $180\text{+}\theta_2$, θ_3 (θ_1, θ_2, θ_3 are the Eulerian angles as defined by Rossmann & Blow [16]) reflects the space group symmetry. The highest peak is located at the correct angular position.

A Fourier transform is also required for the calculation of the cross-vector set with coefficients $F_i(\boldsymbol{h})\cdot F^*_k(\boldsymbol{h})$, $F_i(\boldsymbol{h})$ and $F^*_k(\boldsymbol{h})$ being the calculated structure factor and its conjugate complex of molecules i and k, respectively. Subsequently, however, a number of additional steps are necessary which is prohibitive to the calculation of the correlation map in real space. Even reasonable modification of the target and search maps (e.g. subtraction of the correctly oriented self-vector sets or removal of the origin peak) can be accommodated in an easy reformulation of the Fourier coefficients for the translation function. The only advantage of performing this operation in real space may be seen in the choice of the difference function instead of the product function as a criterion of fit. This, however, is sensitive to the scaling of search and target maps.

For these reasons, little use has been made of the real-space translational search available in PROTEIN. Instead, standard reciprocal-space translation functions have been calculated outside the program system. For this purpose the availability of flexible facilities for data exchange proved to be quite useful.

Even though, the possibility of performing a positional search within PROTEIN is exemplified in Fig. 8, as it demonstrates the modularity of the program system. The only difference to the rotational analysis is the control word `trans` which causes the search model to be

translated instead of rotated (control word `rot`, cf. Fig. 5).

```
SEARCH (PATT [E or F]|E  or  FOUR [E or F]|E
                ( ... )   ! target map
        PEAKS n [G or H or I]|G  ['format']
                ( ... )   ! search map
        TRanslation E of F [P] or [D] or [M]
                  ['descriptive text']
         (SHifts  xs xe dx  ys ye dy
                            zs [ze dz]
          [MOVe x1|0 y1|0 z1|0 [G or A]|G
                x2|0 y2|0 z2|0 [G or A]|G]
          NORMalize c  or  SCALefactor s )   )
```

Figure 8: Summary of commands for performing a positional search

With the new facilities set up for density modification (described in section 3), it should be not too difficult to calculate reciprocal-space translation functions internally as well (by definition of the Fourier coefficients on the command level). The analysis of the resulting 3-dimensional correlation map may be performed (possibly after import) inside PROTEIN in any case.

2.3 Vector verification

A completely different application of the set of real-space search routines is the so-called *vector verification* in a vector map. This, typically, is some kind of difference Patterson map and, again, represents the *target map*. Instead of a "static" search map, *search vectors* are computed "dynamically" when scanning the target map. From each scan position all symmetry related positions (both crystallographic and non-crystallographic) are calculated. The target map is looked up at all positions that correspond to difference vectors between these generated positions. All values are summed (or multiplied - in this case, negative target values are ignored), and the result is saved in the *correlation map*.

This procedure is primarily intended for the automatic interpretation of difference Patterson maps. The resulting map will have peaks at all possible locations of the heavy atoms (also corresponding to the space group dependent choices for the origin of the unit cell). Therefore, it is advisable to carefully choose the scan region. Once major heavy atom sites have been determined, their locations may be used to find minor sites from their difference vectors to the major sites. In this situation the origin of the cell is already fixed. In general, the complete asymmetric unit has to be scanned.

This method eases interpretation of difference Patterson maps significantly. This is particularly useful in cases of high-symmetry space groups or the presence of multiple sites.

During the structure determination of cytochrome *c'* [25] the iron-anomalous scattering turned out to be crucial for the success of the whole analysis. It not only enabled initial phasing, but also allowed discriminating between multiple molecular replacement solutions and, finally, to identify one as correct.

In the following, these data serve as an example for a vector verification procedure. The required target map here is an anomalous difference Patterson map with coefficients $(|F_o^+| - |F_o^-|)^2$. Assuming that the data of the Friedel pairs are available in the reflection file for the native compound `'NATI'`, the definition of the Fourier coefficients for this map is:

```
SF (ampl S (dfpm 'NATI') phase 0 weight 1)
```

where `dfpm` is a control word that selects the anomalous differences.

The resulting map is very clean, as may be seen from a contour plot of the Harker section *z=1/3* (Fig. 9).

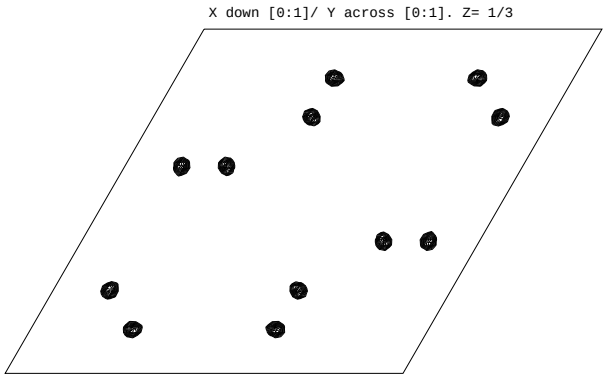

Figure 9: Harker section z=1/3 of the anomalous difference Patterson map for *Alcaligenes denitrificans* cytochrome *c'*

The central portion of the PROTEIN command file for the vector verification is:

```
search (patt E (sym 'P1-' change 2 rad 4)
        vect F 'VV in anomalous DP'
           (shift 0 34 1  0 34 1  0 38 1)  )
```

In order to avoid inadvertent influence from the origin peak, it is simply cut out (all values within a sphere of 4Å around the origin are set to zero). The scan range for the vector verification (control word `shift`) covers an asymmetric unit. A contour plot of the vector verification is shown in Fig. 10. The two large peaks are equivalent and correspond to two different possibilities of enantiomorphs (space groups $P6_122$ and $P6_522$).

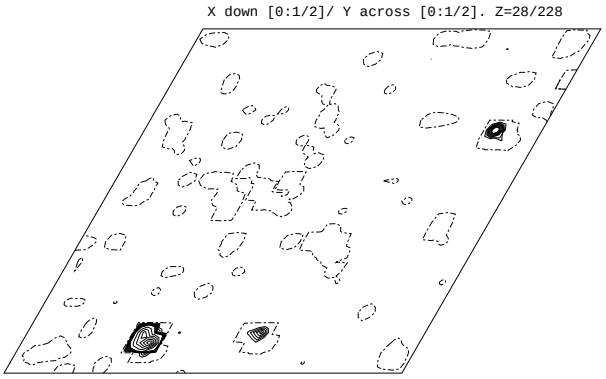

Figure 10: A section of the vector verification map in the anomalous difference Patterson map.

3 Density modification

Density modification techniques aim at the improvement of phases by imposing physically meaningful constraints on the map on a non-atomic level. When the secondary phase information derived therefrom is combined with the primary phase information available from experiment in an iterative way, a density map will be obtained finally which matches both the experimental data and the physical requirements at their best. Such a density map is expected to be superior to the initial one, and usually allows its interpretation and the subsequent refinement of a molecular model.

Use of quite a number of physical constraints can be considered, the most commonly used include:

- Positivity and boundedness
- Map continuity and use of a partial model
- Histogram matching (examination of statistical distribution of density at a given resolution)
- Solvent flatness
- Non-crystallographic symmetry
- Atomicity (only with very high-resolution data)

In all cases, the iterative approach involves the following steps:

0. Calculation of an initial map from experimental data (also phases)
1. Modification of electron density in real space
2. Calculation of structure factors by inverse Fourier transform of density map generated in step 1
3. Combination of structure factors (amplitudes and phases) from experiment and modification
4. Calculation of a new map with these altered structure factors. This map is used as input to step 1 to start a new cycle. Finally it is interpreted

A good overview of the theory and practice of density modification has been given by Podjarny and Rees [28].

3.1 Required tools

In order to use these methods in PROTEIN, implementation of a number of additional tools is required. Following its philosophy of providing elementary tasks (building blocks) which then may be combined with the existing tools to perform crystallographic tasks on the command level, these include:

- Inverse Fourier transform of maps (calculation of structure factors)
- Determination of phase probabilities (figures of merit) for calculated structure factors
- Modification of structure factors in reciprocal space (simple structure factor algebra)
- Modification of map on the basis of density values, optionally by use of restricted areas (masks)
- Combinations of maps (simple map algebra), including rotation and translation, optionally by use of restricted areas (masks)
- Generation of maps from atomic parameters
- Alteration of layout parameters of maps (e.g. expansion to unit cell)

When the steps necessary in density modification procedures are further broken down to even more elementary tasks, a lot of data exchange is required between them. The approach of using solely data files for this purpose is uncomfortable for the user (and also the programmer). Therefore, the mechanism for transferring data from one task to another has been extended by keeping them temporarily in computer main memory. Since PROTEIN had the capability of allocating memory dynamically from its beginning (mainly for providing generality for differently sized problems) and of labeling it internally, it only had to be made explicitly available to the user. Now, he may allocate named memory segments, release them again and use them for the temporary manipulation of maps and (their) Fourier transforms. Initial loading and final saving is ensured through data files:

`READ D 'ftxx'` allocates memory segment `'ftxx'` and loads it with a Fourier transform from file D

`WRITE D 'ftxx'` writes Fourier transform from memory segment `'ftxx'` to Fourier coefficient file D

`READ E 'map'` allocates memory segment `'map'`, loads it with density from map file E

`WRITE E 'map'` writes density map from memory segment `'map'` to map file E

`PURGE 'ftxx' 'map'` releases memory segments `'ftxx'` and `'map'` for further use

Typical usage of "memory-oriented" tasks for density modification would include: (1) read operations to memory, (2) modifications (manipulations), and (3) write operations for saving the results across different jobs.

The basic control words for manipulation of maps or Fourier transforms are `COPY 'from' 'to'` and `MODIFY 'from'` with instructions for modification on the next input level. The first command form leaves the segment `'from'` intact, whereas the second one performs the alterations in place. The supplementary command `CHECK 'from'` allows the calculation of statistical measures from the data contained in memory.

The context-sensitive commands for modification (i.e. possibly different for map and structure factor manipulations) include (unavailable ones in *italics*):

- `ADD number` adds a constant (to each map point or structure factor, which may be complex in the latter case)
- `ADD (table ...)` adds constants from a resolution-dependent table of values (Fourier transforms only)
- `ADD 'mod'` adds another memory segment of the same type, i.e. another map or Fourier transform
- `MULT number` multiplies by a constant (which may be complex for Fourier transforms)
- `MULT (table ...)` multiplies by constants from a resolution-dependent table of values (Fourier transforms only)
- `MULT [C] 'mod'` multiplies by another memory segment of same type, i.e. another map or Fourier transform; this corresponds to convolution in the corresponding Fourier space. The letter C denotes the conjugate complex. Assume, `'ft'` contains a molecular transform, then `COPY 'ft' 'patt'` `(mult C 'ft')` would set up the transform of its Patterson map
- `NORMALIZE` adds and multiplies map values by constants to yield an average of zero and a standard deviation of one
- `REPLACE range target` replaces density values within a specified range by a target value
- `REPLACE range target [I` or `O] 'mask'` replaces density values within a specified range by a target value inside or outside a mask
- `REPLACE range target` *`(INSIDE 'mask'`* or *`OUTSIDE 'mask')`* replaces density values within a specified range by a target value inside or outside a mask
- `ADD 'map'` *`(ROTATE ... TRANSLATE ... INSIDE 'mask')`* adds rotated and/or translated map by specified operators within indicated mask (for NCS averaging)
- *`EXPAND (options)`* changes layout parameters of map, e.g. for expansion to unit cell

All operations are executed immediately and, therefore, may be "chained" on the command level. In this way, powerful modification schemes may be defined on the user command level and deposited for subsequent general usage.

3.2 Applications

Two examples making use of the available map manipulation features will be described in the following.

3.2.1 Determination of molecular envelope

B.C. Wang's algorithm [29] for automatic determination of a molecular envelope from a density map involves the following steps:

- Truncation of the map, i.e. suppression of its negative values
- "Local averaging", i.e. replacing the density at each grid point by the weighted average of the densities at surrounding grid points, within some radius *R*, typically 2.5 times the resolution used. The weighting function in this process is $w = 1 - r/R$ for positive density values and $r < R$
- Suppressing all values below the solvent level and setting all values above solvent level to 1 (the solvent level being determined from a histogram of density values so that the number of grid points below this level corresponds to the expected solvent content of the crystal)

The "local averaging" is equivalent to a convolution between the truncated map (i.e. its positive portions) and the weighting function *w(r)*. The quite compute-intensive task (in Wang's original programs) can be made much faster by use of reciprocal-space methods following the convolution theorem. Instead of averaging in real space, the Fourier transform of the truncated map only needs to be multiplied with the transform of the weighting function [30,31]. This is isotropic and only dependent on $\sin\vartheta/\lambda$ and can be expressed analytically [31]. A Fourier synthesis of the resulting transform directly yields the "averaged" map.

Suppose the Fourier transform of the weighting function is available in tabular form (this has been done in an external program for a number of typical averaging radii). The sequence of necessary steps may be "programmed" with the PROTEIN command language:

```
!== 1 = Truncate (suppress negative values)
!       map within memory area 'mpin'.
read e 'mpin' modify 'mpin' (suppr -10000 0)
!
!== 2 = Backtransform map and perform
!       averaging by convolution (multiply
!       Fourier transforms of truncated
!       density and weighting function.
!       Calculate averaged map by FFT.
!       Data exchange partially via files
!         (for compatibility reasons)
```

```
files (c 50 (stat 'scratch') d 51 (stat
       'scratch') e 52 (stat 'scratch') )
write e 'mpin' (comment 'truncated map')
four i read d  'fttr' ! FT of truncated map
copy 'fttr' 'ftwt' ( ! multiplication of FT's
  mult (table @'"MPIPROT"/exe/ftwt_r6.dat'))
write d 'ftwt'  purge 'fttr' 'ftwt'
four (grid 64 34 72 comment 'averaged map')
!
!== 3 = Statistics on averaged map including
!       histogram to determine density cutoff
!       for solvent level corresponding
!       to the protein content in the cell
read e 'mpav' modify 'mpav' (normalize)
!
!== 4 = Apply solvent level to alter map:
!       Density values below cutoff are set
!       to 0., otherwise 1. The resulting
!       molecular envelope (mask) may be used
!       for further modification procedures.
copy 'mpav' 'envl' (replace -0.75 10000. 1.
                    replace -10000. -0.75 0.)
! files (c 50(del) d 51 (del) e 52(del)
```

Steps 2 and 3 look fairly complicated due to the (still necessary) use of files for certain tasks. Upon extended implementation of labeled memory segments, intermediate files may be avoided completely. Generation of the averaged map in step 2 would then be formulated more simply:

```
four 'mpin' 'fttr' ! Fourier transform of map
copy 'fttr' 'ftwt' ( ! multiplication of FT's
  mult (table @'"MPIPROT"/exe/ftwt_r6.dat'))
four 'ftwt' 'mpav' (grid 'mpin' layout'mpin')
```

Finally, in step 4, the assignment of a cutoff level might be directly expressed as solvent content, which would eliminate step 3 completely.

When deposited in a command procedure, the automatic calculation of a molecular envelope might look the following way:

```
files (e 11 (name 'mir25.fle')) resol 2.5
@'molenv.cmd' !input map: file E, output maps
              !in 'mpin' (truncated)
              !'mpav' (averaged, av=0, std=1)
              !'envl' (molecular envelope)
check 'envl' write f 'envl'  !save envelope
purge 'mpin' 'mpav' 'envl'
```

3.2.2 Solvent flattening

The existence of a uniform solvent region implies strong constraints on the structure factor phases [32]. Therefore, imposing flatness on the solvent (i.e. setting the density outside a molecular boundary to a mean value) leads to improvement of the phases. In general, the effect is more pronounced in loosely packed crystal structures, where the volume occupied by the solvent may be quite considerable.

The iterative process of solvent flattening requires the initial definition of a molecular volume (envelope). Quite a number of different approaches have been reported for this task. An automatic procedure has been demonstrated in the preceding section. The molecular volume typically is represented as a 3-dimensional mask with ones at grid points inside the molecule and zeroes otherwise. The following steps are applied repeatedly until convergence is reached:

- Set the density outside the molecular envelope (i.e. solvent region) to the average in this area.
- Optionally truncate the protein density inside the molecule (apply positivity);“histogram matching” would be performed at this instance additionally.
- Back-transform the modified map and combine the resulting phases with the starting (experimental) phases. Here, several possibilities exist; one commonly used is based upon the assignment of phase probabilities for these phases (Sim-weighting [33]). The combination itself may be performed according to the algorithm proposed by Hendrickson and Lattman [34].
- Calculate a new map from these “combined” phases and repeat the whole process. Optionally a new envelope may be calculated from the improved map.

These steps are formulated in a PROTEIN command procedure (an M.I.R. map of BPTI served as an example):

```
!  Input: electron density on file E
!         mask for molec. vol. file F
!         reflection file on file A
!  Output: solvent flattened map on file 50
!
! == Flatten the solvent region outside mask
!    and truncate region of heavy atom.
read f 'mask' read e 'map'
modify 'map' (replace -1000. -18. 0.025
       replace o 'mask' -10000 10000 0.025)
!
! ==  Write modified map to file and
!     calculate structure factors
files (d 51 (stat'scratch') c 52 (stat
       'scratch') e 50 (stat'scratch')
       b 54 (stat'scratch'))
write e 'map' four i
!
! == Merge structure factors into reflection
!  file for combination with M.I.R. phases
copy a b (merge (HLset 3))
    files (interch a b)
sf (ampl (fo 'NATI') phase (HLset 2 HLset 3)
    weight (fom) )
!
! ==  Calculate new map from M.I.R. phases
!     and solvent flattening.
four (grid 64 34 72  scf 100)
!
!  After swapping of reflection files ready
!   for a new cycle
files (interch a b)
```

The repetition of the whole process simply is (without an automatic criterion for convergence):

```
@'solvflat.cmd' @'solvflat.cmd'
@'solvflat.cmd' @'solvflat.cmd' ...
```

The above procedure implies knowledge of the average value within the solvent area (a specific number has been defined with the `REPLACE` subcommand). This has to be

determined in a preceding step by performing an analysis (CHECK) of the initial map within the solvent area. For this purpose the mask has to be inverted (i.e. inside the molecule 0, outside 1) and applied to the density. The statistical analysis (which also calculates an average) ignores zero values.

```
read f 'mask' modify 'mask' (mult -1. add 1.)
read e 'map'  modify 'map' (mult 'mask')
check 'map'
```

Fig. 11 and 12 show one section of an M.I.R. map of BPTI at 2.5A before and after 16 cycles of solvent flattening.

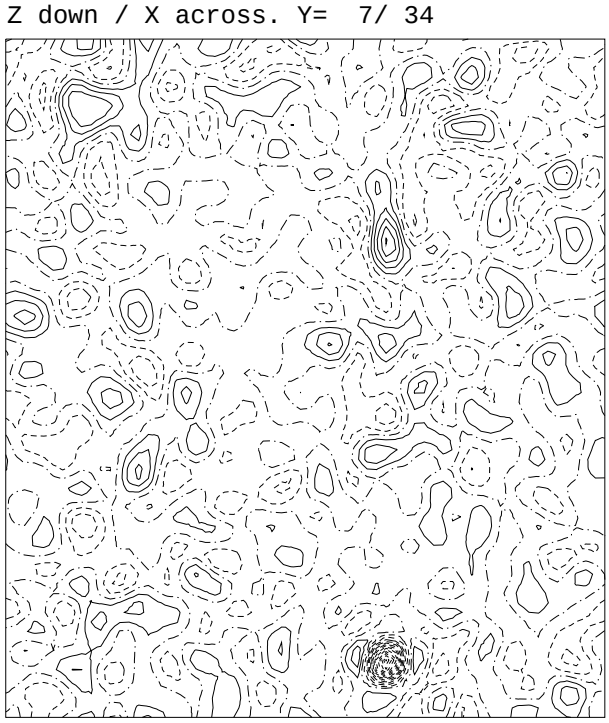

Figure 11: One section of the 2.5A M.I.R. map of BPTI before solvent flattening. The heavy atom area has a deep hole.

4 Availability

PROTEIN originally [6] was developed on an IBM mainframe. During the era of minicomputers with 32-bit architectures, it has been adapted for VAX/VMS. With the advent of cost-effective RISC-workstations, it has been ported to a variety of platforms. It is now available for prominent UNIX systems incl. Silicon Graphics, DEC Alpha (asked for primarily), IBM RS/6000, Sun, Hewlett Packard, as well as OpenVMS on VAX and AXP, on particular request. The program has to be licensed (contact W.St.) and is distributed in executable format with an extensive user manual and a set of examples. The Web page mentioned in the header documents information about the package's capabilities, availability, new features, examples, a list server for distribution of news, etc.

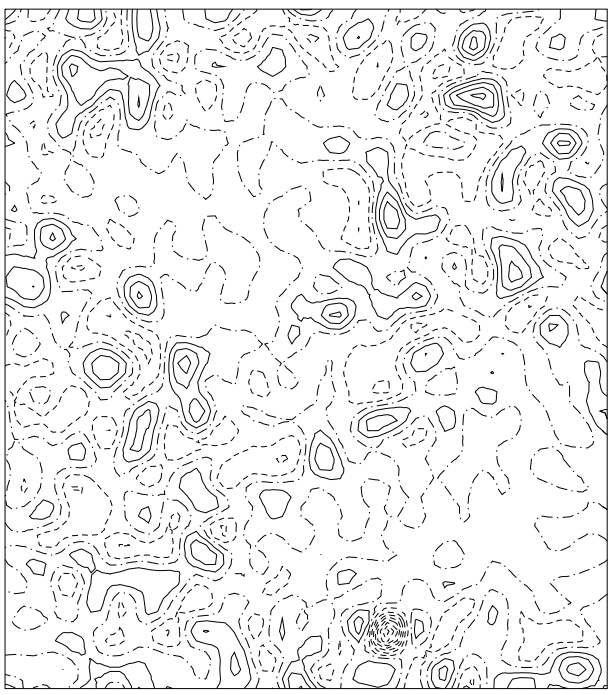

Figure 12: The same section as in Fig. 11 after 16 cycles of solvent flattening. The heavy atom and solvent areas have been flattened out resulting in an enhancement of the features within the molecule.

References

[1] Collaborative Computational Project, Number 4, "The CCP4 Suite: Programs for Protein Crystallography", Acta Cryst., Vol. D50, pp. 760-763, 1994.

[2] S.R. Hall, J.M. Stewart, and R.J. Munn, "XTAL: New Concepts in Program System Design", Acta Cryst., Vol. A36, pp. 979-989, 1980.

[3] J.M. Stewart, D.M. Collins, K.D. Watenpaugh, E. Prince and S.R. Hall, "Programs for phasing by entropy maximization as implemented in Xtal3.2: A crystallographic software system", Acta Cryst., Vol. D49, pp. 100-106, 1993.

[4] A.T. Brünger, X-PLOR, Version 3.1, A System for X-Ray Crystallography and NMR, Yale University Press, New Haven, CT, USA, 1992.

[5] D.E. McRee , "Practical Protein Crystallography", 386 pp., ISBN 0-12-486050-8, Academic Press, San Diego, 1993.

[6] W. Steigemann, "Die Entwicklung und Anwendung von Rechenverfahren und Rechenprogrammen zur Strukturanalyse von Proteinen am Beispiel des Trypsin-Trypsininhibitor Komplexes, des freien Inhibitors und der L-Asparaginase", Ph.D. Thesis, Technical University Munich, Germany, 1974.

[7] W. Steigemann, "Recent advances in the PROTEIN program system for the X-ray structure analysis of biological macromolecules", pp. 115-125 in "Crystallographic Computing 5: From Chemistry to Biology", Eds. D. Moras, A.D. Podjarny and J.C. Thiery, Oxford University Press, New York, 1991.

[8] W. Steigemann, "Density modification and manipulation in the PROTEIN program system", in "American Crystallographic Association, Annual Meeting", Vol. 27 (ISSN 0596-4221), Abstract 2m.5.B, p. 54, 1995.

[9] T.A. Jones, "A Graphics Model Building and Refinement System for Macromolecules", J. Appl. Cryst., Vol. 11, pp. 268-272, 1978.

[10] T.A. Jones, J.-Y. Zou, S.W. Cowan and M. Kjeldgaard, "Improved Methods for Building Protein Models in Electron Density Maps and Location of Errors in these Models", Acta Cryst., Vol. A47, pp. 110-119, 1991.

[11] D. Turk, "MAIN: A Computer Program for Macromolecular Crystallographers, now with Utilities You always Wanted", in "American Crystallographic Association, Annual Meeting", Vol. 27 (ISSN 0596-4221), Abstract 2m.6.B, p. 54, 1995.

[12] D.E. Tronrud, L.F. Ten Eyck and B.W. Matthews, "An efficient general-purpose least-squares refinement program for macromolecular structures", Acta Cryst., Vol. A43, pp. 489-501, 1987.

[13] W.A. Hendrickson, and J.H. Konnert, "Stereochemically Restrained Crystallographic Least-squares Refinement of Macromolecule Structures", pp. 43-57 in "Biomolecular Structure, Conformation, Function and Evolution", Ed. R. Srinivasan, Vol. I, Pergamon Press, New York, 1979.

[14] L.F. Ten Eyck, "Crystallographic Fast Fourier Transforms", Acta Cryst., Vol. A29, pp. 183-191, 1973.

[15] M.G. Rossmann (Editor), "The Molecular Replacement Method", Gordon and Breach, New York, 1972.

[16] M.G. Rossmann and D.M. Blow, "The Detection of Sub-Units Within the Crystallographic Asymmetric Unit", Acta Cryst., Vol. 15, pp. 24-31, 1962.

[17] W. Hoppe, "Die Faltmolekülmethode und ihre Anwendung in der röntgenographischen Konstitutionsanalyse von Bifforin ($C_{10}H_{20}O_4$)", Z. Elektrochemie, Vol. 61, pp. 1076-1083, 1957.

[18] P.M.D. Fitzgerald, "MERLOT, an integrated package of computer programs for the determination of crystal structures by molecular replacement", J. Appl. Cryst., Vol. 21, pp. 273-278, 1988.

[19] J. Navaza, "AMoRe: an Automated Package for Molecular Replacement", Acta Cryst., Vol. A50, pp. 157-163, 1994.

[20] A.T. Brünger, "Extension of Molecular Replacement: A New Search Strategy Based on Patterson Correlation Refinement", Acta Cryst., Vol. A46, pp. 46-57, 1990.

[21] W.L. DeLano and A.T. Brünger, "The Direct Rotation Function: Rotational Patterson correlation Search Applied to Molecular Replacement", Acta Cryst., Vol. D51, pp. 740-748, 1995.

[22] R. Huber, "The Programmed "Faltmolekül" Method", pp. 96-102 in "Crystallographic Computing", Ed. F. R. Ahmed, Munksgaard, Copenhagen, 1969.

[23] C.E. Nordman, "Vector Space Search and Refinement Procedures", pp. 29-38 in "Transactions of the American Crystallographic Association", Vol. 2, 1966.

[24] J. Sevcik, E.J. Dodson and G.G. Dodson, "Determination and Restrained Least-Squares Refinement of the Structures of Ribonuclease Sa and its Complex with 3'-Guanylic Acid at 1.8 A Resolution", Acta Cryst., Vol. B47, pp. 240-253, 1991.

[25] E.N. Baker, B.F. Anderson, A.J. Dobbs and E.J. Dodson, "Use of Iron Anomalous Scattering with Multiple Models and Data Sets to Identify and Refine a Weak Molecular Replacement Solution: Structure Analysis of Cytochrome c' from Two Bacterial Species", Acta Cryst., Vol. D51, pp. 282-289, 1995.

[26] P.M.D. Fitzgerald, "Molecular replacement", pp. 333-347 in "Crystallographic Computing 5: From Chemistry to Biology", Eds. D. Moras, A.D. Podjarny and J.C. Thiery, Oxford University Press, New York, 1991.

[27] R.A. Crowther and D.M. Blow, "A Method of Positioning a Known Molecule in an Unknown Crystal Structure", Acta Cryst., Vol. 23, pp. 544-548, 1967.

[28] A.D. Podjarny and B. Rees, "Density modification: theory and practice", pp. 361-372 in "Crystallographic Computing 5: From Chemistry to Biology", Eds. D. Moras, A.D. Podjarny and J.C. Thiery, Oxford University Press, New York, 1991.

[29] B.C. Wang, "Resolution of Phase Ambiguity in Macromolecular Crystallography", pp. 90-112, Methods in Enzymology, Vol. 115: "Diffraction Methods for Biological Macromolecules", Eds. H. Wyckoff, C.H.W. Hirs and S.N. Timasheff, Academic Press, New York, 1985.

[30] A.D. Podjarny, J.L. Sussman, T.N. Bhat, E.M. Westbrook, M. Harel, A. Yonath and M. Shoham, "Comparison of Different Density Modification Methods for Improving the Image of a Protein Map at High Resolution", Acta Cryst., Vol. A40 (suppl.), p. C14, 1984.

[31] A.G.W. Leslie, "A reciprocal-space method for calculating a molecular envelope using the algorithm of B.C. Wang", Acta Cryst., Vol. A43, pp. 134-136, 1987.

[32] G. Bricogne, "Geometric Sources of Redundancy in Intensity Data and Their Use for Phase Determination", Acta. Cryst., Vol. A30, pp. 395-405, 1974.

[33] G.A. Sim, "The distribution of phase angles for structures containing heavy atoms. II. A modification of the normal heavy-atom method for non-centrosymmetrical structures", Acta Cryst., Vol. 12, pp. 813-815, 1959.

[34] W.A. Hendrickson, and E.E. Lattman, "Representation of Phase Probability Distributions for Simplified Combination of Independent Phase Information", Acta Cryst., Vol. B26, pp. 136-143, 1970.