Users' Guide

The TNT Refinement Package

Dale E. Tronrud

This manual is consistent with Release 5-F of TNT. Detailed information about the package and methods used can be found in the following references. These papers can also be viewed at

http://www.uoxray.uoregon.edu/dale/

- Tronrud, D. E., Ten Eyck, L. F., & Matthews, B. W. (1987). An Efficient General-Purpose Least-Squares Refinement Program for Macromolecular Structures. *Acta Crystallogr A,* **43**, 489–501.

- Tronrud, D. E. (1992). Conjugate-Direction Minimization – An Improved Method for the Refinement of Macromolecules. *Acta Crystallogr A,* **48** (November), 912–916.

- Tronrud, D. E. (1996). Knowledge-Based B-Factor Restraints for the Refinement of Proteins. *J App Cryst,* **29** (2), 100–104.

- Tronrud, D. E. (1997). The TNT Refinement Package. in Macromolecular Crystallography, Part B, Eds Charlie Carter, and Robert Sweet, Volume **277** in Methods in Enzymology, pp 306-319.

- Tronrud, D. E. (1999). The Efficient Calculation of the Normal Matrix in Least-Squares Refinement of Macromolecular Structures. *Acta Crystallogr A,* **55**, 700–703.

Up-to-date information about TNT can be found at

http://www.uoxray.uoregon.edu/tnt/

Contents

Conventions iv

1 TNT Refinement **1**

2 Setting Up a Project **3**

Importing Your Model 3

Coordinate File Editing 4

Sequence File Editing 4

Control File Editing 5

Importing Structure Factor Files 5

Selecting a Test Set 6

Testing Your Files 7

Checking the Scale Factors 8

3 The TNT Control File **11**

General Data Statements 12

Crystallographic Data Statements 12

Stereochemistry Data Statements 13

Noncrystallographic Symmetry Data Statements 15

Refinement Data Statements 15

4 The TNT Sequence File **19**

Single Residue Molecules 19

Single Chain Models 20

Links to Other Asymmetric Units 22

Multiple Chain Models 23

The CHAIN Statement 23

5 Standard Geometry Definition **29**
The Geometry Statement 30
Geometry Libraries 35
IUPAC Labeling Conventions 38
Geometry Definition Example 39

6 Defining Noncrystallographic Symmetry **47**
Defining Structural Correspondences 48
Local Violation of ncs 49
Defining ncs Transformations 49
Noncrystallographic Constraints 51
Noncrystallographic Restraints 52

7 Special Problems in Structure Definition **55**
Fragmented Structures 55
Static Disorder 57
Atoms on Special Positions 58
Molecular Disorder 58

8 TNT Shell Commands **61**
Utility Commands 61
Command: TNT 61
Command: CLEANUP 63
Model Evaluation Commands 63
Command: GEOMETRY 63
Command: NCS 64
Command: RFACTOR 64
Command: DSCREEN 64
Command: DIFFCOR 64
Command: DIFFPLOT 65
Noncrystallographic Symmetry Commands 65
Command: GATHER 65
Command: SCATTER 66
Communicating with the Graphics Program 66
Command: MAKE_MAPS 66
Command: PEKPIK 66
Command: TO_PDB 67

Command: FROM_PDB 67
Structure Factor File Commands 67
Command: CORRECT 67
Command: HKL_SEGREGATE 68
Command: REPORT_HKL 68
Command: HISTOGRAM 68

9 Running Refinement **69**
Choosing Weights 71
Convergence Criteria 73

10 Examples **75**
Protein, All Atom Refinement 76
Protein, Rigid-Body Refinement 77
Protein, Special Geometry 78
Nucleic Acid, Noncrystallographic Symmetry 80
Protein, Rigid-Body Refinement of Two-Domain Molecules . . 82
Protein, Rigid-Body Refinement of Two Domain Molecules . . 84

Conventions

There are a number of conventions used in this document. One set of conventions use the typeface to indicate what kind of object is being described. These conventions are

Bold This is the name of a script, e.g. **tnt**.

`Teletype` This is a file name, e.g. `tnt/data/formfactor.dat`. This typeface is also used to display exact dialogs between the user and the computer. In such an example the text typed by the user is in bold.

First letter capitalized This is the name of a program, e.g. Rfactor.

Another series of conventions are involved in the specification of syntax. When describing a data statement or command there has to be a method to specify that something is optional or something else can be repeated. These conventions are

$< \ldots >$ Substitute a value. The text within the broken brackets describes the value. In the input line the broken brackets are not entered.

N(...) Repeat the contents of the parenthesizes zero or more times.

[...] The contents of the brackets are optional.

{ ... | ... } Choose one of the options separated by "|". There may be more than two options to choose from.

Chapter 1

TNT Refinement

The last step in the determination of a macromolecular structure is refinement. Refinement is the process whereby the model is altered to best match the facts known about the structure. Our knowledge about a structure will usually come from the diffraction of X-rays from a crystal and the facts of stereochemistry — the bond lengths, bond angles, etc. — expected for this type of molecule. Occasionally we will have other sources of information: experimental phases, similarity to a known structure, self similarity, or neutron diffraction. The final model should be consistent with everything known about the structure.

The work of refinement is divided between the computer and the crystallographer. The computer is very good at fine tuning a model to match the observations while the crystallographer is good at recognizing gross errors in the model and building entirely new solutions. The purpose of the TNT package of programs is to perform the automated optimization of the model and provide tools for use during the model rebuilding phase.

TNT does not attempt to correct gross errors in a model. It will not move atoms out of density to place them in "better" density nor will it add or delete atoms. TNT will alter the parameters of the model to improve its fit to the observations and it will provide clues as to the location of gross errors.

In TNT a model is stored in a coordinate file (whose named ends

in '`.cor`'). The structure of the model, the sequence and connectivity of all chains, is described in a sequence file ('`.seq`'). The details of the refinement — the identity of the restraints, their weights, and which parameters to vary and how to vary them — are stored in a control file ('`.tnt`'). Most of refinement is performed by editing the control file and executing the **tnt** command.

After a series of refinement cycles you should examine your model along with the current 2Fo-Fc and Fo-Fc maps (generated by **make_maps**). Each residue should be examined with particular attention paid to those residues which are most likely to be in error.

When the model is optimized to match the observations the observations are treated as restraints. While the overall agreement between the model and its restraints might be quite good, some individual atoms might agree badly. If the observations were measured without error and the type of model is sufficient to describe a molecule the model should be able to agree with all the observations at the same time. If some observations are not met by the model trouble is indicated. Portions of the model which do not fit the observations should be examined carefully on a graphics system.

TNT will provide you with lists of the atoms which are in greatest violation of the restraints. It will list the worst examples of bond lengths, bond angles, etc. with the **geometry** command. The **ncs** command will list the worst violations of the noncrystallographic symmetry, and the **dscreen** command will list the atoms which lie nearest to features in the difference map. These lists are indicators of the trouble spots in the model.

A separate type of error is the omission of atoms. Places where the model should contain atoms, but does not, can be found by looking for peaks in the Fo-Fc map. The **pekpik** command will generate marker atoms at the site of each of the highest 30 peaks in the difference map. These sites might be the location of alternative side chain conformations, the binding of solvent (either water or something else). While you should also examine each peak and try to interpret its cause there is no need to place atoms in every peak. Some peaks simply cannot be explained.

Chapter 2

Setting Up a Project

To begin refinement with TNT you have to reformat your model and your observations. Coordinates are stored in ATOMC format and structure factors are stored in HKL format. Both of these formats are described in Appendix A of the TNT Reference Manual.

Importing Your Model

In all likelyhood your coordinates will start out in PDB format. The **from_pdb** command will convert PDB files to TNT files. Because the PDB format contains information which TNT stores in several different files the output from **from_pdb** will be a `.cor` file, a `.seq` file and a `.tnt` file. Each file will have the same name as your PDB file. If the creation of the sequence or control file would overwrite an existing file a new version is not created. That is, if you already have a TNT control file **from_pdb** will not create another.

> Some programs write PDB files without including the cell constants. Cell constants are required for proper placement of the molecule in crystal's unit cell. You can provide the cell to TNT by either editing the PDB file or by creating a TNT control file containing a CELL statement (A prototypical TNT control file can be copied from

`$refroot/tnt/doc/prototype.tnt`). You should also complain to the author of the offending program.

Because TNT places more information in its files than the PDB file contains you will have to edit some of these file. Usually the coordinate file will be okay. The sequence file might require some editing and the control file will require a great deal. These issues are discussed below.

Coordinate File Editing

In TNT the type of an atom is specified in a completely separate field than the atom name. In PDB format, the atom name field specifies both pieces of information. That method is not specific enough for refinement. If you had an iron atom, does it have a +2 or +3 charge? The scattering is different but the information is not in the PDB file. If an element has only one possible ionization state **from_pdb** will produce the proper entry in your coordinate file. If there is more than one possible state you will have to edit your coordinate file to change the atom type.

Many programs do not write PDB format files correctly. One of the most common errors is to misplace the atom name of metal atoms. If your PDB format file contains this error it is quite possible, for example, that a nickel atom will be misidentified as nitrogen. You should inspect the metal atoms of your TNT coordinate file to ensure that the proper elemental type is written in the first column after the keyword `ATOMC`. If this mistake is made and refinement is attempted with the errant file the B factor of the affected atom will refine to a very low value.

Sequence File Editing

Because other formats of coordinate files do not include sufficient information about the connectivity of the molecule, **from_pdb** will not be able to produce a complete sequence file. If the disulfide bonds are listed in the PDB file your sequence file will contain them. Otherwise

you will have to edit the file to add them yourself. The PDB file contains no mechanism for describing unusual cross links, or breaks in the main chain. You will have to add those to the sequence file too.

Refinement itself cannot change the connectivity of a model, but during model building you can. If you change the basic form of your molecule you will have to edit your sequence file to reflect these changes.

If, to describe your molecule, you use a residue type or linkage type for which no geometry restraints are supplied with TNT you will have to define your own restraints. How this is done is described, and examples are given, in Chapter 5 on Page 29.

Control File Editing

The TNT control file will require a lot of work. The starting file created by **from_pdb** will contain only the most basic information. It will contain the cell constants, `INCLUDE` the sequence file, and a first guess for the weights to use in refinement. A complete control file must contain much more. You must edit this file to reflect the space group, supply the name of the structure factor file and resolution limits, and include any special statements required to define unusual chemical groups or situations.

Chapter 3 describes many of the statements you can use in this file.

As you progress with your refinement you will edit this file many times. As it stands, the prototype results in a stereochemically restrained refinement of the positions and temperature factors of your atoms, with torsion angles restraints turned off. (I don't believe in torsion angle restraints. They are better used to validate the final model.)

Importing Structure Factor Files

You will also have to get your observed structure factors into HKL format. This process is more difficult because there are so many different formats which the many data collection software programs produce.

There is no general purpose format conversion program for reformatting structure factor files supplied in TNT. You will have to write a program to convert whatever format you have into TNT's or find an existing program elsewhere. The format is described in Appendix A of the TNT Reference Manual.

These days most diffraction data sets are reduced with the programs Denzo or Mosfilm. The output of either can easily be converted to a `.mtz` format file. The CCP4 program Truncate can be used to convert from intensities to amplitudes and then the program mtz2various will write a basic TNT hkl format file.

There are two difficult points which remain. The asymmetric unit of data presented to TNT programs must conform to their expectations and the reflections must be supplied in the proper order. If some of your reflections fall outside of the box defined for your space group they will be ignored. No error will be issued.

Once you have your set of Fobs written in the TNT HKL format you can use the **correct** command to fix these last two problems.

When the conversion is complete the next step is to add remarks (REM cards) to the top of the file. Every HKL file should include remarks which describe the molecule, who collected the data, how it was collected, when it was collected, and other relevant information like cell constants and resolution limits.

Selecting a Test Set

If you want to calculate free R's during refinement you will have to select a test set at this time. Since I wish to ensure that the test set is never biased to the model, I separate the test and working sets into independant files using the **hkl_segregate** command. This command will randomly select 10% of your reflections and place them in one file while all the others are placed in another file. The name of the working set HKL file must be entered on the FO statement in your TNT control file.

You will need a script to calculate the free R. A prototype can be created from `$refroot/doc/r-free`. This file must be edited to replace

the token "???.hkl" with the name of your test set HKL file.

If your crystal is isomorphous to the crystal your model was originally refined against (i.e. You are solving a mutant or inhibitor structure) you cannot select a test set in the default fashion. Since your starting model has already been refined against a diffraction data set which is very similar to the new data set you *must* select for your test set exactly the same reflections used in the original refinement. Such a test set can be selected by giving the name of the old test set as the second argument to the **hkl_segregate** command. If the current crystal diffracts to a higher resolution than the original you will have to be crafty. If the original refinement was performed against all the data you *cannot* reserve a test set. The model is already biased toward all the reflections and any free R calculation attempted now would give an anonmolusly low free R.

Testing Your Files

Finally your newly reformatted data files must be tested. You should ensure that the programs properly understand your model. You should check the worst items of bad geometry with the **geometry** command and calculate the R-value of your model with the **rfactor** command. If you have wildly bad bond lengths or bond angles either your model is in error or the stereochemistry has been incorrectly defined. In either case the error must be corrected before refinement begins. It is quite important to check the bad contacts too. Any overlooked geometry restraints will result in inappropriate bad contacts being identified. Often forgotten disulfide bonds will be found in this fashion.

The check by **rfactor** involves some subtle checks as well as the obvious calculation of the R-value itself. Of course, the R-value must be reasonable. If your model was produced via MIR model building your R-value should be less than about 45%. A model produced by MR could have an R-value in the upper 50%'s and still be good. At this point it is too early to condemn such a model. A Molecular Modification model should be below 25% R-value unless large changes occurred during the modification.

In addition to the R-value some other values should be checked. The completeness of the observed data set should be consistent with your expectations based on the progress of data collection. If the completeness is too low or the completeness of the "matches" is not the same as that of the observed data (which it must be because the calculated structure factors are 100% complete) it is likely that you have the wrong asymmetric unit in reciprocal space. This error is corrected with the **correct** command.

If you have reserved a test set, and your model is truely unrefined, the starting working R and free R's should be roughtly equal. Some fluctuations are to be expected because of the small size of the test set, but if your test set contains at least 1000 reflections the free R should be no more than 2the working R. Of course, if your model was refined to an isomorphous crystal's diffraction pattern and your R's are equal you have a problem. This situation indicates that your test set is not a true test set — in such cases you must select as you test set exactly the same reflections as the original refinement.

Checking the Scale Factors

The scale factors also should be checked for reasonableness. Correct scaling is **important**! Bad scale factors cause errors in every structure factor, and every point in your map. The quality of the scale factors depends upon the resolution of the diffraction data. Sometimes, as the resolution drops, the scaling parameters go bad earlier and sometimes later.

You have to judge the parameters based on their values and to do that you need a basic understanding of the meaning of the scaling parameters. TNT uses two scale factor parameters not used by other packages. These parameters provide a simple model of the scattering of the bulk solvent. All four of TNT parameters are described below.

'B' is the number which must be added to each atom in the model to get the average B to match the observed structure factors. If B comes out the be 3 it is probably okay. If it is 27 it means that all the B's in your model have to be increased by 27, putting the overall average in

the 40's. If you believe that to be true, fine. If not you have to set B to a reasonable value, usually zero. At low resolutions (3Å or lower) the R-value is very insensitive to this parameter. It could assume almost any value and the R-value will come out the same. In these cases B should always be set to zero.

'Ksol' is the ratio of the average solvent density to the average protein density, electron density that is. Protein, being what it is, and water, being what *it* is, the ratio should be about 0.7. If your mother liquor is denser Ksol will be larger. It is a rare mother liquor that is denser than protein so Ksol shouldn't be larger than 1.0. It certainly should never be negative. Where there are no high resolution data the precision required of Ksol becomes less. 0.7, 0.8, 0.9, 1.0 are all about the same. When the data are lacking and the program can't come up with a reasonable number you simply set it to a value of your choice.

'Bsol' is the average temperature factor of the bulk solvent. It should be large and positive. It seems to hover in the neighborhood of 100 to 300. Sometimes it will come out to be very large ($\sim$ 1000). This usually occurs when you have not given enough low resolution data. The program chooses a solution which pushes all the solvent scattering into the unobserved portion of reciprocal space. The way to avoid this problem is to include the low resolution data.

Sometimes Bsol is very small. In most cases this is because there is no model for the ordered solvent in your molecule and the program is trying to model the scattering from these atoms with the disordered solvent parameters. In this situation you have to force Bsol to be about 150 or so to keep it from interfering with these ordered atoms.

'K' is the conversion factor to take the observed F's from whatever units they are measured in to electrons per unit cell, the units of the calculated F's. Its value cannot be predicted nor can you simply look at it and decided it correctness. The only check you have on it is that if the other parameters are correct it has to be too. However if K is larger than 10 it means that you have lost one digit of precision of your observed structure factors in the formatted HKL file. This means that you should multiply all your data by whatever factor is required to preserve the precision of your original data file before you write it to HKL format.

During refinement you want the individual temperature factors of the model to absorb the signal which causes the parameter B to be non-zero. The model cannot adjust to this signal if the signal has already been eliminated. B must be fixed with a value of zero in all refinement. This requirement is built into the **tnt** command.

If the B factor offset is too large (> 5) normal TNT refinement will not be able to adjust the individual B's sufficiently for the error to be corrected. When this occurs it is best to use the script `$tntbin/add_b` to add the value of the B scale factor to each individual B factor in the model. To make this correction type "**$tntbin/add_b init.cor 5.36**", with your value of B substituted. The corrected model with be written to `add_b_init.cor`. Rename this file to `init.cor` and proceed with refinement.

Additional restrictions must be placed on the scale factors if the data sets do not cover large areas of reciprocal space. The inner resolution limit should be set to the lowest resolution possible, say 20Å, for a good solvent model to be achieved. To properly determine all the scaling function parameters the outer resolution limit should be better than 2Å. This second criterion is often more difficult to achieve. If your crystals do not diffract to better than 2.5Å resolution you will have to fix B at zero even when calculating maps. This is done by adding a "`SET B 0.0`" statement to your control file.

If you do not have data better than 5Å you will have to fix Ksol at a reasonable number (say 0.8). This is done by including "`SET B 0.0 KSOL 0.8`".

If you are working with just a thin shell of reciprocal space you should set all parameters to zero except K. This is done with "`SET B 0.0 KSOL 0.0 BSOL 0.0`".

Chapter 3

The TNT Control File

The TNT control file contains all the information required to describe your project which is not properly included in either the coordinate file or the sequence file. Since this is a "none of the above" situation it is hard to describe in a simple fashion what goes in this file. Basically, if you need to provide some additional information to TNT you will enter the data in the control file.

You never enter the name of your control file when you run a TNT command. All TNT commands presume that your TNT control file will be located in the current directory and will be the only file which ends with '`.tnt`'

The control file is easy to use and usually not very large. Each line begins with a keyword which describes the data which will follow. If the data will not fit on one line you may continue on the next line by terminating the first with a '\' character (or alternatively a '-' which must be preceded by a space). You may add a comment by either starting the line with the keyword NOTE or placing the '#' or '!' character before the comment.

If some collection of data statements become too large to conveniently contain in the control file you may place them in a separate file. In this case you would use the INCLUDE keyword in the control file to cause the new file to be read. For example, if the new file were called `rigid_body.dat` you would add to your control file the line "`INCLUDE`

rigid_body.dat".

The next section describes many of statements which can be placed in your control file. The TNT Reference Manual describes other possibilities but those statements will be used less frequently.

General Data Statements

```
CELL <a> <b> <c> <Alpha> <Beta> <Gamma>
```

This statement gives the cell dimensions in Ångstrom and degrees.

```
OPTION {SET | CLEAR} N(<Option name>)
```

This statement either sets or clears the options named. Different programs have different options which can be turned on or off by this mechanism. By default all options are cleared when the program starts. The most common option is "VERBOSE".

```
WEIGHT N(<Restraint class> <Value>)
```

WEIGHT statements are used to specify a weight for each restraint type.

Crystallographic Data Statements

```
FO  <File name> [FORMAT {HKL | PACKED | MAP}]
```

This statement gives the name of the file which is the source of the observed structure factors. If a map file is given the observed structure factors are calculated by inverting the map.

```
RESOLUTION <Inner limit> <Outer limit>
RESOLUTION <Outer limit>
```

This statement specifies the inner and outer resolution limits of the crystallographic data in Ångstrom. The order of the resolution limits on the statement is unimportant. If the inner limit is not given it is presumed to be equal to infinity, which means that all data from F(000) to the outer limit will be used.

```
SET [K <Value>] [B <Value>] [KSOL <Value>] [BSOL <Value>]
    [B11 <Value>] [B12 <Value>] [B13 <Value>]
    [B22 <Value>] [B23 <Value>] [B33 <Value>]
```

This statement allows the user to set the value of any of the scaling function parameters. Any parameters which are not explicitly set are calculated by optimizing the match between the two data sets.

If any anisotropic scale factor is set to zero they all are. If KSOL is set to zero then BSOL is also set to zero.

A SET statement clears all information about previous SET statements and resets any scale factors which have already been determined.

Stereochemistry Data Statements

```
ASSUME RESIDUE_TYPE <Name> WHEN_CONTAINS N(<Name>)
```

This statement allows you to define a default residue type for any residue which contains all of the atoms listed, and no more. The puddle of water that is contained in most coordinate files presents a continuing maintenance problem in TNT's sequence file. Without the ASSUME statement you would have to edit the sequence file every time you added or removed solvent.

The following example will, I hope, clarify this statement's use. If your model has a great deal of DMSO, instead of creating individual RESIDUE statements for each one you could add to your control file the statement

ASSUME RESIDUE_TYPE DMSO WHEN_CONTAINS S C1 C2 O

Built into TNT is the assumption that a residue which contains only an atom named "OH" is of type "HOH". This is the PDB convention for water molecules.

```
EXCLUDE  [<Chain>|]<Residue>[:<Atom>] -
         [<Chain>|]<Residue>[:<Atom>]
```

This statement causes any close contacts between atoms in one group with any atom in the other group to be ignored when deciding what the bad contacts are. A group is defined by giving the chain name, residue name, and atom name of the atom whose contacts are to be ignored.

Wild cards may be placed in any position. The default residue and atom names are wild cards, but the default chain name is the "chain with no name". To specify an entire residue one gives the residue name. To specify a particular atom one gives the residue name and the atom name separated by a colon. To specify all atoms with the same name leave the residue name blank and provide a wild card ("*") as the chain name. Here are some examples.

```
EXCLUDE   143     167
EXCLUDE   67:O    SOL1:O65
EXCLUDE   *|:O    *|:C
```

In the first example all contacts between any atoms in residue 143 with atoms in residue 167 will be ignored. The second example states that if a bad contact is found to exist between atom O of residue 67 and atom O65 of residue SOL1 that contact will be ignored. In the third example the program will ignore all contacts between atoms named O and atoms named C anywhere in the structure.

Here is an example of a use of the EXCLUDE statement which uses the oligomer definition of Cro given as an example in the "TNT Sequence File" chapter (page 25).

```
EXCLUDE O| O|
EXCLUDE A| A|
EXCLUDE B| B|
EXCLUDE C| C|
```

These EXCLUDE statements will cause all non-bonded contacts within CRO chains to be ignored leaving only interchain contacts to be listed and refined.

Noncrystallographic Symmetry Data Statements

```
CLUSTER <Name> N([RESIDUES <Residue range>])
               CHAINS N(<Chain>)
```

The CLUSTER statement is used to define the chains which are related by noncrystallographic symmetry, and the portions of those chains for which the correspondence is valid.

The CHAINS option will continue reading chain names until the end of line is reached. Therefore all RESIDUES options must come before it.

Refinement Data Statements

```
COMBINE <Parameter code> N(<Atom code>)
```

This statement creates a subset of the structure that will be treated as a rigid body. The positional shifts of each atom in this class are combined so that the relative positions of the atoms within the group are unaffected by the shifts to be applied. Several rigid bodies can be refined simultaneously by entering several COMBINE statements. Each COMBINE statement generates a single rigid body.

The parameter code can be:

XYZ	Coordinates alone are affected.
B	Temperature factors alone are affected.
OCC	Fractional occupancies are affected.

The atom code can be either:

```
<Chain name> | <Residue name> : <Atom name>   -- or --
<Chain name> |  <Residue 1>   - <Residue 2>
```

In the first case a single atom is specified. If the residue name or atom name slots are left blank they are assumed to be a wild card. If the chain name is left blank it is assumed that the model contains a single chain.

The second form allows the selection of all atoms within a residue range. Both residue names must be given. If the chain name is left blank it is assumed that the model contains a single chain. A space is required before the hyphen ('−').

One may mix and match residue ranges and wild card specifications on the same COMBINE card.

If no atom code is given the parameters of the specified class for all atoms in all chains are combined into a single rigid body.

If non-positional parameters are combined, the shift applied to every atom is the average of the shift that would have been applied to each atom. To force all the atoms in a group to have the same temperature factor or occupancy while allowing the parameter to vary you would first set all the atoms equal to each other and then refine with a COMBINE B or COMBINE OCC along with a description of the group.

The behaviour of COMBINE statements can be modified by setting the MOSES option. By default, any atoms not mentioned on a COMBINE statement is left as a free atom. Mixing rigid bodies and free atoms usually does not work well. When the MOSES option is set all atoms not specifically assigned to a rigid body on a COMBINE statement will be placed in the group it is closest too.

Because of a limitation in the current implementation, COMBINE statements cannot be used along with curvatures. The steepest descent minimization method should be used with rigid body refinement. This is accomplished by giving the third parameter of the **tnt** command as 'sd'. The command would look like "**tnt 1 5 sd**".

`CONSTANT <Parameter code> N(<Atom code>)`

This statement causes the specified parameters to remain constant.

The qualifiers are as described for COMBINE except that one can specify all the parameters of the atoms (XYZ, B, and OCC) by typing "ALL". For example:

```
CONSTANT   OCC
CONSTANT   ALL   1 - 316
CONSTANT   XYZ   O|1:SG   B|1:SG
```

The first example fixes the occupancies of all atoms in the structure. The second example fixes all parameters for residues 1 through 316, traced through links, for the default chain. The last statement causes the positions atom SG of residue 1 of chain O and the same atom in chain B to be held constant.

`RANGE  <Parameter code> <Minimum value> <Maximum value>`

This statement sets a limit on the values a particular class of parameters may have. The parameter code must be either B or OCC. The statement need not be used as defaults are provided for the minimum and maximum values of B factors and occupancies. These defaults are 1.0 and 100.0 for B factors and 0.0 and 1.0 for occupancies. If a value is not specified on the RANGE statement the default is used. For example:

```
RANGE   OCC   0.5    1.0
RANGE     B     ,   40.0
RANGE     B   3.0     ,
```

In the first example occupancies are allowed to range only from 0.5 to 1.0. The second example allows thermal parameters to range from the default (1.0) to 40.0. The third example allows B's to range from 3.0 to the default upper limit (100.0).

Chapter 4

The TNT Sequence File

The TNT sequence file contains information about your model which other crystallographic packages either store in the coordinate file or simply assume from hints in the coordinate file. For example, a Protein Databank file contains the the amino acid type of each residue in your protein but does not explicitly state which residues are connected by peptide bonds, which are connected by other types of links, or which are not connected at all.

The sequence file contains the type of each residue and all the information required to describe the links between residues. There are only three statement types used for this purpose — CHAIN, LINK, and RESIDUE. Since models with a single chain are so common they have been treated as a special case.

Single Residue Molecules

While our interest is focused on the big molecules in our crystals, most of the molecules in the unit cell are small. They are the solvent molecules; mostly water molecules. Since there are a large number of these molecules to track, and they are created and deleted often while the model is refined, listing RESIDUE statements for each one is cumbersome. TNT has a mechanism where the residue type of a single

residue molecule can be deduced from the names of the atoms it contains. The deductive rules are entered on ASSUME RESIDUE_TYPE statements (see page 13).

For single residue molecules you have the choice of either putting the proper ASSUME RESIDUE_TYPE statements in your control file or building RESIDUE statements for each occurrence in your sequence file.

Single Chain Models

Models containing only a single chain are very simple to describe. The sequence file consists of a list of residues in the model, each with its own type and links. All of this information is provided on RESIDUE statements. They have the following form

```
RESIDUE <Residue name> <Residue type> -
                            N(<Residue name> <Link type>)
```

There will be one RESIDUE statement for each residue in the model. Following the keyword 'RESIDUE' is the name of the residue. This is followed by its type. Each residue must have a unique name

The last part of the RESIDUE card specifies the target and link type of any connections between this residue and any other residues.

In the following example residue "1", an isoleucine, is connected to residue "2" by a peptide bond (named PEPTIDE). Residue "2", threonine, is connected to residue "3" by a peptide bond. Residue "3", glycine, is connected to residue "4" by a peptide bond, and so on.

```
RESIDUE   1  ILE   2  PEPTIDE
RESIDUE   2  THR   3  PEPTIDE
RESIDUE   3  GLY   4  PEPTIDE
RESIDUE   4  THR   5  PEPTIDE
RESIDUE   5  CYS   6  PEPTIDE
```

Because residues are recognized by names and not numbers, "numbers" may be left out of the sequence, and additional residues may be inserted once the sequence has been numbered, as shown in the next example.

```
RESIDUE   50    GLY   51    PEPTIDE
RESIDUE   51    THR   53    PEPTIDE
RESIDUE   53    CYS   54    PEPTIDE
RESIDUE   54    THR   55A   PEPTIDE
RESIDUE   55A   VAL   55B   PEPTIDE
RESIDUE   55B   GLY   56    PEPTIDE
RESIDUE   56    VAL   57    PEPTIDE
```

A disulfide bond is shown in the following example. It connects residues 13 and 67.

```
RESIDUE   12   GLY   13   PEPTIDE
RESIDUE   13   CYS   14   PEPTIDE   67   SS
RESIDUE   14   ALA   15   PEPTIDE
```

There are two ways to terminate a polypeptide chain. The linkage type CTERM is used at a normal C-terminus. The extra oxygen of the carboxyl acid group is placed in a separate residue and the two final residues are linked with CTERM. Often a peptide model comes to an end not because of chemistry, but because the density becomes so weak that the chain can no longer be traced reliably. In this case there would not necessarily be an extra oxygen atom — the chain simply ends abruptly.

Because of the way the geometry of amino acids is defined all amino acids must be linked to something on their C-terminal end. If the model stops the last amino acid is linked to something else (anything else) with a BREAK link (I know, "BREAK link" sounds a little strange but you try to think all this stuff up.). The target of this link could be the amino acid on the far side of the gap or just a dummy residue created simply to be a target.

The final example shows how to include a break, a C-terminus, metal ions, and a heme group in the connectivity file.

```
RESIDUE   305     VAL     306     PEPTIDE
RESIDUE   306     ALA     NULL    BREAK
RESIDUE   NULL    NULL
RESIDUE   314     GLY     315     PEPTIDE
RESIDUE   315     VAL     316     PEPTIDE
RESIDUE   316     TRP     COOH    CTERM
RESIDUE   COOH    OXY
RESIDUE   ZINC    ZN++
RESIDUE   CAL1    CA++
RESIDUE   CAL2    CA++
RESIDUE   CAL3    CA++
RESIDUE   CAL4    CA++
RESIDUE   HEME    HEME
```

In this example, the protein contains four calcium ions, one zinc ion, and a heme. There is a break in the chain between residues "306" and "314". The residue "NULL" has been created simply to have something for "306" to be connected. The additional oxygen atom present at the C-terminus is treated as a separate group with the name "COOH" which is bonded to the C-terminal residue with the geometry specified for "CTERM".

Links to Other Asymmetric Units

In certain rare cases a chemical link between two residues cannot be specified on a RESIDUE statement. The links defined on a RESIDUE statement must connect two residues within the same polypeptide. A link to a symmetry related molecule can not be defined there.

The LINK statement was created to define these types of links. The form of the statement is quite simple:

```
LINK <Residue name> <Target name> <Link type>
```

The statement merely contains the name of the two residues and the type of link between them. You do not have to specify the symmetry operator, TNT will determine that by itself.

What would the LINK statement look like if we had a protein where residue 66 was linked by a disulfide bridge to its image across a two-fold? Simply "LINK 66 66 SS".

Multiple Chain Models

When we have multiple chains in a model we cannot identify a particular residue without also stating its chain. In essence the residue name must be extended to include the chain name. This is done by combining the chain and residue names by placing a '|' between them. Residue 123 of chain A becomes A|123.

Sometimes a crystal will contain two molecules with the same chemical structure. When this happens we would say that both molecules are of the same type. In TNT each chain has a type. The chain's type defines its amino acid sequence. The RESIDUE statements in the sequence file define the connectivity of a chain type and the type of each chain is defined on the CHAIN statement.

The CHAIN Statement

The information contained on the chain level includes the name and type of each chain in the structure, and the covalent links that connect it to other chains. All of this information is defined with the CHAIN statement. The covalent links may connect the chain to a symmetry image of another chain as well as to a direct image.

A CHAIN statement must be entered for each chain. The form of the CHAIN statement is:

```
CHAIN <Chain name> <Chain type>  N(<Chain link>)

<Chain link> :==
 <Residue name> <Chain name>|<Residue name> <Link type>
```

The first pair of tokens after the keyword gives the name of the chain and its type. The only information known about the chain type

is its sequence. This information is presented on a series of RESIDUE statements, each of which mentions that chain type.

Each CHAIN statement allows the definition of links to other chains in the structure. The link is defined by first giving the name of the source residue. Then the chain and residue name of the target of the link is specified in the standard fashion by separating the two names with a "|". Finally, the link's type is given. This sequence of information may be repeated as many times as necessary to define all the links.

TNT will assume that each link connects the residue to the nearest symmetry image of the target. It is not required that the particular symmetry operator be specified.

If a chain has a link to a symmetry image of itself that link should be defined on the CHAIN statement as though the connection was to another chain. Bonds across symmetry axes cannot be defined on RESIDUE statements.

Examples of Chain Definitions

Let us consider an oligomer of composition $\alpha_3\beta_2\gamma_1$ which has no interchain connections. This oligomer has six different chains (which I will call A1, A2, A3, B1, B2, C for obvious reasons) of three different types and will require six CHAIN statements. Because there are no links between chains the CHAIN statements are very simple.

```
CHAIN   A1   ALPHA
CHAIN   A2   ALPHA
CHAIN   A3   ALPHA
CHAIN   B1   BETA
CHAIN   B2   BETA
CHAIN   C    GAMMA
```

To complete the definition of this structure one would also include the RESIDUE statements defining the sequence for the ALPHA, BETA, and GAMMA chain types.

If we consider the Insulin tetramer the example becomes a little more complicated.

```
CHAIN   A1   A      7 B1|7 SS      20 B1|19 SS
CHAIN   A2   A      7 B2|7 SS      20 B2|19 SS
CHAIN   A3   A      7 B3|7 SS      20 B3|19 SS
CHAIN   A4   A      7 B4|7 SS      20 B4|19 SS
CHAIN   B1   B
CHAIN   B2   B
CHAIN   B3   B
CHAIN   B4   B
```

Here we have four A chains and four B chains. The A chains are linked to each corresponding B chain by two disulfide bonds (link type "SS"). The first line reads: The chain named A1 is of type A and its residue 7 is linked the residue 7 of chain B1 by a SS bond and its residue 20 is linked to residue 19 of chain B1 by a SS bond. Each B chain has an internal disulfide bond as well. This link would be defined on the RESIDUE statements that define the chain type B.

Multichain Model Example

To show further how one might handle oligomeric proteins I have a larger example of structure definition. It is taken from the Cro structure. Cro crystallizes as a tetramer of chemically identical monomers related to each other by non-crystallographic symmetry. First the CHAIN statements are given. They are very simple because there are no interchain links and all the monomers are of the same type. Next I have given the first few RESIDUE statements defining the amino acid sequence of the chain type CRO. The sequence need only be given once because even though there are four chains (O, A, B, and C) there is only one chain type (CRO).

```
CHAIN  O  CRO
CHAIN  A  CRO
CHAIN  B  CRO
CHAIN  C  CRO

RESIDUE CRO|1    MET   2    PEPT
RESIDUE CRO|2    GLU   3    PEPT
```

```
RESIDUE CRO|3    GLN   4    PEPT
RESIDUE CRO|4    ARG   5    PEPT
RESIDUE CRO|5    ILE   6    PEPT

ATOMC N -23.5711  -21.5563  -17.6677  47.89 1.00 N    1 O
ATOMC C -24.9299  -22.0883  -17.5977  28.88 1.00 CA   1 O

ATOMC N -31.6302  -17.6350  -27.8797  28.57 1.00 N    1 A
ATOMC C -31.4544  -19.0311  -27.6180  29.61 1.00 CA   1 A

ATOMC N -28.8195  -34.1503  -19.6577  44.24 1.00 N    1 B
ATOMC C -28.6735  -32.8969  -20.4065  59.62 1.00 CA   1 B

ATOMC N -39.9969  -26.5889  -24.4465  44.30 1.00 N    1 C
ATOMC C -38.9848  -25.8469  -23.6391  34.11 1.00 CA   1 C
```

Nucleic Acid Structure Example

The sequence for DNA and RNA is defined in a manner very similar to that of proteins. This is understandable because they both are polymers. The names of the bases are ADE, GUA, THY, CYT, URA. The link between the bases is SUGPHOS for RNA and dSUGPHOS for DNA. A special link is required on both ends of a nucleic acid chain. On the 5′ end we need 5'END for RNA and d5'END for DNA and on the 3′ end we need 3'END and d3'END. In proteins the final link is to a residue which contains only the final carbonyl oxygen. In a nucleic acid the final residue is empty. Using these rules the sequence file for Dickerson's dodecamer is

```
CHAIN A DNA
CHAIN B DNA
RESIDUE DNA|1        CYT   2 d5'END
RESIDUE DNA|2        GUA   3 dSUGPHOS
RESIDUE DNA|3        CYT   4 dSUGPHOS
RESIDUE DNA|4        GUA   5 dSUGPHOS
RESIDUE DNA|5        ADE   6 dSUGPHOS
```

```
RESIDUE DNA|6        ADE   7 dSUGPHOS
RESIDUE DNA|7        THY   8 dSUGPHOS
RESIDUE DNA|8        THY   9 dSUGPHOS
RESIDUE DNA|9        CYT  10 dSUGPHOS
RESIDUE DNA|10       GUA  11 dSUGPHOS
RESIDUE DNA|11       CYT  12 dSUGPHOS
RESIDUE DNA|12       GUA  13 d3'END
RESIDUE DNA|13       DUMMY
```

Chapter 5

Standard Geometry Definition

The method of standard geometry definition used in TNT is very easy to understand and to modify. There are eight geometrical relationships which can be set to ideal values for a standard group — bond length, bond angle, torsion angle, pseudorotation within 5 membered rings, planarity around trigonal atoms, general planarity, B factor correlation, and chirality of an atom. Chirality is included only to warn the user when a group has the wrong handedness; it is not considered a restraint and does not affect the shifts of the atoms.

Geometry restraints are defined for each type of residue and each type of link between residues. For example, there are geometry restraints associated with the residue type THR (threonine). These would include the bond length between the atoms CB and OG1. Other restraints are associated with the PEPTIDE link. These include the bond length between the atom C of the previous residue and the atom N of the following residue.

Defining the standard geometry of a molecule consists of simply defining the geometric restraints for each residue type and linkage type in your model. Most of this work has been done for you. The libraries distributed with TNT are described on page 35. If you have something unusual in your model you should read the example on page 39.

One of the best things about TNT is that a prosthetic group, such as a heme molecule, is treated just like any other standard group. It is very simple to define the standard values for such groups. In addition, atoms for which there are no restraints such as metal ions or water molecules, do not appear in the geometry declarations at all.

The Geometry Statement

```
GEOMETRY <Name> {BOND | ANGLE | TORSION | PSEUDO |
                 TRIGONAL | PLANE | BCORREL | CHIRAL} -
         <Value>  <Sigma>   N(<Atom name>)
```

The GEOMETRY statement is used to supply an ideal value for some type of restraint. The restraint is identified by the residue type and the names of the involved atoms. The order of the atom names is usually significant.

The syntax of the GEOMETRY statement is the same regardless of which type of restraint is being defined. First is the word GEOMETRY. This is followed by the name of the object being defined (e.g. ALA, GLU, PEPTIDE). This object can be either a residue type or a linkage type. (ALA and GLU are both residue types where PEPTIDE is a linkage type.) Next is the name of the type of restraint being defined (either BOND, ANGLE, TORSION, PSEUDO, TRIGONAL, PLANE, BCORREL, CHIRAL). The next number given is, in all cases except that of a PLANE, the standard value for this restraint. (With a PLANE restraint this number is the number of atoms in the plane and the standard value is assumed zero.) The second number is the standard deviation from ideality as seen in small molecule structures. Last is the list of the names of the atoms involved in this restraint.

When defining a geometry restraint in a link there is always the question of which residue each atom belongs to. When the bond between the atom C and atom N in the PEPTIDE link is defined there is a problem in that there is an atom named C in both residues, as well as two atoms named N. To remove this ambiguity we recognize that each

link has a source residue and a target. In a PEPTIDE link between residues 10 and 11 the source of the link is 10 while the target is 11.

In the geometry restraint an atom from the source residue has the prefix '−' added to its name, and an atom from the target residue has a '+' added. The default is '−', therefore generally one refers to the source residue by typing the atom names directly, and to the target residue by placing a '+' before the atom name.

Here are some examples of geometry declarations broken down by restraint type.

BOND

```
GEOMETRY PEPTIDE    BOND    1.45    0.02    N, CA
GEOMETRY PEPTIDE    BOND    1.52    0.02    CA, C
GEOMETRY PEPTIDE    BOND    1.33    0.02    C, +N
GEOMETRY CTERM      BOND    1.25    0.02    C, +OH
GEOMETRY SS         BOND    2.036   0.03    SG, +SG
GEOMETRY CYS        BOND    1.54    0.02    CA, CB
GEOMETRY CYS        BOND    1.43    0.02    CB, SG
```

The first word after "GEOMETRY" is the name of the standard group being defined. The first number is the ideal bond length, and the second number is the standard deviation from ideality seen in small molecule structures. Last come the names of the two bonded atoms.

Note that PEPTIDE, CTERM, and SS are kinds of links while CYS is a part of the declaration of a cystine. In the PEPTIDE group the N, CA, and C refer to atoms in the preceding residue, and the +N refers to the N atom of the next residue.

ANGLE

```
GEOMETRY PEPTIDE    ANGLE    112      3    N, CA, C
GEOMETRY PEPTIDE    ANGLE    121.2    3    CA, C, O
GEOMETRY PEPTIDE    ANGLE    115.6    3    CA, C, +N
GEOMETRY CTERM      ANGLE    118      3    CA, C, O
```

```
GEOMETRY CTERM      ANGLE    118      3    CA, C, +OH
GEOMETRY SS    ANGLE    104.5  3    +CB, +SG, SG
GEOMETRY ALA   ANGLE    112    3    N, CA, CB
GEOMETRY ALA   ANGLE    111    3    C, CA, CB
```

Bond angles are defined in the same way as bond lengths. The ideal bond angle and standard deviation are both in degrees. The three atoms involved are listed in the order in which they are bonded to each other (i.e. the middle atom in the list is the atom at the vertex of the angle).

TORSION

```
GEOMETRY PEPTIDE    TORSION   2160   30    N, CA, C, +N
GEOMETRY PEPTIDE    TORSION   3060   20    C, +N, +CA, +C
GEOMETRY PEPTIDE    TORSION   2180   10    CA, C, +N, +CA
GEOMETRY SS     TORSION    2090    10    CB, SG, +SG, +CB
GEOMETRY CYS    TORSION    3060    15    N, CA, CB, SG
GEOMETRY MET    TORSION    3060    15    N, CA, CB, CG
GEOMETRY MET    TORSION    3060    15    CA, CB, CG, SD
GEOMETRY MET    TORSION    3060    15    CB, CG, SD, CE
```

While most aspects of these statements should be clear by now, the first four-digit number is special in that it actually contains two pieces of information. The thousands digit gives the number of evenly spaced minima in the 360 degree range of possible values. The three digits following this digit (but including the sign of the number) is the value of one of the minima measured in degrees.

For example, the first torsion angle defined uses the number 2160. This means that there are 2 minima and one of them is at 160 degrees. The next number on the line is the standard deviation in degrees. It is followed by a list, in bonding order, of the four atoms involved.

PSEUDOROTATION

```
GEOMETRY PRO    PSEUDO    2000    20    CA, CB, CG, CD, N
```

The pucker of a 5 membered ring is described by a pseudorotation angle (Altona, C., Sundaralingam, M., JACS (1972). **94**, 8205-8212). Its standard value is encoded just like that of a torsion angle. There is a multiplicity and a particular value. The line ends with a list of the five atoms of the ring, listed as they occur in the ring. The first atom in the list determines the origin against which the phase shift of the pseudorotation angle is measured.

TRIGONAL

```
GEOMETRY CTERM    TRIGONAL  0  0.02    C, CA, O, +OH
GEOMETRY ASP      TRIGONAL  0  0.02    CG, CB, OD1, OD2
GEOMETRY ASN      TRIGONAL  0  0.02    CG, CB, OD1, ND2
```

This type of restraint is just a subset of the planarity restraint, and is designed to be used for planarity around a trigonal carbon. The first number in the list is the standard value for the r.m.s. deviation from the plane, which should always be zero.

PLANE

```
GEOMETRY PEPTIDE  PLANE   5  0.02    C, CA, O, +N, +CA
GEOMETRY ARG      PLANE   5  0.02    CD, NE, CZ, NH1, NH2
GEOMETRY HIS      PLANE   6  0.02    CB,CG,ND1,CE1,CD2,NE2
```

The first number is the number of coplanar atoms. The next number is the standard deviation of the r.m.s. deviation from the plane in Ångstroms, followed by a list of the coplanar atoms. The order of the names is unimportant.

BCORREL

```
GEOMETRY PEPTIDE  BCORREL    3.70   7.44     CA   C
GEOMETRY ARG      BCORREL   10.59  16.52     CB   CG
GEOMETRY HIS      BCORREL    3.37   3.82     CB   CG
```

The first number is the average increase in temperature factor the second atom shows relative to the first. The second number is the standard deviation for this observation.

CHIRAL

```
GEOMETRY CYS    CHIRAL    1    1    CA, N, CB, C
GEOMETRY THR    CHIRAL    1    1    CB, CA, CG2, OG1
GEOMETRY ILE    CHIRAL    1    1    CB, CA, CG2, CG1
```

The chiral atom is named first in the list, followed by the names of three of the atoms bound to it. The order of the final three names specifies the chirality of the first. The rule is:

- Identify the atom of lowest priority (as defined by the IUPAC). This is usually a hydrogen atom. If you don't know the priority simply pick one of the atoms.
- Orientate, in your mind or on the graphics system, the central atom so that the atom you picked is pointing away from you.
- List the three other atoms in clockwise order.

If the priority of the three atoms as listed on the GEOMETRY statement is (1,2,3) or a permutation the central atom has an R configuration. If the order is (3,2,1) then the configuration is S.

Geometry Libraries

Several standard geometry library files are distributed with TNT. Different chemical groups are defined in different files. To use one of these libraries simply INCLUDE it when you run the program Geometry. All of them are stored in the directory `$tntdata`. `tntgeo_v010.dat` is INCLUDEd in all scripts supplied with TNT.

- `tntgeo_v010.dat`

  This file is a container file supplied for convenient use. It combines together a number of files which contain the standard geometry for many types of chemical groups. The values in these files were entered over 15 years ago and, while complete, are not completely current. These are the default values used in TNT refinement because of their completeness.

  The individual components of `tntgeo_v010.dat` are:

  - `protgeo.dat`

    This library contains the geometry restraints for the twenty amino acids, the peptide bond (PEPTIDE), the C-terminus (CTERM), and the disulfide bond (SS). The amino acids are identified by the usual three letter codes. To maintain compatibility with other programs the amino acid proline can be identified either as PRO or CPR. (Often CPR is used to identify a proline with a cis peptide bond. TNT treats an amino acid the same regardless of the conformation of the peptide so this distinction is not necessary.)

  - `bcorrel.dat`

    This file contains the B correlation restraints for the groups defined in `protgeo.dat`. When you wish to restrain the thermal parameters of your model because of the lack of high resolution data you should INCLUDE this file in addition to `protgeo.dat`. I have yet to create a corresponding file for the other geometry libraries, due to the lack of high resolution models from which the proper restraints can be derived.

- `nuclgeo.dat`

  This file contains the geometry restraints for DNA and RNA. The five bases are identified by three letter codes. The sugar-phosphate backbone is linked with SUGPHOS in RNA and DSUGPHOS in DNA. The first base (at the 5′ end) must be linked to the second via a 5'END or D5'END linkage. At the 3′ end the final base must be linked to an empty residue via a 3'END or d3'END linkage. The atom names used in this library are those of the IUPAC, not those of the PDB. The atoms in the sugar rings are identified with primes ('), not stars (*).

- `cofactor_geo.dat`

  This library contains the geometry restraints for ATP, ADP, AMP, CMP (i.e. cyclic AMP), dUMP, NAD, and NDP (i.e. NADPH). The atom naming convention is that of the PDB. The atoms in the ribose sugars are identified with stars (*). The PDB has chosen very strange names for the atoms in these groups, please read their documentation when attempting to use this library.

- `othergeo.dat`

  This library contains geometry restraints for a number of small compounds. These include

  * BENZENE Benzene
  * BME Reduced β-Mercaptoethanol. Use SS to link two BME residues or a CYS residue to a BME.
  * CBZ The N-terminal blocking group carbobenzoxy. Use CBZLINK to link a CBZ residue to the N-terminal of a peptide.
  * DELTA Lyscine residue with all atoms past CD removed.
  * DMSO Dimethyl Sulfoxide
  * DTT Self-oxidized DTT
  * DTT2 One half of a DTT. This residue type is used when a DTT residue sits on a crystallographic two-fold. Use the link DTTLINK to connect the DTT2 to itself.

    * EPSILON Lyscine residue with all atoms past CE removed.
    * GAMMA Lyscine residue with all atoms past CG removed.
    * HEDS Two BME residues linked together. The atoms are named using the PDB convention.
    * MSE Seleomethionine
    * SO4 Sulfate (-2)

  – `chlorgeo.dat`

    This library contains the geometry definitions for bacteriochlorophyll-a.

- `heme_co_geo.dat`

  This library contains the geometry definitions for carbomethoxyheme (COHM).

- `csdx_protgeo.dat`

  A new search of the Cambridge Structure Database was performed by Engh and Huber (Engh, R.A., Huber, R. Acta Cryst (1991) A47, 392-400). The results of this survey have been used to construct a new geometry library for proteins. The restraints are consistent with the CSDX library in the XPLOR refinement package. However it is not consistent with the nucleic acid, and co-factor libraries described above. The result of mixing the two sources of restraint information is difficult to predict. If you have a simple protein this library can be used with no fear of problem.

  Since the choice of restraint library will cause slight differences in the resulting model you should note the name of the library in your Protein Databank submission and in your paper.

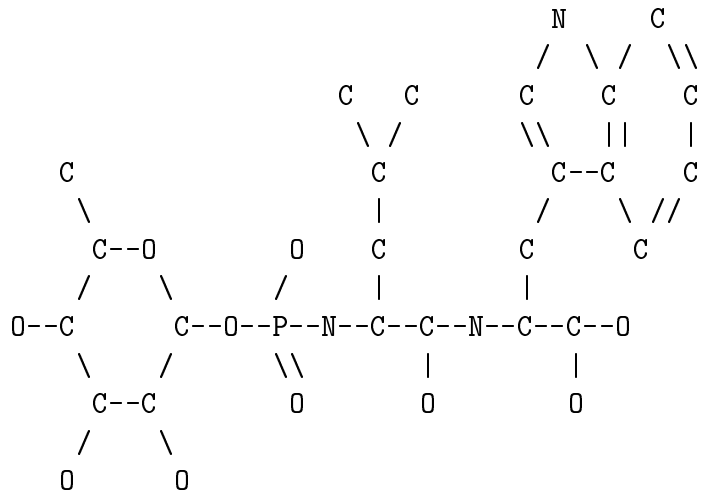

Figure 5.1: Unprocessed Phosphoramidon

IUPAC Labeling Conventions

The IUPAC has defined the labels for certain atoms in a couple amino acids in terms of an artificial chirality. These residue are valine and leucine. The file `iupacgeo.dat` contains the definitions of these "chiral centers" so that an inversion of the side chain of one of these residues can be monitored. If a valine or leucine beta carbon inverts the atom labels should be switched to keep the PDB people happy. This library also contains the definition of the appropriate "chi" angles which the IUPAC uses to define the atom labels in ASP, GLU, PHE, and TYR. If the "chi" angle is greater than 90 degrees for one of these amino acids the labels should be exchanged. Since these restraints were designed to identify IUPAC convention matching they do not reflect the proper standard values. These torsion angles cannot be used in refinement.

Geometry Definition Example

Even though most of the geometry restraints you will ever need are supplied with TNT you may encounter the need to define the restraints for a group yourself. If you are working with enzyme-inhibitor complexes you will certainly have to. However, even in the case of a simple protein you may encounter the binding of an unusual solvent molecule or discover some strange chemistry.

The definition of stereochemistry restraints requires that you analyze the molecule to determine which parts can be described using the given libraries. Then restraints for the unique parts must be be created. These steps are illustrated with the following example.

Figure 5.1 shows the structure of the phosphoramidon inhibitor of Thermolysin. This is a natural product which is a reasonably good inhibitor. We will step through the process of defining the standard geometry of this rather strange compound.

The first step when defining a new structure or group is to see if one can break it into smaller, known groups. (Because we often deal with proteins, the natural units of structure are referred to as "residues", but it should be understood that a "residue" can be any collection of atoms that the heart desires.) Examining the above structure one will note that the atoms following the phosphorus (on the right) make up a leucine and a tryptophan linked by a normal peptide bond and ending in a normal C-terminus as found in proteins. Because these groups are already in the standard geometry library we will split each of them into separate residues. In proteins the C-terminus is handled by putting the extra oxygen in a separate residue, whose type is unimportant because it contains no restraints. The type of the link between the last amino acid and the residue containing the final oxygen is called CTERM. The names of these three residues — the leucine, the tryptophan, and the terminal oxygen — can be anything and I choose PEP1, PEP2, and MYEND.

The group of atoms to the left can be split several different ways or left as a single residue. It seems reasonable to keep the rhamnose sugar in one unit but what should be done with the pseudo-phosphate? It is

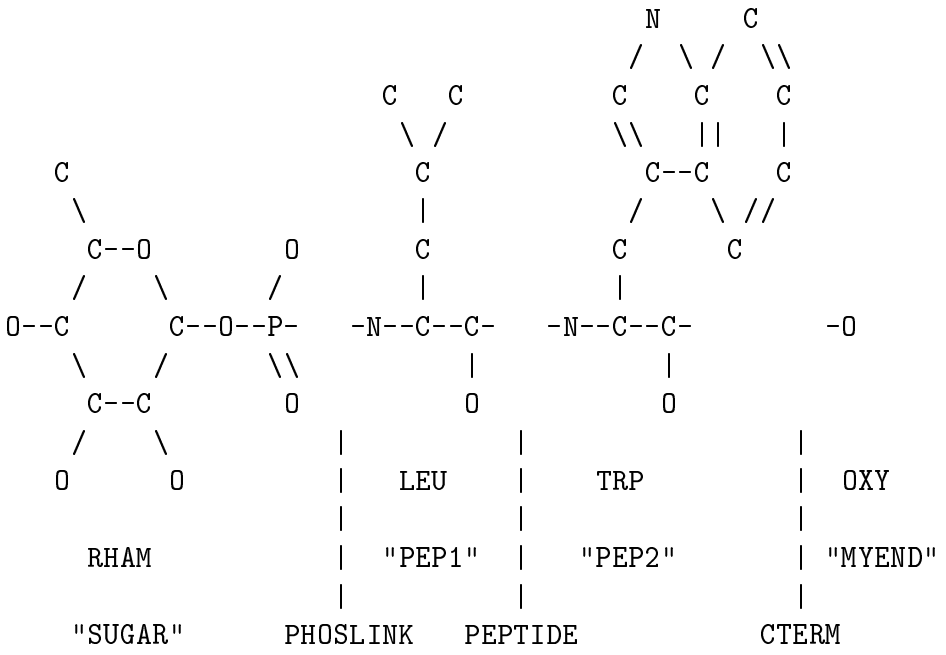

Figure 5.2: Phosphoramidon with residues and linkages named

a bad idea to make groups that are less than three atoms wide because no geometry restraint may span more than two residues and if a group were two atoms wide it would not be possible to define the torsion angles that would stretch from the preceding residue through the little residue and into the following residue. Therefore I have chosen to place the phosphorus and its two oxygen atoms into the same residue with the rhamnose sugar. I have named this residue SUGAR and named its type RHAM. Its link to the leucine (PEP1) will be called PHOSLINK.

In Figure 5.2 I have redrawn the inhibitor after splitting it in the manner I discussed above and I have labeled the parts. The names in quotes are the residue names and must be unique within the chain. These names are included in the list of atomic coordinates (i.e. on the ATOMC statements) and tell the program that a given atom belongs in the residue that is specified. The names under the vertical lines are the

names of the linkage types that connect the residues at these specified points. The other names specify the type of each residue. These names are included on the GEOMETRY statements that specify the geometry restraints of these groups.

With the structure now broken into residues and every part and link named we can write the RESIDUE statements that define to the program the type of each residue and the type of connection each residue makes.

```
RESIDUE   SUGAR   RHAM   PEP1     PHOSLINK
RESIDUE   PEP1    LEU    PEP2     PEPTIDE
RESIDUE   PEP2    TRP    MYEND    CTERM
RESIDUE   MYEND   OXY
```

Now that the residues of the inhibitor have been defined each atom must be given a name. The atom name must be unique within that residue but other residues in the structure may have atoms with the same name (e.g. CA). Because LEU, TRP, and CTERM are already defined in the standard geometry library for proteins, the atom names in those residues must conform to the same standard as the library. The standard geometry library uses the international convention for naming atoms within amino acids.

The residue type RHAM is not known to the library and we can choose any naming convention we wish. I choose to name the sugar ring's atoms in the normal fashion of numbering the atoms in the carbon chain starting from the most oxidized atom. Each oxygen was given the same number as its carbon. The phosphorus atom can easily be called P. The two oxygen atoms attached to P cannot be called O1 and O2 because those names are in use so I decided to call them PO1 and PO2. These are not the greatest names because some programs (but none in this package) determine the atom type by looking at the first letter of the atom name and these atoms would be improperly identified as phosphorus.

In Figure 5.3 I show the inhibitor with all the atoms given their names. These names must be used whenever an atom is referred to by the data given to a program. These names will be on the ATOMC

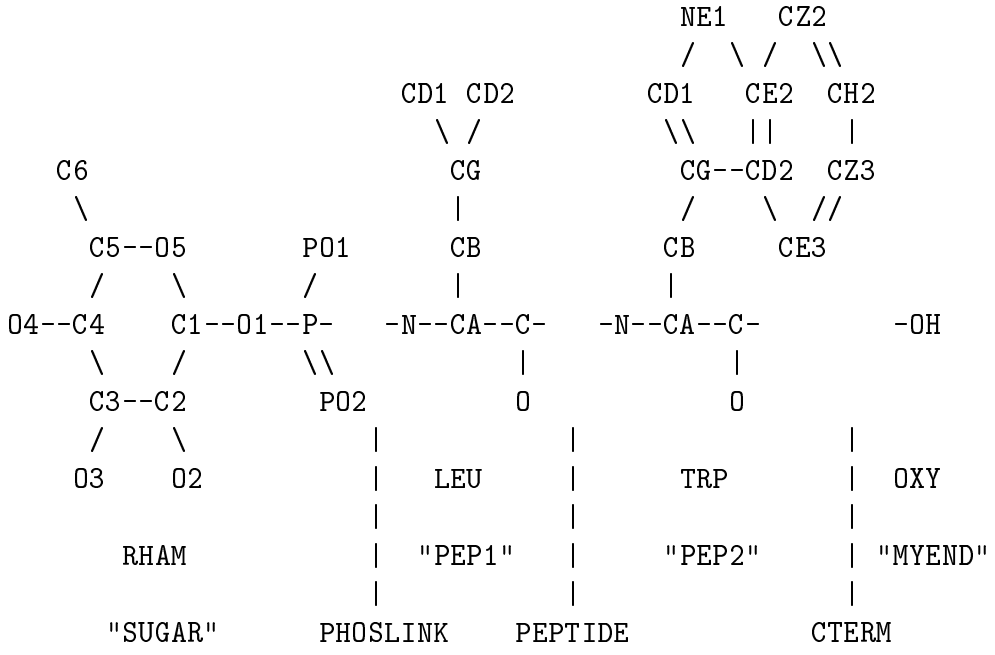

Figure 5.3: Phosphoramidon with atom names

statements defining the coordinates of these atoms and will also be used on the GEOMETRY statements that define the restraints to be applied.

Now we must look at the residue types and linkage types and determine which still need to be defined and which we are finished with. LEU, TRP, PEPTIDE, and CTERM are in the protein library and need not be worried about any more. OXY contains a single atom and does not need any restraints. Therefore it does not need to be worried about either. All that need be defined are RHAM and PHOSLINK.

Here we must come to grips with the difference between a residue type and a linkage type. A residue type contains the definitions of a collection of geometry restraints that involve only atoms within a single residue. A linkage type contains the definitions of restraints that span two residues but may also contain restraints that involve only atoms within the first residue of the link or only atoms within the second residue. Therefore restraints must be defined in the linkage type if they span two residues, but many restraints can be placed in either the residue type *or* the linkage type. (If a restraint was placed in both it would end up with twice the weight). The decision as to where to specify the restraint is arbitrary, but some rules of convenience can be formulated.

When defining the geometry of a polymer composed of different kinds of monomers some restraints are common to all monomer types (e.g. within the backbone) and some restraints are unique to their particular type of monomer (e.g. the side chains). A good rule is to place only the unique restraints in the residue type and to place the general restraints in the linkage type, which is the same for all monomers. Consider a protein: each amino acid type has a CA to C bond restraint that is independent of the type of amino acid while the NE1 to CE2 bond restraint is unique to tryptophan. In these examples the NE1 to CE2 bond would be defined in the TRP residue type, while the CA to C bond would be defined in the PEPTIDE linkage type even though both atoms are in the same residue. If this rule were not followed and the CA to C bond was defined in each amino acid type everything would work fine but this bond length would have to be defined separately for each amino acid, twenty different times.

Unfortunately this rule does not help us much with our present problem. We do not have a polymer composed of rhamnose sugars and strange phosphates and cannot define what would be a general linkage in such a polymer. However, rhamnose might come up in some future structure determination (It is a common carbohydrate) and it might be nice to have its standard geometry ready should the need arise. Therefore I decided to place geometry restraints that deal only with the rhamnose sugar into the RHAM residue type and to place all the restraints dealing with the phosphate and the link to PEP1 into PHOSLINK.

With my plan in mind I went in search of a small molecule structure determination of a rhamnose sugar molecule. My major source of such information is a book called "Tables of Interatomic Distances and Configurations in Molecules and Ions" published by the Chemical Society of London. (Of course, the Cambridge Structural Database is ideal for this task if you have ready access to it.) After leafing through many pages I found the structure I desired and wrote the geometry definition below. (Obviously one must ensure that the atom names are consistent with those used on the ATOMC statements.)

```
GEOMETRY    RHAM    BOND    1.42    0.02    C1, O5
GEOMETRY    RHAM    BOND    1.37    0.02    C1, O1
GEOMETRY    RHAM    BOND    1.51    0.02    C1, C2
GEOMETRY    RHAM    BOND    1.47    0.02    C2, O2
GEOMETRY    RHAM    BOND    1.53    0.02    C2, C3
GEOMETRY    RHAM    BOND    1.41    0.02    C3, O3
GEOMETRY    RHAM    BOND    1.52    0.02    C3, C4
GEOMETRY    RHAM    BOND    1.40    0.02    C4, O4
GEOMETRY    RHAM    BOND    1.56    0.02    C4, C5
GEOMETRY    RHAM    BOND    1.45    0.02    C5, O5
GEOMETRY    RHAM    BOND    1.52    0.02    C5, C6
GEOMETRY    RHAM    ANGLE    109    3    O5, C5, C4
GEOMETRY    RHAM    ANGLE    110    3    O5, C5, C6
GEOMETRY    RHAM    ANGLE    114    3    C6, C5, C4
GEOMETRY    RHAM    ANGLE    109    3    C5, C4, C3
GEOMETRY    RHAM    ANGLE    110    3    C5, C4, O4
```

```
GEOMETRY    RHAM    ANGLE    108    3    O4, C4, C3
GEOMETRY    RHAM    ANGLE    111    3    C4, C3, C2
GEOMETRY    RHAM    ANGLE    114    3    C4, C3, O3
GEOMETRY    RHAM    ANGLE    106    3    O3, C3, C2
GEOMETRY    RHAM    ANGLE    110    3    C3, C2, C1
GEOMETRY    RHAM    ANGLE    108    3    C3, C2, O2
GEOMETRY    RHAM    ANGLE    104    3    O2, C2, C1
GEOMETRY    RHAM    ANGLE    113    3    C2, C1, O5
GEOMETRY    RHAM    ANGLE    109    3    C2, C1, O1
GEOMETRY    RHAM    ANGLE    109    3    O1, C1, O5
GEOMETRY    RHAM    ANGLE    120    3    C1, O5, C5
```

Often such a book will not list the standard deviations for each geometry restraint and this book is no exception. Here I have used the numbers that Lynn Ten Eyck started using which are 0.02Å for bond lengths and 3 degrees for bond angles. We also use 15 degrees for torsion angles and 0.02Å for both types of planarity. The book did not list the torsion angle values or the handedness of each chiral center. Therefore I looked at the structure and guessed that the torsion angles should all be three-valued and staggered. This guess seems correct to me but I am not greatly concerned because I don't normally refine torsion angles and could choose not to define them at all. I also examined the structure of the molecule and wrote the definitions of the chiral centers. This completes the definition of RHAM.

```
GEOMETRY  RHAM    TORSION    3060    15    O5, C5, C4, C3
GEOMETRY  RHAM    TORSION    3060    15    C6, C5, C4, C3
GEOMETRY  RHAM    TORSION    3060    15    C5, C4, C3, C2
GEOMETRY  RHAM    TORSION    3060    15    C4, C3, C2, C1
GEOMETRY  RHAM    TORSION    3060    15    C3, C2, C1, O5
GEOMETRY  RHAM    TORSION    3060    15    C2, C1, O5, C5
GEOMETRY  RHAM    CHIRAL    1    1    C1, O5, O1, C2
GEOMETRY  RHAM    CHIRAL    1    1    C2, O2, C1, C3
GEOMETRY  RHAM    CHIRAL    1    1    C3, O3, C2, C4
GEOMETRY  RHAM    CHIRAL    1    1    C4, O4, C5, C3
GEOMETRY  RHAM    CHIRAL    1    1    C5, O5, C6, C4
```

Now we arrive at the difficult task of defining PHOSLINK. (Difficult not because of the program but because of not knowing what numbers to plug in.) I checked the CRC and found that it said that P-N bond lengths are 1.4910Å but had no mention of what structure that number came from or what the bond angles should be. I looked in the International Tables (Volume III) and found a table that included $P_4N_4(CH_3)_8$ (P-N of 1.60(0.03)Å) and $HPO_3{\cdot}NH_2$ (P-N of 1.78(0.06)Å). I could not figure out the Lewis dot structure for the first one and it didn't have any oxygen in it so I ignored it. I found $HPO_3.NH_2$ in the big book in the library and used the geometry from there to define PHOSLINK. This is certainly not the best structural analog because the atoms bonded to the phosphorus have no carbon atoms bonded to them. All of the P-O bond lengths did not seem to vary much between different structures so I defined those bond lengths with the normal sigma. I gave the P-N bond length a larger sigma because I was quite uncertain about its real value. It is interesting to note that during refinement the P-N bond length shortened to about 1.5Å. This possibly implies that incorrect geometry definitions can be overcome if the crystallographic data are good enough.

```
GEOMETRY  PHOSLINK    BOND    1.52    0.02    -O1, P
GEOMETRY  PHOSLINK    BOND    1.52    0.02    P, PO1
GEOMETRY  PHOSLINK    BOND    1.52    0.02    P, PO2
GEOMETRY  PHOSLINK    BOND    1.78    0.06    P, +N
GEOMETRY  PHOSLINK    ANGLE    115    3    -O1, P, PO1
GEOMETRY  PHOSLINK    ANGLE    115    3    -O1, P, PO2
GEOMETRY  PHOSLINK    ANGLE    115    3    PO1, P, PO2
GEOMETRY  PHOSLINK    ANGLE    103    3    -O1, P, +N
GEOMETRY  PHOSLINK    ANGLE    103    3    PO1, P, +N
```

This is the definition of the geometry restraints in PHOSLINK. I did not define torsion angles because I was not sure what they should be and I wasn't going to refine them anyway. Note that some atom names have a "−" in front of them and some do not. The program treats both classes the same and looks for those atoms in the first residue in the linked pair (SUGAR). The atom names beginning with a "+" are in the second residue, PEP1.

Chapter 6

Defining Noncrystallographic Symmetry

Noncrystallographic symmetry is present whenever you have collected diffraction data from a crystal containing two or more copies of a molecule where these copies are not related by the normal symmetry of the crystal. The diffraction data provide unique "snapshots" of each copy. The knowledge that the underlying pattern is the same in each copy introduces a great deal of additional information, which can be used to solve the structure.

> The crystallographic symmetry also provides a great deal of information. However, because the relationship between the copies is exact we always choose to use this type of symmetry as constraints, and because of its mathematical nature we can apply the constraint in reciprocal space. During data collection it is assumed that the differences between symmetry related reflections are due entirely to noise. The intensities are averaged to remove this noise.
>
> In effect we are using noncrystallographic symmetry the same way. The way we incorporate this information is different because the noncrystallographic symmetry does not pass through to reciprocal space in so simple a fashion. The restraints, or constraints, must be applied in real space.

> A knowledge of noncrystallographic symmetry introduces restrictions on the phases of the reflections in a fashion that crystallographic symmetry does not. In a diffraction pattern the reflections are far enough apart that each of their phases are uncorrelated to that of their neighbors. Noncrystallographic symmetry allows one to determine what the intensity of a point between two reflections would be if it could be measured. The higher the copy-number of the symmetry the greater the number of points between each pair of reflections and the greater the information about their phases.

TNT is not a phasing program, nor is its goal to perform density modification. TNT improves the agreement of a model to a set of observations. Noncrystallographic symmetry can be used in this process in two ways; The symmetry can be forced to be exact (a constraint) or can be used as a suggestion that the molecules should be similar to one another (a restraint).

Defining Structural Correspondences

To either constrain or restrain to noncrystallographic symmetry the portions of the molecules which correspond to one another must be defined. In TNT this information is defined on the CLUSTER statement.

A cluster is a collection of chains, some portion of which have identical conformations. A cluster may contain many chains. Parts of each chain may belong to different clusters but no single residue may belong to more than one cluster — The clusters must not overlap.

As an example consider a case where a hemoglobin molecule has crystallized in a form with one tetramer ($\alpha_2\beta_2$) in the asymmetric unit. The two α chains have the same structure, as do the two β chains. However, even though the α and β chains are similar there are significant differences. One would not want to restrain them to each other. We are left with two clusters, one for the α chains and another for β chains. The CLUSTER statements for this case would simply be

```
CLUSTER ALPHA CHAINS A1 A2
CLUSTER BETA  CHAINS B1 B2
```

Local Violation of ncs

It may happen that, due to crystal contacts, there is found to be a structural difference between the two α chains in the region of residues 100 to 125. These residues should not be restricted by the noncrystallographic symmetry, because the symmetry does not apply. You can use the RESIDUE modifier on the CLUSTER statement to specify only those residues which should be restricted. In this new case the CLUSTER statement for the ALPHA cluster would have to be altered, resulting in the new statements:

```
CLUSTER ALPHA RESIDUE 1 - 99 RESIDUE 126 - COOH CHAINS A1 A2
CLUSTER BETA CHAINS B1 B2
```

If you have noncrystallographic symmetry of this type and you execute the **gather** command, only the residues covered by the noncrystallographic symmetry will be gathered. The new ALPHA chain will only contain residues 1 thru 99 and 126 to the end. Chains A1 and A2 will retain residues 100 thru 125.

Defining ncs Transformations

If TNT is given a full set of coordinates, and the CLUSTER statements defining the areas of correspondence, it can calculate the transformations which relate one copy to another. In the case where only the unique coordinates are given, as when the **scatter** command is to be used, the transformations must be supplied by the user.

The transformations are defined to move a particular chain into superimposition with the prototype location for that cluster. While the prototype may be located anywhere in space, usually one places the prototype at the location of one of the chains. In the case of hemoglobin

discussed in the last section, the ALPHA prototype would be located at A1. The transformation of A1 to ALPHA would then be the identity operator. The transformation of A2 to ALPHA would be the same as that of A2 to A1.

The transformations are defined jointly with the CLUSTER and TRANSFORMATION statements. The NCS option on the CLUSTER statement gives the name of the transformation which relates one chain to the prototype. The actual matrix values are associated with that transformation name on the TRANSFORMATION statement. There will be one CLUSTER statement for each cluster and one TRANSFORMATION statement for each chain in each cluster.

As an example, consider the structure of the M6I mutant form of T4 lysozyme. This protein crystallizes with four copies in the asymmetric unit. There is a variable hinge-bending angle between the N-terminal and C-terminal domains. However, the first 10 amino acids move with the C-terminus. The CLUSTER statements for this structure are

```
CLUSTER NTERM RESIDUE 11 - 59 CHAINS A B C D
CLUSTER CTERM RESIDUE 1 - 10 RESIDUE 60 - 162 CHAINS A B C D
```

The noncrystallographic symmetry operators are defined as follows.

```
CLUSTER NTER NCS  A NTE-A    B NTE-B    C NTE-C    D NTE-D
TRANSFORMATION NTE-A    SYSTEM TNT      -
    MATRIX    1.00000  0.00000  0.00000 -
              0.00000  1.00000  0.00000 -
              0.00000  0.00000  1.00000 -
    SHIFT XYZ     0.0000    0.0000    0.0000 -
    SHIFT  B     0.00 -
    SHIFT OCC  0.0000
TRANSFORMATION NTE-B    SYSTEM TNT      -
    MATRIX    0.65484  0.73147  0.19009 -
              0.74936 -0.66109 -0.03759 -
              0.09817  0.16706 -0.98105 -
    SHIFT XYZ    17.0624    9.1073   10.8902 -
    SHIFT  B   -10.76 -
```

```
    SHIFT OCC  0.0000

...

CLUSTER CTERM NCS  A CTE-A    B CTE-B    C CTE-C    D CTE-D
TRANSFORMATION CTE-A    SYSTEM TNT      -
    MATRIX    1.00000  0.00000  0.00000 -
              0.00000  1.00000  0.00000 -
              0.00000  0.00000  1.00000 -
    SHIFT XYZ     0.0000    0.0000    0.0000 -
    SHIFT  B     0.00 -
    SHIFT OCC  0.0000
TRANSFORMATION CTE-B    SYSTEM TNT      -
    MATRIX    0.66181  0.72421  0.19370 -
              0.73855 -0.58551 -0.33426 -
             -0.12866  0.36428 -0.92236 -
    SHIFT XYZ    17.7401    9.5472   11.6293 -
    SHIFT  B    -5.33 -
    SHIFT OCC  0.0000
...
```

Noncrystallographic Constraints

The most powerful fashion to use noncrystallographic symmetry is to reduce the number of parameters in your model with constraints. The correspondence of the separate images of the molecule is assumed to be perfect and the model does not allow any variation between copies. Basically the number of parameters in the model decreases by the number of images in the asymmetric unit.

When using noncrystallographic constraints your coordinate file will contain only a single copy of each molecule. This copy is called the prototype. You must supply the transformations which will be applied to the atoms in the prototype to generate each individual molecule: reconstructing the asymmetric unit.

Refinement of a model with constrained ncs is somewhat more com-

plicated than basic refinement. This complications arrises because the coordinate file does not contain all the atoms in the asymmetric unit of the crystal. To proceed you must generate a TNT coordinate file containing only the unique atoms, and an "ncs" file containing the ncs operators which generate the rest of the atoms. The easiest way to do this is to start with a PDB file containing all atoms in the asymmetric unit. After using **from_pdb** to convert this data to TNT's format edit the TNT control file to add the proper CLUSTER statements. You can then use the **gather** command to determine the ncs operators and average the redundant atomic positions. **gather** writes two files: `gathered.cor`, which contains the coordinates, and `gathered.ncs`, which contains the operators. It would be best to rename these files to something more memorable.

You are now nearly ready to start constrained refinement. The remaining step is to create a refinement script. In all other cases you can run TNT refinement using one of the scripts supplied in `$tntbin`. When running constrained ncs refinement you must customize the refinement script to your project. Make a copy of `$refroot/tnt/doc/tnt_cycle_cons_ncs`. Inside this file certain locations are marked where you must make changes.

Once these changes have been made you can start your refinement. Type the command "**tnt 1 30 pcg tnt_cycle_cons_ncs**" and away you go.

Noncrystallographic Restraints

When there are real differences between the copies of the molecule, you do not want to use the noncrystallographic symmetry as constraints. To allow the data to tell you where the differences are you must allow some variation in your model. These requirements require that you incorporate the noncrystallographic symmetry as restraints.

You will have to choose a weight to set the relative importance of these observations to your other observations. Since the NCS program does not have a standard deviation for the noncrystallographic symme-

try "observations" built in you should set the weight to roughly

$$\frac{1}{\text{expected rms agreement}^2}.$$

If you believe the agreement is, in truth, exact you still cannot expect the agreement to be better than the error in the positions of the atoms. If this error is about 0.2Å the weight would be about 25. In practice one usually must over weight the NCS term to get the expected agreement. I have found that a weight of 40 usually gives reasonable results but there is no guarantee that this weight will work in you case. As usual you will need to choose the weight which gives you the statistics which you believe.

TNT contains a standard refinement script for ncs restraints. It is named `$tntbin/tnt_cycle_ncs`. To restrain ncs place the appropriate CLUSTER statements and ncs weight in your control file and type "**tnt 1 20 pcg $tntbin/tnt_cycle_ncs**".

Chapter 7

Special Problems in Structure Definition

Fragmented Structures

In many structure determination problems one produces a model where the polypeptide chain is broken into fragments even though it is known that the "real" molecule is a single piece. This usually happens because the loops on the surface of a molecule cannot be seen in early electron density maps. Because of the flexibility of TNT there are many ways one can model this situation. This section will discuss two possibilities.

The simplest way to handle this problem is to define the sequence of your protein ignoring the issue of what you can and cannot see. Write RESIDUE statements which describe the chemical structure of your molecule. The parts you can't see, and haven't built, will be indicated as missing but refinement will continue unaffected. As you build more residues into the map you do not have to edit the sequence file.

The other method is to write a sequence file that describes not the molecule you have, but the molecule you have built. This sequence file will be easier to construct from a PDB file but that sequence file will require some editing to become functional.

As an example, suppose that you have two fragments; one numbered

from 100 to 125 and the other numbered from 140 to 160. Your sequence file will look like:

```
RESIDUE 100 ALA 101 PEPTIDE
RESIDUE 101 GLY 102 PEPTIDE
              .
              .
              .

RESIDUE 125 GLY 140 BREAK
RESIDUE 140 GLU 141 PEPTIDE
              .
              .
              .

RESIDUE 160 TYR NULL BREAK
RESIDUE NULL NULL
```

In the standard TNT geometry library you will see that the link BREAK defines the geometry for a blunt end at the C terminus. It should be used whenever you have a C-terminus without the extra oxygen atom of a "real" carboxy terminus. Because BREAK is a link it must link to something. It doesn't matter what you link it to because it defines no restraints involving the "linkee".

If you don't want to connect your fragments directly, as I have done, you can connect them all to a NULL or dummy residue, a residue with no atoms and no restraints. All residues must be defined on a RESIDUE statement. Therefore, if you create a dummy residue of some sort you must include a RESIDUE statement which defines its type. I would suggest that you link fragments together if you have some confidence of the connectivity, and link them to a dummy residue if you do not.

Even if you cannot see all of the atoms in the residues next to the break, you can still define the break in this fashion. You may receive warnings that some atoms cannot be found, but in this case the warnings can be safely ignored.

Static Disorder

Often one will encounter a side chain for which the density is a superposition of two different conformations. It is quite easy in TNT to model these side chains. When there are alternative locations for a particular atom the different possibilities are marked by adding a code to the end of the atom's name with a period between the original name and the code.

This sounds more complicated than it is. An example will make the point more clear. Let us say that the distal three atoms of an ASP amino acid occurs in three conformations. We will use the letter A to indicate the first, with B and C indicating the other two. The names of the three atoms are CG, OD1, and OD2. Therefore the names of the disordered atoms become CG.A, CG.B, CG.C, OD1.A, OD1.B, OD1.C, OD2.A, OD2.B, and OD2.C.

TNT will automatically recognize that this side chain is disordered and appropriately apply the stereochemistry restraints for an aspartic acid.

It is very important that you do not use an atom name with a disorder indicator along with the same atom name without one. If OD1 is disordered in a particular residue there can be no atom named, simply, OD1. If any versions of an atom are marked by a letter they all must be.

When one considers stereochemical interactions between residues it is presumed that atoms marked with .A will interact with other atoms marked .A but not with atoms marked with other letters. If the atoms 123:OD1.A and 132:NH1.A are too close together they will be pushed apart because of the bad contact. However 123:OD1.A and 132:NH1.B can be superimposed and the contact will still not be considered to be bad.

In addition, if residues 123 and 124 are linked in the sequence file by a PEPTIDE bond and each are entirely disordered, the .A version of 123 will be considered to be connected to the .A version of 124 and the .B versions of 123 will link to the .B version of 124. Again no bad contacts will be refined between the .A's and .B's.

Atoms on Special Positions

When an atom falls on a special position — a two-fold or higher symmetry axis — a particular problem must be handled. Because all atoms are expanded by the symmetry of the space group when structure factors are calculated the electron density for an atom in a special position will appear too strong. For a three-fold axis the density will be three times stronger than that of a single atom in that position. To correct for this duplication the atom's occupancy should be set, and fixed, at $1/n$ where n is the "fold" of the axis. The occupancy of an atom sitting on a three-fold should be set to 1/3.

Molecular Disorder

When a crystal contains two or more molecules which are very similar, they sometimes crystallized in a disordered fashion. Two different kinds of molecule may occupy the same lattice position with low occupancy. This situation occurs most often in crystals containing nearly palindromic DNA.

Models can be constructed for such crystals using TNT, but it requires some work and creativity. And speaking of creative limitations, it is almost impossible to intelligibly describe a solution to this problem in general. I will contrive a very specific case and describe its definition and you will have to extrapolate the principle to your case.

Let us assume a crystal containing dsDNA located on a crystallographic two-fold axis. This DNA is nearly palindromic, with the sequence on one strand of AAATGT and the other ACATTT. Call the first chain W and the second C (Watson and Crick). The molecule crystallizes such that W and C appear superimposed.

You cannot model this situation as a single molecule with statically disordered bases because each conformation consists of a chemically different species. You have to consider this DNA strand as a superposition of two different compounds. From that starting point you will want to apply tricks to reduce the number of parameters to the minimal set, because these crystals usually have a limited resolution.

The crudest model for you situation would be to place both a W and a C strand, superimposed, each with a 50% occupancy. Symmetry will generate the other two (half occupancy) strands and complete the model. This is the most general model and if I had 1.0A data I would consider it.

Since I am at low resolution I want to reduce the number of parameters. How far I can go depends upon the amount of conformational change induced by the difference in sequence. If only the nature and location of the bases are affected I could force the sugar-phosphate backbone to be identical and remove the parameters that allow the backbones to differ. This is likely what I want to do, but I should worry that the backbone does move and watch for it. After all, many papers have been written about sequence specific changes to the backbone.

To reduce the parameters I will have to resort to tricks. What I need to do is to create a new chain, named M, whose sequence file contains monstrous, two-headed, bases at the locations of symmetry breakdown. In the most conservative case I define a single chain with sequence A(AC)AT(TG)T and refine the parameters of eight bases.

Now you probably want to know how to define the monstrous sequence file. I will have to do some geometry definition hacking to pull it off. First start with the definition for the perfect palindrome

```
RESIDUE   M|1   ADE   2      D5'END
RESIDUE   M|2   ADE   3      DSUGARPHOS
RESIDUE   M|3   ADE   4      DSUGARPHOS
RESIDUE   M|4   THY   5      DSUGARPHOS
RESIDUE   M|5   THY   6      DSUGARPHOS
RESIDUE   M|6   THY   NULL   D3'END
```

(This is just standard DNA.) Now I have to add the second heads to 2 and 5. We put the extra base atoms in residues 2A and 5A and link them to 2 and 5. The link for each is different because one is a purine and the other a pyrimidine and since the atom names differ in each the definitions of the bond lengths and angles must also differ. The new RESIDUE statements will be

```
RESIDUE   M|2A   CYT   2   ULINK
RESIDUE   M|5A   GUA   5   YLINK
```

Next I have to create definitions for ULINK and YLINK. These links will contain all the geometry definitions you find in `$tntdata/nuclgeo-.dat` that are included in each base that contain some atoms in the base and some atoms in the sugar. If a restraint only contains atoms in the base, ignore it. If a restraint only contains atoms in the sugar, ignore it. Only those restraints which have atoms from both should be put in the link.

Once this has been done I still have a few tasks. All the atoms in 2A and 5A are have 0.5 occupancy and those atoms in the base in 2 and 5 also have 0.5 occupancy. The program must also be told to ignore all bad contacts between the two bases that we know are never really clashing. This is done with the two statements

```
EXCLUDE   M|2A   M|2
EXCLUDE   M|5A   M|5
```

Of course, if the sugar also changes conformation I have to elaborate this further.

Chapter 8

TNT Shell Commands

The majority of tasks required for the refinement of a macromolecular model can be performed by TNT's shell commands. These commands are defined as aliases in the script `$tntbin/logicals`, which should be "sourced" every time you log in.

TNT's commands are listed in Table 8.1. The main commands for refinement are bf tnt and **cleanup**. **tnt** does the actual refinement while **cleanup** is used to remove temporary files if refinement is interrupted. The rest of the commands are used to check agreement of the model to the restraints, work with noncrystallographic symmetry, produce data files for your model building program, and to generate TNT format files from what you have. Each command is described below.

Utility Commands

Command: TNT

tnt <**first cycle**> <**last cycle**> [<**method**>] [<**script**>]

A series of refinement cycles is executed with the **tnt** command. This command centralizes all the steps which are common to all refinement cycles. It reads the file `init.cor` and produces a file named `cycX.cor`,

Utility	**tnt**	run refinement
	cleanup	cleanup after job crash
Model Evaluation	**geometry**	list worst stereochemistry
	ncs	list worst ncs violations
	rfactor	show R value
	dscreen	list atoms w/ worst map fit
	diffcor	list atoms w/ largest shifts
	diffplot	plot each atom's shift
Noncrystallographic Symmetry	**gather**	averages all copies
	scatter	generates all copies
Generating Files	**make_maps**	calculate 2Fo-Fc and Fo-Fc maps
	pekpik	produce PDB file w/ model and peaks
	to_pdb	produce PDB file w/ model
	from_pdb	convert PDB file to TNT
Structure Factor Files	**correct**	standardize HKL file
	hkl_segregate	herd out a 10% test set
	report_hkl	display interesting things
	histogram	plot scale factors vrs resolution

Table 8.1: TNT Refinement Commands

where X is the cycle number, for each cycle. When the cycles are complete a new `init.cor` is created which contains the most recent coordinates.

The first two parameters are required. They simply provide the starting and ending cycle numbers. The default <method> is 'pcg' which indicates the Preconditioned Conjugate Gradient method (also known as the Conjugate Direction method). When performing rigid body refinement one must use the Steepest Descent method, indicated by 'sd'. The fourth parameter allows you to define an alternative refinement script. For example, if you wish to restraint noncrystallographic symmetry you should specify the script `$tntbin/tnt_cycle_ncs` .

You may try out the **tnt** command by running some refinement on one of the supplied examples. Change your directory to `$refroot/tnt-/doc/examples/phrm`. Copy the initial coordinate model, `add_b_phrm-.cor`, to `init.cor`. Then type "**tnt 1 30 | tee test.log**".

The first cycle will be called number 1 and the last cycle to run will be 30. Compare the contents of `test.log` with `standard.log` to verify the correct operation of TNT.

Command: CLEANUP

cleanup

The **cleanup** command deletes the temporary files created during a cycle of refinement. If you interrupt a refinement run the simplest way to get ready to restart refinement is this command.

Model Evaluation Commands

Command: GEOMETRY

geometry <coordinate file name>

The **geometry** command will list the atoms found in the worst stereochemical restraints in the model.

Command: NCS

ncs <coordinate file name>

The **ncs** command will list the atoms found in the worst pairs of non-crystallographic symmetry related atoms.

Command: RFACTOR

rfactor <coordinate file name>

The **rfactor** command will describe the overall agreement between the observed structure factors and those predicted by the model. It prints the completeness of the observed data, the scale factors relating the two data sets, and the R value itself. It also provides a break down of the R value in 10 shells of resolution.

Command: DSCREEN

dscreen <coordinate file name>

The **dscreen** command lists the atoms for whom the diffraction data suggests the position, temperature factor, or occupancy has the greatest error. These are atoms very near features in the Fo-Fc map.

Command: DIFFCOR

diffcor <coordinate file name> <second file name>

The **diffcor** command compares two coordinate files and writes up to three lists of atoms with large shifts. The atoms with largest positional shifts are written to the file `screen.cor`. Those with the largest shifts in their B factors are listed in `screen_b.cor`, and the atoms with large occupancy shifts are listed in `screen_occ.cor`. An additional file, called `diff.cor`, is produced which contains the shifts of all the atoms.

Command: DIFFPLOT

diffplot <coordinate file name> <second file name>

The **diffplot** command compares two coordinate files and displays a plot of the shifts required to move from the second to the first.

Noncrystallographic Symmetry Commands

These commands are useful when refining a model with constrained (or "strict" ncs). There are several programming issues which has not been written into the refinement scripts described in this guide. To refine a model with constrained ncs you should turn to Page 51.

Command: GATHER

gather <coordinate file name>

The **gather** command will take a full model (e.g. one that contains a separate occurrence of each molecule in the asymmetric unit) and average it over the noncrystallographic symmetry to produce a coordinate file with only the unique parts of the asymmetric unit represented. For example, assume we have a model with four identical chains (named O, A, B, and C) in the asymmetric unit and the noncrystallographic symmetry cluster CRO is defined to cover all residues of each chain. After the **gather** command the new coordinate file will contain one chain, named CRO, which is the average of all four chains. In addition to the new coordinate file, **gather** will write out a small file which contains the TRANSFORMATION statements which describe the rotation and translation required to overlay each chain on the prototype.

The files written by the **gather** command are `gathered.cor` and `gathered.ncs`.

Command: SCATTER

scatter <coordinate file name> <ncs file name>

The **scatter** command will take the prototype for a crystal structure and expand it by the noncrystallographic transformations to generate the full asymmetric unit of the crystal. Both a coordinate file containing the prototype and another file containing the transformations are required. The ncs transformation file was probably written by the **gather** command.

The file written by the **scatter** command is named `scattered.cor`.

Communicating with the Graphics Program

Command: MAKE_MAPS

make_maps <coordinate file name>

The **make_maps** command produces two map files in DSN6 format. One is named `2fo-fc.dsn6` and the other `fo-fc.dsn6`. The contents of the maps should be obvious from their names.

Command: PEKPIK

pekpik <coordinate file name>

The **pekpik** command produces a PDB format coordinate file which contains all the atoms in the starting coordinate file as well as marker atoms placed at the locations of the highest 30 peaks in the Fo-Fc map. All the marker atoms are placed in a residue named PEAK. The highest peak is marked with atom PEAK:1.

Command: TO_PDB

to_pdb <coordinate file name>

The **to_pdb** command creates a PDB coordinate file from the TNT coordinate file.

Command: FROM_PDB

from_pdb <coordinate file name>

The **from_pdb** command reads a PDB format coordinate file and writes, at least, a TNT coordinate file. If a sequence file did not exist one will be created. If there was no TNT control file one will be created which contains only limited information. You will have to edit this control file prior to executing any other TNT commands.

If the PDB file contains a proper CRYST1 card a TNT control file is not required for this command. This is the only TNT shell command which will run without a control file. If the PDB file does not contain the cell constants on a CRYST1 card you must create a TNT control file which supplies this information.

Structure Factor File Commands

Command: CORRECT

correct <HKL file name>

The **correct** command is used to finish the conversion of your diffraction data file to TNT format. You must convert your data file to the general appearance of the TNT HKL format (described in Appendix A of the TNT Reference Manual) prior to using the **correct** command. The **correct** command will read your file, move the data to the proper

asymmetric unit, and write the sorted reflections to a new file. The new file should be used in all subsequent TNT operations.

The output of the **correct** command is given the same name as the input file but prepended with "`corrected_`".

Command: HKL_SEGREGATE

hkl_segregate <HKL file name> [< HKL prototype name>]

Once you have a proper HKL format file you may wish to segregate it into a test and working set. The command **hkl_segregate** performs that task. If you issue the command giving only the name of you HKL file two new files will be created, one containing 10% of the data and the other containing the rest. The second argument can be used to name a test set that you want to match. When refining a crystal which is isomorphous to some previous project you will want to use the same reflections in the new test as as the old.

Command: REPORT_HKL

report_hkl <HKL file name>

The **report_hkl** command will print a short description of the contents of the named HKL file. This description will include the number of reflections, the resolution limits, and the Wilson B.

Command: HISTOGRAM

histogram <coordinate file name>

The **histogram** command creates a plot of TNT's scaling verses empirical scaling. The scale factors are displayed as a function of $\sin\theta/\lambda$. Any systematic deviation should be investigated as a possible problem with the data collection or reduction.

Chapter 9

Running Refinement

The course of refinement will depend upon the errors in the model. When a structure is solved by Molecular Replacement the relative position of the atoms are usually quite good but their absolute position could be in error by a great deal. This is because the rotation function and translation function solutions could be in error by several degrees and maybe an Ångstrom. These are very large errors and normally could not be corrected except for the strong phasing power of the essentially correct core of the molecule. This situation calls for rigid-body refinement, the motion of the core of the molecule as a single unit. In addition, the domains of the molecule might have shifted in the current molecule relative to those of the prototype. Just to be sure, after the overall rigid-body refinement the molecule should be broken down into rigid domains and refined again.

Usually rigid-body refinement is performed using only the data to 4.0 Ångstrom resolution.

When a structure is solved by Molecular Modification, the structure of a molecule which is a modified form of a known structure in an isomorphous crystal, often the cell constants will have changed. This change will require that refinement begin with rigid-body refinement as well. In this case the goal is to principally translate the molecule not to rotate it but the method is the same. If the modification causes a change in the orientation of the domains, rigid-body domain refinement will be required as well.

Whenever performing rigid–body refinement you should consider setting the weight for nonbonded contacts to zero. In a Molecular Replacement model the loops on the surface are quite likely misplaced. Forcing the center of the molecule to be displaced because of a surface bad contact might not be a good idea.

Models derived from these methods will be quite good once the overall corrections to orientation have been made. One may start immediately using the higher resolution data because the expected size of the shifts required to correct the model is small. With a Molecular Replacement model you may want to start at 2.5Å resolution while a Molecular Modification model probably can start at the limit of the data collected.

When the initial model has been constructed by building into a map, such as in MIR and MAD structure determination, overall adjustments in orientation will not be required. However such models will have many misplaced atoms and probably many places where the agreement with the geometric restraints is poor. Because large shifts will be required to correct the model only the lower resolution data should be used at first. The best strategy appears to be to start with the resolution limit of your map. A round of loosening up the geometry and tightening down should be performed. (This step will be discussed in more detail later.) It might pay to loosen and tighten a second time before extending resolution.

If the model improved a great deal during refinement you should enter model building without adding more data. If the refinement did not change the model much, and you think it fits the data you are using well enough, you should extend the resolution and run more refinement. If the agreement between your model and your data cannot be made satisfactory it is likely that something is seriously wrong. Either the model or the data must be corrected before preceding.

Rigid-body refinement is performed by placing one or more COMBINE statements in your control file, fixing the B factors and occupancies with two CONSTANT statements, and running the refinement by typing "**tnt 1 5 sd**".

To run free atom refinement simply remove the COMBINE state-

ment(s), remove the "CONSTANT B" card and type "**tnt 1 30**". Of course you may use whatever cycle numbers you want.

There are two questions left to be answered. How many cycles should I run and what weights should I use? The first question is easy. About 30 cycles seems to get the atoms with low temperature factors to where they are going. Additional cycles will continue to move atoms, in some cases quite a bit, but those atoms will be the ones with very high B values and their position is very imprecisely defined by the diffraction data anyway.

Choosing Weights

The determination of proper weights is essentially an empirical process and therefore requires some experimentation. In other words, you will have to guess.

Basically, the importance of each kind of restraint you place on your model can be individually set using a weight. The larger the weight the more important that restraint becomes. Since all the weights are relative increasing one is the same as dropping all the others.

I recommend that you decide your weights based on trial and error. Choose weights which give you the quality of answer you want. When your TNT control file was created by **from_pdb** it contained the weights

```
WEIGHT RFACTOR  0.0010
WEIGHT BOND     0.8  ANGLE 1.3  TORSION 0   PSEUDOROTATION 1
WEIGHT TRIGONAL 2    PLANE 5    CONTACT 10  BCORREL 1
```

You will need to fine–tune these to match your situation. After a trial run of refinement compare the r.m.s. errors for the geometry restraints of the model to the target values in Table 9.1.

If the r.m.s. error for a class of restraints is too large I increase its weight. If the model is too consistent with the observations its weight must be reduced.

Restraint type	Target
Bond Lengths	0.02
Bond Angles	3.00
Torsion Angles	–
Trigonal Atoms	0.02
Planar Groups	0.02
B Correlation	6.00

Table 9.1: Target r.m.s. Value for each Restraint Type

These r.m.s. values are measure the level of precision of the geometry library. A high quality model, whose stereochemistry has not been restrained, generally agrees with the TNT restraint library with these r.m.s. errors.

Usually the only problem will be the RFACTOR weight. Since there is no objective target values for the diffraction data term, we can only set the RFACTOR weight by monitoring the stereochemical restraints' r.m.s. errors. If all these restraints are too tight the RFACTOR weight must be increased. If they all are too loose the RFACTOR weight must be dropped.

Usually I change the RFACTOR weight by factors of two to ten. For example, if the bond length r.m.s. were 0.08 instead of 0.02 (and all the other geometric restraints were proportionately large) I would decrease the RFACTOR weight by a factor of ten. If the bond length r.m.s. were instead 0.01 I would double the RFACTOR weight.

When the model is in bad shape it is a good idea to adapt the model to the diffraction data first and then bring the model into agreement with ideal geometry. This is done by performing a series of cycles with a high RFACTOR weight and another series with lower weight. The final weight is chosen so that the geometry statistics at the end are just what you want.

If the starting model is good, say the R value is less than 25%, then there is no reason to follow this complicated procedure. In this situation you may simply refine with the RFACTOR weight set to give

proper geometry. Usually the same weight for the RFACTOR term can be used each time you return from model building.

The B correlation (BCORREL) module restrains the B factors of the model to a library of standard restraints. These restraints were developed by an analysis of the relationship between the B factors of atoms bonded to each other in models refined to 1.6Å diffraction data. If your data extends to 1.6Å it makes no sense to apply these restraints and you should set the BCORREL weight to zero. If your diffraction data is not of this quality you should set the weight to one. With very low resolution data you may need to increase the weight to force the model to reach the target for this restraint.

Convergence Criteria

The process of model building followed by refinement can seem to repeat endlessly. Somewhere it must end. A good time to end would be when the new model, ready for model building, is so similar to the one from the last round that the new map will show nothing new. The Fobs are always the same — the new map will be distinguished by the difference in Fcalc and ϕcalc (the calculated phase). In my experience, I have never found a difference in a model when the R value between the two sets of Fcalc is less than 3% and the r.m.s. phase difference is less than 10 degrees. These parameters should be monitored each round of model building and can be used to judge when refinement should stop.

Chapter 10

Examples

This chapter provides examples of TNT control files from a number of projects. Each example is introduced with several keywords which describe the basic features used in the example. The control file is listed along with a brief description.

Protein, All Atom Refinement

Goose Lysozyme

```
CELL 38.3 65.7 45.2 90 116 90
INCLUDE /usr/local/tnt/data/symmetry/p21b.dat

FO glnatob.hkl
RESOLUTION 1.6

INCLUDE gans.seq

EXCLUDE   3:OD1   3:C
EXCLUDE  97:OD1  97:C
EXCLUDE 108:OD1 108:C
EXCLUDE 168:OD2 168:C
EXCLUDE 182:CG  182:O

CONSTANT OCC

WEIGHT RFACTOR  0.0090
WEIGHT BOND     0.8  ANGLE 1.3  TORSION 0
WEIGHT TRIGONAL 2    PLANE 5    CONTACT 10  BCORREL 0
```

Goose Lysozyme crystals diffract quite well. The control file shown here varies both the positions and temperature factors of each atom. The occupancies are held fixed because even with 1.6Å resolution data these parameters are not well defined. You will almost never vary occupancy parameters.

Because of the presence of the high resolution data it was decided not to impose the B correlation restraint. The weight of the BCORREL term was set to zero.

The criteria TNT uses to identify a bad contacts seems to be too restrictive in the case of certain atoms in aspartic and glutamic acids. The EXCLUDE cards in the example were placed there because these "bad" contacts were determined by the experimenter to be, in fact, proper.

Protein, Rigid-Body Refinement

FGF

```
CELL 30.9,33.4,35.9,59.5,72.0,75.6

FO fgfnat.hkl
RESOLUTION 20 4
SET B 0.0 KSOL 0.8

INCLUDE fgf.seq

COMBINE    XYZ
CONSTANT    B
CONSTANT   OCC

WEIGHT RFACTOR 0.0010
WEIGHT   BOND     0.8  ANGLE 1.3  TORSION 0
WEIGHT   TRIGONAL 2    PLANE 5    CONTACT 0  BCORREL 1.0
```

Fiberblast Growth Factor crystals grow in space group P1. Since P1 is the default space group one is not required to enter symmetry operators in the control file.

This control file indicates rigid-body refinement of the positions of the atoms in the model. Both the temperature factors and occupancies are held constant. Since the resolution limit was reduced to 4Å TNT was unable to properly scale the data sets. The SET statement was used to force B equal to 0.0 and K_{sol} equal to 0.8. Given this restriction TNT generates proper values for the other scale factors.

The weights of the stereochemical restraints are unimportant except for that of the bad contacts. In the early stages of a Molecular Replacement rigid-body refinement the loops on the surface of the molecule are unreliable and it is probably not wise to use these overlaps in the positioning of the molecule. The CONTACT weight should be set to zero.

Protein, Special Geometry

Thermolysin:Phosphoramidon

```
CELL 94.1 94.1 131.4 90 90 120
INCLUDE /usr/local/tnt/data/symmetry/p6122.dat

FO tln_phrm.hkl
RESOLUTION 2.3

INCLUDE tln.seq

RESIDUE  SUGAR  RHAM  PEP1  PHOSLINK
RESIDUE  PEP1   LEU   PEP2  PEPTIDE
RESIDUE  PEP2   TRP   MYEND CTERM
RESIDUE  MYEND  OXY
RESIDUE  SOL2   WTR

CONSTANT OCC

INCLUDE phrmgeo.dat

WEIGHT RFACTOR 0.0010
WEIGHT   BOND     0.8  ANGLE 1.3  TORSION 0
WEIGHT   TRIGONAL 2    PLANE 5    CONTACT 10  BCORREL 1
```

The complexes of the endoprotease Thermolysin with a number of inhibitors have been refined. Usually the binding of an inhibitor will not affect the data stored in the sequence file. Therefore, the sequence file of the apoenzyme is INCLUDEd in this control file and the RESIDUE statements specific to the inhibitor are added. In addition the stereochemical restraints are defined by INCLUDEing the file `phrmgeo.dat`.

This control file also varies both the positions and temperature factors of all the atoms. At a resolution of 2.3Å one would usually set the BCORREL weight to one.

All the files required to perform refinement on this example are supplied in `$refroot/tnt/doc/examples/phrm`. A description of these particular `RESIDUE` statements is given starting on Page 39.

Nucleic Acid, Noncrystallographic Symmetry

bDNA

```
CELL 24.87 40.39 66.2
INCLUDE /usr/local/tnt/data/symmetry/p212121.dat

FO 1bna.hkl
RESOLUTION 2.3
SET BSOL 150

INCLUDE bdna.seq

EXCLUDE H|25:O H|86:O
EXCLUDE H|54:O H|86:O
EXCLUDE H|62:O H|77:O
EXCLUDE H|63:O H|77:O

NOTE Define NCS symmetry for only the core.  There are
NOTE significant differences at the ends.  It is probably
NOTE not good to restrain this, just monitor it.

CLUSTER BDNA RESIDUES 4 - 9 CHAINS A B

CONSTANT OCC

WEIGHT RFACTOR 0.03  NCS 0
WEIGHT BOND     0.8  ANGLE 1.3  TORSION 0   PSEUDOROTATION 0
WEIGHT TRIGONAL 2    PLANE 5    CONTACT 10  BCORREL        0
```

This TNT control file varies the position and temperature factor of each atom in the model. A number of problems arise with this refinement. First, the quality of the diffraction data is low. This results in a failure to properly scale the observed and calculated data without forcing the parameter B_{sol} equal to 150.

Second, a large number of bad contacts are discovered between various water molecules. EXCLUDE statements were entered to cause TNT to ignore the worst. Before beginning a serious refinement these water molecules should be examined manually and corrected. (The water molecules in the original PDB file are unsupported by the density map. I would remove them all, refine the DNA, and build back those whose density returned. I have done this and very few return.)

There are two chains in the asymmetric unit which are chemically identical. Therefore a CLUSTER statement has been added to describe the noncrystallographic symmetry. However the similarity of the two chains is poor. Either the current model is incorrect or the ncs does not hold. Since we doubt the ncs we set the weight for the NCS term to zero. The only effect of the CLUSTER statement is that it allows the use of the **ncs** command to monitor the similarity of the two chains.

You may have noticed that the weight for the B correlation restraint has been set to zero in spite of the poor quality of the diffraction data. This is done because a library for temperature factor correlations in nucleic acids does not exist at this time. You cannot use the BCORREL restraints with DNA or RNA.

All the files required to perform refinement on this example are supplied in `$refroot/tnt/doc/examples/bdna`.

Protein, Rigid-Body Refinement of Two-Domain Molecules

T4 Lysozyme (M6I)

```
CELL 72.200   73.800  150.500  90.00  90.00  90.00
INCLUDE /usr/local/tnt/data/symmetry/p212121.dat

FO r-free/1m06i_exp.hkl
RESOLUTION 2.2

INCLUDE ../../t4l.seq
RENAME CHAIN_TYPE NULL TO T4L
CHANGE RESIDUE_TYPE OF T4L|6 TO ILE

CHAIN A T4L
CHAIN B T4L
CHAIN C T4L
CHAIN D T4L

ASSUME RESIDUE_TYPE HOH WHEN_CONTAINS OH

CLUSTER NTERM RESIDUE 11 - 59 CHAINS A B C D
CLUSTER CTERM RESIDUE 1 - 10 RESIDUE 60 - 162 CHAINS A B C D

COMBINE XYZ A|11 - 59
COMBINE XYZ A|1 - 10 A|60 - 162
COMBINE XYZ B|11 - 59
COMBINE XYZ B|1 - 10 B|60 - 162
COMBINE XYZ C|11 - 59
COMBINE XYZ C|1 - 10 C|60 - 162
COMBINE XYZ D|11 - 59
COMBINE XYZ D|1 - 10 D|60 - 162

CONSTANT B
CONSTANT OCC
```

```
WEIGHT RFACTOR 0.0005
WEIGHT BOND     0.8  ANGLE 1.3  TORSION 0
WEIGHT TRIGONAL 2    PLANE 5    CONTACT 0  BCORREL 1.0
```

This is the control file which performs rigid-body refinement on the model of the T4 lysozyme mutant M6I. There are four molecules in the asymmetric unit, each with differing angles between the N and C terminal domains. Each of the eight domains is refined as a distinct group. The interesting part is that the first ten amino acids move with the C terminal domain, not the expected N terminal domain. The CLUSTER statements define the ncs which permits the use of the **ncs** command to monitor the similarity of the various domains. The COMBINE statements create the rigid bodies.

To prevent the, possibly, incorrect location of the surface loops of the molecule from causing the domains to be misplaced, the weight for bad contacts has been set to zero.

Note the way the sequence of this mutant is defined. First the sequence file the the wild type enzyme is read. Since this sequence is defined in terms of the "chain with no name" the chain type is renamed to be clearer. Then the residue type of residue number 6 is changed to reflect the mutation. Each chain is defined to have the resulting sequence.

The advantage of this scheme is that the sequence of the majority of the protein does not have to be recreated for each mutant. In addition, it is very clear from looking at the control file that the only mutation in this protein is at residue 6. Often T4 lysozyme mutants are made in a non-wild type background and it can be confusing if this is forgotten.

Protein, Rigid-Body Refinement of Two Domain Molecules

HEWL

```
CELL 28 62.9 60.5 90 90.8 90
INCLUDE /usr/local/tnt/data/symmetry/p21.dat

FO hewl.hkl
RESOLUTION 4
SET KSOL 0.8    ! Data will not scale properly without
                !  this SET card even to the full
                !  resolution of 2.2A.
INCLUDE pdb1lym.seq

CLUSTER N RESIDUE 1 - 91 CHAINS A B
CLUSTER C RESIDUE 92 - OXY CHAINS A B

COMBINE  XYZ  A|1 - 91
COMBINE  XYZ A|92 - OXY
COMBINE  XYZ  B|1 - 91
COMBINE  XYZ B|92 - OXY

CONSTANT   B
CONSTANT  OCC

WEIGHT RFACTOR 0.007
WEIGHT BOND 1.0 ANGLE 1.0 TORSION 0.0
WEIGHT TRIGONAL 5.0 PLANE 10.0 CONTACT 0.0 BCORREL 1
```

This control file indicates rigid-body refinement for each of the two domains of hen egg white lysozyme. In this crystal form there are two molecules in the asymmetric unit. As usual they are named A and B.

The first COMBINE statement defines the N terminal domain of chain A, containing residues 1 through 92, as one rigid-body. The

second domain contains residues 92 through OXY and is made a rigid-body with the second COMBINE statement. (OXY is the name of the final residue of a polypeptide chain and contains the lone oxygen at the C terminus.) The other two COMBINE statements define the equivalent rigid bodies for chain B.

The resolution limit was dropped to 4Å and therefore a SET statement forcing the value of K_{sol} was required. As the note in the control file states this SET statement is even required with all the data. This fact indicates some problem with the data set and should be noted in the control file in this fashion to prevent the problem from being forgotten. Both the B factors and the occupancies are held constant because the Steepest Descent method will only allow one kind of parameter to vary at a time.

Index

A

add_b script, 10
ASSUME statement, 13
Atom type, 4
ATOMx statement
    example of, 25

B

B
    validation, 8
B correlation, 34
Bond angle, 31
Bond length, 31
Break, 21
    example of, 56
Bsol
    validation, 9

C

C-terminus
    break, 21
    normal, 21
CELL statement, 12
CHAIN statement
    example of, 24–26
    usage, 23
Chirality, 34
cleanup command, 63
CLUSTER statement, 15
    example
        defining transformations, 50
        entire chains, 48
        residue ranges, 49, 50
COMBINE statement, 15
command
    cleanup, 63
    correct, 67
    diffcor, 64
    diffplot, 65
    dscreen, 64
    from_pdb, 67
    gather, 65
    geometry, 63
    histogram, 68
    hkl_segregate, 68
    make_maps, 66
    ncs, 64
    pekpik, 66
    report_hkl, 68
    rfactor, 64
    scatter, 66
    tnt, 61
    to_pdb, 67
CONSTANT statement, 17
correct command, 67
Crosslinks
    defining a, 22, 24
Cystine bridge
    defining a, 21, 25

D

diffcor command, 64
diffplot command, 65
Disorder
    atomic, 57
    molecular, 58
Disulfide bond
    defining a, 21, 25

DNA geometry definition, 26
dscreen command, 64

E

Engh & Huber Parameters, 37
EXCLUDE statement, 14

F

Fixing parameters, 17
FO statement, 12
Free R, 6
from_pdb command, 67

G

gather command, 65
geometry command, 63
GEOMETRY Statement
    detailed discussion, 30

H

histogram command, 68
HKL file
    creating, 5
hkl_segregate
    usage, 6
hkl_segregate command, 68

I

Ideal geometry
    definition of, 29

K

K, 9
Ksol
    validation, 9

L

Limiting parameters, 17
LINK statement, 22
Linking residues
    to symmetry images, 24
    within a chain, 20

M

make_maps command, 66
Metal atoms, 21
Molecular disorder, 58
Moses option, 16
Multiple conformations, 57

N

ncs command, 64
Noncrystallographic symmetry, 47
    constraints, 51
    restraints, 52
    structural correspondences, 48

O

OPTION statement, 12

P

PDB Format, 3
pekpik command, 66
Peptide bond
    defining a, 20
    example of, 25
Planarity, 33
Pseudorotation angle, 33

R

R free, 6
r-free script, 6
RANGE statement, 17
report_hkl command, 68
RESIDUE statement
    example of, 25, 26
    usage, 20
RESOLUTION statement, 13
rfactor command, 64
Rigid bodies and solvent, 16
Rigid-body refinement, 15

RNA geometry definition, 26

S

Scaling Fo's and Fc's
    B, 8
    Bsol, 9
    K, 9
    Ksol, 9
scatter command, 66
script
    add_b, 10
    r-free, 6
    tnt_cycle_cons_ncs, 52
    tnt_cycle_ncs, 63
SET statement, 13
    usage, 10
Solvent
    in rigid bodies, 16
Special positions, 58
Standard geometry
    definition of, 29
    finding values for, 44
    libraries, 35
Static disorder, 57
Steepest Descent, 17

T

tnt command, 61
tnt_cycle_cons_ncs, 52
tnt_cycle_ncs, 63
to_pdb command, 67
Torsion angle, 32
TRANSFORMATION statement
    example, 50
Trigonal planarity, 33

W

WEIGHT statement, 12
Weights
    Starting guess, 71
Wilson B, 68