HIGH-PRESSURE X-RAY CRYSTALLOGRAPHY AND CORE HYDROPHOBICITY OF T4 LYSOZYMES

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Marcus David Collins

January 2006

HIGH-PRESSURE X-RAY CRYSTALLOGRAPHY AND CORE HYDROPHOBICITY OF T4 LYSOZYMES

Marcus David Collins, Ph.D.

Cornell University 2006

While a great many protein structures are now known, considerably less is known experimentally about how these molecules reversibly fold and unfold into nearly unique, active structures. For more than 50 years, the central hypothesis has been that micro-scale phase separation between oil-like non-polar and charged or polar amino acid residues drives the formation of a “hydrophobic” protein core. A great deal of evidence supports this hypothesis, but the unfolding of proteins as a response to pressure contradicts it. A possible solution is the suggestion that pressure induced unfolding is a different process from thermally or chemically driven unfolding. Separately, little is known about the structural response of proteins to pressure, although pressure has clear and sometimes large effects on protein activity and stability. For these reasons, we have chosen to study the cavity-containing mutant L99A of T4 Lysozyme, along with the wild-type protein, under pressures up to 2 kbar, a compression of about 0.1%. The protein is remarkably incompressible, and the presence of a cavity has almost no impact on the pressure response over the wild-type lysozyme. Instead, four water molecules cooperatively fill the cavity, interacting with each other and the protein roughly equally. We believe the cavity to be half full with water near 2 kbar, suggesting that despite its hydrophobic nature, the interior of a protein maintains strong electrostatic interactions.

BIOGRAPHICAL SKETCH

Marcus Collins and his twin brother Nathan were born in Portland, Oregon in August of 1977. Raised by an electrical engineer and a nurse, he developed an early interest in science, declaring at the age of 3 that he would like to become an "astronomical chemist", whatever that might have meant to him. He spent a great deal of time in the woods hiking and skiing. He graduated from the University of Washington having climbed most of the Cascade volcanoes and with College and Departmental Honors in Physics. He was awarded an NSF Graduate Fellowship to study at Cornell University beginning in 1999. His present study of biological systems is a complete fluke, as he had no interest in biology in high school. At this point, his entire career appears to be some sort of cosmic coincidence, leaving him to wonder what the punchline will be.

You have opened a new door, and I share this with you,

for I have been where you are now.

This world is more beautiful than I believe we can possibly comprehend. It has been a privelege to spend my days watching the tickings and tockings of an infinite universe. I dedicate this work to two people who have shared with me the equally infinite joy of watching it all.

For Nathan, who will always be a physicist and more importantly my friend.

and

For Erin, whom I love dearly.

ACKNOWLEDGEMENTS

Words fail to describe the impact Sol Gruner has had on my life. For his constant patience, bizarre humor, and guidance I am eternally in his debt. Professors Neil Ashcroft, Jim Alexander, Persis Drell and Lois Pollack have shown me much compassion and care. Dr. Mark Tate has been a source of constancy and support throughout my time at Cornell. Professor (!) Peter Abbamonte has been a good friend as well. To all of them, my deepest thanks. I have greatly enjoyed my colloboration with Dr. Michael Quillin and Professor Brian Matthews of the University of Oregon, and Gerhard Hummer of the National Institutes of Health. I would not be half the person I am were it not for the strength, inspiration, and guidance of my family: Dr. Wilfred and Mildred Gamon, Philip and Ellen Collins, and my parents, John and Kathryn Collins. Chris Mathieson, Matthew Van Sickle, and David Burdick have guided me and consoled me for many, many years. I used to enjoy talking to Raphael Kapfer about his interesting work pressurizing proteins. I dearly wish that he could share in what I have learned. Gil Toombes and Nozomi Ando have had great impact on this work and I will miss them both. The assistance of Paul Urayama, Marian Szebenyi, Richard Gillilan, David Schuller, and Andrew Stewart was invaluable as I began this work. The following people deserve no less mention: Gene Yore; Dr. Wai Wong; Dr. Larry Gulberg, who encouraged me in high school; Prof. David Boulware, who was my early mentor; Prof. Luke Keller, Dr. Donald Barry, Aaron Blake, Adam Kravetz, and Matthew Steven, who have all shared many a good adventure with me.

Let us begin.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Please note that the symbol or abbreviation you are seeking may instead be found in the following table of symbols. Single capital letters should often be read as atomic element names or as the sequential names of secondary structure elements.

→	Mutation, when found between two amino acid abbreviations
2D	Two-dimensional
A, Ala	Alanine
A	Adenosine or Adenine (if in context of nucleic acids)
Å	Ångstroms, 0.1 nanometers
AMBER	Assisted model building with energy refinement, an MD package
Arg	Arginine
Asp	Aspartic acid
Asn	Asparagine
Be	Beryllium
BME	β-mercaptoethanol
bR	Bacteriorhodopsin
°C	Temperature in degrees celsius
C, Cys	Cysteine
C	Carbon (usually of the carboxyl group of an amino acid)
C_α	The α, or central, carbon of an amino acid
C_γ,C_δ, *etc.*	Side chain carbon atoms, in IUPAC naming conventions
cal	Calorie, 4.184 Joules
CCD	Charge-coupled device
CCP4	Colloborative Computational Project Number 4
CESR	Cornell Electron Storage Ring
CHESS	Cornell High Energy Synchrotron Source
CO	Carbon monoxide
CP	Cytoplasmic
CT	C-terminal domain
D	Debye, unit of dipole moment, 3.355×10^{-30} Coulomb-meters
DAC	Diamond anvil cell
FTIR	Fourier-transform infra-red spectroscopy
Glu	Glutamic acid
^{1}H, ^{2}H	Hydrogen or deuterium, respectively
HEWL	Hen egg-white lysozyme
I, Ile	Isoleucine
iff	If and only if
in.	inch, 25.4 millimeters
K	Temperature in degrees Kelvin
keV	kilo-electron Volts, 1.6022×10^{-16} Joules
L, Leu	Leucine
L99A	WT* T4 Lysozyme with Ala subsituted for Leu at position 99

M	Molar
MAD	Multiple anomalous diffraction
MC	Main chain
mg	milligrams
mL	milliliters
mM	millimolar
mm	millimeter
MPa	Megapascals
mt{...}	Name of a diffraction dataset, see Table 3.1
μL	microliters
N	Amino nitrogen of the peptide backbone
NMR	Nuclear magnetic resonance
NT	N-terminal domain
O	Carbonyl oxygen of the peptide backbone
Pa	Pascals
PDB, pdb	Protein Data Bank
PDL	Perl Data Language
Phe	Phenylalanine
pmf	Potential of mean force
RMS	Root-mean-square
SANS	Small angle neutron scattering
SAXS	Small angle X-ray scattering
SC	Side chain
Ser	Serine
SNase	Staphylococcal nuclease
T, Thr	Threonine
TLA	Three letter acronym
Tyr	Tyrosine
U	Uridine or Uracil
UV	Ultra-violet
V, Val	Valine
WT	Wild-type
WT*	A mutant of WT T4 Lysozyme with cysteines replaced by Thr or Ala
wt{...}	Name of a diffraction dataset, see Table 3.1

LIST OF SYMBOLS

A comment on notation. Any document must choose some way of organizing its symbols. The following table lists verbatim as many symbols as feasible that are used in this text, standard and otherwise. However, I have tried to adhere to a few conventions which will help the reader and prevent them from constantly referring to this table. Vectors are universally denoted in italic boldface, *e. g.* $\boldsymbol{x}$. Note, however, that it is the convention of this thesis to denote sets of scalars as vectors. This may lead to confusion primarily in sets of structure factor amplitudes. Thus, $\boldsymbol{F}$ is a *set* of structure factor *amplitudes*, not a single, complex structure factor. Similarly F_{h} is just the amplitude, with the sans-serif h referring to a particular point in reciprocal space. Only when using the notation $F(\boldsymbol{Q})$ should the reader understand this to mean the *complex* structure factor at a point in three dimensional reciprocal space denoted by the *vector* $\boldsymbol{Q}$.

Subscripts indicate either an integer index, a set of integer indices (as with $\mathsf{h} \equiv \{h, k, l\}$, where h, k, l are integers) or indicate a class of variable, *e. g.* F_o is an *observed* structure factor amplitude.

Care should be taken to note the font of both the symbol and its subscripts. There are only so many letters in the Greek and Roman alphabets, so a sans-serif symbol (*e. g.* D) should be read as distinct from a Roman (D), italic (D), or boldface (**D**) symbol, and so on. Boldface and sans-serif symbols generally, but not always, denote matrices. Roman letters are most often used for units, elements, *etc.* Italic letters generally, but not always, indicate simple scalar variables. In special contexts they may denote complex numbers.

Finally, some symbols are not to be taken as only one character. For instance, g and $g(r)$ do not refer to the same thing at all. Context is unavoidably important, so the reader must not use this table blindly.

Please note that the symbol you seek may instead be listed in the previous table of abbreviations. The character Δ is generally used to denote a change, and so is not used for alphabetization here; thus $\Delta\gamma$ would be listed under 'g' for gamma, the latinized spelling of the Greek letter γ.

*	Denotes a complex conjugate
$<>$	Denotes an average, type depending on context
∇	The gradient operator
∂	Partial derivative operator
$\circ$	The convolution operator: $A \circ B = \int A(x)B(y-x)\mathrm{d}x$
$\emptyset$	The empty or null set
$\iff$, *iff*	If and only if
A_{ij}	Lennard Jones parameter for interactions between ith and jth atoms
a_i	Integers
α_c	Structure factor phases calculated from a model
B	Crystallographic "temperature" factor, Debye-Waller factor
ΔB	Error in model temperature factors

B_{ave}	Average crystallographic temperature factor
B_{ij}	Lennard Jones parameter for interactions between ith and jth atoms
β	The inverse of temperature times Boltzmann's constant, $1/k_BT$
C	Completeness, percentage of theoretical diffraction spots actually observed
$\mathbf{C}$	The covariance matrix of parameter estimation and model fitting
ΔC_P	Change in heat capacity
χ^2	Goodness-of-fit
D	Debye, unit of dipole moments = 3.355×10^{-30} Coulomb meters
D	$< \exp[-(\Delta B s^2/4)] \cos 2\pi s \Delta x >$
$\mathbf{D}$	Curvature matrix
D_{ij}	Components of the distance difference matrix, Eqn. 5.3
d	Real-space distance between Bragg diffracting planes
δ	Change in unit cell lengths
δ_{ij}	Kronecker δ-function, equal to 1 *iff* $i = j$, zero otherwise
ΔE	Internal energy difference of two states, often with a subscript indicating a specific contribution
ϵ	Dielectric constant
$\boldsymbol{\epsilon}_i$	Real space basic vectors of a lattice
$\boldsymbol{\epsilon}_i^*$	Reciprocal space basic vectors of a lattice
ε	Multiplicity of a diffracted reflection
$\mathcal{F}$	Denotes a Fourier transform
$F(\boldsymbol{Q})$	Complex structure or scattering factor at reciprocal space position $\boldsymbol{Q}$
$\boldsymbol{F}_c, \boldsymbol{F}_o$	Vectors of calculated or observed x-ray scattering amplitudes
F_p	Structure factors calculated from a partial (incomplete) atomic model
$F_{c,\mathsf{h}}, F_{o,\mathsf{h}}$	Calculated and observed scattering amplitudes with indices $\mathsf{h} \equiv h, k, l$.
f	A number between zero and one
f_0, f', f''	Components of atomic scattering factors
$\boldsymbol{f}_i$	Force on the ith atom
$\boldsymbol{G}$	Reciprocal space lattice vector
ΔG	Gibbs free energy difference between two states
$\Delta\Delta G$	Change in ΔG upon mutation
$G(\lambda_r)$	Contact solvent density at λ_r
g	Factor to put F_o and F_c on a common scale
$g(r)$	Two particle correlation function
γ	Equilibrium value of a dihedral angle
ΔH	Enthalpy difference between two states
h, k, l	An integer, in the context of diffraction; Miller indices of diffraction spots
h	The Miller indices of a diffraction spot
$I, I(\boldsymbol{Q})$	Intensity (of x-rays)
I	Mass inertia tensor
I_γ	The γth eigenvalue of I
I_0	Zeroth modified Bessel function

i	An integer or $(-1)^{1/2}$, depending on context
K_a	Equilibrium constant for an acid titratable group deprotonation
K_l	Force constant for covalent bonds
K_l	Scale factor used in merging diffraction data, see Section 4.2.1
K_n	Force constant of the nth term of the dihedral angle potential
K_θ	Force constant for bond angles
k	A reaction rate, often with a subscript indicating the relevant reaction
$\boldsymbol{k}_i$,$\boldsymbol{k}_s$	Wavevector of incident or scattered x-rays
k_B	Boltzmann's constant, 1.3806×10^{-23} J/K
κ_T	Isothermal compressibility, $-\partial \ln V/\partial p$
$L(\boldsymbol{x}, \boldsymbol{F}_o)$	Likelihood function of $\boldsymbol{x}$ given $\boldsymbol{F}_o$
l, l_{eq}	Bond length, equilibrium bond length
l	Index of a *layer* or diffraction image
λ	Wavelength of light (usu. x-rays)
λ_r	Scaled size of a cavity
$\lambda_0, \lambda_1, \lambda_2$	Langrange multipliers
m	Mosaicity of a diffraction dataset or crystal
m_i	Mass of the ith atom
μ	Chemical potential
μ^{ex}	Excess chemical potential $\mu - \mu^{ideal}$
$\Delta\mu$	Change in chemical potential
$\mathbf{N}, N_{ij}$	The normal matrix of parameter estimation, and its components
N_A	Avagadro's number, 6.022×10^{23}
N	Number of water molecules in a cavity
N_{occ}	Number of atoms in a model
n_{obs}	Number of observed diffracted reflections
O	Observable quantity averaged over many observations
$\boldsymbol{O}, O_i$	Vector of observables or one particular observable
ω	Ratio of outer to inner diameters
p	Pressure
$p(\boldsymbol{x})$	Prior probability of model parameters $\boldsymbol{x}$
P_{act}	Actual measured pressure
p_{ins}	Histogram of water insertion energies
P_{nom}	Nominal (rounded) pressure
p_{rem}	Histogram of water removal energies
P_{yield}	Pressure at which an object deforms inelastically
$P(\boldsymbol{x}, \boldsymbol{F}_o)$	Probability of $\boldsymbol{x}$ given observations $\boldsymbol{F}_o$
p_N	Probability of having N water molecules in a cavity
pH	Negative logarithm, base 10, of the hydrogen ion concentration
ϕ, ψ	Backbone angles of peptides
ϕ_d	Dihedral bond angle
pK_a	$-\log K_a$
Q	Magnitude of the scattering vector

$\boldsymbol{q}$	Scattering vector, $\boldsymbol{q} = \boldsymbol{k}_s - \boldsymbol{k}_i$
q_i	Electrical charge of the ith atom
q_{ins}, q_{rem}	Canonical partition functions of cavity water before insertion or removal of a water molecule
R	The molar gas constant, Avagadro's number times Boltzmann's constant
R	Reliabitliy index, in the context of structure refinement
R	A rotation operator or matrix
$\boldsymbol{R}$	Real space lattice vector
R_{cav}	Radius of a cavity
R_{free}	Reliability index based on data not used for refinement
R_g	Radius of gyration
$R_{\gamma=1,2,3}$	Radii of gyration along principal axes of the inertia tensor
R_{merge}	Reliability index after merging diffraction data
R_s	Solvent radius
r	resolution limit of a diffraction dataset, in Å
r	In spatial context, the magnitude of a real space vector
$< \Delta r^2 >$	An atom's mean squared displacement from its equilibrium position
$\boldsymbol{r}$	A position in real space
$\dddot{\boldsymbol{r}}$	Third time derivative of position
$\{\boldsymbol{r}_i\}$	A set of positions in real space
r_{ij}	Distance between ith and jth atoms
$r_{i,\alpha}$	αth component of the position vector of the ith atom
$\rho(\boldsymbol{r})$	Electron density as a function of position $\boldsymbol{r}$
ρ_{bulk}	Bulk number density of water
ΔS	Entropy difference between two states
s	$2\pi \sin\theta/\lambda$, a common measure of scattering vector amplitude
σ	Standard uncertainty, often with subscript indicating of what
σ_{DPI}	Cruickshank's Diffraction-component precision index
T	Temperature
$\boldsymbol{t}$	Real space translation vector
Δt	Time step (for MD simulation)
θ, θ_{eq}	Angle between two bonds, its equilibrium value
θ_s	Half the angle between the incident and scattered wavevectors
U, U_N	Potential energy, potential for a particular water occupancy N
ΔU	Energy of insertion of one water molecule
V	Volume of a system
$\mathcal{V}$	Potential energy
V_{cav}	Cavity volume in an unfolded protein
V_{cell}	Volume of the crystallographic unit cell
ΔV	Volume difference of two states, a subscript may indicating which system
$\Delta V^\ddagger$	The activation volume of a reaction
v	Solute volume
$\boldsymbol{v}$	Velocity

w_{h}	Weighting factor of diffracted reflection h
$x(v)$	Arbitrary function of solute volume
x, y, z	Real space coordinates
$\boldsymbol{x}, x_i$	Vector of model parameters, the ith model parameter
Δx	Coordinate (model) errors
δx	The “jump” of a parameter which will optimize the model fit to data
Ξ	Grand canonical partition function
$\delta\xi$	The residual for a given model
Y	The isotropic yield strength of a material
y_i	A generic function
z	Activity
ζ	Angle between rotation axis or Be-cell axis and $\boldsymbol{k}_s$

Chapter 1

Introduction

In the last ten years the determination of a protein structure has become almost routine, at least if crystals of the protein can be grown. Concurrently, it has become clear that the old adage *structure equals function* must be revised. Even with a structure in hand, much work is necessary to understand how a protein functions. New methods are needed to elucidate how proteins function, and the ideas that have guided us thus far will need revision.

My work has attacked the problem of protein structure by examining the response to pressure of a cavity-containing mutant of T4 Lysozyme. To start this story, I want to ask three questions. First, why should I want to apply pressure to a protein? Why have I chosen to study a cavity-containing lysozyme? Finally, what motivated the use of crystallography in these experiments?

Once these questions are answered, I will examine in some detail the hydrophobic model of protein structure and folding. Since my experiment is simply to make a thermodynamic perturbation to a protein, we need to first understand the basic thermodynamic properties of proteins. We also need to see where present models fail to explain the available data, which will provide the final motivation for my experiments.

1.1 Why pressure?

Our intuitive experience with biology is limited to one atmosphere, but an enormous fraction of the biosphere is at a large hydrostatic pressure. Earth's oceans have a pressure gradient of roughly 1 bar per 10 m depth, due simply to the weight of water above. The average pressure of the ocean floor is roughly 380 bar and life is found in some of the ocean's deepest trenches where pressures exceed 1 kbar[1, 2]. That life can survive under such extreme conditions is itself fascinating.

We need not invoke extreme biological conditions to see that pressure is a useful tool. Pressure, like any thermodynamic variable, can be used to perturb a system and study its underlying properties. So while pressure may seem remote to life at one atmosphere, it is in fact little different from chemical concentration or temperature.

1.1.1 Pressure effects on proteins

The first pressure experiment on a protein appears to have been the denaturation of albumin from eggs, by P. W. Bridgman, in 1914[3]. Since that time a great amount of work has been done.

Pressure has been implicated in a variety of functional changes for proteins[4, 5]. It has been shown to reverse anasthesia[6]. Almost all proteins unfold under modest pressures[7]. Pressure affects the spectroscopic[8] and kinetic properties of proteins[2, 5].

The compressibility of proteins is quite small, so that at 1 kbar a typical protein compresses only 0.1 to 1%[9]. How is it that such small changes can have such large effects on proteins? Proteins are classic soft systems with many internal degrees of freedom. Soft systems have *energy landscapes* with many energy levels spaced closely together[10]. Even near the true lowest energy state there is a large density of states which samples a significant conformational space. Moreover, the functional states of the molecule are also low lying relative to the true ground state. Thus seemingly small thermodynamic perturbations can significantly affect protein behavior.

1.1.2 Pressure as a thermodynamic variable

In all cases in this work, pressure will be applied to proteins which are surrounded by water–even in the protein crystals described in later chapters, liquid water surrounds the well ordered proteins in the crystal. Thus even though the proteins in the unit cell of such a crystal are in different orientations, the surrounding water ensures hydrostatic conditions and eliminates the problem of pressure gradients in the system. We wish to lay out the basic thermodynamics of such systems.

Under conditions of prescribed temperature and pressure, the relative probabilities p_0/p_1 of a system being in two given states (denoted $0, 1$) may be determined from the relative free energies of those states, namely,

$$\frac{p_0}{p_1} = e^{-\beta \Delta G}, \tag{1.1}$$

where β is the familiar $1/k_B T$ temperature factor ($k_B T \approx 2.5$ kJ/mol at room temperature) and ΔG is the difference in pressure and temperature dependent Gibbs free energies of the two states,

$$\Delta G = \Delta E - T\Delta S + p\Delta V. \tag{1.2}$$

ΔE and ΔS are the differences in internal energy and entropy of the two states. We are here most concerned with last term $p\Delta V$, the pressure multiplied by the change in volume of the *total* system, which includes both the proteins and their surrounding water. There may be (sometimes important) pressure dependences of both ΔS and ΔE.

The kinetics of transitions between the two states may be affected by pressure in an analagous fashion:

$$\left.\frac{\partial \ln k}{\partial p}\right|_T = -\frac{\Delta V^{\ddagger}}{RT}, \tag{1.3}$$

where k is a reaction rate corresponding to a transition between two states separated by a transition state with *activation volume* $\Delta V^{\ddagger}$.

Pressure favors states of our total system with smaller volumes and speeds reactions with transition states smaller than the initial state. A perhaps more subtle point is that in some sense pressure has an effect opposite to that of temperature. While increasing temperature favors states of larger entropy (that is, more atomic wiggling about), increasing pressure favors states of lower volume (most likely less atomic wiggling about). Pressure tends to decrease the volume of a system while temperature tends to increase the system's volume. This analogy is limited since temperature plays a critical role in determining the distribution of states regardless of the details of pressure, volume, or entropy, and because pressure will potentially change ΔS and ΔE. Nonetheless, it is a useful rule of thumb.

Another use of pressure is to observe the "soft" modes of a system. In proteins these might be implicated in protein function (for instance, see the work on hen egg-white lysozyme by Kundrot and Richards[11, 12] or more recently by Refaee *et al.*[13], and the work by Paul Urayama on myoglobin[9, 14].) Computer simulation has shown that the T4 lysozyme active site (to be discussed briefly below) fluctuates strongly[15]. Such fluctuations may correspond to fluctuations in the total system volume, which is related to the isothermal compressibility κ_T by

$$< V - < V >>^2 = k_B T \kappa_T V, \tag{1.4}$$

where V is the system volume. We might then expect regions of high compressibility to be related to functional modes of the protein.

Perhaps most importantly, pressure can help us explore states which, while normally inaccessible, are nonetheless interesting for understanding protein stability and function. Much debate has arisen over the role of water in cavities or in the hydrophobic cores of proteins. While the unfavorable interactions of non-polar "hydrophobic" parts of a protein and water are thought to be the principal driving force in protein folding[16], these interactions remain poorly understood. Walter Kauzmann has pointed out more than once[17, 18] that the hydrophobic model has serious flaws when we consider its pressure behavior (see below). This question is especially topical as the focus of protein folding theory and experiment turns to the the role of water in folding and structure prediction[19–23].

In the absence of a consensus in the theory, it is especially important to understand empirically the interactions of water with the various amino acids comprising protein molecules. While a temperature experiment may tend to increase the likelihood of water penetration into the core, thus permitting a measurement of the associated free energy, it is more likely to first denature the protein. At the very least, any substantial increase of temperature will disrupt the protein structure, making interpretation of the experiment in terms of the original structure more difficult. Finally, temperature and chemical perturbations frequently damage protein crystals.

Pressure allows us a wider working range than other thermodynamic probes. Proteins retain their native structure over a wide enough pressure range (generally 1 to several kilobar) to observe interesting phenomena besides unfolding, such as spectroscopic changes associated with ligand binding (e.g. [8].) Pressure does not damage crystals as readily as temperature or chemical perturbations[9, 12, 14, 24]. Increasing pressure may favor the hydration of the protein interior in general, something to be discussed later in this chapter. Similar statements can be made about, for instance, the spectroscopic states of oxygen binding proteins and their relation to structure[8]. By favoring states which are important to protein function or structure, but are only transient or minimally populated at ambient conditions, we can learn a great deal about the molecule. By using pressure, we can favor these states with minimal perturbation to the overall structure of the protein.

How much can we favor these relevant but depopulated states with pressure? That is, "What is ΔV?" There is no *a priori* answer to this question. Yet much is already known about the volumetric properties of proteins and related molecules.

1.1.3 Volume properties of proteins

Proteins are covalently linked polymers. Their folded structures are stabilized by a combination of hydrogen bonds, disulfide and ionic "bridges" between parts of the chain, steric constraints of the backbone and side chains, and by hydrophobic effects[16, 25]. This last is most important, but least understood, and we will return to it often.

Covalent bonds are least susceptible to pressure effects, as their ΔV of formation is small[4]. Similarly, hydrogen bond formation has a small associated change in volume. We will not be concerned with these in our work.

Dissociation of salts into ions, or the ionization of amino acid residues of a protein, are all associated with non-trivial volume changes[26]. The ionized species always have a lower volume, due to charge-dipole interactions which constrain the fluctuations of bulk water. Since pressure can therefore affect the pK_a values[1] of buffers, we must carefully choose buffers for high-pressure biological experiments.

[1] The acid dissociation constant for the buffer.

Table 1.1 lists ΔV for various biochemical reactions.

Similarly important to our work is the pressure effect on hydrophobic interactions. While the details of the hydrophobic effect should be saved for later, it is interesting to note that transfer of hydrocarbons from their neat liquids or simple organic solvents into water at standard temperature and pressure has a large, negative volume[17, 18]. As the pressure of the system increases, this volume becomes less negative, until finally it becomes positive, usually in the range of 1 to a few kilobar.

1.2 Selecting probes for high pressure experiments

A wide array of probes have been used to study proteins at high pressure, including nuclear magnetic resonance (NMR)[28, 29], fluorescence, small angle x-ray or neutron scattering (SAXS/SANS)[30, 31], fourier transform infrared spectroscopy (FTIR)[2, 9, 30], and optical spectroscopy[8]. What are the advantages and disadvantages of crystallography relative to these other probes?

The power of NMR to solve protein structures continues to grow[32, 33]. But for now it is perhaps best suited to studying dynamics, such as the access of solvent to the backbone studied by ^{1}H/^{2}H exchange studies (*e. g.* [34]), dynamics of side chains[33, 35], or of fluctuations of the backbone (as in ubiquitin[36]). As a structural technique it is more limited. Its weakness lies in that one only measures distances between spin-labelled groups in the protein. There is frequently little or no orientation information, and usually a high constraint to observation ratio is needed to solve a structure. The protein must also be specially prepared; in dynamics studies for instance, side-chain methyl groups must be uniformly sub-

Table 1.1: Volume changes associated with various biochemical reactions. *hydrophobic* refers to disocciation of hydrophobic solutes; *unfolding* refers to protein unfolding. Adapted from [1, 9] and [27].

Reaction type	*Reaction*	ΔV @ 25° C *ml/mol*
ionic	$H^+ + OH^- \rightarrow H_2O$	21.3
	$HPO_4^{2-} + H^+ \rightarrow H_2PO_4^-$	24.0
	protein-COO^- + H^+ → protein-COOH	10
	protein-NH_3^+ + OH^- → protein-NH_2 + H_2O	20
H-bonding	poly(L-Lysine) → helical poly(L-Lysine)	-1.0
	poly(A + U) → helical poly(A + U)	1.0
hydrophobic	methane (CH_4) in hexane → CH_4water	-22.7
unfolding	myoglobin (pH 5, 20° C ≈ 3.5 kbar)	-98
	WT T4 lysozyme (pH 3.6, ambient P and T)	-30

stituted by ^{13}C methyls, without substituting other carbon atoms. Nonetheless it remains the only way to solve a structure in solution, rather than in a crystal. I also find it to be the most convincing method for observing dynamics in a real protein.

UV fluorescence or absorbance is often used to probe global structure of a protein, usually through the fluorescence shift upon exposing aromatic side chains, such as tryptophan, to water after unfolding[37]. SAXS/SANS experiments observe changes in global structure by measuring the radially averaged density autocorrelation function, from which the radius of gyration of a protein can be derived (e.g. [20, 31]). FTIR experiments monitor secondary structure and occasionally other properties of proteins[38]. In all cases the experiments are relatively straightforward, but provide limited information.

Optical spectroscopy is exceptionally useful to study ligand binding if a signal is present, as for CO or oxygen binding in hemoglobin[8], or in the many fluorescent proteins now available. Ligands can also help to assay unfolding, *e. g.* the spectroscopic shift of a heme group upon unfolding of the surrounding protein in metmyoglobin[17].

All of these techniques are useful probes of the general structure of a molecule, but lack detail. Which helix unfolded? Was the tryptophan UV fluorescence quenched by some other mechanism?

Current thinking suggests that protein function is conferred first by structure and then by the fluctuations of that structure. Proteins are miniature machines. To understand the function of proteins, we must understand exactly which atoms are going where. Probing the structure with less than direct means is akin to a mechanic attempting to diagnose your car's problems by looking at the color of the exhaust. It is possible, even aesthetically pleasing, but difficult and prone to error.

It has become increasingly possible to simulate protein molecules on the computer, and to compare results of simulations with experimentally determined parameters such as the radius of gyration[31] or noble gas binding constants[15, 39]. This places an additional demand on the experimentalist, namely to put the simulation to a full test by directly determining atomic coordinates. Especially in pressure experiments, crystallography allows us to directly observe local compressibilities, deformations, and volumes. (How small a disturbance we may accurately detect is a question for Chapters 4 and 5.) Simulations and other models may then be tested directly.

Crystallography also provides us with the ability to the appearance of bound ligands or water molecules which attach to a protein in response to some thermodynamic impetus. Even other structural techniques may struggle to see something that is not part of the original atomic model. X-ray diffraction data, by directly sampling the electron density, can in the final stages of refinement highlight areas where there is something missing from the model. We will find this fact very important in the analysis of T4 lysozyme under pressure.

Crystallography does have limits. The very nature of the experiment precludes any direct knowledge of unfolding processes or the unfolded state. Despite some inferences that can be made about dynamics from thermal B-factors, it remains a static technique[2]. Pressure allows us to push these limits somewhat. By combining Equation 1.4 with parts of the protein identified by crystallography to deform

[2] Though this is changing as new theoretical and experimental crystallographic methods gain more of a following.

very little under pressure, we may identify parts whose fluctuations are large in the folded state. These regions may provide insight into unfolding or functional processes.

1.3 T4 lysozymes

We are concerned here with guiding principles of protein structure and function, not with the particulars of a given molecular or biochemical system. It is important to choose a system which is representative yet straightforward to work with, already well characterized, and a protein which we believe will respond to pressure in some interesting fashion.

I have chosen bacteriophage T4 lysozyme. This enzyme is produced by *Eschericia coli* following infection with bacteriophage T4[40]. Late in infection, it cleaves cell wall saccharide bonds, allowing replicate phage particles to escape and continue infecting other bacteria[41]. Its functional purpose is similar to that of the more familiar hen egg white lysozyme (HEWL), although its activity against various cell walls differs from HEWL.

The structure of T4 lysozyme was first solved by Brian Matthews[40], and is similar in some respects to HEWL. The structure is shown schematically in Figure 1.1. There are two domains, one a mix of β-sheet and α-helix (herein the N-terminal domain), the second a bundle of α-helices comprised of residues 1-13 and 80-162 (the C-terminal domain.) The active site rests more or less between these two domains, and a long α-helix connects them.

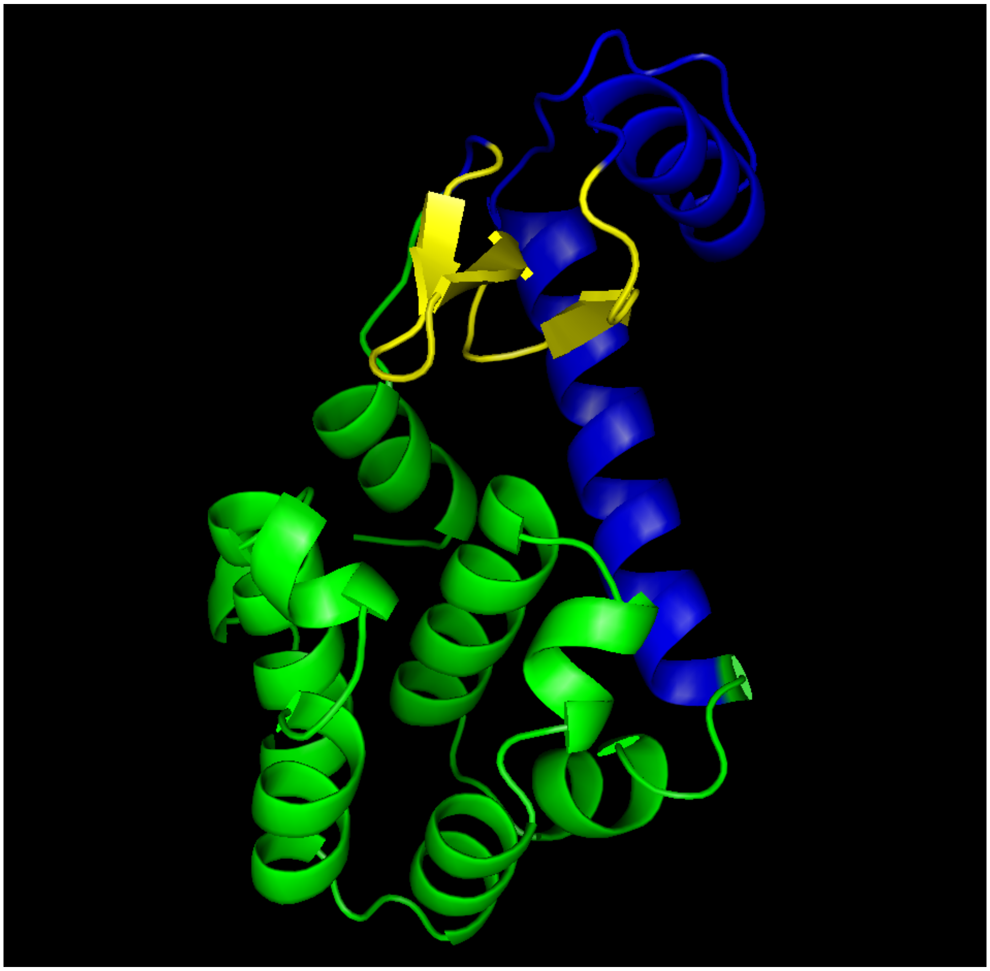

Figure 1.1: Schematic structure of wild-type phage T4 lysozyme, showing ten α-helices, a β-sheet (in yellow), and loops. In this view, we are facing the active site (between the β-sheet and the C-terminal domain (green). PDB code 1L63, available from www.rcsb.org.

1.3.1 Structural and thermodynamic studies

A dizzying array of studies have explored the structural stability and activity of T4 lysozyme. Early work suggested that residues far from the active site could affect activity[42]. Site-directed mutagenesis, beginning with Thr157→Ile[43], has provided a wealth of structural and thermodynamic information about this molecule. That first study showed that this one mutation could lower the unfolding temperature of the molecule by more than 10 degrees Celsius. Some substitutions affect stucture but (surprisingly) not stability, such as mutations of Proline 86[44]. Others affect stability by improving hydrophobic interactions[45], or by introducing charges or dipoles which interact with helix dipole moments[46].

This last case is an interesting example of *rational design* of proteins. Aspartic acid residues were substituted for native residues at the N-terminal ends of two helices B and J. The authors chose mutations based on sites in the crystal structure where substitutions would not be hindered sterically. The negative charge of these acidic residues interacts with the intrinsic electrostatic dipole moment of the helix, reducing the internal energy.

In this work we are continuing studies of cavity containing mutants originally produced to study hydrophobic interactions that stabilize the protein interior. As mentioned above the free energy of forming a cavity has a volume dependent part $p\Delta V$, but also a change in internal energy ΔE, and in principle a change in the entropy of neighboring side chains. Originally only a few mutants were produced[47], but they were followed by more extensive studies[48, 49]. Many of these mutants were found by crystallography to collapse partially or even completely and with a range of thermodynamic and structural effects[47–49]. In the case of the Leu99

→ Ala (L99A) mutant studied here, the cavity is actually slightly larger than one would predict by removing the corresponding atoms out of a computer model of the wild type protein.

There are a number of caveats, but the major conclusion from these studies was that the mutation-induced change in Gibbs free energy difference between the folded and unfolded states $\Delta\Delta G$ is well estimated with two contributions. One contribution accounts for the difference in free energies of transfer of the original and subsituted amino acids from a non-polar to polar environment, as from octanol to water. This depends only on the specific mutation and adds roughly -2 kcal/mol for the L99A mutation. A second term accounts for non-polar interactions (which may include changes in entropy, as well as the more obvious loss of Van der Waal's interactions), with a value of -22 cal mol^{-1}Å^{-3}[48]. Then, for Leucine to Alanine substitutions and in the absence of pressure, we have approximately

$$\Delta\Delta G = -2\,\mathrm{kcal}\cdot\mathrm{mol}^{-1} - (22\,\mathrm{cal}\cdot\mathrm{mol}^{-1}\text{Å}^{-3})\Delta V_{cav}, \tag{1.5}$$

where ΔV_{cav} is the cavity volume introduced by the mutation L→A. For the L99A mutant $\Delta\Delta G \approx 8\,k_B T$, which already suggests a great deal about the mechanical properties of the protein. This energy is quite large, and we cannot imagine a liquid supporting such a cavity under any positive applied pressure. This observation is interesting in the context of the debate over whether the interior of a protein is liquid or solid and will be discussed below and in the final chapter. At a pressure of 2 kbar (200 MPa), we can roughly double the free energy of the cavity:

$$pN_A = 200\,\mathrm{MPa}\ \times 6.02\times 10^{23}\mathrm{mol}^{-1} \approx 120\,\mathrm{J}\cdot\mathrm{mol}^{-1}\text{Å}^{-3}\ \approx 30\,\mathrm{cal}\cdot\mathrm{mol}^{-1}\text{Å}^{-3}$$

Thus pressurizing this protein should provide an interesting measure of the liquid versus solid nature of the protein core.

Another interesting feature of this cavity is its accesibility to the outside solvent ([50] and see below). Might pressure affect this pathway? Might we force water into the cavity? (The $p\Delta V$, ΔE_{vdw} and ΔS terms associated with such an event are all relevant.) These are obvious questions to ask, both of which are intimately related to our understanding of protein structure.

1.3.2 T4 lysozyme under pressure

Very little is known about bacteriophage T4 lysozymes under pressure. The noble gas studies[39] already mentioned were carried out under modest pressures up to several tens of atmospheres. The results of that work have also been modelled using molecular dynamics[15]. These studies will be discussed in some detail later. Here it is good to note that simulation and experiment find the same binding sites for Xenon, and reasonably good agreement for the occupancy fractions.

The only study above these pressures was by amide hydrogen deuterium exchange NMR[34]. In this tedious experiment, the WT* (cysteine free "psuedo wild-type") mutant was rehydrated in deuterated water (D_2O) and pressurized at 9 steps between 0.1 and 200 MPa, for a range of times up to about 3 hours. During this time, deuterons exhange with backbone amide hydrogens, an effect reflected in the 2D NMR spectra. The samples were then transferred quickly to an NMR instrument for ^{15}N-^{1}H measurements. By examining the kinetics of the H-D exchange reaction, some conclusions can be drawn about the conformational flexibility.

The data are noisy but two main points are clear. The first is that there are many pathways into the core of the protein, manifested in the lack of correlations between reaction activation volumes of neighboring residues. Second, and most interesting, the activation volumes corresponding to structural rearrangement in the region of the cavity are almost all small and negative. This may indicate that water penetration is more routine that we might normally expect.

There are problems with this experiment. Perhaps most difficult is the tacit assumption that the structure has not changed, implying zero change in chemical shifts of the ^{15}N NMR spectra. How good an assumption this is remains to be seen. Also, the activation volume determined from the data is modelled as a combination of three terms[34],

$$\Delta V^{\ddagger}_{obs} = \Delta V_s + \Delta V^{\ddagger}_{k_{\mathrm{OH}}} + \Delta V_{K_w}. \tag{1.6}$$

$\Delta V^{\ddagger}_{k_{\mathrm{OH}}}$ is the activation volume of the exchange reaction itself, and ΔV_{K_w} is the volume change upon ionizing water[3]. ΔV_s represents the structural component of the process. It amounts to an estimate of protein structural change needed to allow water in prior to exchange. Since the experiment measures $\Delta V^{\ddagger}_{obs}$, the accuracy with which ΔV_s is determined depends on the knowledge of the second and third terms on the right hand side of Equation 1.6. Constant values are used for both $\Delta V^{\ddagger}_{k_{OH}}$ and ΔV_{K_w}. This seems reasonable for ΔV_{K_w}, but we might wonder if $\Delta V^{\ddagger}_{k_{\mathrm{OH}}}$ varies from site to site, and if so how much. Finally, we are left to wonder how the results might be different for the L99A mutant.

Pressure perturbation calorimetry has been carried out on T4 lysozyme mutant WT*[27] and on other mutants (private communication of the authors.) This technique measures the slight changes in heat of unfolding a protein under modest pressures. The most striking finding for us is that the difference of *thermal* unfolding volumes is quite similar to the difference in cavity volumes introduced by mutation. This implies that whatever the unfolding mechanism, the unfold-

[3] OH^- ions dominate over water in the exchange process.

ing volume does not seem to depend directly on the solvation properties of the substituted residue. Instead the unfolding volume appears to depend only on the physical size of the residues. This experiment, despite its name, is not carried out at large pressures, so it is not sampling the pressure unfolded state, but rather the volume difference of the folded and thermally unfolded state. We must therefore take care not to extrapolate from these results to the high pressures of our experiments.

1.4 Hydrophobic models and protein folding

The definition of "hydrophobic" is somewhat vague. In fact it describes an extraordinarily broad class of phenomena which are loosely connected by the observation that certain materials are not easily miscible with water. The most commonly cited example is the demixing of water and hydrocarbon oils, such as hexane or octanol. Walter Kauzmann[51] was the first to suggest that perhaps protein structure might be explained by such phase separation on a microscopic scale. Kauzmann's observation has had great success in explaining how and why proteins should rapidly collapse into a non-random, folded state[16, 25]. The model suggests that the dominant force in protein folding is the transfer of hydrophobic, low polarizability amino acid side chains from water to a non-polar environment. Experiment generally confirms the intuition that non-polar "hydrophobic" residues like Leucine or Valine are primarily found in solvent inaccessible regions of the protein[16, 52]. The model successfuly predicts certain features of the entropy and enthalpy of unfolding as a function of temperature[16], but as Kauzmann himself pointed out[17, 18], it has some curious flaws. Before considering those flaws, we need to understand some basics of hydrophobicity.

1.4.1 Hydrophobic solutes in water

Several important features characterize the solvation of small hydrophobic species such as methane in water. At room temperature, this solvation is disfavored by a large entropic penalty, and a much smaller enthalpic penalty[16]. Solvation of small nonpolar species is also associated with a large increase in the heat capacity of the solution[16]. We will now consider these features, and their temperature dependence.

The entropic penalty is certainly the most discussed feature of hydrophobicity, and one which continues to be a matter of heated debate, though there seems to be little objective doubt as to the nature of this penalty. First note that the concept of an entropic *penalty* for mixing is in itself odd. The translational entropy of mixing is always positive. We must look at other degrees of freedom of water to find an answer.

In the classic picture, water becomes orientationally ordered, forming a hydrogen bonded "clathrate" cage around a hydrophobic solute (*e. g.* a model from Harold Scheraga's group[53, 54] which has recently been revisited[55]). This model considers the energy levels E_i of a system, where the index i indicates the (integer) extent of hydrogen bonding of each water molecule. Then one simply calculates the equilibrium distribution over these energy levels, subject to some reasonable constraints. I would disagree with the authors that the agreement with experimental data is excellent, since for many thermodynamic parameters available experimental data indicate linear dependence on temperature, while the model has non-linear temperature dependence. The more recent model[55] does much better in this regard.

Such clathrates do in fact form, for instance around methane, but only at somewhat elevated pressures[56]. (The reasons for this will be discussed later in this chapter.) While such a clathrate picture can guide a picture of the underlying energy levels in the system, it is unclear to me that those energy levels map back only to a clathrate picture, and whether the energy spectrum used in the clathrate model[53–55] is realistic. Typically these models are compared to x-ray and neutron diffraction data (*e. g.* [57]) though the authors admit there are large uncertainties in the pair correlation functions derived from that sort of scattering[55, 57]. Recent XAFS[4] experimental evidence, supplemented with computer simulation[58] reveals that the hydration around Krypton in water is not well modelled by a clathrate picture. Unlike x-ray or neutron measurements, the XAFS data are in principle easier to interpret, especially when the distribution of water around a guest solute is the object of study.

I find more recent classical field theories both more elegant and appropriate, though they are necessarily more complicated. Moreover they do a somewhat better job of modelling experimental data. I will not review these theories in detail here, as they involve extensive field-theoretic and information theory calculations, the details of which will not be important later. Instead I point the reader to good reviews and applications[56, 59–61]. Lawrence Pratt has written an excellent review of the subject[62] and two other papers[56, 63] have excellent references. The goal here is to point out the flavor of the theory, for later comparison to the situation in our pressure-hydrated protein cavity.

The central question to ask is simple: given the ordinary structure of water,

[4] X-ray absorption fine structure

what is the probability of finding a void between water molecules large enough to insert a solute? In its simplest form, this approach treats all particles as hard spheres (other interactions can be added in a straightforward manner), but as we'll see uses experimental input to distinguish water from other solvents. The distinguishing feature of water is in its experimental two-particle correlation function, an observation that led to the first modern theory of hydrophobicity[64]. Alternatively, the correlation function can be derived from simulation (*e. g.* , [56, 62]).

It is convenient to write the excess chemical potential of a non-interacting hard sphere solute in water,

$$\mu^{ex} = -k_B T \ln p_0 = 4\pi\rho k_B T \int_0^{\lambda} G(\lambda_r)\lambda_r^2 \mathrm{d}\lambda_r, \tag{1.7}$$

where $\lambda = R_{cav} + R_S$, the sum of cavity and solvent radii, ρ is the number density of water, p_0 is the probability of the solute volume being empty, and $G(\lambda_r)$ is the "contact value of the solvent density at the surface of the exclusion volume"[56]. (Or, the value of the pair correlation function at the contact radius λ_r.) Equation 1.7 is at first mysterious, but is easy to derive. Take an empty cavity already of radius r. Then the probability of a spherical shell around the cavity being empty is $1 - 4\pi r^2 \rho G(r)\mathrm{d}r$. The probability that the cavity and the shell are empty is the product of the two separate probabilities:

$$p_0(r + \mathrm{d}r) = p_0(r)(1 - 4\pi r^2 \rho G(r)\mathrm{d}r) = p_0 + \frac{\partial p_0}{\partial r}\mathrm{d}r.$$

This can be recast in the form $\partial \ln p_0/\partial r = -4\pi r^2 \rho G(r)$, from which it is trivial to derive Equation 1.7.

Equation 1.7 describes the radius dependent surface tension of water around the non-interacting solute. The function G is peaked about some λ_r. What turns out to be different about water is not the position of this peak, but its width: it is considerably narrower than the peak for a similarly sized nonpolar solvent[62]. A more recent approach[56] matches a simple model for the probablilites p_N of finding N water molecules in the solute volume to known information about the moments $< N^k >$ (the brackets indicate a thermal average), determined from Monte Carlo simulation. Others[61] have constructed more traditional theoretical models as well.

Once the probability p_0 is known it can be used to calculate thermodynamic quantities. In these pictures, the problem is entirely entropic: the excluded volume of the hard sphere solute limits the density fluctuations of the solvent, and therefore its entropy. But there is no particular reason to limit ourselves to hard spheres. Information about interactions is contained in $G(\lambda)$. Hydrogen bonding further limits density fluctuations, narrowing the first peak in $G(\lambda)$ and making solvation even more unfavorable. Some of the details will be considered below in section 1.4.2.

As the solvent size grows, this fluctuation-entropic picture will eventually fail. For a sufficiently large spherical cavity, water lining the cavity will no longer be able to satisfy all of its hydrogen bonding potential. Indeed at this point, the excess chemical potential of the solute goes from a solute volume dependent regime to a surface area dependent regime, and the solvation penalty becomes increasingly enthalpic. One group calculates that the transition between these regimes is near 1 nm, curiously close to the size of hydrophobic groups in proteins[61].

The solvation entropy also has an interesting temperature dependence. For small hydrophobic solutes, ΔS for transfer from a gas to water nearly converges to zero near 400 K[16, 56, 65]. The information theory model (discussed below in Section 1.4.2) maximizes the Shannon information entropy contained in the probabilities p_N of having N water molecules in a given volume. It is able to predict the entropy convergence reasonably well. This prediction arises again from the fact that the chemical potential of solvation, Equation 1.7, is almost entirely determined from properties of the solvent. In particular it has been shown[56] that the first term in an approximation to equation 1.7 is

$$\mu^{ex} \approx T\rho_{bulk}^2(T)x(v) \tag{1.8}$$

where T is the temperature, ρ_{bulk} is the density of water, and $x(v)$ is a function of solute volume v. Since $x(v)$ only scales μ with no temperature dependence of its own, μ^{ex} is peaked at the same temperature for all v. Thus $S = -\partial\mu^{ex}/\partial T$ converges to zero at a common temperature for all nonpolar solutes. Including more terms in the approximate expression 1.8 introduces small changes which slightly improve the modelling of experimental data[56].

The temperature dependence of entropy is associated with the large increase in the heat capacity upon solvating nonpolar species in water (denoted ΔC_p),

$$\Delta S(T) = \Delta S(T_S) + \int_{T_S}^{T} \frac{\Delta C_P}{T}\mathrm{d}T, \tag{1.9}$$

where T_S is often taken to be the temperature of entropy convergence. This implies a similar temperature dependence of the enthalpy of solvation:

$$\Delta H(T) = \Delta H(T_H) + \int_{T_H}^{T} \Delta C_P \mathrm{d}T, \tag{1.10}$$

where T_H is defined similarly to T_S. Among other things, this implies that the solubility of nonpolar species has a minimum as a function of T, which turns out to be quite a bit above room temperature[16].

1.4.2 Hydrophobic interactions

The models discussed above have significant advantages in that they are more predictive than the old schematic picture of clathrate formation. In particular, protein folding in a hydrophobic model depends not only on the existence of a free energy minimum upon phase separation of hydrophobic residues, but on a relatively long-ranged attractive force between hydrophobic groups in water. Tranfer experiments measure the difference in solubility between an oily solvent (such as octanol) and water; low solubility of the hydrophobic group in water compared to oil does not necessarily imply the existence of an attractive interaction between hydrophobic groups in water. In any case, the form of the interaction is needed if we are to calculate protein folding kinetics and pathways.

We are interested in the *potential of mean force* (pmf) between two or more hydrophobic groups in water. This pmf is just the free energy of the whole system for a given configuration of the weakly-interacting, hard sphere "hydrophobic" solutes, generally plotted against some spatial separation of the hydrophobic objects.

One approach[56, 59], determines the pmf from information theory. Simulation is used to generate many configurations of water, from which the first and second moments of the number of water molecules N in a volume v are calculated:

$$\begin{aligned} < N > &= \rho_{bulk} v, \\ < N(N-1) > &= \rho_{bulk}^2 \int_v \mathrm{d}\boldsymbol{r} \int \mathrm{d}\boldsymbol{r}' g(|\boldsymbol{r} - \boldsymbol{r}'|). \end{aligned} \tag{1.11}$$

These quantities constrain the probabilities p_N since $< N^k >= \sum_N p_N N^k$. Maximizing the information entropy represented by the p_N, subject to the constraints in Equation 1.11, yields $p_N = \exp(\lambda_0 + \lambda_1 N + \lambda_2 N^2)$. The Lagrange multipliers λ_i are chosen to satisfy the moment equations. Other models of the p_N are shown to model available data and simulations less well.

The key finding from this approach is that there is a solvent-separated minimum in the pmf, at least for small solutes. The calculation finds two minima in the potential between two methane molecules, one at 0.39 nm, and a second at 0.73 nm. This pmf implies that there are attractive "hydrophobic" interactions, but that they are interrupted by a free-energy barrier which peaks near 0.55 nm separation. This will be important when we consider pressure unfolding.

The chief limitation of this model is that it assumes Gaussian statistics for density fluctuations[61], which appears to be a very good estimate for small length scales, but breaks down at larger length scales. To address this issue, Lum *et al*[61] have constructed a theory which treats the slowly varying, long ranged part of the density fluctuations separately from the short length scale part. This yields some interesting results.

First, they show that nonpolar, noninteracting plates or cylinders will attract each other in water, with ranges of tens of Ångstroms. To be clear, they model a system where in vacuum the plates or cylinders would not interact in any way (although later they add in explicit polarizability). Thus the observed attraction is *solely a property of water*, upon which the plates or cylinders place particular boundary conditions. Perhaps more interestingly, they find two solutions for the water density between the hydrophobic objects. They point out that there is a kinetically frustrated first-order transition between a high- and low-density state of the water between the hydrophobic surfaces, which they refer to as a drying transition. In this model, hydrophobic objects lead to decreased density at their surfaces over some range away from the surface. When these ranges overlap for two hydrophobic objects, there is higher water density (and pressure) outside than in between, and the objects are pressed together. Finally, the model predicts that the crossover from small-solute to large-solute behavior (discussed above) is near 1 nm.

This model predicts hydrophobic association, at least for sufficiently large objects (for which $R_{cav} \gtrsim 0.5\,\mathrm{nm}$ in equation 1.7). It may also be useful in understanding the pressure-dependent behavior of proteins.

1.4.3 Hydrophobicity in proteins

The most frequent response I get to the question "How do we know that hydrophobicity is the driving force behind protein structure and stability?" is that x-ray crystallography has shown that most hydrophobic residues are buried far from water in most proteins. Indeed this is a good observation, and does constitute an important part of the available evidence[16]. It is not, however, the best evidence. The thermodynamic properties discussed above give us a good starting point.

Most amazing is the fact that the specific entropy[5] of unfolding for proteins converges to a common value quite near 400 K[16, 66]. There is also a concurrent convergence of the specific enthalpy of unfolding, at a temperature very close by[65]. In fact much of the thermodynamic behavior of proteins can be predicted reasonably well based on the transfer of model hydrocarbons into water[65, 67]. This is the best evidence for a hydrophobic microphase separation model of folding, and is the basis of the "liquid hydrocarbon" model of the protein interior (discussed further below and in the final chapter).

Similarly, the stability of proteins *does* have a maximum at a finite temperature (generally between 0 and 40° C)[2]. Cold denaturation is an observed phenomenon,

[5] Entropy per mole.

and this also agrees well with hydrophobic models.

Another important piece of evidence are the cavity studies on T4 Lysozyme[39, 47, 48, 68] and on barnase[69]. A major conclusion from these studies was that the stability difference, $\Delta\Delta G$, between wild-type and cavity-containing proteins included a cavity-volume or area independent term equal to the difference in free energies of transfer from oil to water of the wild-type and substituted residue. For instance, in the L99A mutant of lysozyme studied in this thesis, Leucine has approximately 2 kcal/mol higher a free energy of transfer from hydrophobic solvents to water than Alanine[47]. Thus the L99A mutant is 2 kcal/mol *less* stable than the wild type lysozyme, even before we consider the direct effects of the cavity. This effect is conserved over many different hydrophobic substitutions, and is good evidence that hydrophobic effects play an important role in protein folding.

For all this evidence, and as has been discussed in Chapter 1, the model has some problems. These will be discussed next.

1.4.4 Unresolved problems

The biggest problem in the hydrophobic picture of proteins is that proteins are not small solutes, and neither are their amino-acid side chains. In fact, it appears unclear whether hydrophobic stabilization in proteins is volume dependent or area dependent[61]. This fact may help explain the experimental ambiguity between area or volume dependence of the change in protein stability $\Delta\Delta G$ in the cavity studies[47–49] that led to our present work.

One of the most curious issues, reviewed by Barry Honig[67], is that while many thermodynamic properties of proteins are well modelled by a liquid hydrocarbon model, the actual stability of folded proteins is not. The heat capacity change on unfolding correlates fairly well with buried hydrophobic surface area, but the free energy of unfolding does not correllate with heat capacity, and therefore with hydrophobicity of the core. Since unfolding entropy apparently does correllate with hydrophobic effects, it must be that the enthalpy of unfolding becomes *less* positive as the core becomes *more* hydrophobic. This issue is difficult, and will be discussed somewhat in Section 7.1.3.

While that issue is particularly challenging to understand, a number of problems with the hydrophobic model are amenable to studies with high pressure crystallography.

Pressure unfolding of proteins

From a thermodynamic perspective, one of the biggest problems is the pressure denaturation of proteins, wholly unaccounted for in a simple hydrophobic model[17, 18] in which the protein pressure unfolds as it would at high temperatures. Mixing of hydrocarbons and water is associated with a large, negative volume (as much as -100 mL/mol) at low pressures, but at high pressures this volume becomes small and positive. Zipp and Kauzmann[17] showed for myoglobin that the contribution to unfolding volume from hydrophobic effects would have to be large and *positive* at low pressures and small and *negative* at higher pressures[6]. This is exactly opposite the behavior of short hydrocarbon chains.

Whatever the pressure unfolded state may be, it is not the same as the thermally or chemically unfolded state. This can be demonstrated by, among other things, careful measurement of the radius of gyration of the various unfolded states[30,

[6] Notably the hydrophobic contribution to the unfolding volume is a derived and somewhat subjective quantity, see Zipp and Kauzmann[17] for details.

31]. Staphylococcal nuclease (SNase) unfolds from a compact state (radius of gyration $R_g \approx 18\,\text{Å}$ at ambient conditions) to a much larger state ($R_g > 45\,\text{Å}$) when thermally unfolded. In contrast, the pressure unfolded state has a radius of gyration closer to 35 Å[30]. The pressure unfolded state also retains a great deal of its secondary structure[2, 31]. If nothing else, the pressure dependence of ΔV indicates that there is a change in the character of the unfolded state.

Better knowledge of the pressure-unfolded state is needed if we are to fit it into our picture of protein stability. Given that folded proteins have near optimal packing densities[70], we must wonder how it is possible to reduce their size by unfolding the protein. Intuitively, this should produce the opposite result.

Molecular simulation provided the first suggestion that under pressure water molecules penetrate into the "hydrophobic core" of the protein. Hummer and colleagues[56, 59] began by examining the thermodynamically averaged attractions between two methane molecules in water, using their information theory model discussed above. As mentioned before, it turns out that this system has two stable states, one where the two methane molecules are in contact, and a solvent-separated state. As pressure is increased the equilibrium shifts from the contact state to the solvent separated minimum. The kinetic barrier between the two states also changes. Based on their simulation, the authors hypothesized that the pressure unfolded state included a solvated protein interior and a partial loss of native structure.

More recent simulation work has again implicated internal hydration in pressure unfolding. In their study of pressure and temperature denaturation of a small protein fragment, Paschek and Garcia[71] showed similar behavior for a β-hairpin fragment of protein G. Vaitheeswaran *et al.*[72, 73] have continued by exploring the energetics of water molecules in non-polar cavities formed from random carbon spheres, fullerenes or nanotubes. They have shown that for sufficiently large cavities, where multiple water molecules can share some hydrogen bonds, the cavity will be filled.

Connecting these results to experiment is difficult: the latest simulation results for cavities are for cavities which are closed[72], so that no water can actually enter or exit. Instead, unphysical trial moves allowed in Monte Carlo simulations are used to determine the relative thermodynamic energies of the states in question. We could not construct such a situation in a physical experiment. Moreover, the simultations found that for water-filling of fullerenes to be favorable, the cavity inside the fullerene needed to be larger than any single cavity regularly found in a real protein[72].

Paliwal *et al.*[31] have observed the radius of gyration of Staphylococcal nuclease (SNase) by small angle neutron scattering (SANS) as a function of pressure, and also made molecular dynamics (MD) computer simulations of the protein. The simulations were able to reproduce the SANS data quite well, though at a higher pressure scale. The MD simulations suggest that penetration of water into the core of the protein is significant, and that much of the secondary structure is retained on unfolding. This work made an important step forward, but direct observation of internal hydration of the protein remained out of reach.

Kinetic pressure unfolding experiments, particularly those of Cathy Royer and Roland Winter over many years, have shown a variety of interesting behaviors[2, 20, 30, 74–77]. Their measurements indicate a range of transition state volumes and behaviors, some of which suggest that the key step in the pressure unfolding process is hydration of the protein core[20]. What connection this has to folding in general remains to be seen, but a connection might be made if Hummer's hypothesis can be verified. In that case, we might generally expect a kinetic barrier between a state with many buried water molecules and the dehydrated, folded state.

All of this highlights a critical physical issue in protein folding: *just how does the hydrophobic effect work in proteins?* While a great deal of evidence suggests that the hydrophobic effect is dominant in folding, very little effort has been made to include the effect explicitly in protein structure prediction. One may look to the pioneering work of Harold Scheraga or Peter Wolynes for evidence that, in fact, water has been treated until recently as a nuisance to be avoided. This approach is understandable given the complexity of hydrophobic effects (e. g. [60–62]), and has been somewhat successful. However as research begins on transition states and intermediates in folding, we must be cautious not to judge the validity of these intermediates based on the accuracy of the final, folded state. Only recently have researchers explicitly considered solvent effects in folding[19, 23, 78], with novel and quite interesting results. In all cases long-ranged interactions electrostatic and hydrophobic interactions between amino acid side chains were shown to be water mediated. One of the studies[19] found that water was expelled from the protein core only after most of the protein had folded. In the other case[23, 78], water appeared to *smooth* the inherently rough energy landscape of a protein, making the folding process *less* frustrated. Work from David Chandler's group[60, 61] has shown that water-mediated hydrophobic interactions can lead to long ranged attractions between model cylinders with hydrophilic outer faces and micelle formation of surfactants. These theoretical results motivate experiments on the interactions of water and protein.

Polarity and polarizability of protein interiors

Another prediction of the hydrophobic model is a small polarity and polarizability of the protein interior, since the key feature of hydrophobicity is a relative inability of the solute to interact with water electrostatically. (Hence the frequent interchange of *nonpolar* and *hydrophobic.*) Indeed, if we take the liquid hydrocarbon model literally, we would expect an enormous penalty for transferring water into a hydrocarbon protein interior. For some oils, this can reach $7\,k_BT$[79]! Polarizability of the protein interior is not easy to measure. It has been measured in Staphylococcal nuclease through a clever, but quite indirect, method[80]. A titratable acidic residue is buried in the hydrophobic core of the protein, and the pK_a of the residue measured. pK_a values are sensitively dependent on the dielectric constant of the surrounding medium, since the medium will screen the charge present upon deprotonation of the residue.

The result is striking: the pK_a measurements indicate that the static (zero-frequency) dielectric constant is quite high[80, 81]. Why this should be has remained somewhat unclear[82, 83], but it has been suggested that water penetrates the hydrophobic core of the nuclease and electrically screens the titratable residue[82].

We might also suspect that the hydrophobic model is at least incomplete because of the presence of a large number of dipoles inside a protein, namely, the peptide backbone of the protein. Each peptide group has a dipole moment larger than that of water, a fact which is crucial to helix formation. As mentioned above this also leads to a large polarity of helices, which can be exploited to stabilize proteins in their folded states[46]. The interior of a protein cannot really be like liquid oil.

Cavities and water in proteins

Determining the presence, volume and solvent accessibility of a cavity in a protein is challenging; it will be left for Section 5.2.3. Here, all we need to know is that a cavity (or void) exists when a ball of a given radius can be placed inside the protein, but cannot escape to the outside solvent via any static path.

Virtually all proteins have cavities, usually totalling about 1 per cent of the total protein volume[52, 84]. The $p\Delta V$ energy of even a 150 Å cavity is quite small at 1 atmosphere; thus volume effects on the system are small at this pressure. As we increase pressure, the protein must find some means to reduce that volume and pack more densely. We would expect that if the interior is strongly hydrophobic, water will remain excluded, despite the obvious decrease in volume if the cavity is filled. Determining the structural relaxations of protein cavities is thus helpful to understand the hydrophobic effect in proteins.

A few experiments have identified water buried deep in the hydrophobic core of a protein at ambient pressure, though some have been marred by experimental ambiguities. X-ray crystallography experiments that followed the polarizability experiments mentioned above found water buried in the core of a Staphylococcal nuclease mutated to have a charged residue in the core[81, 82]. This result may have been an artifact of crystal freezing[83]. NMR experiments can in principle detect buried water, as in human interleukin-1β[85]. Others[86] have suggested potential flaws in this observation, though some later crystallographic evidence may have vindicated the original result[87].

The role of water in cavities (and of cavities in general) is poorly understood.

Some see water in cavities as stabilizing[88] based on the argument that any attractive interaction (here, van der Waals forces) is stabilizing. This ignores the free energy lost when that water molecule is removed from bulk solvent. Buried water is ubiquitous in proteins[52, 84], but for it to be buried at ambient pressure it was thought necessary that the water hydrogen bond directly to the protein[52, 89]. This does not preclude the possibility of water residing in non-polar cavities[69, 72], but Zhang and Hermans[52] suggest that, even in the case of a large cavity with many water molecules, interactions with the protein are still needed to stabilize the cavity hydrated state.

This point might seem moot, but water is believed to play functional roles in a number of proteins, often in non-polar cavities. Bacteriorhodopsin[90, 91], cytochrome p450[92], and heme-copper oxidases[90] are all examples of proteins where internal hydration is thought to be crucial to their function, and where the presence or absence of water in hydrophobic cavities has been somewhat controversial. Water also can catalyze structural change[68]. Moreover, dehydration of core-forming residues is increasingly thought to be a rate-limiting step in folding[19, 20], at least for small proteins of 100 residues.

Without some experimental determination of the free energy of transferring water from solvent to the interior of the protein, it is difficult to make much of a conclusion about the role of water in protein structure. Indeed it is difficult to say how "hydrophobic" the interior actually is. No direct measurement has yet been made of the polarity of the protein interior.

Mechanical aspects of proteins

Finally we would like to consider the mechanical properties of proteins. In particular, debate over the fluidity and packing properties of proteins persists[65, 93]. Crystallographic B-factors, which measure the mean-square thermal displacements of atoms, are almost universally lower for main chain atoms than side chains. Cavity mutations generally support this conclusion[48, 94]. Recent NMR data[95] appears to indicate that the creation of a cavity within a protein (by mutations which remove part or all of one or more amino acid side chains) does increase the fluctuations of nearby side-chains but that these fluctuations are collective rather than independent. The cavity of L99A T4 lysozyme is known to be transiently accessible to solvent without large backbone fluctuations[35, 50]. Indeed the presence of large cavities in proteins argues that the main chain must be quite rigid. It is thus interesting to note that cavity-lining side-chain B-factors in L99A T4 Lysozyme (see below and Chapter 5) are smaller than for any other side chains in the molecule.

Pressure is the best thermodynamic tool to test mechanical properties of proteins. It is especially interesting to ask whether or not a cavity which is transiently accessible at ambient pressure remains accessible or even open at higher pressure.

1.5 Goals and Organization

We have begun with a well characterized protein, which contains a large cavity, empty at ambient conditions. There are three possibilities when we pressurize the molecule: there will be no response, the cavity and surrounding protein could deform, or solvent (water, in our case) could fill the cavity. The outcome of the experiment will depend sensitively on the interactions of atoms surrounding the cavity with each other and with water. We cannot, in one simple experiment, determine what exactly those interactions are, but we can determine the relative free energies of the states we observe. Through detailed modelling, we may further determine the relative importance of various interactions.

Therefore we lay out three goals for this thesis. First we seek to determine the outcome of a pressure experiment by crystallographic structure determination. Second, we will model the system and determine which of its features are most important in determining the outcome of the pressure experiment. Finally we will attempt to understand how pressure experiments fit in to the broader understanding of protein folding and function.

The remainder of this thesis is organized in essentially chronological order, moving from the early considerations of how to generate high pressures for a crystallographic experiment, to our final conclusions about protein structure. First I will discuss the high pressure techniques used in our work. I will not discuss in any great detail other methods, except to understand why we chose those used here. In Chapter 3 I will discuss the preparation of protein crystals for diffraction experiments in the cell described in Chapter 2, and consider data collection. Chapter 4 describes the computational tools used to determine protein structures from x-ray diffraction data, and carefully considers what one can learn from those structures. While it may seem esoteric, and the mathematical results will not be used in great detail, this chapter is particularly important to understand how much confidence we may place in our results, and where to put our emphasis.

Subsequent chapters will consider the slight structural changes observed (Chapter 5) and more importantly the observation of water in the high pressure structure of T4 lysozyme mutant L99A (Chapter 6). We will conclude with a discussion of the implications of these findings for protein structure in Chapter 7.

Chapter 2

High Pressure Equipment

In this chapter I consider the apparatus used in my high-pressure experiments. High-pressure protein crystallography demands much of our equipment. We must maintain sufficient pressure to observe some interesting phenomena, while permitting an unobstructed x-ray path over the large angular range needed to collect a complete diffraction data set.

Most of the high-pressure equipment used for this thesis was inherited from the earlier work of Paul Urayama and others in our group[9, 96, 97]. Paul's thesis[9] provides somewhat more detail, particularly in the choice of materials, than the reader will find below. Another good text is *High Pressure Technology*[98]. Rather than provide a lengthy description of choices that were not mine, I will review the equipment used and the basic principles of its operation.

I will first present basic methods of generating pressure and sealing high-pressure systems. Afterwords I will describe the crystallographic cell.

2.1 High-pressure basics

Before anything else, we must establish a reasonable pressure range for our experiments. Everything else–materials, seals, fittings, and even the method of pressurization–is ultimately designed around this choice.

2.1.1 Pressure range

In our experiment we have an obvious choice of the relevant pressure range, found by comparing the $p\Delta V$ energy of the L99A cavity to the stability of the molecule. Solving the equation $\Delta G = 0 = \Delta H + p\Delta V$ using published values of ΔH[48] and a guess of the unfolding volume ([27] and private communication with the authors) yields a pressure of roughly 1000 bar at pH 3 and about 3300 bar at pH 5.7, in both cases at room temperature. Our crystallographic experiments are carried out at higher pH[1], so the unfolding pressure is likely somewhat larger still.

(Solving this equation for the WT* molecule yields a pressure of more than 10,000 bar, highlighting an interesting problem. At such pressures water will freeze at room temperature, greatly changing the experiment. We are fortunately not concerned with this effect here.)

We would ideally carry out an experiment over the full pressure range of interest, which would extend up to the unfolding pressure where $\Delta G = 0$. 5000 bar (500 MPa) is thus a reasonable guess of our maximum pressure of interest. We were limited in our experiment to much lower pressures by the requirements of a suitable crystallographic cell.

2.1.2 Materials

Stainless steel has been the material of choice for almost all equipment used in the experiments described here. It combines good chemical resistance and more than adequate strength. It is not easily machineable, but high-pressure parts are com-

[1] The solid state physicist may find this statement odd; I remind the reader that protein crystals contain a large fraction of liquid water, so that hydrogen ions may diffuse in and out of the crystal proper, and pH in the crystal is then defined by equilibrium with a solution which surrounds the crystal.

mercially available (ours are purchased from High Pressure Equipment Company, Erie, PA). Flexible stainless steel syringe tubing is needed for connections to the crystallographic cell, but is also commercially available (Small Parts Incorporated, Miami Lakes, FL).

The pressurization medium should be as chemically inert and as incompressible as possible. Water is not a good choice as it becomes extremely corrosive at high pressure. We have chosen Fluorinert FC-77, a fully fluorinated hydrocarbon (3M Company, www.3m.com), originally designed for use in the electronics industry.

Most seals I used were *cone* seals, which require no further materials. We do not use traditional o-ring seals, but we do use a variant of a *Bridgman* seal (see below) which does use an o-ring. O-rings can be made of soft metal such as copper; we use Viton rubber.

2.1.3 High pressure seals

Cone seals

A cone seal is the most common seal in our apparatus. It consists of a female inner cone whose cone angle is slightly larger than the cone angle of the male outer cone piece (Figure 2.1). When the two pieces are compressed together a line seal forms. A collar on the male piece allows a threaded nut to compress it into the female cone. While cone seals are durable, they rely on the deformation of the steel cones in order to seal. Care must be taken not to deform the cones permanently by over-tightening the nut. Torque-limiting wrenches should be used. 1/4 inch nominal outer diameter seals should be tightened to 25 foot-pounds; 1/8 inch seals should be tightened to 75 inch-pounds. Periodic maintenance of the seals, including refacing the cones with a machine lathe, is necessary. Cone seals are useful to approximately 7 kbar.

Bridgman seals

The Bridgman seal is somewhat more clever, and very useful when it works. The seal is formed by creating a pressure differential across a plug which compresses a seal (the two can be one and the same.) A classic Bridgman seal is shown in Figure 2.2. Since the seal area supporting the plug is smaller than the plug area facing the pressurized medium, it follows that in equilibrium the pressure in the seal is higher than in the medium. Therefore the medium cannot push through the seal. It is not actually necessary to have the plug piece shown; it is only necessary for there to be a differential pressure across the seal. Because the seal is generated by the applied pressure, it is extremely robust and can withstand pressures up to 50 kbar[9].

Paul Urayama has made a variation on this seal which we use in our Beryllium crystallographic high pressure cell[9]. It will be described in Section 2.2.

2.1.4 Pressure generation and measurement

In our work, we have only used liquid pressurization methods. Gas pressurization can be used, but has several distinct disadvantages for what we do. Chief among these disadvantages are the greatly increased danger of working with pressurized gas, and decreased control over the pressure we achieve. Here we discuss the considerations and methods of pressurization.

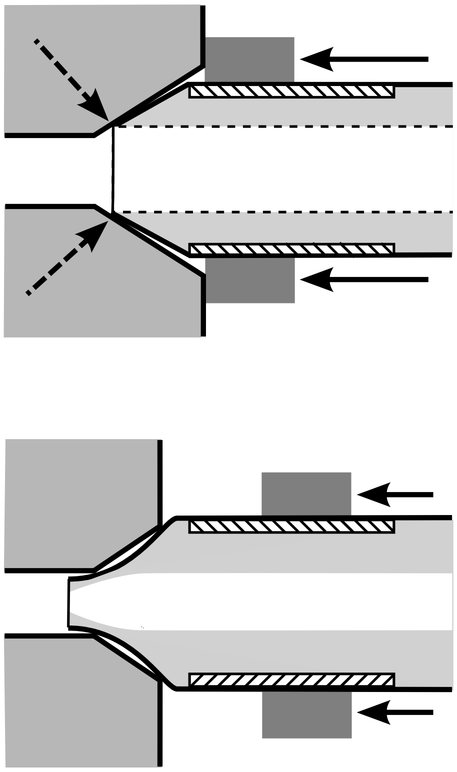

Figure 2.1: Cone seals. Solid arrows indicate applied force onto an adjustable collar. Dashed arrows indicate a proper line seal. (Top) A properly fitted cone seal. The adjustable collar (dark grey), threaded onto the end of the tubing, sits just against the face of the female cone piece, allowing the male cone to deform enough to seal, but preventing great deformation. (Bottom) If the collar is not in the correct position, the male cone will be damaged, or (not shown) the male cone will not contact the female cone and no seal is made. Adapted from [98].

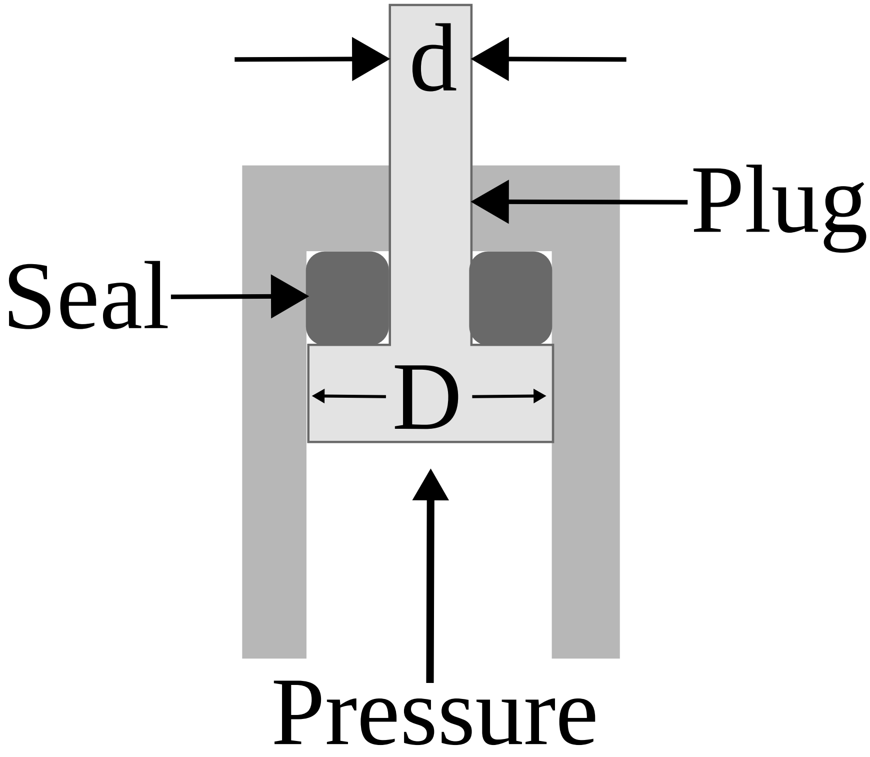

Figure 2.2: A schematic Bridgman seal. A plug compresses some sealing material. Since the area supporting the seal $\propto D^2 - d^2$ is less than the area facing the pressurized medium $\propto D^2$, it follows that the pressure in the seal is higher than that of the medium, and the medium is unable to force itself out. Adapted from [98].

Gas *versus* liquid pressurization

A gas high pressure pump is generally motor driven and built to achieve some ultimate pressure. Additional pressure regulators are necessary to control the pressure on a fine scale. More concerning is the energy stored in the gas while it is compressed. Remembering our familiar $p\Delta V$, we know that the work stored in compressing a material will be larger the more it compresses. While liquids are relatively incompressible, gases reduce their volumes by roughly p/p_{atm}, potentially several thousand times. Were the pressure containment system to fail, it would fail explosively. A beryllium high pressure cell pressurized with gas to 2 kbar must be considered to be a loaded gun.

On the other hand liquid pressurization stores very little energy in the material, and can be safely and easily controlled manually. Since most liquids are relatively incompressible, failure of the system would at worst result in a small drop of the fluid leaking out. Equipment for liquid pressurization is much more portable. Only a small hand crank press, a few fittings and some tubing are needed.

Particular to this experiment is the concern that we might force gas into the L99A cavity. It is experimentally known that nobel gases do bind to the cavity, so gas pressure was avoided because of the possible artifacts it may induce.

Pressure generation

The pressure generation system is shown in Figure 2.3, up to the crystallographic cell. A press is connected to two valves. One valve opens to a reservoir of pressurization medium, while the other opens to the crystallographic cell.

The press we use (Hi-P Model 50-6-15) is a metal cylinder with an approximately 1 cm bore down its axis, through which a piston is forced to compress the

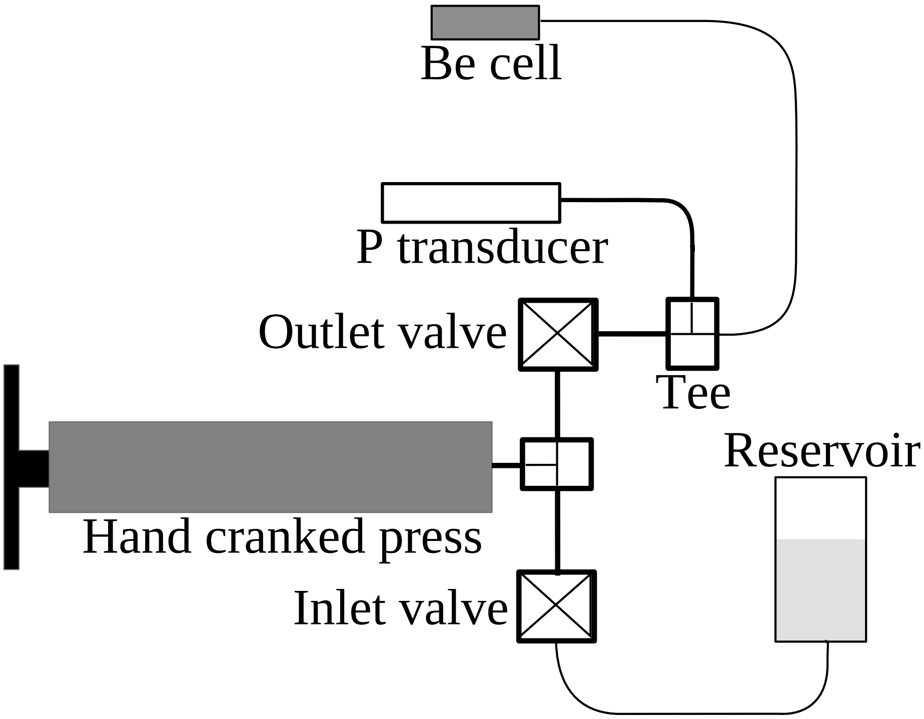

Figure 2.3: Schematic drawing of the pressure apparatus. A fluid reservoir is connected through an inlet valve to the press. The press is also connected to an outlet valve which is connected by stainless steel capillary tubing to the crystallographic cell.

pressurization medium. The piston is held in place by external threads, and hand cranked to compress the medium. The press itself is operable to 2.07 kbar. (This choice of equipment provides us with some measure of safety: the press will fail before the Be cell yields.) Periodic maintenance on the piston seal is necessary to prevent leaks from developing.

Valves rated to ∼ 4 kbar are used to permit easy filling and operation of the system. Tubing was generally 1/4 or 1/8 inch outer diameter. Small inner diameter stainless steel tubing is used for flexible connections. It is hard soldered into a commercial 1/8 inch tube so that it can be easily connected via cone seals to other components. This tubing can be quite small and can fail much more easily than other components. Great care should be used with any flexible tubing in high-pressure experiments. The valves and connections are cone-sealed to each other.

To fill the pump, one must first drive the piston fully inward. Then one opens the inlet valve (to the pressurization medium reservoir) and closes the outlet valve. Reversing the piston fills the piston chamber. It is often helpful at this point to close all valves, drive the piston in to achieve a small pressure (10 bar) and quickly open the inlet valve. This helps to purge gas from the system. To apply pressure to the sample, one closes the inlet valve, opens the outlet valve, and turns the piston inward.

Pressure measurement

A Sensotec (Columbus, OH) Model UHP transducer measured pressure using a strain-dependent resistor. It is specified to have 0.5% accuracy and to operate up to 100 kpsi, well beyond the range of our experiment. Transducers are read out with a Sensotec SC2000 signal conditioner. The transducer readout slowly relaxes[2] over 5-10 minutes even in a system with no leaks. The pressure readout stabilizes after this time; if it does not the system has a leak. In all of my experiments, the sample was left to equilibrate much longer than this relaxation time, and the pressure was recorded after the readout had stabilized.

[2] It is not entirely clear why this is, but it appears to be due to heating of the sensor element: more rapid pressurization leads to a greater "overshoot" in the initial pressure reading.

2.2 Crystallographic Cell

Our crystallographic cell is based upon the cell originally used by Kundrot and Richards[11] in their high pressure experiments on hen egg-white lysozyme. Several improvements were later made by Paul Urayama. The most important of these changes was a new sealing design which was suitable to at least 2 kbar. Kundrot and Richards' design leaked near 1 kbar[11].

The cell is machined from a beryllium-beryllium oxide alloy, Brush-Wellman grade I-250. Beryllium in general has relatively high strength, but the addition of some BeO provides additional strength. This alloy is 97.0% beryllium minimum, with the remainder BeO. The yield strength is 4480 bar. Due to its low atomic number (Z=4) it has low x-ray absorption. I-250 beryllium has an absorption path length (for which the transmission is 1/e) of roughly 1.5 cm for 1 Å x-rays. Paul Urayama's thesis[9] describes other grades of beryllium and describes the reasons for the particular choice used here. I-250 showed the best blend of absorption and strength.

Beryllium metal is relatively safe, but its compounds, particularly the oxide, can be hazardous. Contact between beryllium and water should thus be avoided to avoid oxide formation. Inhalation is the primary danger, and the oxides easily form powders. Inhaled dust or particulate beryllium oxides can, over time, lead to a condition called chronic berylliosis, an eventually fatal condition. Large doses at one time can lead to acute berylliosis, a form of anaphylaxis. Beryllium is also considered to be a carcinogen. To minimize these risks any beryllium should be stored dessicated in an air and water tight container, and should be cleaned and dried after use. Protective equipment such as gloves, goggles, and a particulate respirator should be worn.

Beryllium metal is brittle and difficult to machine. Along with the concerns of toxicity, this limits the number of facilities that are properly equipped to machine the metal. Our original cells were machined by Alfa Machine Company, of Monmouth Junction, NJ. More recently we have had cells machined by Brush-Wellman, who specialize in beryllium products for a wide range of applications from nuclear energy to aviation.

The cell and its accessories are shown in Figure 2.4 on the following page.

The cell itself is a simple cylinder, one inch (25.4 mm) long, one quarter inch (6.35 mm) in diameter. A 0.75 inch (19 mm) dead end bore will contain the crystal for diffraction measurements. 1/4-28 threads on the open end provide for connection to the high pressure system. Because the beryllium threads are brittle, we use an adapter to limit the frequency of threading the cell itself. The adapter has a receptacle for the cell, and its other end is identical to the open end of the cell itself. The adapter threads into a base-piece which is hard soldered to a flexible stainless steel tube. The base piece has a small post so that it is easily mounted to a standard crystallographic goniometer.

The seal between the adapter and cell is a modified Bridgman seal. The adapter has an internal cone, and the open end of the cell is chamfered. A Viton o-ring is placed in the adapter cone and the chamfer compresses the o-ring when the cell is threaded into the adapter. Because the high pressure side of the o-ring has greater surface area than the metal-o-ring contact, the pressure in the seal is higher than in the pressurization fluid. The seal is easily made with only finger-tightening of the cell.

For a cylinder having ratio of outer to inner radii ω, the maximum pressure

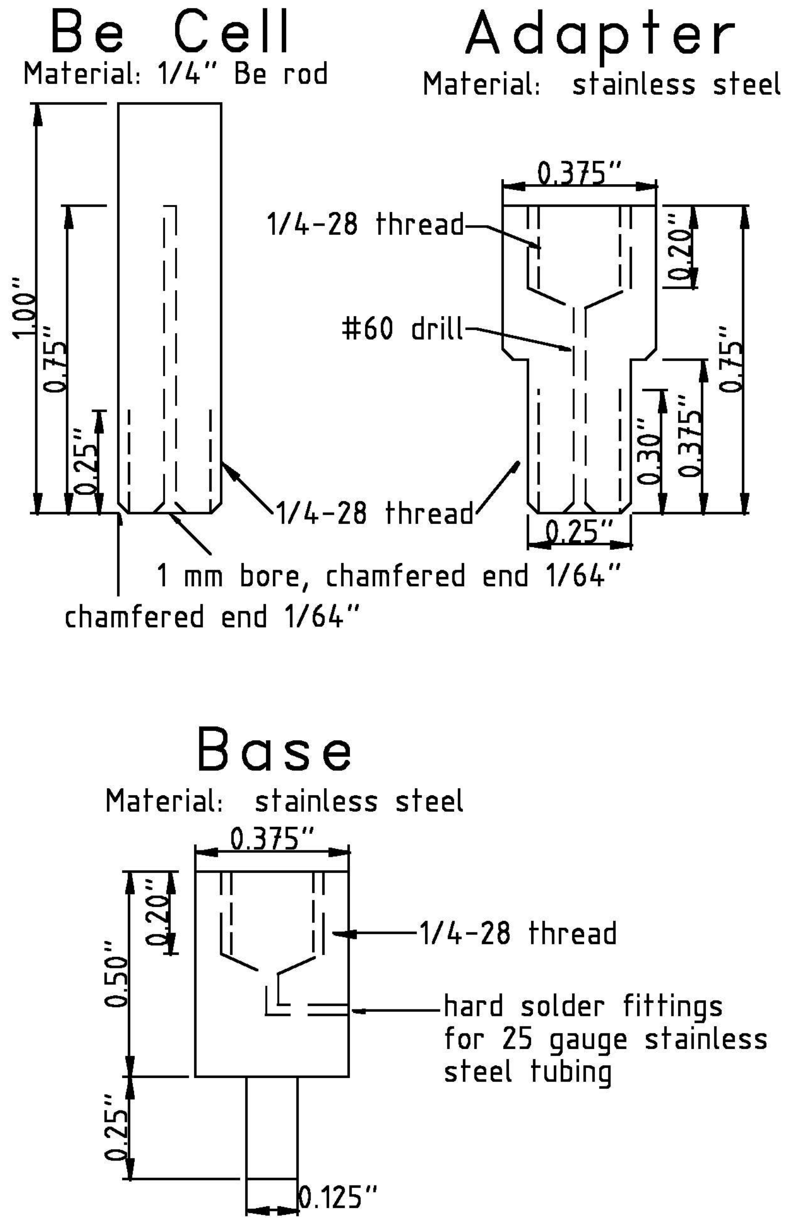

Figure 2.4: The crystallographic cell. A base piece with capillary stainless steel tubing connects to the optional adapter, and to the Beryllium cell itself. Adapted from [9].

which can be applied before deformation is[98]

$$P_{yield} = \frac{Y}{\sqrt{3}} \frac{\omega^2 - 1}{\omega^2} \tag{2.1}$$

where Y is the isotropic yield strength of the material. In our design, with $\omega = 6.35$, $P_{yield} = 0.56Y$, which for Beryllium I-250 is 2500 bar.

2.3 Other cells and methods

Other methods have been used to determine crystallographic structures under pressure. Paul Urayama[9, 14], along with Rafael Kapfer and Chae Un Kim[24], developed a method known as *high pressure freezing.* A group led by Roger Fourme at the European Synchrotron Radiation Facility (ESRF) has developed a diamond anvil cell for protein crystallography, and demonstrated its use on lysozyme[99, 100] and on cowpea mosaic virus[101].

Paul Urayama's high pressure freezing method involved pressurizing a protein crystal in liquid isopentane before freezing it by immersing the crystal's high pressure container in liquid nitrogen[102]. The crystal had to be carefully removed from the system and excess solid isopentane removed. Careful study[9] suggests that the pressure effects are frozen in as long as the crystal is never warmed above the water glass temperature. Thus all operations after freezing must be performed under liquid nitrogen or a cold stream.

In a more recent variation, the protein crystal is pressurized in helium gas before flash freezing in liquid nitrogen. One must first coat the crystal with an oil to prevent it from dehydrating under pressure; water easily evaporates into high pressure helium. The crystal is placed in a pressure vessel, supported on a crystallographic "loop" which itself rides in a brass and iron caddy. The crystal is held above the bottom of the vessel with a magnet external to the pressure tubing. Gas pressure is applied, and the sample allowed to equilibrate. It is then dropped, by removing the magnet, into the bottom of the pressure vessel which has been pre-cooled in liquid nitrogen. Here the crystal freezes. The crystal is recovered and stored in liquid nitrogen, and tranferred to a nitrogen cold stream for data collection.

In principle this technique could be extended to substantially higher pressures than the beryllium cell, which could be quite useful as we'll see in Chapter 6. I was concerned that helium gas would enter the cavity at relatively low pressures and prevent either cavity collapse or water filling. As of this writing, this possibility has not been tested. Because it was clear from the beginning that the beryllium cell would not have such problems, I chose that cell over high-pressure freezing for my work.

Diamond anvil cells (DACs) have long been used for high pressure studies on minerals (e.g. [103]) and liquids (e.g. [104]). They consist of two diamonds pressed into opposite sides of a metal gasket. The means of compressing the sample volume varies; the most common is to hold the two diamonds in facing plates and tighten the plates against each other with common machine screws. Various gasket metals are used, depending on the experiment. Water becomes extremely corrosive in the kilobar range, and so Rhenium gaskets or similarly resistant materials must be used. A DAC is able to achieve high pressures by using modest pressures on the large outer face of the diamond to push on a much smaller supporting area. Pressures in excess of 100 GPa have been achieved. Pressure-dependent ruby fluorescence is typically used to calibrate the internal pressure.

While the sample in a DAC is visible when pressurized, the useable diffraction cone is limited, necessitating higher x-ray energies to obtain high resolution data. In most cases multiple protein crystals are required to obtain a complete dataset. The thick diamonds required make an intense source mandatory. Until recently, DACs struggled to achieve fine (ca. 100 bar) pressure steps reproducibly. Mohammed Mezour and colleagues[100] have overcome this problem by using an inflatable metal membrane pressing against one face of the cell to finely control the pressure.

Despite its difficulties, a DAC is currently the only demonstrated method of achieving pressures above 2 kbar in a crystallographic cell. High-pressure freezing could be easily extended to 4 kbar. Above these pressures the equipment required becomes substantially more cumbersome, and DACs become the optimal choice.

Chapter 3

Experimental Methods: Crystallographic Data Collection

In the previous chapter, I outlined equipment for high-pressure protein crystallography. Here I will discuss the crystallographic preparations for high-pressure protein experiments. First crystals must be grown. They will have to be mounted specially for high-pressure studies. Finally data is collected on a synchrotron x-ray beamline. Refinement to an atomic protein structure will be discussed in the next chapter. At each step but crystal growth, high-pressure experiments require adaptations from routine protein crystallography. As much as possible, specific protocols will be discussed, to make this text as useful as possible to the reader.

3.1 Crystal growth

3.1.1 Basic techniques

For all experiments discussed in this work, I have grown crystals by the *hanging drop method*[105]. This method is straightforward and relatively easy to reproduce. The difficulty in protein crystal growth is to quickly achieve crystal nucleation and simultaneously grow crystals of good quality. The former requires strong association between protein monomers, the latter a weaker association. Thus the range of conditions under which suitable crystals will grow is generally narrow, and sometimes seems nonexistent.

The hanging drop method is sufficiently easy that many experiments can be set up quickly and a wide range of conditions can be tested. The method works by slowly evaporating a drop of protein solution so that the protein slowly approaches its nucleation point. The rate of evaporation is controlled by placing the drop in vapor contact with a second solution of slightly higher precipitant concentration (see below) and allowing the drop and second solution to reach equilibrium. The exact parameters will depend on the protein to be crystallized. Exact methods also vary according to the whims of the crystallographer. I describe my methods below.

I begin with a solution of precipitants, usually but not always salts of fairly high concentration, called a *mother liquor*. I next prepare the protein solution in a buffer of well known pH, and often with antioxidants, antibacterial chemicals, or other stabilizers. The mother liquor should be filtered to remove particulate matter. Generally the protein cannot be filtered as it will bind onto the filter material, resulting in almost total loss of the protein. Instead I centrifuge the protein solution for 10-15 minutes at 10,000-15,000 rpm in a microcentrifuge, sedimenting any particulate matter[1].

Once the solutions are ready, I prepare a commonly available *well plate* (*e. g.* from Hampton Research, Aliso Viejo, CA, or Nextal Biotechnologies, Montreal, QC, Canada). I fill each well with roughly 1 mL of mother liquor. I next pipette small drops of the protein and mother liquor solutions onto a thin glass cover slip and mix them by repeatedly pipetting the mixed drop up and down. Care should be taken to prevent contamination of the stock protein and mother liquor solutions.

[1] Here, as elsewhere in crystallography, it can be unclear how much of this is simple superstition and how much is actually necessary. The reader may assume that if I have included a step here, it is because it was important in achieving final results of good quality and reproducibility.

The final drop size is usually 2-20 μL. Silanized glass slides help to prevent protein crystals from sticking to the slide, facilitating later removal.

A bead of vacuum grease is placed around the rim of the well, so that it may be sealed. The slide is inverted and the drop suspended above the well solution. To avoid drying out the well or the drop, we must make sure that the grease seal is continuous and robust[2]. I like to run the back end of good tweezers around the edge of the glass slide to compress the grease seal.

Figure 3.1 shows a finished hanging drop crystallization tray.

A wide variety of parameters can be important to the success of the experiment. The concentration of solutions is of course critical, but sometimes the volume of the drop, the temperature of the plate, or vibration (either too much or too little) can be important. In our case, some tinkering with the original recipe was frequently necessary to produce good crystals. The exact procedures will be discussed next.

3.1.2 Phage T4 lysozyme

Our protein stock solution was provided by Professor Brian Matthews and Dr. Michael Quillin of the University of Oregon. Both the mutant L99A and psuedo wild-type WT* were provided at concentrations of roughly 15 mg/mL. WT* has no engineered cavities and has had all cysteines removed for stability (Cys54→Thr and Cys97→Ala; the mutant is often called TA* for this reason). L99A is a further mutant of WT* where Leucine 99 has been replaced by Alanine, producing the cavity described in Chapter 1. I used solutions as they were for the experiments descibed in this thesis.

[2] The Nextal plates use a simpler rubber gasket and threaded cover slide assembly, making this process considerably easier. However, the cost is proportionally larger.

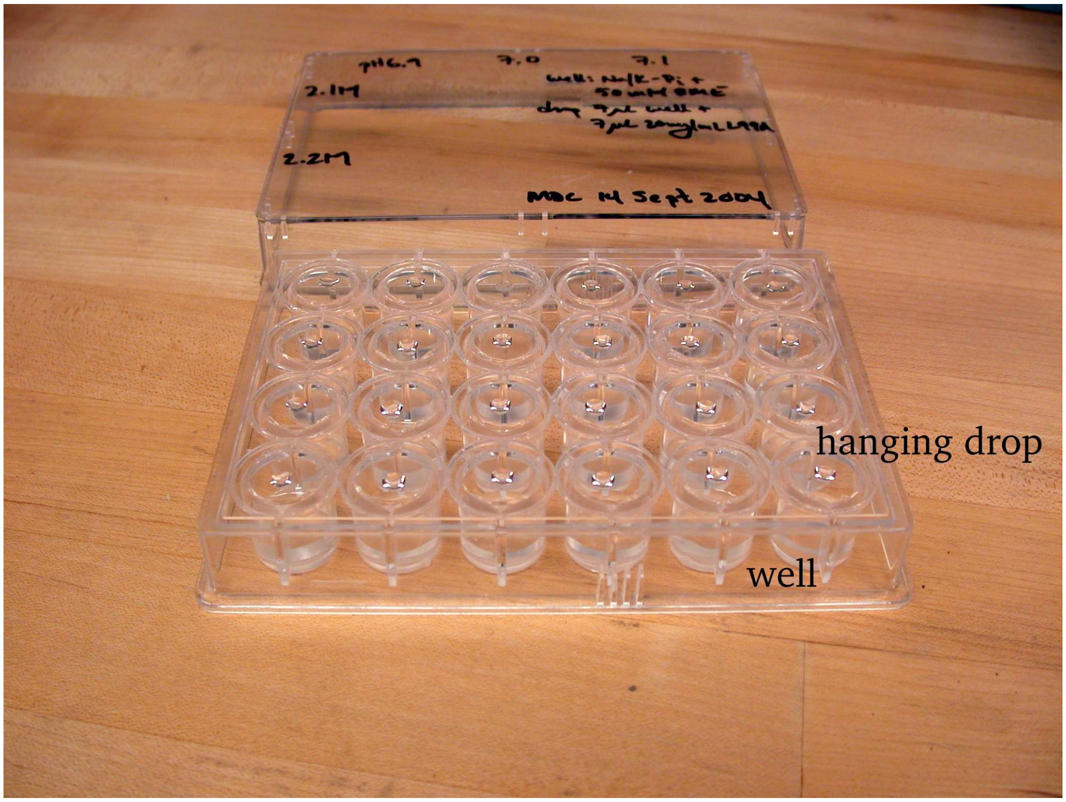

Figure 3.1: A finished hanging drop crystallization tray. The drop hangs on a glass slide above a well filled with a more concentrated precipitant solution. Each plate has 24 wells, and the plate has a cover to protect the slides. Each plate should be dated and carefully labelled with its contents.

Crystals of phage T4 lysozyme are usually grown in a 2.0 to 2.2 molar (M) phosphate buffer at pH values between 6.8 and 7.2. I prepared phosphate buffers at a given pH by mixing 4.0 M solutions of dibasic potassium phosphate and monobasic sodium phosphate, followed by dilution to the appropriate concentration. I measured the concentration of stock solutions using analytic acid titration with a calibrated pH probe. I also occasionally measured concentrations of the mother liquor itself. It is necessary to first dilute a small sample (usually 100 μL) of the stock solution as pH probes do not work accurately at very high ionic strengths.

I added β-mercaptoethanol (BME) to a final concentration of 50 mM. Before the WT* mutant was produced, this was used in Matthews' lab to prevent oxidation of thiols at residues 54 and 97 in the lysozyme[106]. In our case, these thiols have been removed to avoid oxidation. I added the BME as it aids particular intermolecular contacts necessary for crystallization.

Next I prepare the hanging-drop well plate. One milliliter of the mother liquor is added to each well of the plate. Usually each plate I prepared had wells with one of three different pH values and one of two buffer concentrations. Alternatively, I placed stock 4 M buffer/precipitant in the wells, and added water to dilute to the appropriate concentration and volume (1 mL), adding the BME afterwords. In this case, it is beneficial to mix the solutions well by nutating the plate for at least 30 minutes. If the rim of each well does not already have a grease bead, it should be added now.

A small (5-10 μL) drop of protein solution is placed on a clean, silanized glass slide. I add an equal volume drop of precipitant solution to the protein solution, and the drop is mixed by pipetting up and down three to five times. Finally I invert the slide and seal it above the matching well, either with grease or by threading the cap snugly.

The recipe I originally tried called for incubating the crystals at 4° C for 3 to 4 weeks. On occassion this worked well, but frequently I found it was necessary to leave the trays at room temperature for a few days to nucleate crystals. In some cases, the crystals were put back at 4 °C to grow, and in other cases left at room temperature. Alternatively, I added $\sim 100\,\mu$L of 4 M stock buffer to the wells to concentrate them and increase the nucleation rate. All three protocols produced good crystals of 0.5 to 1.0 mm size in about one month. It was never clear why it was necessary to nucleate crystals at room temperature or with higher well solution concentration, but I have no reason to suspect that the crystals are any different than those grown purely at cold temperatures.

3.2 Preparing crystals for diffraction experiments

After a crystal is grown, they must be prepared for x-ray diffraction measurements. A number of general points apply regardless of pressure. High-pressure experiments in the beryllium cell described in Chapter 2 require special loading of the crystals, described below.

3.2.1 General considerations

Crystals of protein are different from their inorganic or small molecule cousins in that they are held together by weak forces. Where smaller molecules and in particular inorganic crystals often have strong ionic or even covalent bonds, protein crystals are largely held together through dispersion forces, hydrogen bonds, and salt bridges. They are easily fractured by overly aggressive handling, rapid temperature changes, or rapid changes in pH or chemical concentrations. Protein crystals have an exceptionally high water content, anywhere from 30-80%[9]. As a result, dehydration can rapidly destroy the crystal. Whenever they might be exposed directly to air, protein crystals should be kept in a drop of their mother liquor, and a pipette and mother liquor should be on hand to hydrate them rapidly should the drop begin to dry out. Changes in pH or buffer concentration may be necessary; these should be performed by slowly adding the new buffer to the drop. Each type of crystal is different; some learning curve is unavoidable when attempting such changes.

Crystals of T4 lysozyme have the additional curious property that they are less dense than their mother liquor, so that they float. It is tempting to suggest that this is due to the (empty) cavity volume of the L99A mutant, but both the WT* and L99A variants float in their mother liquor. In fact the mother liquor is a more than two molar salt solution, making its density larger than that of the protein. This property makes the crystals somewhat tedious to handle in the loading process.

3.2.2 Loading the Beryllium pressure cell

The process of loading the Beryllium pressure cell has improved greatly since its initial use in our laboratory. I will now describe the modern procedure. Crystals will be loaded first into a glass x-ray capillary (Charles Supper Company, Natick, MA) which is then inserted into the Be cell. I first prepare a capillary of approximately the correct length. One must be sure that the capillary is not so long that moving a crystal inside will become difficult, and not so short that it will become unrecoverable in the cell. Also the capillary should be neither so wide that it will not fit, nor so narrow that the sample is loose in the cell or excess pressurization medium (usually of higher x-ray absorption) is present. For a cell inner diameter 1.0 mm, I have found that 0.9 mm nominal diameter capillaries are best.

In practice, the capillary (Figure 3.2) should be shortened so that it is easy to reach all interior points with crystal handling tools. A Delrin or similar plastic block with a through-hole of 1.0 mm diameter (the same as the inner dimension of the cell) is useful here. The block should be roughly 0.5 inches longer than the depth of the hole in the pressure cell (that is, 0.5 inch + 0.75 inch =1.25 inch). I pass the capillary through the hole until it is stopped by the capillary flare, and break it cleanly at the opposite end with good tweezers. I next flame seal the capillary by passing the bottom end through a butane lighter flame (see Figure 3.2) while rolling it between clean, bare fingers. After sealing, I verify that the capillary will still fit in the cell by placing it in the Delrin block. I verify that the working length of the capillary (the distance along which it is smaller in diameter than the inner dimension of the pressure cell) is approximately 0.25 inches longer than the inner depth of the cell. This will ensure that some part of the capillary will extend out of the beryllium cell when it is loaded, facilitating sample removal.

The crystal must sit in some immobilizing material but remain bathed in its mother liquor. Following Paul Urayama, I use Sephadex G-200, a large carbohydrate that will take up roughly twenty times its mass in water. This will serve as a soft, semi-rigid support for the crystal in the cell. Care should be taken to fully hydrate the Sephadex, as poor hydration will result in an osmotic shock to the crystal and deterioration on the scale of hours to a day. I use a positive displacement pipette (Drummond Scientific Company, Broomall, PA) to transfer the mother-liquor soaked Sephadex into the capillary and then centrifuge it to the

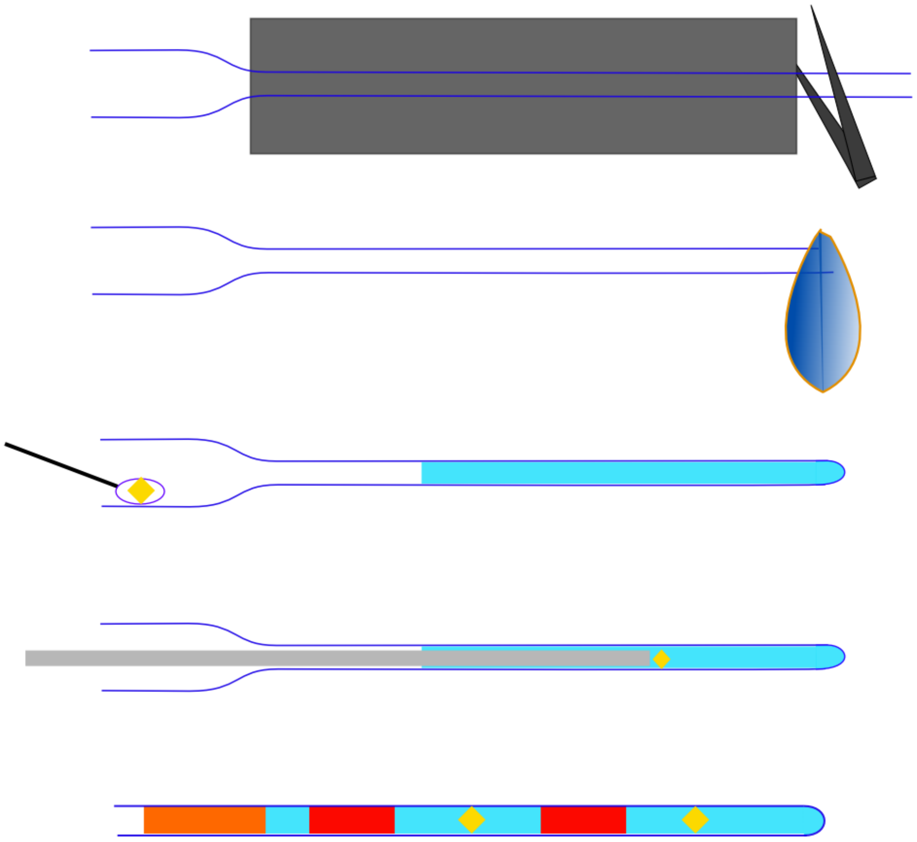

Figure 3.2: Steps in preparing a capillary for the high pressure cell. (Top) First the capillary is shortened to make it easier to reach the contents inside, and to ensure that it will fit in the cell. A jig described in the text is useful; tweezers are used to break the capillary where it emerges from the jig. (Top center) The capillary is flame sealed. Butane lighters work well for this purpose. (Center) After using a centrifuge to deposit the Sephadex-mother liquor mixture at the bottom of the tube, crystals are most easily transferred using standard crystallographic loops. (Bottom center) The crystal is gently pushed down the capillary using a smaller diameter glass capillary or a piece of copper wire. (Bottom) A finished capillary, showing Sephadex gel holding two crystals, separated by small pieces of thin copper wire, and sealed using grease.

bottom. If it does not centrifuge easily (in a benchtop centrifuge at only 1000 rpm for a few minutes at most) it is probably not sufficiently hydrated.

Harvesting the crystal and moving it into the Sephadex is the most sensitive step of the loading process. It is not complicated, but takes practice. A pipette method has been described[9], but I find it much easier[3] to use a standard crystallographic "loop" (e.g. those from Hampton Research). The crystal is harvested normally by scooping the crystal from its drop with the loop, and then it is touched to the inner wall of the capillary (Figure 3.2), preferably along with some mother liquor to keep it well hydrated. Once the crystal is inside the capillary, dehydration becomes less of a concern.

Various methods can be used to move the crystal into the Sephadex material. For robust crystals which do not float, the crystal can be centrifuged into the capillary. Phage T4 lysozyme floats in its mother liquor so something else will have to be done. I have found that the best method is to use a 0.3 mm glass x-ray capillary with a slightly larger glass bead at the end (either it came this way, or I produced the bead using the same method as to seal an open capillary). An extremely straight piece of small-diameter copper wire can be used, but in this case care is needed to avoid breaking the fragile glass capillary. Gentle force is used to gradually move the crystal down the tube. This takes some practice, but will produce good results.

To make later location of the crystal in the Beryllium cell more straightforward, small pieces of thin copper wire can be placed on either side of the crystal. I have found that it is quite easy to load 2-3 crystals in one capillary, making data collection at the x-ray beamline much more efficient.

[3] Kudos to Buz Bartow for this idea.

The capillary is then sealed using grease[4] pre-warmed to body temperature. The grease is injected into the capillary using a small hypodermic syringe. Care should be taken to avoid leaving excessive air bubbles in the capillary.

Finally, the capillary flare is broken off with tweezers, sizing the overall length to be about 0.25 inches longer than the internal length of the beryllium cell.

A loaded capillary is shown at the bottom of Figure 3.2. It is placed into a cell pre-filled with pressurization medium (see Chapter 2 for details). The cell is then attached to the high pressure press and pressurized.

3.3 Data collection

All X-ray data collected for use in this thesis were obtained at the F1 station of the Cornell High Energy Synchrotron Source (CHESS). Particulars of the data collection are noted below.

3.3.1 F1 station equipment

CHESS F1 station uses radiation from positrons in the Cornell Electron Storage Ring (CESR) passing through a 24 pole wiggler magnet. The resulting X-rays are first passed through a vertically focussing high-heat load white-beam mirror, and then into a horizontally focussing silicon crystal monochromator 20 meters from the source. The beam next passes through a vertically focussing mirror. The fully focussed beam has a 1 mrad horizontal and 0.4 mrad vertical divergence, and is 0.2 mm vertical by 2 mm horizontal at the focus. A 100 μm collimator just before the sample sets the beam size and divergence. Current operating parameters

[4] I use Apezion-N vacuum grease, but it is expensive.

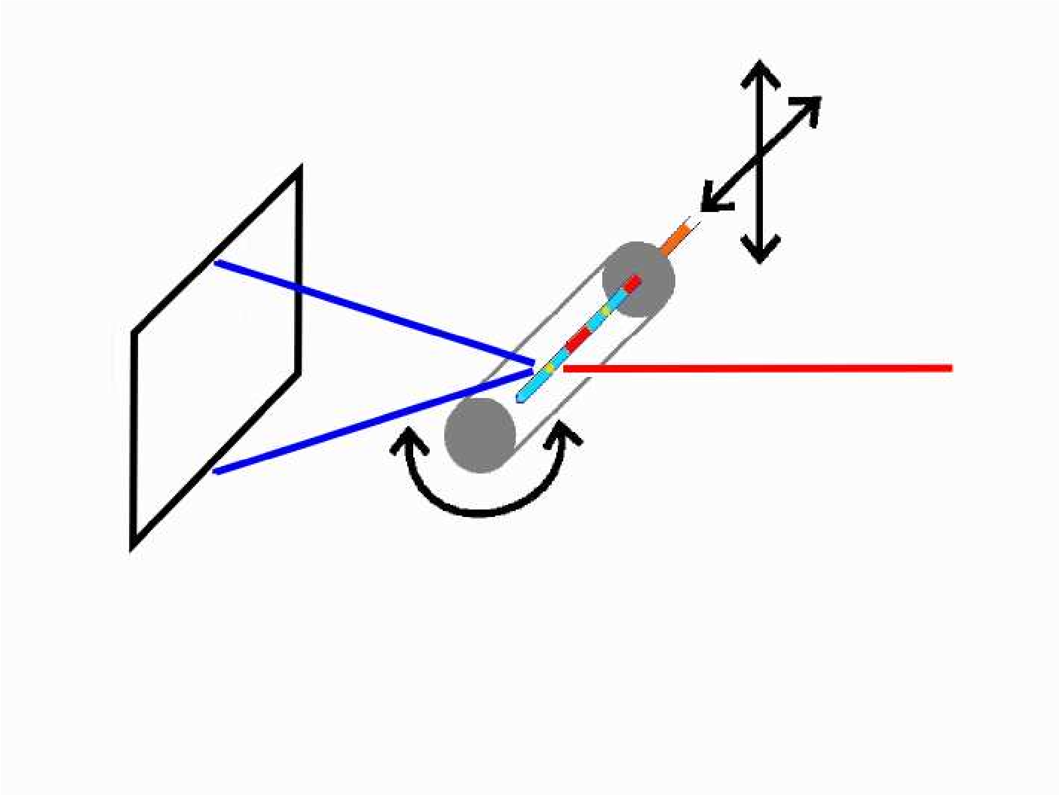

Figure 3.3: Experimental geometry. Incident X-rays (red) impinge upon the sample inside the beryllium high pressure cell. X-rays scatter (blue) at various angles and are detected on a CCD camera (outlined in black at left) to produce an image. The sample can be moved in the plane perpendicular to the beam to best align the crystal, and is rotated about the axis of the Be cell during data collection.

can be found at http://www.chess.cornell.edu/aboutus/east/f1.htm on the World Wide Web.

The sample cell is mounted on an air-bearing rotation stage, fitted with a standard crystallographic goniometer head, whose axis is perpendicular to the beam and in the horizontal plane. The rotation stage can be translated in three dimensions for crystal alignment. All of this equipment sits on a motorized optical bench. Beam alignment is performed by translating and rotating the bench relative to the fixed x-ray beam. Figure 3.3 shows the sample geometry on the beamline.

CHESS staff tune the X-ray energy to $40\,eV$ above the Bromine absorption edge, 13.514 keV, with a typical flux of 3×10^{11} photons per second through a $300\,\mu$m diameter collimator.

The F1 station uses a pair of ADSC Quantum 4 CCD-based detectors. I used only one of these for my experiments, as neither the crystal-limited diffraction resolution or unit cell size warranted use of both. The detector is set roughly 20 cm from the sample, so that diffraction to ~ 1.9 Å can be collected.

The CCD was controlled using ADSC's custom software. The station equipment and point detectors are controlled using a combination of SPEC[107] and in-house software written by CHESS staff scientists.

3.3.2 Choosing collection parameters

A number of parameters must be set before data collection begins. When the first test images from a particular crystal are taken, we note the quality of the diffraction spots. Ideal diffraction spots are small in size, circular in shape, and intense but do not overload the detector. Low resolution spots are important in refinement, so these should be given as much care as high resolution spots. Features of reciprocal space called lunes can be seen if the image is of good quality. The resolution, limited by thermal motion of atoms as well as the intrinsic mosaicity and disorder of the crystal, should also be noted. Crystals yielding resolution less than 2.5 Å were discarded, since I knew in advance that crystals of better quality should be available.

A crystal of good quality and well centered in an x-ray beam will produce an image like that in Figure 3.4. We will need to collect a minimum amount of data to adequately sample reciprocal space for future refinement of the structure. To do

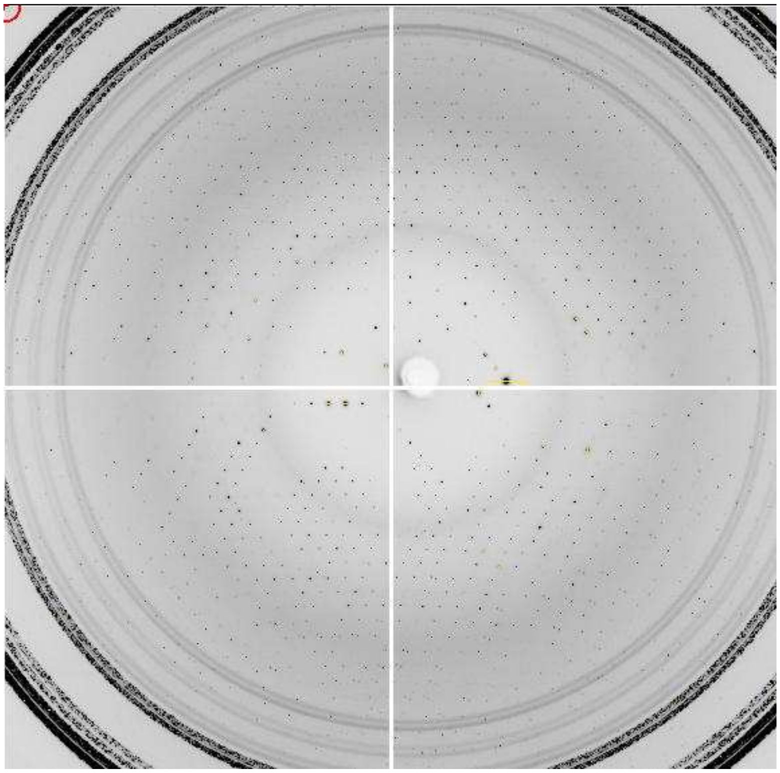

Figure 3.4: A typical X-ray diffraction image. This is the first in the mt2k8 series (see text). Be powder diffraction rings (Section 3.3.4) are visible, becoming strong near 2.0 Å.

this we will collect images over an angle of rotation perpendicular to the the beam of $360/n$ degrees, where n is the order of the highest symmetry axis[5]. Modern software is capable of determining the orientation matrix of the crystal from a few images or even just one, especially if the space group is already well known; thus we need not set the crystal to a known orientation before collecting data.

The exposure time is set to achieve maximum resolution possible for the crystal without overexposing the stronger low resolution spots or subjecting the crystal to excessive radiation damage. Radiation damage becomes apparent as the reso-

[5] In this case the crystals are in space group $P3_22_1$, so n is 6.

lution in each image decreases, the individual spots become larger and less clear, and mosaicity increases. We may take still or *oscillation* images[105]. Oscillation images are taken while the crystal rotates smoothly from one angle to another, usually separated by one to two degrees. The advantage of larger oscillation angles is that we sample a wider wedge of reciprocal space (Chapter 4). One must be careful not to sample so wide a wedge that reflections overlap on the detector. The disadvantage is that each reflection spends less time in a diffraction condition, for a given length of exposure. In practice, this loss is moot: having fewer images simplifies later merging of the data, and exposure time can be increased if necessary. It is only a concern for weak reflections, where decreasing the background noise by taking shorter exposures may help the signal to noise ratio.

3.3.3 Alignment of the crystal in the pressure cell

The Beryllium pressure cell is optically opaque, so we cannot align the crystal in the x-ray beam using the optical techniques common in protein crystallography. In past experiments using the Beryllium cell, careful measurement of the crystal location in the capillary, and the capillary in the cell, was used to find the crystal once on the crystallographic x-ray beamline. The cell was moved a calculated distance from its end, and the crystal should be in the beam. When working with large crystals, this method works reasonably well. With crystals of plate or needle like habits, or crystals that are smaller than the inner diameter of the capillary, this method of location is extremely tedious.

Instead I have introduced bits of wire into the cell which absorb X-rays more strongly than anything else in the cell. The pressure cell is mounted on the beamline, and a point detector (usually an ion-chamber) is placed between the crystallographic x-ray detector and the sample. The Be cell is centered vertically on the beam, using the point detector and the absorption of the cell itself. The cell is scanned along its axis to locate the bits of wire (Figure 3.5). Combined with careful measurements as in the past, this is an accurate and quick means of locating crystals. Some T4 lysozyme crystals are also visible as local absorption *minima*, an observation we attribute to the strongly electron dense mother liquor.

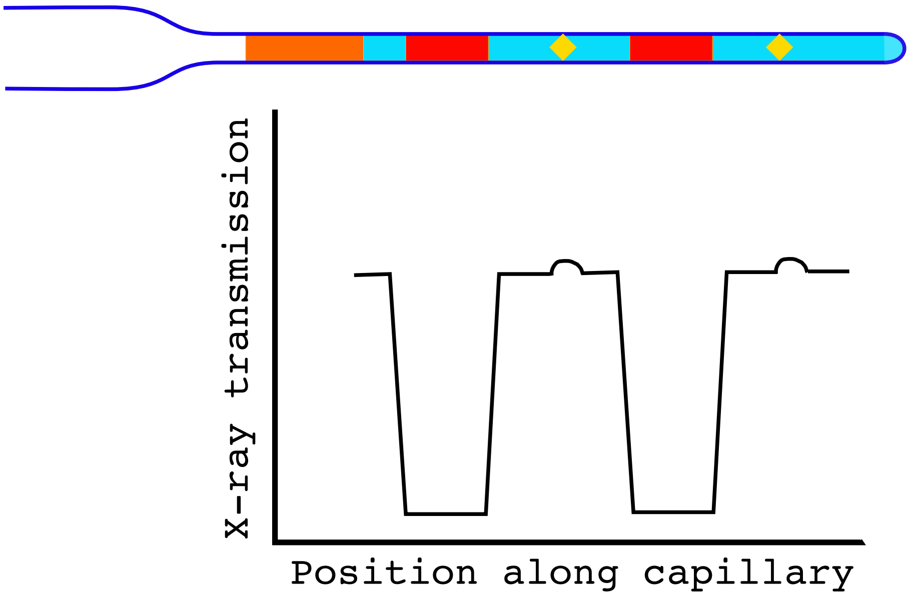

Figure 3.5: A capillary loaded with two protein crystals and showing the X-ray transmission corresponding to each part of the capillary. In our case, where the mother liquor is unusually electon-dense, the crystals are visible as small peaks in transmission.

To ensure that the crystal is centered on the crystallographic oscillation axis (see below), images are collected at 0, 90 and 180 degrees rotation. Course alignment is sufficient, and this procedure will also allow quick screening of crystal quality. Once the crystal is in place, collection of X-ray diffraction data begins.

3.3.4 Other beryllium cell issues

Despite its low atomic number, some absorption and scattering from the Be cell is inevitable (Figure 3.4). Diffraction rings from beryllium are visible at roughly 2.4 Å, and are become strong at 2.0 Å. This limits the useful resolution of our experiments to 2.0 Å.

The Be cell is cylindrical, not spherical, and therefore there is a scattering-angle dependent absorption correction to be applied to the data (Figure 3.6). We minimize this correction by rotating the sample about the cell axis and keeping the cell axis perpendicular to the incident beam. In this geometry the absorption correction depends only on the angle ζ between the axis of the cell and the scattered x-ray. At 2.0 Å resolution and beam energy $\sim$ 13.5 keV, there is at most a 0.32 mm or 12% spread in path length through the Beryllium. At this energy the attenuation length through pure Beryllium is $\sim$18 mm, resulting in a less than 2% spread in absorption corrections from the unscattered beam to the highest resolution diffraction spots. The ratio I/σ_I, diffracted intensity to uncertainty, is $\sim$10 at the highest resolution in each dataset, so that the uncertainty is much larger than this absorption correction.

If the crystal is off center there is also a correction due to asymmetry of the crystal position in the cell. We were able to avoid this situation in all cases.

3.4 Data collected

The data used for this thesis were collected in July and November of 2004. Another synchrotron experiment was performed prior to the final data collection, during which I worked out many of the details that would make later experiments suc-

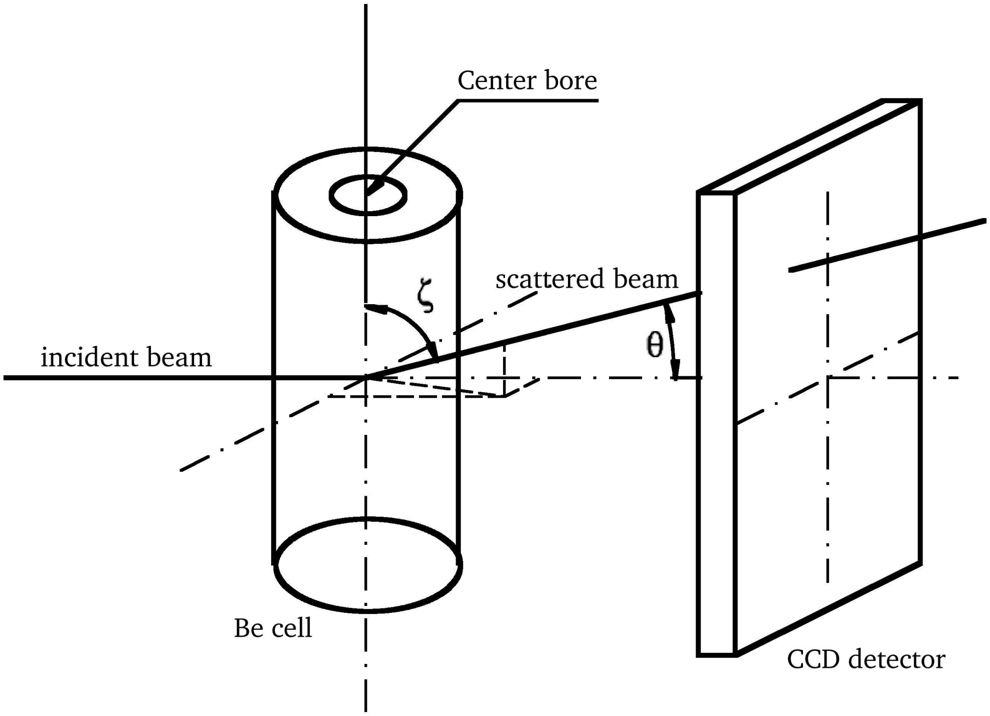

Figure 3.6: Beryllium cell scattering geometry. Adapted from [9]

cessful. Much of the procedure described above was developed between that first experiment and July of 2004.

A total of 10 data sets were refined for the L99A mutant at four different nominal pressures (0, 1, 1.5 and 2 kbar), and six for the psuedo wild type WT* molecule (at 0, 1 and 2 kbar). The important data collection and reduction parameters specific to each dataset are listed in Table 3.1 on the next page. Refinement parameters will be discussed in the following chapter.

In general, data were collected over as large a range as the alignment and quality of the crystal would permit. The crystal was rotated through 2 degrees during the collection of each image, and through between 60 and 120 degrees during the collection of a complete dataset. Exposures were between 15 and 20 seconds; the same exposure time is used for all images in one dataset. Collection was terminated when radiation damage became apparent through decreased resolution in successive images. The crystals tend to have a right triangular cylindrical shape, so that three faces are rectangular and the other two faces are triangular. Crystals used were all approximately 0.8 mm in their largest dimension (the base of the triangle) and 0.5 mm perpendicular to the longest dimension.

Table 3.1: Crystallographic data collected on phage T4 lysozyme. P_{nom}, nominal pressure of each dataset; P_{act}, measured pressure (uncertainty 0.05 kbar, see Ch. 2). r, resolution in Å. C, percent completeness. m, mosaicity in degrees. R_{merge} (in percent) is a measure of the quality of the dataset (see Chapter 4 for definition.) *wt* represents data collected on the WT/TA* psuedo wild-type lysozyme, *mt* the L99A mutant. July 2004 data are suffixed with *a* or *b*. November 2004 data are suffixed with numerals which are not necessarily sequential.

Name	P_{nom} (kbar)	P_{act} (kbar)	r	C	m	R_{merge}
mt0k1	0	0.10	2.4	97.3	0.19	6.6
mt0ka	0	0.00	2.2	90.7	0.10	5.8
mt0kb	0	0.10	2.3	94.8	0.14	7.3
mt1k6	1	1.07	2.1	94.8	0.16	11.0
mt1k7	1	1.03	2.2	94.3	0.18	5.5
mt1k9	1	1.07	2.1	94.6	0.15	5.2
mt1.5k1	1.5	1.46	2.1	96.1	0.25	5.5
mt2k1	2	1.90	2.1	92.9	0.13	3.4
mt2k3	2	1.90	2.2	91.3	0.17	3.9
mt2k8	2	1.95	2.1	94.8	0.16	3.6
wt0ka	0	0.31	2.2	95.3	0.11	5.2
wt1k6	1	1.04	2.1	95.3	0.17	7.0
wt1k7	1	1.04	2.1	96.8	0.21	4.1
wt1ka	1	1.01	2.2	94.6	0.11	5.2
wt2k1	2	1.93	2.0	90.5	0.13	5.5
wt2k2	2	1.93	2.0	90.5	0.15	5.7

Chapter 4

Refinement of Crystallographic X-ray data

This chapter describes the process of constructing an atomic model of a protein from raw diffraction images. First we must consider how x-rays are diffracted from crystalline matter. The data collected must then be reduced to a workable set. Finally the data will be modelled using standard statistical methods. I will describe these methods in some detail, as they are fundamental to understanding what this "refinement" produces. The end of the chapter describes practical aspects of refinement, such as indicators of progress and accuracy, and specific protocols used for this work.

4.1 Basic diffraction theory

Before I discuss structural refinement from x-ray diffraction data, I will review the basic principles of diffraction. Nielsen and Morrow have written a good text for further study[108].

4.1.1 Quantum mechanical basis

All scattering in principle derives from transitions between quantum mechanical states. In this section I will derive the the "cross-section" (to be defined later) for x-rays scattering from matter. I will assume that the reader is familiar with Dirac notation and second quantization. Nielsen and Morrow[108] cover the topic reasonably well in an appendix, and any text on advanced Quantum Mechanics should also cover the subject (Sakurai's *Advanced Quantumm Mechanics*[109] is my favorite). Sturm gave a concise and readable review of various scattering processes for x-rays and other probes[110].

We begin with the eigenstates $|n\rangle$ of a Hamiltonian $\mathcal{H}_0$ each having energy E_n so that $\mathcal{H}_0 |n\rangle = E_n |n\rangle$. Next imagine that there is a perturbing (or interaction) Hamiltonian $\mathcal{H}_I$ of as yet unspecified form; we need only assume for what follows that to first order $\mathcal{H}_I$ perturbs only the energies, and not the eigenstates, of the system. Using a first order approximation in time dependent perturbation theory, we can derive *Fermi's Golden Rule*:

$$w_{i\to f} = \frac{2\pi}{\hbar}|\mathcal{H}_{if}|^2 \delta(E_f - E_i) \tag{4.1}$$

which describes the transition rate between eigenstates $|i\rangle$ and $|f\rangle$ of $\mathcal{H}_0$ in the presence of the perturbing Hamiltonian $\mathcal{H}_I$. $\mathcal{H}_{if}$ are the *matrix elements* $\langle f| \mathcal{H}_I |i\rangle$. The Dirac δ-function ensures energy conservation. It should be taken as implied that to determine the full scattering rate (or the scattering cross section, which will be defined properly below), Equation 4.1 is to be integrated over all possible initial and final states consistent with the experiment of interest. It is not, incidentally, necessary to restrict ourselves to the first order approximation; for instance, the second order terms yield the transition rates for so-called resonant diffraction.

Our next step is to describe and evaluate the matrix elements $\mathcal{H}_{if}$. We will choose a gauge for the (real) vector potential $\boldsymbol{A}$ such that $\nabla \cdot \boldsymbol{A} = 0$ and so that the scalarset potential is zero. (In this case, the operator forms of $\boldsymbol{A}$ and momentum commute, and the fields are transverse). From correspondence to classical mechanics it can be shown that the appropriate Hamiltonian for a singly charged particle interacting with light is

$$\mathcal{H} = \frac{(\boldsymbol{p} - e\boldsymbol{A})^2}{2m} + \mathcal{V} + \mathcal{H}_{rad}, \tag{4.2}$$

where $\boldsymbol{p}$ is the momentum of the particle having charge e and mass m. $\mathcal{V}$ is the potential energy of the system in the absence of the externally applied electromagnetic field. $\mathcal{H}_{rad}$ is the energy in the radiation field itself.

Dropping the radiation field self-energy, Equation 4.2 can be separated into a part which describes the system in the absence of the applied electromagnetic field, and an interaction part $\mathcal{H}_I$, given by

$$\mathcal{H}_I = -\frac{e}{m}\mathbf{p} \cdot \mathbf{A} + \frac{e^2}{2m}\mathbf{A}^2. \tag{4.3}$$

Note that I have now written the momentum and vector potential in a new typeface as $\mathbf{p}, \mathbf{A}$ to denote that they are properly thought of as quantum mechanical operators.

Upon quantizing the electromagnetic radiation field (see Sakurai[109]), the vector potential is written

$$\mathbf{A} = \sum_p \sum_{\boldsymbol{k}} \sqrt{\frac{\hbar}{2\epsilon_0 V \omega_{\boldsymbol{k}}}} \left[\boldsymbol{\epsilon}_{p,\boldsymbol{k}} \mathfrak{a}_{p,\boldsymbol{k}} \mathrm{e}^{i\boldsymbol{k}\cdot\boldsymbol{x}} + \boldsymbol{\epsilon}_{p,\boldsymbol{k}} \mathfrak{a}^\dagger_{p,\boldsymbol{k}} \mathrm{e}^{-i\boldsymbol{k}\cdot\boldsymbol{x}} \right] \tag{4.4}$$

where $\boldsymbol{\epsilon}_{p,\boldsymbol{k}}$ is the polarization vector for polarization state p and wavevector $\boldsymbol{k}$, V is the volume of some confining box containing the sample, $\boldsymbol{x}$ is a position in space, $\omega_{\boldsymbol{k}}$ the frequency of radiation ($\omega_{\boldsymbol{k}} = c|\boldsymbol{k}|$ where c is the speed of light), and ϵ_0 is the vacuum dielectric constant. $\mathfrak{a}$ and its conjugate $\mathfrak{a}^\dagger$ are the annihilation and creation operators for photons. The normalization is chosen so that the energy in the radiation field, written in harmonic oscillator form $E = \hbar\omega\mathfrak{a}^\dagger\mathfrak{a} + 1/2$ for discrete photons, is consistent with the classical energy of the field. $\mathfrak{a}^\dagger\mathfrak{a}$ is a *number operator*. By this I mean that the electromagnetic field is an eigenfunction of the operator $\mathbf{a}^\dagger\mathbf{a}$ with eigenvalues equal to the number of quanta in a given mode of the field (that is, the number of photons having a particular wavevector and polarization).

Thus the function of the vector potential operator $\boldsymbol{A}$ is to create and destroy quanta of the electromagnetic field, which we call photons. Returning to Equation 4.3, we can immediately see that scattering requires both an incident photon to be destroyed and a scattered photon to be created. The first term, corresponding to emission and absorption processes, is not of further interest here, so long as our x-ray beam energy is sufficiently different in energy from any transitions between electronic states of $\mathcal{H}_0$. (This is not always true in protein crystallography, where for instance the absorption properties of particular atoms can be used to extract phase information as will be discussed briefly in later sections.) For first order scattering processes with no atomic absorption or emission (and hence no $\mathbf{p} \cdot \boldsymbol{A}$ term), we may write Equation 4.1 as

$$w_{i\to f} = \frac{2\pi}{\hbar} |\, \langle f; \boldsymbol{k}_f, \boldsymbol{\epsilon}_f | \, \frac{e^2}{2m} \mathbf{A} \cdot \mathbf{A} \, | i; \boldsymbol{k}_i, \boldsymbol{\epsilon}_i \rangle \, |^2 \delta(E_f - E_i + \hbar\omega_f - \hbar\omega_i) \tag{4.5}$$

Note that generally the states $|n; \boldsymbol{k}, \boldsymbol{\epsilon}_p\rangle$ denote the n^{th} electronic state combined with a photon state. Here, $|i\rangle$ and $|f\rangle$ denote the initial and final electronic states of the system, subscript i and f indicate the initial and final wavevectors, frequencies and polarizations of the electromagnetic field, and E_i, E_f are the initial and final energies of the electronic states. The Dirac δ-function enforces total energy conservation. Again, to determine the experimentally observed cross section, we must integrate over all initial and final states consistent with the experiment.

I will next make use of the fact that the photon states $|\boldsymbol{k}, \boldsymbol{\epsilon}_p\rangle$ can be written as a creation operator acting on the zero-photon state $|0\rangle$: $|\boldsymbol{k}, \boldsymbol{\epsilon}_p\rangle = \mathbf{a}^\dagger_{p,\boldsymbol{k}} |0\rangle$. Two of the terms resulting from the integral $\langle \boldsymbol{k}_f, \epsilon_f | \mathbf{A} \cdot \mathbf{A} | \boldsymbol{k}_i, \epsilon_i \rangle$ will thus be of the form $\langle 0 | \mathbf{a}\mathbf{a}^\dagger \mathbf{a}^\dagger \mathbf{a}^\dagger | 0 \rangle$. Letting all of the operators act to the right, this is the product of a zero photon state with a two photon state, which is identically zero. Making use of the fact that $\mathbf{a}_{\boldsymbol{k}}$ and $\mathbf{a}^\dagger_{\boldsymbol{k}'}$ commute if $\boldsymbol{k} \neq \boldsymbol{k}'$, the nonzero terms have the matrix elements

$$\langle f, 0 | \, \mathbf{a}_{p_f, \boldsymbol{k}_f} (\mathbf{a}^\dagger_{p', \boldsymbol{k}'} \mathbf{a}_{p, \boldsymbol{k}} \mathrm{e}^{i\boldsymbol{q}\cdot\boldsymbol{x}} + \mathbf{a}^\dagger_{p, \boldsymbol{k}} \mathbf{a}_{p', \boldsymbol{k}'} \mathrm{e}^{-i\boldsymbol{q}\cdot\boldsymbol{x}}) \mathbf{a}^\dagger_{p_i, \boldsymbol{k}_i} \, | i, 0 \rangle \tag{4.6}$$

where I have introduced $\boldsymbol{q} = \boldsymbol{k}' - \boldsymbol{k}$. Only in the case that $\boldsymbol{k}' = \boldsymbol{k}_f$ and $\boldsymbol{k} = \boldsymbol{k}_i$ (or vice versa) is the matrix element non-zero. The simple way to see this is to observe that you can't destroy a photon that isn't already there. The net result is that for all first-order scattering, the matrix elements squared take on the form $| \langle f | \, \mathrm{e}^{-i\boldsymbol{q}_{fi}\cdot\boldsymbol{x}} \, | i \rangle |^2$ where $\boldsymbol{q}_{fi} = \boldsymbol{k}_f - \boldsymbol{k}_i$; since the electronic states' amplitudes squared are just the charge densities, the relevant scattering matrix elements give Fourier transforms of some charge density, though what charge density we have not yet specified.

We now gather all of the important constants out front. The net result is

$$w_{i\to f} = \frac{2\pi}{\hbar} (\boldsymbol{\epsilon}_f \cdot \boldsymbol{\epsilon}_i)^2 \left[\frac{e^2 \hbar}{2m\epsilon_0 V} \sqrt{\frac{1}{\omega_{\boldsymbol{k}_f} \omega_{\boldsymbol{k}_i}}} \right]^2 | \langle f | \, \mathrm{e}^{-i\boldsymbol{q}_{fi}\cdot\boldsymbol{x}} \, | i \rangle |^2 \delta(E_f - E_i + \hbar\omega), \tag{4.7}$$

where $\omega = \omega_f - \omega_i$ We collect a factor of $r_o = e^2/4\pi\epsilon_0 mc^2$, the classical radius of the electron, to arrive at

$$w_{i\to f} = \frac{2\pi}{\hbar} (\boldsymbol{\epsilon}_f \cdot \boldsymbol{\epsilon}_i)^2 r_0^2 \left[\frac{2\pi\hbar c^2}{V(\omega_{\boldsymbol{k}_f} \omega_{\boldsymbol{k}_i})^{1/2}} \right]^2 | \langle f | \, \mathrm{e}^{-i\boldsymbol{q}_{fi}\cdot\boldsymbol{x}} \, | i \rangle |^2 \delta(E_f - E_i + \hbar\omega). \tag{4.8}$$

We recognize $(\boldsymbol{\epsilon}_f \cdot \boldsymbol{\epsilon}_i)^2 r_0^2$ as the Thomson scattering cross section for a single free electron, exactly as calculated from classical electrodynamics.

We are interested in the scattering cross section, the number of photons scattered from a single particle into a small range in $\boldsymbol{k}$-space about some final wavevector $\boldsymbol{k}_f$, divided by the incident flux and the volume in phase space into which we scatter. Equation 4.8 describes the scattering *rate* from a well defined initial state into a particular final state. We will assume the incident photons all have a precisely defined wavevector and energy (and for synchrotron radiation, polarization), but clearly this is not the case for the scattered photons.

We need to consider the volume of phase space into which the photons can scatter, and the density of states in that volume. The allowed values of $\boldsymbol{k}$, in a box of side L, are obtained by enforcing periodic boundary conditions at the sides of the box. Thus $\boldsymbol{k} = (2\pi/L)(n_x\hat{x} + n_y\hat{y} + n_z\hat{z})$, where the n are integers and $\hat{x}, \hat{y}, \hat{z}$ are orthogonal real space unit vectors. The number of photons scattered from an initial state $|\boldsymbol{k}_i\rangle$ into a small volume $k_f^2 \mathrm{d}k_f \mathrm{d}\Omega$ of $\boldsymbol{k}$-space is then

$$I_{sc} = N_0 \sum_f w_{i\to f} \frac{V}{(2\pi)^3} k_f^2 \mathrm{d}k_f \mathrm{d}\Omega \tag{4.9}$$

where N_0 is the number of incident photons, $V = L^3$ and k_f is the magnitude of $\boldsymbol{k}_f$. We are interested in the quantity $\mathrm{d}^2\sigma/\mathrm{d}\Omega\mathrm{d}E_f$, the fraction of photons scattered into a certain angular region $\mathrm{d}\Omega$ and energy region $\mathrm{d}E_f$ around a particular $\boldsymbol{k}_f$ (and its associated E_f). The incident photon flux is the number of incident photons N_0, divided by the volume V in which they reside times their velocity c, and the differential in energy is just $\mathrm{d}E_f = \hbar c k_f$. With these values, the doubly-differential scattering cross section is $I_{sc}/(N_0\hbar c^2 \mathrm{d}k_f \mathrm{d}\Omega/V)$, or

$$\frac{\mathrm{d}^2\sigma}{\mathrm{d}\Omega\mathrm{d}E_f} = r_0^2(\boldsymbol{\epsilon}_i \cdot \boldsymbol{\epsilon}_f)^2 \frac{k_f}{k_i} \sum_f |\,\langle f|\, \mathrm{e}^{-i\boldsymbol{q}_{fi}\cdot\boldsymbol{x}}\, |i\rangle\,|^2 \delta(E_f - E_i + \hbar\omega). \tag{4.10}$$

I have implicitly assumed that neither is there absorption in the sample nor do photons scatter more than once. Both have similar effects, since both limit the depth into a sample that the incident beam penetrates. (Multiple scattering implies that the scattered beam is sufficiently strong that the second scattering from this beam is detectable. This has two effects: the scattered intensity is redistributed in the $\boldsymbol{k}$ space, and the incident intensity varies significantly with depth in the sample–the same net effect as absorbtion.) This assumption has the interesting consequence that an infinite sample would scatter an infinite number of photons, so that more photons are scattered than are incident on the sample, which cannot be true. The proper treatment is discussed by Batterman and Cole[111], but for protein crystallography Equation 4.10 is an excellent approximation.

I have not yet made any comment about whether the scattering is *elastic*, that is, whether $|\boldsymbol{k}_i| = |\boldsymbol{k}_f|$. There is no need for this constraint, though it is frequently arbitrarily and separately enforced in derivations of scattering. Equation 4.10 contains both the Compton and Thomson scattering in one unified form. Fortunately, unless the initial and final electronic states are the same, the matrix elements in Equation 4.10 are generally (though not always) small. Thus, unless the Fourier transform of the ground state electron density is small, elastic scattering from a sample dominates over all other scattering processes.

This raises another important issue. We do not, in a typical crystallography experiment, have any energy resolution at all. (It is entirely possible to measure the scattering cross section as defined in Equation 4.10, as has been done for water[112] and other materials.) We instead measure the cross section defined in Equation 4.10 integrated over all possible final photon energies. We use the fact that $\sum_f |n\rangle \langle n| \equiv 1$, and integrate over final energies to find that the measured scattered intensity is

$$I(\boldsymbol{q}_{fi}) \propto \frac{\mathrm{d}\sigma}{\mathrm{d}\Omega} \propto \langle i| \, \mathrm{e}^{-i\boldsymbol{q}_{fi}\cdot(\boldsymbol{x}-\boldsymbol{x}')} \, |i\rangle \tag{4.11}$$

which is the Fourier transform of the density autocorrelation function.

Finally, I have only considered the electronic states of a system. Indeed, all charged particles scatter photons, but since the mass of the scattering particle enters twice in the denominator of Equation 4.10 (there as part of the classical electron radius), heavier particles like protons scatter much less than lighter particles like electrons.

4.1.2 Scattering factors

Given an electron density $\rho(\boldsymbol{r})$, the scattering factor is defined as

$$f(\boldsymbol{q}) = \int_V \rho(\boldsymbol{r}) \mathrm{e}^{i\boldsymbol{q}\cdot\boldsymbol{r}} \mathrm{d}^3 r, \tag{4.12}$$

where $\boldsymbol{q} = \boldsymbol{k}_s - \boldsymbol{k}_i$ is the difference between the scattered and incident wavevectors, and the integral is over the scattering volume (the intersection of the photon beam and the sample.) Simply put, the scattered amplitude in $\boldsymbol{q}$ space (or *reciprocal* space) is the Fourier transform of the electron density of the specimen.

A crystal is the convolution of some basic repeat unit (often several copies of one protein in different orientations) with a lattice, defined by *lattice vectors* $\boldsymbol{R} = \sum_i a_i \boldsymbol{\epsilon}_i$, where $\boldsymbol{\epsilon}_i$ are the real-space *basic vectors* ($i = 1, 2, 3$) and a_i are integers. If the scattering factor of the basic repeat unit is $F(\boldsymbol{q})$, then the scattering from a crystal will be

$$F_{cryst}(\boldsymbol{q}) = F(\boldsymbol{q}) \sum_{\boldsymbol{R}} \mathrm{e}^{i\boldsymbol{q}\cdot\boldsymbol{R}}. \tag{4.13}$$

We may consider particular atomic models for the electron density if we wish, but these can be generated easily from Equation 4.12, by producing a model of each atom's individual electron density. These are frequently more easily characterized by the *atomic scattering factors*

$$f(\boldsymbol{q}) = f_0(\boldsymbol{q}) + f'(\boldsymbol{q}) + if''(\boldsymbol{q}). \tag{4.14}$$

To construct $F(\boldsymbol{q})$, we simply sum up these contributions analogously to Equation 4.13. The first term f_0 is from Equation 4.12, and the second and third terms account for *anomalous scattering*. In the derivation of Equation 4.12 it was assumed that all electrons are free, which is not true for core shell electrons in real atoms. The energy of x-ray photons may also be quite near the quantum mechanical binding energies of these electrons, so that absorption and near-resonance must be considered. f' and f'' account for these effects.

Our experiment measures the differential scattering cross section, $I(\boldsymbol{q})$ (Equation 4.11), which is proportional to the squared *amplitude* of the scattering factor F. This gives rise to the *phase problem* encountered in inverting diffraction data to real-space electron density. It will be discussed in more detail later in this chapter.

4.1.3 The reciprocal lattice and symmetry

In Equation 4.13, the sum over all lattice vectors in a perfect crystal having N cells is, assuming N to be very large:

$$\sum_{\boldsymbol{R}} \mathrm{e}^{i\boldsymbol{q}\cdot\boldsymbol{R}} = N \iff \boldsymbol{q}\cdot\boldsymbol{R} = 2\pi n, \tag{4.15}$$

where n is an integer, otherwise the sum is essentially equal to zero[1]. Defining the *reciprocal lattice basic vectors*

$$\boldsymbol{\epsilon}_i^* = 2\pi \frac{\boldsymbol{\epsilon}_j \times \boldsymbol{\epsilon}_k}{\boldsymbol{\epsilon}_i \cdot (\boldsymbol{\epsilon}_j \times \boldsymbol{\epsilon}_k)}, \tag{4.16}$$

[1] For a finite crystal, the modulus of the sum is actually $\sin(N\pi\xi)/\sin(\pi\xi)$ where ξ measures the distance in reciprocal space from a particular Bragg reflection h, so that $\boldsymbol{q}\cdot\boldsymbol{R} = 2\pi(n+\xi)$. The sum goes to N as $\boldsymbol{q}\cdot\boldsymbol{R} \to 2\pi n$. This is a straightforward application of geometric sums; see Nielsen and McMorrow[108] for further discussion.

such that $\boldsymbol{\epsilon}_i \cdot \boldsymbol{\epsilon}_j^* = \delta_{ij}$, we find solutions to equation 4.15

$$\boldsymbol{G} = \sum \boldsymbol{G}_i \boldsymbol{\epsilon}_i^* = h\epsilon_1^* + k\epsilon_2^* + l\epsilon_3^*. \tag{4.17}$$

The integers h, k, l are the *Miller indices* of the reciprocal lattice point. Equation 4.15 is satisfied if $\boldsymbol{q} = \boldsymbol{G}$.

The Laue condition Equation 4.15 is not particularly easy to visualize on its own. Instead, consider a sphere of radius $|\boldsymbol{k}_i| = |\boldsymbol{k}_s| = 2\pi/\lambda$ tangent to the origin of reciprocal space $\boldsymbol{q} = \boldsymbol{0}$. (See Figure 4.1.) Here λ is the wavelength of x-rays, which defines the magnitude of the incident and scattered wavevectors. The incident wavevector ends at the origin, and its beginning defines the center of the sphere. The scattered wavevector $\boldsymbol{k}_s$ begins at the center, and its end traces out the sphere. Any point $\boldsymbol{G}$ of the reciprocal lattice which lies on the sphere is a solution to Equation 4.15. For convenience the angle between the incident and scattered wavevectors is $2\theta_s$. This *Ewald construction* leads to a form of the more familiar *Bragg condition* for diffraction:

$$G = \frac{2\pi}{d} = \frac{4\pi}{\lambda}\sin\theta_s. \tag{4.18}$$

The spacing d between Bragg reflection planes is defined by the Laue condition.

Some symmetry properties of diffraction are important in checking the quality of experimental data. From Equation 4.12, we can see that $F(\boldsymbol{G})$ is the complex conjugate of $F(-\boldsymbol{G})$. Thus the amplitudes and intensities of these two diffracted beams are identical. Such related reflections are referred to as *Friedel* or sometimes *Bijvoet* pairs. Checking that their intensities match is a useful check of the unit-cell indexing and data merging steps described later on.

Additionally, any symmetry of the lattice or of the unit cell will introduce symmetries into the diffraction. T4 lysozyme crystallizes in spacegroup P$3_2$21,

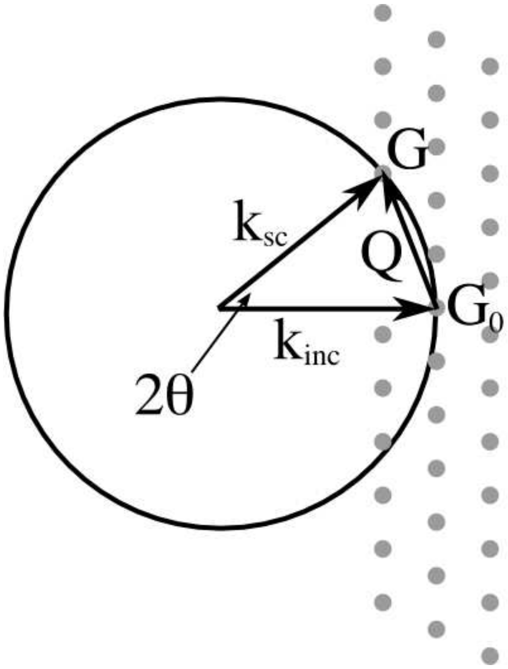

Figure 4.1: The Ewald Sphere. The Laue condition is satisfied when the origin and another point on the reciprocal lattice (grey circles) fall on the sphere of radius $|\boldsymbol{k}_{inc}| = |\boldsymbol{k}_{inc}| = 2\pi/\lambda$.

whose symmetry operations are listed in Table 4.1. Each operation effectively divides the unit cell into *asymmetric units*; P$3_2$21 has six asymmetric units.

The easiest way to determine the symmetry in the diffraction pattern due to a symmetry in the electron density is to observe that for a coordinate transformation $\boldsymbol{r} \to \mathsf{R}\boldsymbol{r} + \boldsymbol{t}$ which preserves the electron density $\rho(\boldsymbol{r}) = \rho(\mathsf{R}\boldsymbol{r} + \boldsymbol{t})$, the corresponding structure factors are

$$\begin{aligned} F(\boldsymbol{q}) &= \int \rho(\boldsymbol{r}) \mathrm{e}^{i\boldsymbol{q}\cdot\boldsymbol{r}} \mathrm{d}^3 r \\ &= \int \rho(\mathsf{R}\boldsymbol{r} + \boldsymbol{t}) \mathrm{e}^{i\boldsymbol{q}\cdot(\mathsf{R}\boldsymbol{r}+\boldsymbol{t})} \mathrm{d}^3 r \\ &= F(\mathsf{R}^T\boldsymbol{q}) \mathrm{e}^{i\boldsymbol{q}\cdot\boldsymbol{t}}. \end{aligned} \tag{4.19}$$

In this transform, R is a rotation matrix and $\boldsymbol{t}$ a translation. Since the translation adds only a phase, it will not affect the symmetry of the diffraction pattern. It can however lead to extinction for reflections along the translation axis[113].

Table 4.1: Symmetry operations of space group P$3_2$21. Real-space coordinates x, y, z are expressed as fractions of the basis vector lengths. $i = -h - k$. From the International Tables for Crystallography, volumes A and B.

Real-space	Reciprocal-space
symmetry	symmetry
$x, y, z \to x, y, z$	$h, k, l \to h, k, l$
$x, y, z \to -y, x - y, z + 2/3$	$h, k, l \to k, i, l$
$x, y, z \to y - x, -x, z + 1/3$	$h, k, l \to i, h, l$
$x, y, z \to y, x, -z$	$h, k, l \to k, h, -l$
$x, y, z \to x - y, -y, 1/3 - z$	$h, k, l \to h, i, -l$
$x, y, z \to -x, y - x, 2/3 - z$	$h, k, l \to i, k, -l$

4.1.4 Temperature factors

Thermal disorder in the crystal results in an overall reduction of the diffracted intensity[105, 108] by the factor $\exp(-2B\sin^2\theta_s/\lambda^2)$. The temperature factor $B = 8\pi^2 < \Delta r^2 >$, where $< \Delta r^2 >$ is the mean squared displacement of atoms in the protein crystal. The overall correction is often referred to as the Debye-Waller factor.

In practice, many kinds of disorder contribute to a reduction in diffracted intensity indistinguishable from the Debye-Waller factor. The intensity $I(\boldsymbol{G})$ is the product of $F(\boldsymbol{G})$ and its complex conjugate $F^*(\boldsymbol{G})$, and can be written in terms of the temperature-adjusted individual atomic scattering factors $f_{DW,i}(\boldsymbol{G})$ and positions $\boldsymbol{r}_i$ as

$$I(\boldsymbol{G}) = F(\boldsymbol{G})F^*(\boldsymbol{G}) = \sum_{i,j} f_{DW,i}(\boldsymbol{G})f^*_{DW,j}(\boldsymbol{G})\mathrm{e}^{i\boldsymbol{G}\cdot(\boldsymbol{r}_i-\boldsymbol{r}_j)}. \tag{4.20}$$

If the sum is over all atoms in a large crystal (in the same sense as in Equation 4.15), then we can approximate that the *average* intensity is

$$< I(\boldsymbol{G}) >= \sum_{i,j} f_{DW,i}(\boldsymbol{G})f^*_{DW,j}(\boldsymbol{G}) < \mathrm{e}^{i\boldsymbol{G}\cdot(\boldsymbol{r}_i-\boldsymbol{r}_j)} >= \sum_i f^2_{DW,i}. \tag{4.21}$$

The averages (denoted by brackets <>) are generally taken over shells of $\boldsymbol{G}$ having magnitude between G and $G + \delta G$. Usually twenty or so such shells are used to cover the entire dataset. Even in this case, the phase angles vary enough so as to be distributed randomly so that the approximation in Equation 4.21 is valid. The factors $f_{DW,i}(\boldsymbol{G})$ are generally assumed to be isotropic and so are essentially constant within each shell. This can then be used to scale measured intensities, by a constant C, to absolute values appropriate for inverting Equation 4.12:

$$< I(\boldsymbol{G}) >_{meas}= C < I(\boldsymbol{G}) >_{abs}= C\mathrm{e}^{-2B_{ave}\sin^2\theta_s/\lambda^2}\sum_i f_i^2(\boldsymbol{G}). \tag{4.22}$$

A *Wilson plot* plots $\ln(< I > / \sum f_i^2)$ *versus* $\sin^2 \theta_s/\lambda^2$ to determine this scaling. Both the overall temperature factor B_{ave} and the scaling constant C are used to put the data on an absolute scale for later refinement. In practice, Wilson plots should be linear over a broad range of data, as in Figure 4.2. If they are not, the data should be examined more closely for indexing or integration errors.

I have not derived the form of the Debye-Waller factor, but were I to do that, we would find a second important term, involving the correlation between atomic displacements[108]. This term gives rise to *thermal diffuse scattering* or TDS for short. It contains information about low-energy elastic modes and also static defects in the lattice[108]. We will not consider it further here, as the TDS is generally weak in protein crystals.

4.2 Data reduction, refinement, and error

4.2.1 Reduction and observational uncertainties

We do not know the intensity of the incident x-ray beam or the absorption of the crystal, which may vary from image to image for a variety of reasons, including changing path length of x-rays through the crystal as it rotates or the decay of the synchrotron electron or positron fill. Thus the images need to be scaled together via some scale factor K_{l} where l is a *layer*, usually a single diffraction image.

We measure structure factor amplitudes F_{hl}, where h stands for a particular x-ray reflection and the measurement is in layer l. (Note that in this section, l *does not* stand for a Miller index.) These are generally corrected for polarization and geometry dependent factors[108] before the process of scaling begins. In most cases, there will be multiple measurements of each reflection. Associated with each amplitude is a standard uncertainty σ_{hl}, derived from Poisson statistics as the square-root of the intensity of the reflection.

Various schemes for image scaling exist, but I will only comment on that used by the program *Scalepack*[114, 115]. It uses a non-linear least squares method to minimize the function

$$\Psi = \sum_{\mathsf{h}} \sum_{\mathsf{l}} \frac{1}{\sigma^2_{\mathsf{hl}}} (F^2_{\mathsf{hl}} - K_{\mathsf{l}} F^2_{\mathsf{h}})^2, \tag{4.23}$$

where F_{h} is a weighted average of the F_{hl}[115]:

$$F^2_{\mathsf{h}} = \frac{\sum_{\mathsf{l}} \frac{1}{\sigma^2_{\mathsf{hl}}} K_{\mathsf{l}} F^2_{\mathsf{hl}}}{\sum_{\mathsf{l}} K^2_{\mathsf{l}} / \sigma^2_{\mathsf{hl}}}. \tag{4.24}$$

Errors are estimated from the covariance matrix of this minimization (see discussion of error in maximum likelihood methods below). These should in principle be much smaller than the Poisson uncertainties of the individual reflections.

The statistic $R_{merge} = \sum_{\mathsf{h}} \sum_{\mathsf{l}=1}^{N} |F_{\mathsf{h}} - F_{\mathsf{hl}}| / \sum_{\mathsf{h}} N F_{\mathsf{h}}$, where N is here the number of images, is a typical indicator of integration and scaling quality[105, 114]. For completeness, R_{merge} is listed for each dataset in Table 3.1. However, note that R_{merge} is an *unweighted* sum, which takes into account neither the layer weights K_{l} nor any error in measurement. It is traditionally used to assess the quality of merging data from separate crystals, and is thus not ideal in measuring the quality of diffraction. Better choices to assess data quality and resolution are the resolution shell averaged values of I/σ_I or F/σ_F. These are plotted in Figure 4.3 for a typical dataset. It is crucial that one indicate the scaling algorithm used in calculating these parameters, because their values can depend significantly on the reduction procedure[114, 115].

At the end of the scaling step, one should examine at least the Wilson plots for the data. One such plot is shown in Figure 4.2 for the mt2k1 dataset. Wilson

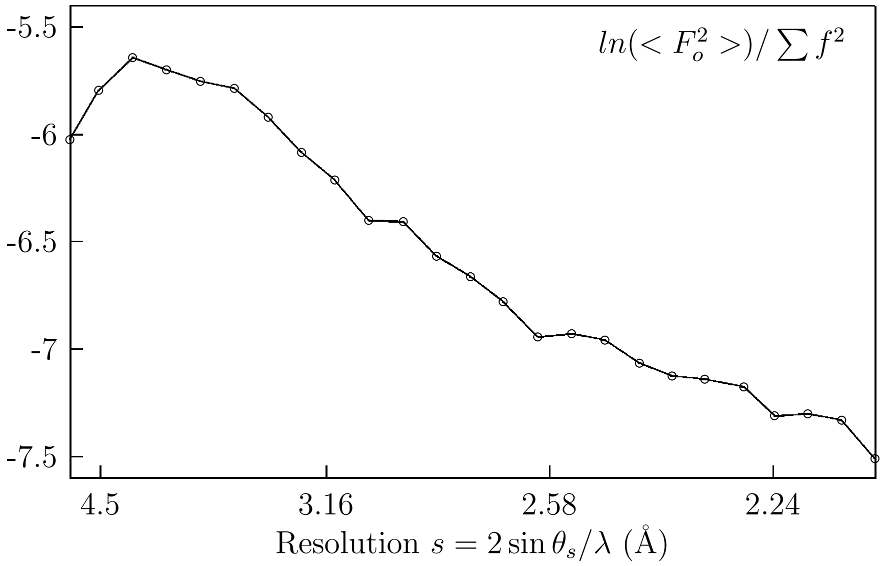

Figure 4.2: A Wilson plot for the mt2k1 dataset.

plots are never perfectly linear, and at low resolution (left side of the figure), the assumptions made for the plot fail. This plot is typical of the data described in this thesis, and indicates adequate data for further refinement.

4.2.2 The phase problem

As noted above, we collect diffraction intensities as data, and wish to invert them to obtain an electron density. The inverse Fourier transform of Equation 4.12 requires the phases of the complex structure factors, which (at present) we cannot measure in normal crystallography experiments. A number of techniques known as *direct methods* use theoretical means to obtain these phases, but the vast majority of crystallogaphy attempts to solve structures without these methods.

More frequently, a small set of phases is obtained by using anomalous diffraction techniques, such as *multiple anomalous diffraction* (MAD), which uses the wavelength dependence of atomic scattering factors in absorbing atoms to construct

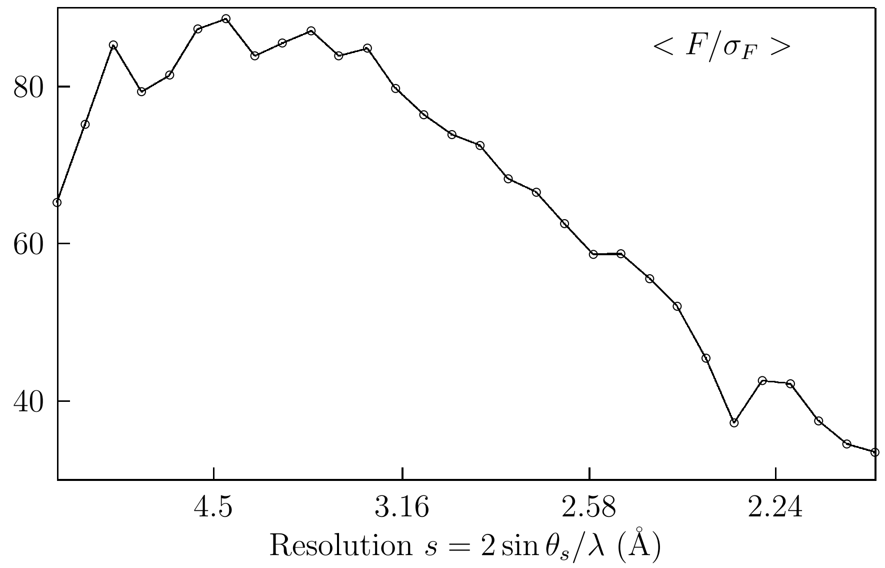

Figure 4.3: Structure factor amplitudes divided by standard uncertainties, versus resolution.

a Patterson map[105] of a small subset of atoms in the molecule. This provides a starting guess at the phases which can be used to bootstrap refinement. It is nonetheless necessary to go through an often laborious process of *model building* in which the molecule is threaded through a probably disconnected and noisy electron density map generated by the Fourier transform of observed structure factor amplitudes convolved with rough guesses at phases. Usually manual intervention is needed. As this technique is not used for our work, we will not consider it further here.

Here we work with proteins of well known structure at ambient pressures. This allows us an in principle easy solution to the phase problem[2]: we calculate them using Equation 4.12 from the electron density of the known structure. We bypass

[2] It is interesting to note that errors in the phases themselves are believed to be quite large[116]. That this does not severely hinder refinement speaks to the robustness of the methods.

model building and proceed directly to refinement. One may question the validity of this approach. We are assuming that the pressure-induced changes in the structure will be sufficiently small so as to not drastically change the structure factor phases and make refinement from the ambient pressure structure to the high pressure structure difficult or impossible. As previously discussed, and documented in the literature[12, 14], pressure-induced changes are on the order of tenths of angstroms. Ultimately, the validity of this assumption will be tested by whether or not we can refine the data to a reasonable structure. As we will see later in this chapter, that is indeed posssible.

For the mt2k1 dataset, the phases change fairly uniformly as refinement proceeds, and mostly independent of resolution $s = 2\sin\theta_s/\lambda$, by 25-30 degrees. Phase errors in well refined structures can be easily this large[116], even if the structure is quite accurate.

4.2.3 Maximum likelihood and error formalism

The statistical method which we use to determine a protein structure from x-ray diffraction data is known as *maximum likelihood.* When we determine the structure of a macromolecule from a set of data, we are really asking *which, out of the ensemble of all possible structures, is the structure most probably represented by the data?* Formally, we maximize the conditional joint probability of a vector of parameters $\boldsymbol{x}$ given a vector of *amplitude* observations $\boldsymbol{F}_o$, $P(\boldsymbol{x}; \boldsymbol{F}_o)$ (It is not necessary to choose only one such structure; using Monte Carlo methods, or other techniques, it can be possible to determine a meaningful ensemble of structures which are consistent with the data, *e. g.* [117].) By Bayes' theorem, this *posterior* *probability* is[118]

$$P(\boldsymbol{x};\boldsymbol{F}_o) = p(\boldsymbol{x})P(\boldsymbol{F}_o;\boldsymbol{x})/P(\boldsymbol{F}_o) = p(\boldsymbol{x})L(\boldsymbol{x};\boldsymbol{F}_o). \tag{4.25}$$

$p(\boldsymbol{x})$ is the *prior probability distribution* of the parameters $\boldsymbol{x}$, which reflects any previously known features of the model (see below). $L(\boldsymbol{x};\boldsymbol{F}_o)$ is the *likelihood function*, a measure of the probability of observed structure factors given a particular model.

Generally we use a variant of Equation 4.25 with calculated stucture factors $\boldsymbol{F}_c$ in place of atomic coordinates (model parameters) $\boldsymbol{x}$, for the simple reason that it is easier to make the comparison in reciprocal space than in real space. (The model with which we calculate the $\boldsymbol{F}_c$ will be considered later.) One occasionally will find a reference to "real space residuals" (*e. g.* [119]) where the experimental data are inverted to generate an electron density which is then compared with the model density. I have not used real space residuals in this work, except in the sense of seeking out places where there may be atoms missing in the model.

The likelihood function L is of chief concern here, and it is here that the various refinement programs diverge. All model fitting routines, no matter how simple, or how hidden, implicitly use the concept of maximum likelhihood. Here we will discuss the likelihood function used in Refmac version 5[118], used for the refinements in this work.

A good example that will familiarize the reader with the concept underlying maximum likelihood is the familiar *least squares residual.* It is the negative log of the probability function

$$P(\boldsymbol{O};\boldsymbol{x}) = \prod_i exp\left(-\left(O_i - y_i(x_1,...,x_n)\right)^2\right), \tag{4.26}$$

which is the joint normal probability that some function y_i of model parameters $\boldsymbol{x}$ yields the observed data $\boldsymbol{O}$. (Note that the components of the vectors are denoted as O_i and x_i in Equation 4.26.) Here no prior probabilities of parameters or uncertainties in the measurements are included, which amounts to the assumption that the uncertainties are all 1, in the units of measurement corresponding to the observations $\boldsymbol{x}$. (Thus the exponential factor in Equation 4.26 is assumed dimensionless) The *log-likelihood* form (here, the least squares residual $\sum(O_i - y_i)^2$) is much more convenient to work with. The refinements described below all use the log-likelihood form in their actual implementation.

4.2.4 The likelihood function

A number of assumptions must be made to make use of Equation 4.25 in model refinement. These are not always well justified, and in some cases it has been shown that refinement under these assumptions leads to underestimated parameter errors, e.g. [116]. Nonetheless we must make such approximations to move forward.

We first assume that the probability $P(F_{o,\mathsf{h}}|; F_{c,\mathsf{h}})$ of observing any one observed amplitude $F_{o,\mathsf{h}}$, with indices $\mathsf{h} \equiv h, k, l$, given the model-calculated amplitude $F_{c,\mathsf{h}}$ is independent of all other amplitudes, so that we may write the joint probability as a product of probabilities for each reflection. This is most likely not the case, but has proven to be a useful approximation[118].

Second, we assume that errors in atomic coordinates are both independent and have the same distribution for all atoms. This *cannot* hold if we apply stereochemical restrains to an atomic model as will be discussed below. All evidence (*e. g.* , [120] or [121]) suggests that this is nonetheless an improvement.

Following Srinivasan and Ramachandran[122], Randy Read[120] and others showed that under these assumptions, the probability of observing a reflection with Miller indices h, k, l to have (complex) structure factor F_p given the model calculated structure factor F_c *in the absence of observational error* is:

$$P(F_p; F_c) = \frac{1}{\pi\varepsilon\Sigma(1-D)} \exp - \left[\frac{|F_p - DF_c|^2}{\varepsilon\Sigma(1-D)}\right], \tag{4.27}$$

for acentric reflections [118]. D is a parameter involving the errors in coordinates, Δx, and temperature factors ΔB, $D =< \exp[-(\Delta B s^2/4)] \cos 2\pi s \Delta x >$. The brackets $<>$ indicate a probability weighted average over the distributions of Δx and ΔB. $\Sigma = \Sigma_j^{N_{atoms}} f_j^2(s)$, where $s = 2\sin\theta_s/\lambda$. ε is the multiplicity of the particular diffracted reflection. Some modification is necessary for centric reflections (for which phases are more restricted), but the general form is the same.

This distribution is actually quite easy to derive[120, 122], although at first glance it seems mysterious. We begin from the probabilities of positions of individual atoms, which is the convolution of the positions themselves and some probability distribution which we take to be the same for all atoms. The probability of each structure factor is determined from the Fourier transform of the probability distribution of atomic positions. The centroid of observed structure factors F_o is then simply some complex multiplier times the corresponding model structure factors F_c. That complex multiplier is D. Thus Equation 4.27 is simply a Gaussian distribution of observed structure factors about an appropriate mean value determined by properly considering how model uncertainties propagate through to the structure factors.

The effects of observational uncertainties σ_{F_o}, estimated in the integration and scaling step, are added in heuristically by letting $\varepsilon\Sigma(1-D) \rightarrow 2\sigma_{F_o}^2 + \varepsilon\Sigma(1-D)$ in Equation 4.27. Though it is imcomplete, there is a reasonable foundation for this approximation[121]

We now return to Equation 4.25 and consider the probability of a given observed amplitude. To do so, we integrate over all phase angles with some probability distribution for phase angles. Assuming all phases are equally likely, we obtain[118, 122]

$$P(|F_o|;|F_c|) = \frac{2|F_o|}{2\sigma_{F_o}^2 + \varepsilon\Sigma(1-D)} \exp\left(-\frac{|F_o|^2 + |DF_c|^2}{2\sigma_{F_o}^2 + \varepsilon\Sigma(1-D)}\right) \times I_0\left(\frac{2|F_o||DF_c|}{2\sigma_{F_o}^2 + \varepsilon\Sigma(1-D)}\right) \tag{4.28}$$

for acentric reflections, where I_0 is one of the hyperbolic Bessel functions. Again, the differences for centric reflections are minimal, and reflect the fact that the distribution for centric structure factors is one dimensional, not two dimensional.

A number of extensions can be made to consider special cases, for instance where phases are better known or where parts of the model are disordered or missing entirely. These are treated in the references[118, 120–123].

It is also the case that before 1997 or so, refinement was based upon simple least squares residuals using either intensities or amplitudes of reflections. This assumes, among other things, that the errors in measurement are the same for all h, k, l and that there are no phase errors. On the other hand, it avoids potentially difficult covariances in fitting, as we have now added in many parameters with similar effects on the model (the D, or their cousins σ_A[123]).

4.2.5 Error formalism

We would like an estimate of the uncertainty in our final, refined model. For simplicity we will ignore the effects of restraints[3], and also assume a least squares

[3] Formally, they enter into the residual just as observations do[124].

residual, something of the form[124, 125]

$$\chi^2(\boldsymbol{x}) = \sum_{\mathsf{h}} w_{\mathsf{h}} \big(F_{o,\mathsf{h}} - F_{c,\mathsf{h}}(\boldsymbol{x})\big)^2, \tag{4.29}$$

where each reflection $\mathsf{h} \equiv h, k, l$ is weighted by the model-independent w_{h} (often involving estimated observational uncertainties), and $\boldsymbol{x}$ represents the model parameters. Our goal is to minimize this residual, using non-linear least squares tactics. The minimization condition is

$$\frac{\partial \chi^2}{\partial x_i} = -2 \sum_{\mathsf{h}} w_{\mathsf{h}} (|F_{o,\mathsf{h}}| - |F_{c,\mathsf{h}}(\{x_i\}|) \frac{\partial F_{c,\mathsf{h}}}{\partial x_i} = 0, \tag{4.30}$$

for all model parameters x_i. Given a starting set of model parameters, how do we arrive at the parameters that minimize the residual 4.29? Sufficiently near the solution, the problem is essentially quadratic and described by a *Hessian* or *curvature matrix.* More generally, the problem must be solved iteratively. In either case, we want to find the jump δx_i which reduces error in the model. There are number of possible methods. One is the linear guess[126],

$$\delta \xi_{\mathsf{h}} = \sum_i \frac{\partial F_{c,\mathsf{h}}}{\partial x_i} \delta x_i, \tag{4.31}$$

where i is a model parameter index, c refers to calculated structure factors, and $\mathsf{h} \equiv h, k, l$ stands for the Miller indices of a particular reflection. Writing this in matrix form we have $\delta \boldsymbol{\xi} = \mathbf{A} \delta \boldsymbol{x}$, where $A_{\mathsf{h},i} = \partial F_{c,\mathsf{h}} / \partial x_i$. Now, $\delta \boldsymbol{\xi}$ is the vector of residuals for each reflection and $\delta \boldsymbol{x}$ the (guessed) vector of parameter changes that will minimize χ^2. Some matrix rearrangement yields

$$\mathbf{N} \delta \boldsymbol{x} = \boldsymbol{b}, \tag{4.32}$$

where $\boldsymbol{b} = \mathbf{A} \delta \boldsymbol{\xi}$. $\mathbf{N}$ is called the normal matrix, and for our purposes its elements are approximated by[125]

$$N_{ij} = \sum_{\mathsf{h}} w_{\mathsf{h}} \frac{\partial F_{c,\mathsf{h}}}{\partial x_i} \frac{\partial F_{c,\mathsf{h}}}{\partial x_j}. \tag{4.33}$$

The normal matrix is particularly important because it is the inverse of the *covariance matrix* $\mathbf{C}$. Standard uncertainties are obtained from the diagonal elements of $\mathbf{C}$ as $\sigma^2(x_i) = C_{ii} \equiv (\mathbf{N}^{-1})_{ii}$.

The normal matrix can also be derived by expanding the residual as[125]

$$\chi^2 = \gamma - \boldsymbol{d} \cdot \boldsymbol{x} + \frac{1}{2}\boldsymbol{x} \cdot \mathbf{D} \cdot \boldsymbol{x} \tag{4.34}$$

where $\mathbf{D}$ is the curvature matrix, $\boldsymbol{d}$ is the gradient of χ^2 (which should be zero at a minimum), and $\boldsymbol{x}$ is again a vector of the model parameters. In this case, where the curvature matrix can be calculated directly, the best guess of a change in parameters is $\delta\boldsymbol{x} = \mathbf{D}^{-1} \cdot [-\nabla_i \chi^2]$. This is identical to Equation 4.32.

Though in principle inversion of the matrix $\mathbf{N}$ permits direct estimation of parameter uncertainty in a model, for a large system this is computationally extremely difficult. Moreover, there is a tacit assumption in the calculation of this matrix that the χ^2 surface is in fact quadratic to some good approximation. Covariance of the model parameters will arise because of our restraints, also skewing our error estimates.

Perhaps most importantly, we do not use a least-squares residual. The actual residual is more complicated and does not follow the simple argument above unless we are in fact very close to the “true” solution of the minimization[118].

Cruickshank[124] discusses these and other difficulties, ultimately coming to the conclusion that even for data sets and refinement procedures where calculation of the covariance matrix is possible and stable, it is extremely impractical. Instead he suggested the *diffraction-component precision index*, or DPI. He has compared his DPI to the results of *full matrix inversion*[4] for superb data and well refined

[4] The process of calculating the covariance matrix.

models. There was substantial agreement between the two error calculations. For completeness, we include the R_{free} form of his DPI here:

$$\sigma_{DPI}(r, B_{ave}) = 3^{1/2}(N_{occ}/n_{obs})^{1/3}C^{-1/2}rR_{free} \tag{4.35}$$

where

- N_{occ} is the number of fully occupied sites (i. e. atoms in the model),
- n_{obs} the number of observations (reflections),
- C the fractional completeness of the dataset to d_{min},
- r the resolution limit of the data, and
- R_{free} is a measure of refinment progress defined in section 4.3.2.

This provides some useful guess of the *average* uncertainty in the model, but does not address individual atomic position uncertainties. We must use it cautiously, especially in areas of poor constraint or regions which have unusual temperature factors.

The derivation of Equation 4.35 is not at all trivial, and will be left to the references[124, 127]. Let it suffice to say that the formula is somewhat *ad hoc*, but with some grounding in the full matrix inversion method noted above. Due to this derivation, this formula should be viewed with some caution when using residuals other than least-squares. Cruickshank has modified his original formula to the form (4.35) in an attempt to address this problem, but verifying its accuracy is difficult.

4.3 Practical aspects of model refinement

4.3.1 The atomic model

I have not yet discussed the most significant limitation of the data, namely their number. A typical high-quality diffraction data set includes 10,000 to 20,000 reflection measurements. A simple cubic unit cell 50 Å on a side contains 125,000 1 Å^3 cubes. Thus the typical protein diffraction dataset cannot constrain a model of the electron density which independently specifies the density in each such 1 Å^3 cube; in such a model there are vastly more parameters than data.

To reduce the number of parameters, the electron density model is generated from a polypeptide chain of spherical atoms. This is of course not accurate, but is a sufficient approximation at the resolution of my work. Similarly, the use of isotropic temperature factors in constructing the model is also acceptable.

A further simplification is to use stereochemical "dictionaries" (*e. g.* [128]) which include information about bond lengths, bond angles, and sometimes torsion angles, all derived from high resolution small molecule and protein structures. These are implemented as restraints in the likelihood functions above.

We need to be careful how much we favor the dictionary over our data. Refinement packages have various means by which this is achieved. A general overall weighting parameter is used in a given refinement. This parameter can be varied, the refinement begun again, and the results compared with other refinements. Tighter constraints, favoring the stereochemical dictionary, are often needed for lower resolution datasets.

In our work, we may reasonably expect that there will be deviations from some kinds of parameters, and less deviation from others. For instance, we may expect that bond lengths will be quite rigid while torsion angles are less rigid. Therefore, we will want more control over how the weighting is implemented. Refmac version 5[118] provides such control. Again, this balance must be determined empirically through repeated attempts at refinement of the structure.

Some effort has been made to include multiple configurations of the protein in the electron density model[129]. Lindorff-Larsen *et al.*[130] have used molecular dynamics methods and NMR data to determine a possible ensemble of states of the protein. In principle one could examine the full distribution of structures and their probabilities defined by Equations 4.25 and 4.28. All such methods are computationally expensive, and the number of states which we can realistically model remains limited by our data.

This is a question of purpose. Do we need to know to 1 Å precision the positions of atoms, or do we need 0.1 Å precision? Until recently, stuctural biology has proceeded without worrying much over this issue. As Quantum Mechanical modelling of photosynthetic proteins (*e. g.* [131]) and studies such as my own move forward, we will require better models and better data.

It must be said that the model itself can be a limiting factor, and our task very much akin to fitting a straight line to exponentially distributed data. There will be some limit to the precision for which we can hope.

4.3.2 Practical refinement indicators

Crystallographers do not generally examine the likelihood function when refining their data (although they probably ought to do so.) Instead, the common progress indicators are the reliability index R,

$$R = \frac{\Sigma_{\mathsf{h} \in \mathsf{W}} F_{o,\mathsf{h}} - g F_{c,\mathsf{h}}}{\Sigma_{\mathsf{h} \in \mathsf{W}} F_{o,\mathsf{h}}}, \tag{4.36}$$

and its companion, the "free" reliability index R_{free}[132, 133]

$$R_{free} = \frac{\Sigma_{h \in T} F_{o,\mathsf{h}} - g F_{c,\mathsf{h}}}{\Sigma_{\mathsf{h} \in T} F_{h}^{o}}, \tag{4.37}$$

The sum is over some subset of the data, either the "working" set W used for refinement, or a "test" set T ($\mathsf{T} \cap \mathsf{W} = \emptyset$) in the case of R_{free}, and g is some Q-dependent factor meant to place the two sets of structure factors on an common scale. It is determined much as the scale factors for merging data (Section 4.2.1).

As we refine the structure, R should decrease if the weighting parameters for restraints are reasonable. We continue a given stage of refinement until R converges to some value. That value should be examined carefully. Values above 0.3 for good data sets of reasonable resolution (say 2 Å) should be considered suspect. Frequently such a large value indicates that something else is wrong, and the refinement will not proceed well from here. It may be necessary to return to model building, or there could be an error in the initial data processing.

Once R converges, we have a second problem, pointed out by Branden and Jones[132], and addressed by Axel Brunger[133]. Once we have begun a refinement, R will of course decrease. It has now been biased by the refinement process, leaving open the possibility of *overfitting*. In some spectacular cases such overfitting has led to left handed helices appearing in final structures[132].

To calculate R_{free} we set aside some data (on the order of 10%). This set must be mutually exclusive with the working data set used for refinement. R_{free} is used for the purpose of making sure the refinement is not running off in some unphysical direction. It has become widely accepted as a good measure of the validity of a structure.

It is important not to rely on R_{free} as an indicator of the progress of refinement, but rather to use it as a measure of the reliability of the final refined structure. Particularly in the early stages of refinement, the sum in Equation 4.37 taken over the test dataset is not in fact very accurate. Only as the refinement converges are these parameters meaningful[124].

Most important to us, R_{free} can itself be biased. Our refinement starts from a previously refined model. Unless our "free" reflections are the same as for that original model, some bias will leak in. In this case, it is often best to examine the deviations and distributions of bond lengths, angles, *et cetera* to determine the progress of refinement. In the final refinements here, I have relied heavily on examining the distributions of bond angles and lengths to determine the quality of refinement.

As it happens, most of the progress indicators yield essentially the same results. Several are of these plotted in Figure 4.4. Small rises in most measures correspond to the addition and removal of water molecules every 5 cycles of refinement. By all measures, the bulk of the refinement is complete after only 5 cycles. Cruickshank's DPI continues to decrease very slowly, as does R. Bond length deviations from the stereochemical dictionary decrease very little, and bond angle deviations decrease very slowly. The negative log likelihood continues to decrease throughout the refinement, but very slowly, despite the fact that R_{free} increases slightly. The extra refinement improves the structure slightly, primarily by properly modelling water surrounding the protein.

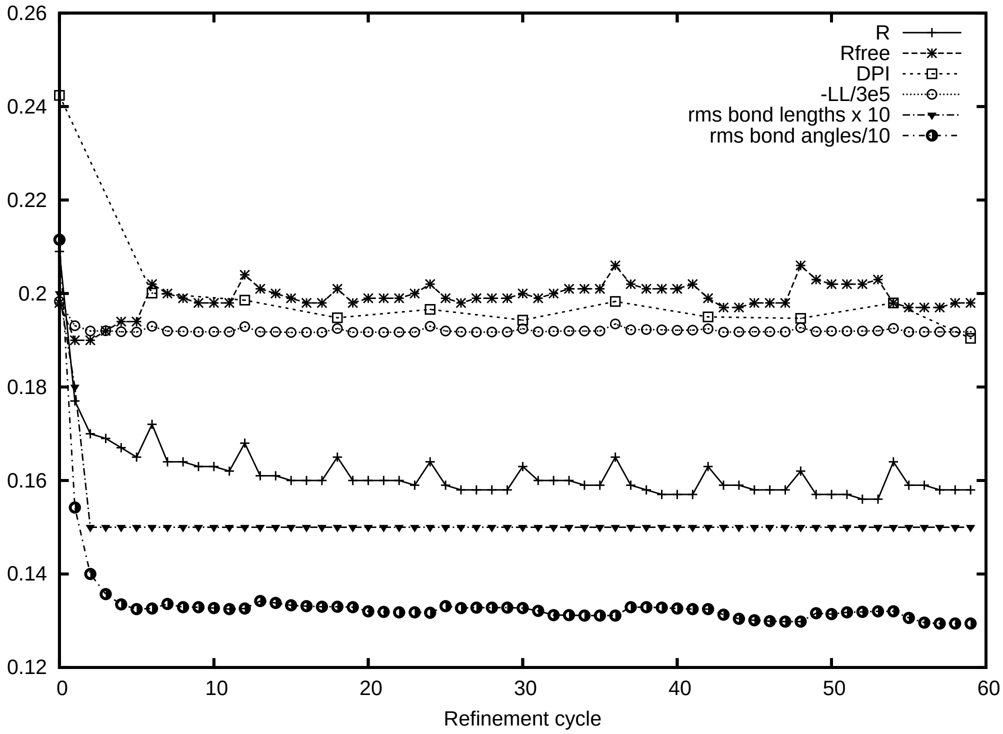

Figure 4.4: Progress of refinement for dataset mt2k1. R and R_{free} are defined in the text. DPI is given by equation 4.35, and is only plotted in Å every 5 cycles of refinement. -LL is the negative logarithm of the likelihood function, Equation 4.28, divided by 300,000 to put it on a convenient scale. The rms bond length (in Å) and bond angle (in degrees) deviations are also scaled so they are visible in this plot.

4.3.3 Ramachandran plots

Another indicator of refinement success is the Ramachandran plot[105]. The peptide bond itself is quite rigid, and planar. Thus two angles, ϕ and ψ determine the backbone configuration of the protein. Figure 4.5 shows the geometry. These angles are not restrained in refinement, but take on a limited subset of values in real proteins due to steric constraints. Thus they are useful for identifying possible problems with the model. Proline must be considered separately, since it is not strictly an amino acid and lacks the same peptide bond. Glycine has no side chain, and thus has much more relaxed constraints on ϕ and ψ. Good structures should have 90% or more of their residues in the allowed regions of the Ramachandran plot[134].

4.3.4 Electron density maps

A picture is worth a thousand statistics, and the numbers can only tell you that something is wrong, not where. At the end of any step in refinement, one must examine the structure carefully against the experimental electron density. Visualization is crucial in building a model of a *de novo* protein structure.

Since the phases of the Fourier components $F(\boldsymbol{G})$ are known only from the model, we will use these for both the experimental and calculated electron density. Thus the electron density is not entirely accurate, and often appears to be quite confusing until well into the refinement process. The "maps", as they are known, are calculated as[105]

$$\rho_{m,n}(x,y,z) = \frac{1}{V_{cell}} \sum_{\mathsf{h}} (mF_{o,\mathsf{h}} - nF_{c,\mathsf{h}}) \mathrm{e}^{-2\pi i(hx+ky+lz)+i\alpha_c}, \tag{4.38}$$

where m and n are generally integers. V_{cell} is the volume of the unit cell, and x, y, z

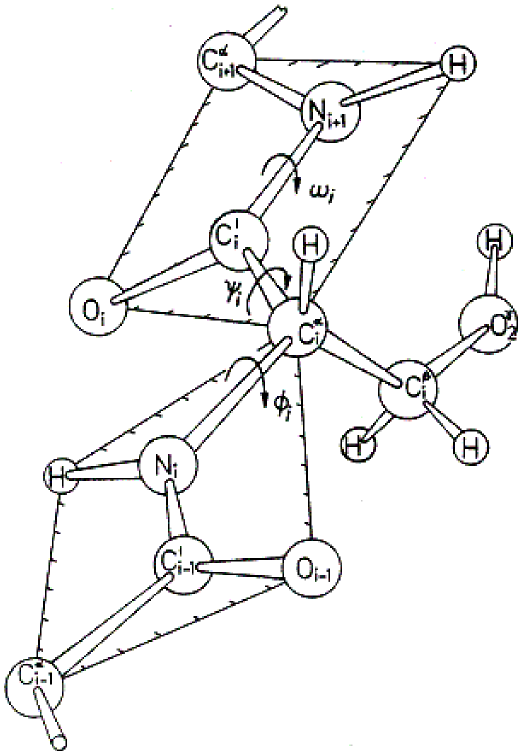

Figure 4.5: The peptide bond. Torsion angles ϕ and ψ are shown. From www.fkem2.lth.se/education/kursinfo/biofysikalisk_kemi/kursmaterial/ laborationer/ramachandran2004.htm

are *fractional* coordinates relative to the unit cell lengths (so that, *e. g.* , $0 \leqslant x < 1$). Setting $m = 2, n = 1$ approximates the experimental electron density, but includes "highlights" where the model disagrees with the data. It is useful particularly in initial modelling. The difference map constructed with $m = 1, n = 1$ is especially useful when looking for small changes (as those due to pressure) or positional errors.

As we'll see later in this dissertation, the electron density maps provide the best way of determining model completeness. If the model is incomplete, both the $2F_o - F_c$ and difference maps should have large (3 standard deviations) positive peaks. This is akin to the calculation of *OMIT* maps, which intentionally leave out parts of a questionable model. The omission biases the map to the remaining structure, so that any remaining density in an ambiguous region should be more reliable.

Other maps are frequently useful. I will leave their details to later chapters, where they are more topical.

Maps are generally displayed as a surface of constant value. This contour value is usually specified in increments of the standard deviation of electron density in the map, σ. Maps can be contoured at absolute electron densities, but we are most often interested in looking at unusual peaks or noisy areas, so scaling in terms of σ is adequate.

4.4 Refinement protocol and results

4.4.1 Procedures

The exact procedure used to refine molecules for this work has been made as automatic as possible in order to minimize errors and make comparisons between solved structures as meaningful as possible. Master scripts generate the actual scripts used by the CCP4 suite to refine protein structures.

Data were integrated and scaled using *DENZO*[114] at the CHESS computing facilities. Structure factors and uncertainties were converted to the CCP4 format and refined using CCP4 software, freely available at www.ccp4.ac.uk. Refinement of each dataset proceeded as follows:

1. Perform rigid body refinement of the molecule using the atomic model available from the Protein Data Bank (www.rcsb.org), either 1L63 for WT* or

1L90 for L99A T4 lysozyme, and Refmac version 5[118].

2. Perform ten cycles of restrained refinement using a final global restraint weight of 0.3 in Refmac version 5, alternating with automatic electron density map refinement using arp_waters (described in detail below).

3. Examine the model and $2F_o - F_c$ and $F_o - F_c$ maps using the program O[135].

When examining the electron density maps, it became clear that some measure of subjectivity could lead to significantly different final results, at least in modelling solvent water molecules. I adopted the use of the program *arp_waters*, as implemented in the CCP4 package, to make possible more objective analysis. The program determines points of correlation between the $2F_o - F_c$ and $F_o - F_c$ maps and subject to user-specified rules adds or removes water molecules from the model. The main parameters are:

1. A cutoff in the $2F_o - F_c$ map, below which water molecules should be removed (set to 1 standard deviation of the map electron density).

2. A cutoff in the difference map above which water molecules should be added (set to 3 standard deviations).

3. Maximum numbers of water molecules to add or remove in one cycle (set to 20 and 10 respectively).

After one cycle of adding and removing water molecules, a form of real space refinement is used to position the added oxygen atoms. It is then necessary to perform restrained refinement again.

The R values change very little when adding and removing individual atoms, highlighting the need to examine the maps and build the model manually. Similarly, the refinement should be monitored frequently to ensure that it is proceeding acceptably.

In this work, an enormous number of refinements was necessary to globally determine the best parameters. For consistency, I wrote a Perl script (www.perl.org) to generate CCP4 refinement scripts. One set of parameters was then used to refine all data sets. The set of parameters which optimized the datasets globally was used to refine the final models discussed in the next two chapters. This procedure should minimize artifacts between models.

In the final models, difference maps $(F_o(p) - F_o(0)) \exp(i\alpha_c(0))$ were calculated, using the ambient pressure model-calculated phases $\alpha_c(0)$, to verify the presence of water molecules in the cavities and any other pressure-induced differences. These maps will be described in more detail in Chapter 6.

4.4.2 Robust model quantities

One key point that is evident from the discussion of error above is that there are not particularly robust methods of estimating coordinate error on an atom by atom basis. Exceptional data and model fitting are required to make such estimates reliable and meaningful. In our case, the assumptions made in refinement, and even the model itself, are not particularly realistic, making the situation seem even more grim. We will be very concerned with uncertainty when we examine high-pressure effects on protein structures, and so we must consider how we will establish meaningful uncertainties.

I have taken a few different approaches to this problem. We can gain some qualitative sense of the robustness of refinement by synthetically adding noise to real data and rerefining the structure. By repeating this procedure many times we can see how noise in the data propogates into our structure. The results are not surprising. Uncertainties increase with decreasing average structure factor amplitudes. With more noise, the refinement is much more sensitive to the restraint weighting parameters, and the atomic positional uncertainties concurrently larger. R and R_{free} do in fact correlate with uncertainty, but this can be masked by fitting with different restraint weighting. Finally, it is crucial that the refinement has fully converged, or any estimation of error is invalid.

A better, more concrete method of error estimation is to first collect more data. With one data set we cannot really make any statistically significant statement. I have collected two or three data sets for each mutant at most pressures, and this will allow real experimental estimates of uncertainty rather than speculative estimates.

Even in the face of large uncertainties, we can reduce the error in measurable quantities by averaging over many atomic positions. Given N observations O_i, with weights (often derived from uncertainties) w_i, the average O is

$$O = \frac{1}{N} \sum w_i O_i, \tag{4.39}$$

so that uncertainties in the individual O_i propogate to the average quantity as

$$\begin{aligned} \sigma^2(O) &= \sum \sigma_i^2 \left(\frac{\partial O}{\partial O_i} \right) \\ &= \frac{1}{N^2} \sum \sigma_i^2 w_i \\ &= \sigma^2 / N. \end{aligned} \tag{4.40}$$

In the last line, we have assumed that all errors are the same and the weights w_i are all one. Then the error of the averaged quantity scales as $N^{-1/2}$. For a helix, which may contain 100 or more atoms, the uncertainty for the center of mass may be quite small. 10 or 20 atoms may even reduce uncertainty enough to make meaningful conclusions. Some changes in the structure may be visible above the uncertainties derived from examining multiple datasets; in other cases it may be necessary to examine averaged quantities.

Finally, there will be situations where the atomic coordinates are simply not the correct object of study. We must remember that the fundamental quantity in x-ray scattering is electron density. By comparing the diffraction data at various pressures, we can examine the electron density for changes and make some conclusions about changes to the atomic structure based on these. This is our only recourse when we search for missing parts of the atomic model, and is a useful fallback if the model has large uncertainties.

4.4.3 Refinement statistics and review

The results of all refinements are listed in Table 4.2. As one can see, the refinements were quite successful based on these statistics. The standard uncertainties listed in the table are a modified form of the DPI discussed above, more appropriate to the maximum likelihood formalism used in Refmac 5. See the program documentation for more details (www.ccp4.ac.uk). These uncertainties hover around the 0.1 Å level, which as we'll see in the next chapter is about what we determine from comparing multiple nominally identical structures.

The Ramachandran plots for all structures, wild-type and mutant, are all virtually the same. In almost all of the structures, residues Ile29 and Phe114 are slightly outside the normal range of ϕ, ψ values. (A typical plot is shown in Figure 4.6 on page 111.) This number of outliers is not at all abnormal, but it is still

Table 4.2: Refinement statistics. r, resolution range used in refinement. R and R_{free} defined in the text. esu is the estimated standard uncertainty based on the Refmac 5 maximum likelihood refinement. Rms bond lengths and angles are the root-mean-square deviations from the optimal values in the stereochemical dictionary (see text).

Name	r (Å)	R	R_{free}	esu (Å)	rms bond lengths (Å)	rms bond angles (°)
mt0k1	52.78-2.40	0.16027	0.21198	0.141	0.020	1.564
mt0ka	52.78-2.40	0.14414	0.19725	0.127	0.019	1.538
mt0kb	52.78-2.40	0.15023	0.21790	0.142	0.021	1.647
mt1k6	52.63-2.10	0.17015	0.21193	0.112	0.017	1.560
mt1k7	52.63-2.15	0.16586	0.21857	0.118	0.018	1.485
mt1k9	52.63-2.10	0.17177	0.23190	0.126	0.019	1.533
mt1.5k1	52.56-2.10	0.16449	0.21365	0.109	0.016	1.441
mt2k1	52.49-2.10	0.15785	0.19819	0.097	0.015	1.294
mt2k3	52.49-2.20	0.15780	0.20834	0.115	0.016	1.398
mt2k8	52.49-2.11	0.16069	0.20365	0.099	0.016	1.394
wt0ka	52.70-2.19	0.15558	0.22860	0.132	0.019	1.596
wt1k6	52.63-2.10	0.16117	0.22212	0.107	0.016	1.446
wt1k7	52.56-2.10	0.15743	0.21417	0.098	0.015	1.357
wt1ka	52.56-2.19	0.15017	0.20298	0.101	0.016	1.363
wt2k1	52.41-2.01	0.16007	0.21022	0.091	0.013	1.308
wt2k2	52.41-2.01	0.15860	0.21314	0.090	0.014	1.280

worthwhile to look at the electron density around these flagged residues. Examining the electron density around Ile29 (Figure 4.7 on page 112) indicates that the density is strong, so that this is little cause for concern.

In Figure 4.6, Ala112 is also barely outside the normal ϕ, ψ range; of all the refined structures, this is the only one in which Ala112 is outside the normal range. The density around residues 112-114 is a bit weak. In this region (the C-terminal end of helix F) the side chains are often in stronger density than parts of the main chain, and appear to constrain the main chain atoms. Helix F is typically less well ordered than the rest of the protein, even in the wild-type protein (private communication with Brian Matthews).

4.5 Summary

In this chapter I have laid out the basic principles of macromolecular crystallography. It is by no means a complete description. I wish to convey the sense that it is, despite much effort by many brilliant scientists, a somewhat tenuous process. We must be careful what we say about a structure, even if the refinement converges to excellent values. That said, the data appear to be very satisfactory. Uncertainties should be small, and the structure reliable. Keeping in mind the vagaries of the atomic model and error estimation, and any suspect positions in our specific structures, we may now begin to examine what actually happened to these molecules under pressure.

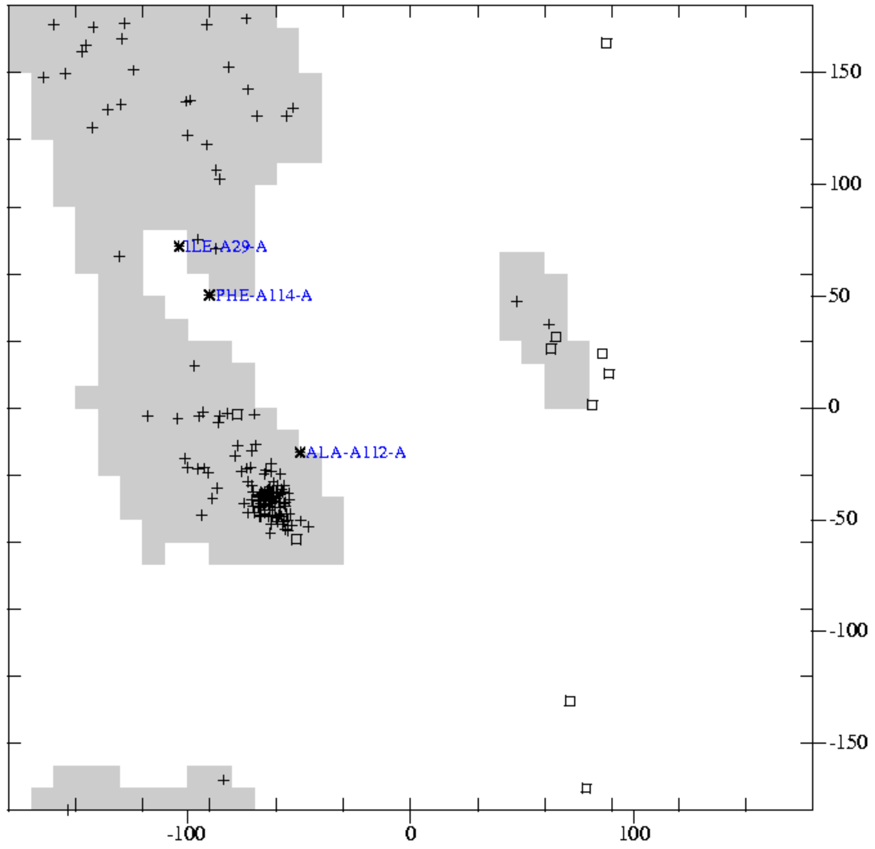

Figure 4.6: Ramachandran plot for structure mt2k8. The grey regions show allowed combinations of ϕ, ψ main chain torsion angles. Outliers are labelled with their residue name and number. Glycines, which are much less restricted, are shown as open squares. Figure made with *MOLEMAN2*, available from http://xray.bmc.uu.se/usf.

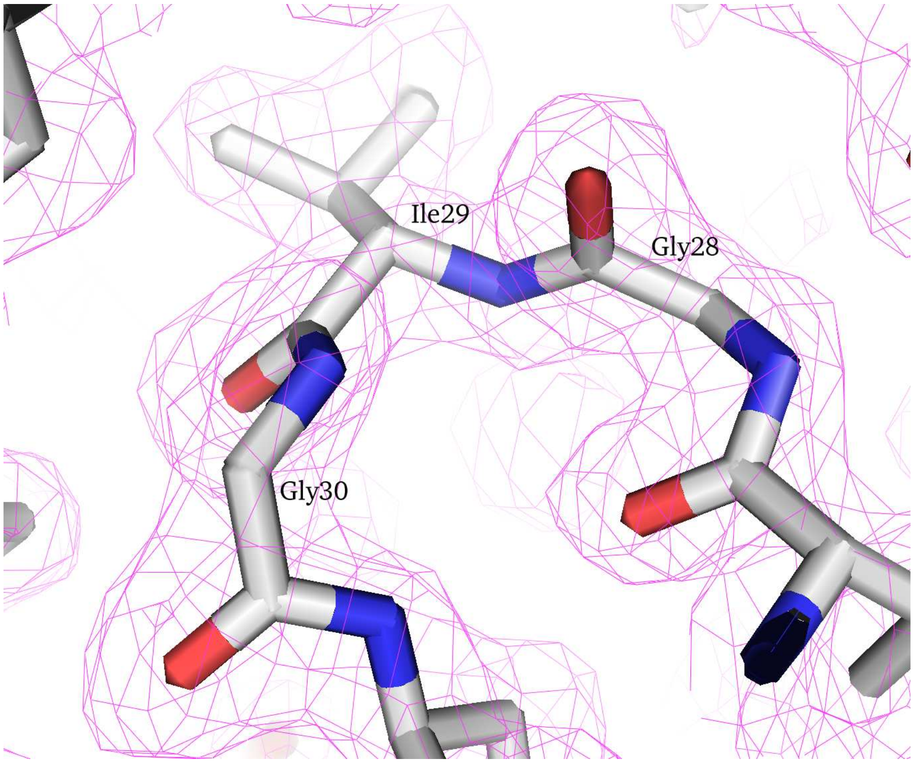

Figure 4.7: Electron density near Ile29 C_α in structure mt2k8. $2F_o - F_c$ electron density contoured at 1σ in magenta. The density is not noisy, is quite smooth, and is well fit by the model, despite the Ramachandran outlier Ile29.

Chapter 5

Structure of T4 lysozymes at high pressure

One possible reaction of a cavity containing protein to pressure is the structural collapse of that cavity. More generally, pressure will cause compression of part or all of the protein. In this chapter I will describe the observed changes in the model structures of the L99A and WT* T4 lysozymes. I begin with a survey of some of what is known about the structure, and follow that with a detailed description of how I compared structures. The observation that the model is incomplete as pressure increases–namely that the cavity fills with water–will be discussed in the next chapter.

5.1 T4 lysozyme at ambient pressure

5.1.1 Static structure

As noted in Chapter 1, T4 lysozyme has ten α-helices, a β-sheet, and is composed of two domains. The N-terminal domain is a mix of helices and sheet, and is somewhat smaller than the seven α-helix C-terminal domain. The domains are connected by a long α-helix (residues 60-80.) The molecule's structure is remarkably robust to mutation, as evidenced by the studies of the Matthews group at the University of Oregon.

The primary cavity of mutant L99A is actually an extension of a roughly 50 Å^3 cavity found in the wild type protein (Figure 5.1 on page 115). The mutation increases cavity volume by roughly 120 Å^3. Otherwise the structure of the L99A mutant and its parent WT* are virtually identical. The backbone rms deviation is only 0.1 Å[48]. The cavity is slightly larger than one would expect if the structure did not relax at all. The largest differences between L99A and WT* are a movement of Val87 0.4 Å away from, and Tyr88 0.5 Å towards, the mutation site[48].

5.1.2 Known modes of structural change in T4 lysozymes

Substantial NMR evidence[34, 35, 136] and simulation suggests that T4 lysozyme has both very rigid and very dynamic parts. Simulation by molecular dynamics[15] has indicated that the C-terminal and the N-terminal domain, as well as the linker helix are each individually rigid, while the three components move relative to each other by as much as 8 Å. The active site is between the C- and N- terminal domains; presumably these fluctuations represent comformational flexibility related to the mechanism of the protein.

A different issue altogether is the fluctuation of side chains. While it is known that the L99A cavity is accessible to noble gases[39] and also to benzene and benzene derivatives[50], there is no solvent accessible path into the cavity in the L99A structure determined by x-ray crystallography. Clearly there must be fluctuations which make the cavity transiently accessible to the outside solvent. Amide exchange studies mentioned in Chapter 1[34] suggest that the hydrophobic core of the WT* mutant is solvent accessible even at low pressures. NMR studies[136] indicate that ligand binding is very rapid.

While the amide exchange studies indicate a wide variety of pathways into the molecule, other NMR studies by Mulder *et al.*[35] have specifically probed for states

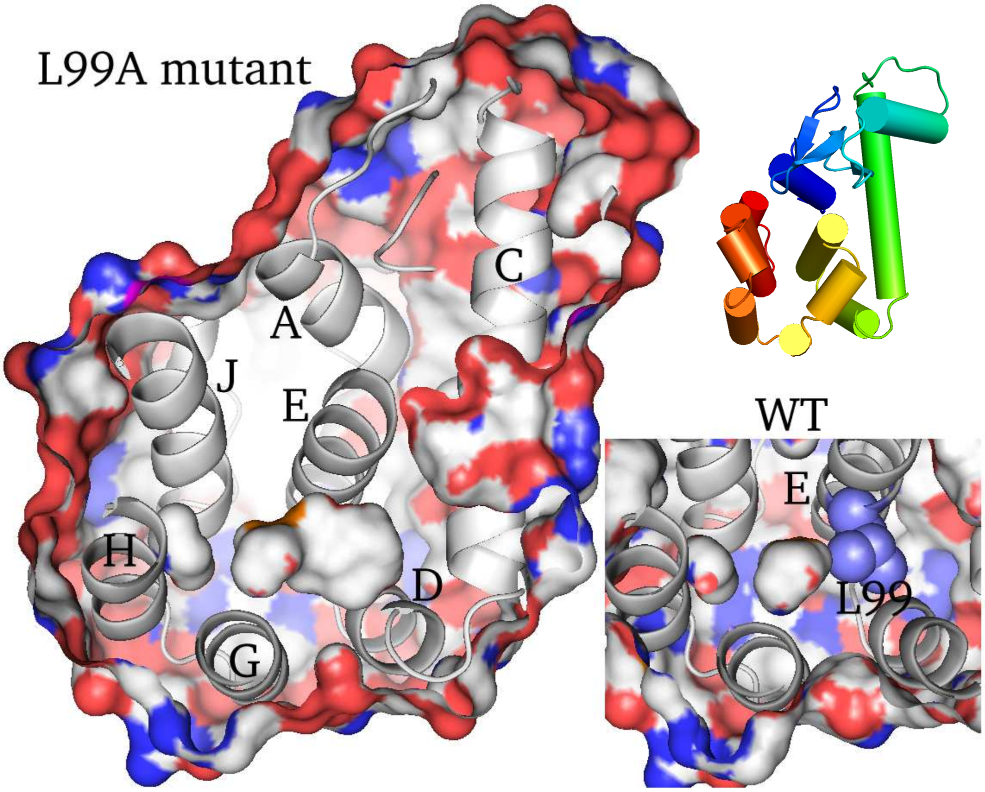

Figure 5.1: Primary cavity of psuedo wild-type and L99A T4 lysozymes. The cavity of L99A (main image) is an extension of a cavity present in the wild type lysozyme (inset). Letters indicate helix designations. Spheres in the inset represent the WT* Leu99 side chain, which is truncated to Alanine in the L99A mutant. This view is through the F helix (not shown) into the cavities; the E helix is shown center, just above the cavity. The cartoon insert shows the orientation of the molecule. See Figure 5.3 for a complete description.

in which there is access to the L99A cavity. They observe a marginally populated excited state ($\sim 3\%$ at room temperature, $\sim 2\,\text{kcal/mol}$ above the ground state) in which the L99A cavity is accessible to surrounding water. The transition involves the C-terminal end of helix E, helices F and I and loops connecting these helices to the rest of the protein. (Figure 5.1 is a view through the F helix (not shown) into the cavity; helix I is a one turn helix between H and J, which are left and upper left of the cavity in the figure. The mutation L99A is squarely in the middle of helix E.)

5.2 Methods of structure comparison

Beyond the basic, somewhat subjective features of structure, we would like to be able to measure specific, quantitative changes. This requires some quantitative notion of what the structure is. The α-carbon backbone positions are often used and are thought to be most robust, but this lacks detail. Feeling that there is no particular reason to choose the α-carbons over, say, the amino nitrogens, I've chosen to make comparisons of the backbone based on the four backbone atoms: C_α, C, N, and O. Side chain comparisons may also provide useful information, but side chains are generally more disordered and therefore subject to more uncertainty. The orientation of helices can also be calculated and compared to observe rotations. Whatever we do, we must be careful to follow the recommendations of Chapter 4, and care must be taken to properly orient the molecules for comparison.

We will also compare the total molecular volume and the volume of cavities in the protein, in particular the void left by the L99A mutation. Calculating and visualizing these volumes is not trivial, but established methods exist.

5.2.1 Orienting molecules for comparison

Any set of coordinates $\{\boldsymbol{r}_i\}$ can be linearly transformed by rotation, translation, scaling and shearing. In general these operations do not commute. The most general unitary transformation consists of a rotation R and a translation $\boldsymbol{r}_0$:

$$\boldsymbol{r}' = \mathsf{R}\boldsymbol{r} + \boldsymbol{r}_0. \tag{5.1}$$

Where R is a 3 by 3 matrix. For the transformation to preserve distances (as for a pure rotation), the matrix R must be unitary. Minimizing the difference between two structures a, b with the residual

$$\sum_i \frac{1}{w_i}(\boldsymbol{r}'_{a,i} - \boldsymbol{r}'_{b,i})^2, \tag{5.2}$$

where the sum is over all atoms i, we determine an optimal alignment of the two structures. The weights w_i must be chosen appropriately. I have generally set them to one, since I cannot reliably say that the uncertainty in one atomic position is greater or less than another specific atomic position. Now the two structures may be compared in terms of their coordinates. Deviations may be calculated and so forth.

Implementation

One goal of my work was to establish more empirical positional uncertainties than have been quoted in the past. This ultimately requires the simultaneous alignment of more than two structures. The following describes the program I wrote for this purpose.

Equation 5.2 was implemented in Perl (www.perl.org). Perl was chosen for its ability to efficiently parse text files, useful for handling Protein Data Bank (www.rcsb.org) structure files. The code is listed in the appendices.

The process is shown schematically in Figure 5.2. First the pdb files are read in and “vetted”. This step involves parsing the PDB file for coordinates, masses, temperature factors, and other important information. Then the program discards atoms which are not to be compared, keeping only a user-selected subset of the structure. The code is very flexible in this regard. Next each molecule’s center of weight (not necessarily mass) is moved to the origin. A transformation (general linear or pure rotation) is calculated and applied to each whole molecule’s coordinates. New coordinates are written in PDB format, and a log file records the transformations made, as well as a root-mean-square deviation for all atoms compared. Optionally, a distance difference matrix can be calculated between structures a, b according to

$$\mathsf{D}_{ij} = |\boldsymbol{r}'_{a,i} - \boldsymbol{r}'_{a,j}| - |\boldsymbol{r}'_{b,i} - \boldsymbol{r}'_{b,j}|. \tag{5.3}$$

The distance difference map has the advantage of being independent of the transformation. However, such maps are not particularly easy to interpret. I have not used them here.

There are two reasons to align molecules in this fashion. The most obvious is to examine differences between molecules at different pressures. We may also wish to align the models derived at one pressure in order to estimate uncertainty in atomic positions. In this case, the models derived in step (d) of Figure 5.2 will be averaged, and the rms deviations from that structure determined. The averaging could be of any sort: we might average positions of atoms, side chain dihedral angles, or even derived quantities such as helical axes (discussed below). Uncertainties can thus be estimated for any parameter.

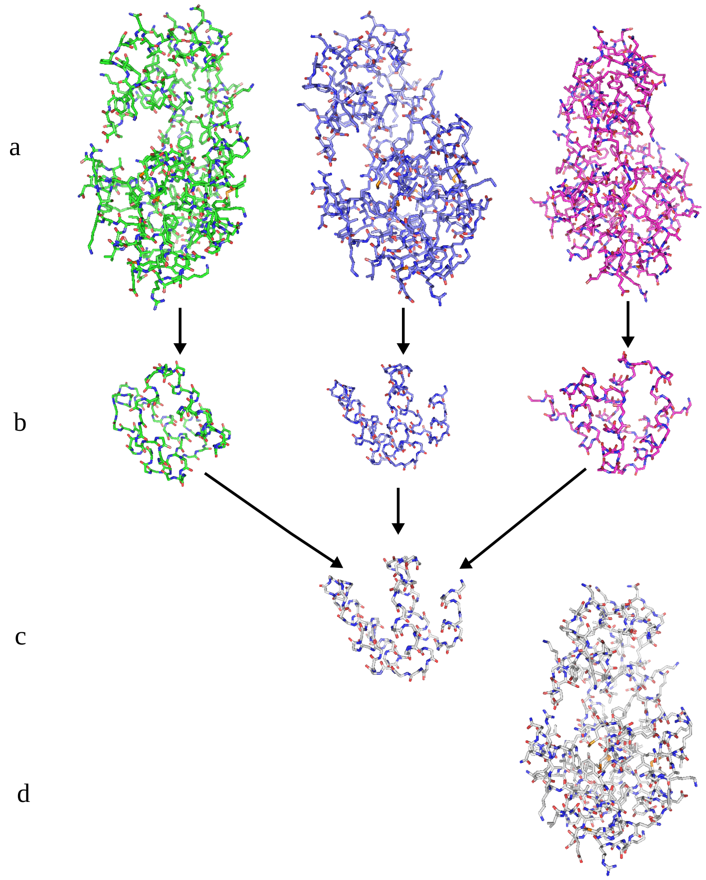

Figure 5.2: Alignment and structural averaging. (a) Three structures from refinement. (b) Part of the molecule is selected for alignment. (c) The selections are aligned. (d) The transformations generated in (c) are applied to the molecules in (a). The structures may now be averaged or compared.

Selective structure comparison

To observe changes in a molecule, one must ask the right questions. For instance, if we expect helices to be extremely rigid internally, but to rearrange relative to each other, we may wish to highlight this effect by aligning two structures at different pressures using only one helix. A second helix will then be shown is its orientation relative to the first and our expectation can be tested. Similarly, it may be interesting to compare the RMS deviations of the backbone to that when side chains are included.

I have tried several such selective alignments. The numbers reported here are based either on whole molecule alignments, or alignments of the C-terminal domain (residues 82 through 162). In all cases, only the backbone atoms were used unless otherwise specified.

Multiple structure comparison

Equation 5.2 can be minimized analytically when it is used for only two structures. There is no simple analytic solution for three or more structures, although the extra data included surely makes the data more robust. To make such comparisons I have used a simplex fitting algorithm (see *Numerical Recipes*[125] for a discussion of the method) from the Perl Data Language[1]. Also, one must decide which residual should be used and reported. We may choose to calculate a pairwise rms for each pair of molecules and then average over all pairs; alternatively, we may choose to average atomic mean square deviations over molecules first, then sum over all atoms to be compared. (The difference arises because the residual is of the root-

[1] PDL::Opt::Simplex. For this and other useful Perl modules, visit the Comprehensive Perl Archive Network at www.cpan.org.

mean-square form. As it happens the two forms do not give substantially different results, and the only reason I chose the latter form was to gain a sense of individual atomic positional uncertainties.)

When there are more than two data sets for a given mutant at one pressure, this procedure is used to align the molecules before real space averaging. Averaged structures at different pressures can be easily compared.

5.2.2 Orientation of helices

Based on the method (and code) of Kumar and Bansal[137], I have written a Perl script to calculate the positions and orientations of helices, and output a visual indicator of this orientation for protein graphics programs. Briefly, the method is as follows: select the first four C_α atoms at the start of a helix. Then calculate the difference vectors between the first and third, and second and fourth C_α positions. The cross product of these two difference vectors defines the local helical orientation. The "box" of four atoms is then shifted one residue along the chain, and the procedure is repeated until we can go no further along the helix. This an effective tool for quickly assessing structural changes in proteins.

5.2.3 Volume calculation

Molecular and cavity volume calculations are in principle a good indicator of exactly how a molecule is compressing. There are a number of ways of calculating volumes, all of which are subject to some caveats. The most pervasive problem is that small changes in atomic positions can lead to large changes in cavity volume. Molecular volumes in general are difficult to define, and so should always treated with skepticism.

All current methods use the concept of a probe sphere which is rolled along the locus of spheres of atoms in the molecule[2]. The locus of points traced out by the center of the probe sphere is called the *solvent accessible surface.* The volume from which any part of the probe is excluded is called the *solvent excluded volume* and is bounded by the *molecular surface.* The radius of the probe is taken to be 1.2 Å in this thesis, and the atomic radii are chosen to emulate Michael Connolly's original program *MSP*[138], which is no longer available. These are approximately the van der Waal's radii of water and the individual atoms, respectively. This choice is consistent with the previous work on T4 lysozyme[48].

The actual method used to calculate that volume varies. The "rolling probe" method, though conceptually simple, is difficult to implement and is not generally used. Here, we will discuss only two implementations, which illustrate *grid based techniques* and more sophisticated *analytical techniques.*

Grid based techniques

A grid based approach is implemented in the program *VOIDOO*[139]. The basic idea is simple. One creates a cubic lattice of some spacing (generally about 0.5 Å), and asks whether each lattice point is within a certain cutoff distance (the probe radius plus the atomic radius) of each atom in the molecule. This implementation uses a "flood-fill" algorithm. Each grid point is given a value zero. Then, if a grid point is within the cutoff distance from any protein atom, it is reassigned the value one. The faces of the lattice ("the outside world") are reset to zero. We now take any point on the faces of the lattice whose value is zero and reset it to two. The

[2] Most software ignores water molecules.

process is repeated for neighbors of this point whose values are also zero, and so on. Since we begin from the outside, by the time the iteration has finished, any point whose value is still zero must be inside a cavity.

An obvious problem is that the grid spacing is quite important. A more insidious problem is that the orientation of the cavity with respect to the grid can affect its apparent volume. Nonetheless VOIDOO still yields reasonable and consistent results. Since it deals explicitly in real coordinates of the cavity wall, VOIDOO is especially useful in visualizing the shape and location of each cavity.

Analytic techniques

Edelsbrunner has developed an interesting and aesthetically pleasing method of determining the volume of protein voids based on an analytically calculated *alpha shape*[140, 141]. The full description is complicated, but again the basic idea is simple. We begin with a point at the center of each atom in our structure. Each point swells into a growing sphere until it intersects the spheres representing its neighbors. When two spheres intersect, their centers define a line segment of the alpha shape; when three mutally intersect, it defines a triangle; four mutally intersecting spheres define a tetragonal pyramid. Each sphere swells until its radius is the radius of the atom plus the radius of the "probe sphere" (*i. e.* 1.2 Å). Those regions of space not inside tetragonal pyramids of the alpha shape are voids. The volume of the real void is the volume of the polygonal void minus a term representing the parts of atoms inside the polygonal representation.

This algorithm resembles the older Voronoi decomposition method, but overcomes certain technical difficulties, especially with the molecular envelope. The chief advantage of the alpha shape method is that its results are independent of molecular orientation or any grid. It only depends on the probe sphere size and the atomic radii.

Michel Sanner's *MSMS* program[142] is mathematically similar to the alpha shapes method, although the program is implemented differently.

5.2.4 Detection limits

Paul Urayama[9] has pointed out that the expected structural changes in proteins under pressure will fall in two categories, bulk properties (such as compression, volume, *et cetera*) and motions of specific atoms or secondary structure elements. We have already seen that the former are considerably more robust, but we nontheless expect any changes will be small. The crystallographically determined volume compressibility of hen egg white lysozyme was $\approx 5 \times 10^{-3}\,\text{kbar}^{-1}$[12], while that for myoglobin was about $9 \times 10^{-3}\,\text{kbar}^{-1}$[9]. For a spherical protein this corresponds to a change in radius of about half a percent for a pressure increase of 1 kbar. For T4 lysozyme, having a volume of about 20,000 Å^3, that corresponds to slightly less than 0.1 Å.

Pressure changes are not likely to be isotropically distributed in such an inherently anisotropic medium. The molecule is made up of secondary structure elements held together with highly oriented hydrogen bonds and steric constraints that depend strongly on the orientation of the main chain and neighboring side chains.

In this work, an effort was made to collect sufficient data on which to base statistical conclusions about uncertainty. In the case of the L99A mutant, there are three data sets at each of three pressures, plus one more at 1.5 kbar. I have somewhat less data for the WT* structure, and so any conclusions about it are less robust. Nonetheless some variance can be estimated there as well. To assert that there is a meaningful structural change, I require that the change is at least at the level of uncertainty in the parameter, and to say anything strongly, that it be considerably larger than the uncertainty. In the former case, we are greatly aided by visual examination of the structure, where it is possible to unambiguously see whether some motion is in fact concerted, or whether it is an artifact.

5.2.5 A note on isotropic scaling

I have chosen not to scale molecules by an overall factor. The reason is simple: it explicitly violates the stereochemical restraints used in refinement, and physically misrepresents any realistic behavior of the molecule. As was discussed in Chapter 1, proteins should compress anisotropically, due to the varying effects of pressure on the different interactions stabilizing the protein in its folded state. In particular we expect covalent bonds to be nearly incompressible over the range of pressure used here, and so isotropic scaling is inappropriate. Moreover, we are attempting to highlight, not dismiss, differences. We hope to attribute changes in volume to structural rearrangement, not changes in bond lengths. I have had to give up this goal in the sense that averaging structures in real space is not inherently consistent with the restraints either. However *average* bond lengths and angles should remain the same after real-space averaging, only the distribution will widen. Moreover this is the only way in which to simultaneously use all of the data in refinement and construct useable estimates of the variance in atomic positions without appeal to some heuristic formula. No such justification can be made for isotropic scaling.

5.3 Observed changes in the structure of T4 lysozyme at high pressure

Both the wild type and L99A mutant lysozymes change almost not at all when subjected to pressure. That the compact C and N terminal domains change so little in the WT* lysozyme is not particularly surprising. While the N and C terminal domains do displace relative to each other, they do not grow closer or farther apart, which we might have expected from molecular dynamics simulation[15]. (The experimentally determined distance between the domain centers of mass changes less than 0.06 Å at 2 kbar.) As we'll see the two molecules are virtually identical at high pressure, but for the few missing atoms in the L99A lysozyme. The details are described below. Interpretation of the results will be left for the final chapter.

5.3.1 Overall changes in unit cell and molecular volume

Crystals of T4 lysozymes grow in spacegroup $P3_221$, with the (unique) c axis being roughly 1.5 times longer than the a and b axes. The unit cell compresses essentially linearly with pressure, with the c-axis being softer than the a or b axes. The c-axis decreases 1.0% from about 96.5 to about 95.5 Å over 2 kbar, while the a- and b-axes decrease 0.5% from about 60.9 to 60.6 Å. At any given pressure, the unit cell parameters cluster tightly. The L99A unit cell is slightly larger than the WT* unit cell, and this difference is maintained throughout the pressure range studied. As pressure increases to 2 kbar, the unit cell volume decreases roughly 2 percent for both the psuedo wild-type and the L99A cavity mutants. Urayama[9] quotes similar changes in unit cell volume for sperm whale myoglobin at 2 kbar. Kundrot and Richards[12] reported a 1.1% decrease in unit cell volume for Hen Egg-White

Lysosyme at 1 kbar.

The L99A molecular surface volume (Table 5.1) decreases from 21,090 Å^3 to 20,890 Å^3 over 2 kbar, a change of slightly less than 1 percent. The WT* volume decreases from 21,070 Å^3 to 20,860 Å^3 at 2 kbar, again slightly less than 1 percent. This is our first indication that the molecule will change very little as we increase pressure.

5.3.2 Changes in cavity volume

Table 5.1 lists the cavity volumes for the nine L99A models using the AlphaShapes package described above. The cavity volumes do not cluster as well as the cell parameters, but this is to be expected given that volume errors are additive (volume is not an average quantity). The cavity volume decreases on average as pressure rises from 1 bar to 2 kbar, though by barely more than 3 percent. This is interesting, since the molecular volume changes by only 1 percent, and the unit cell volume decreases by only 2 percent. I am reluctant to put much faith in this difference, because of the extreme difficulty in assigning any meaningful uncertainty to these volumes. Probably the unit cell volume is robust, and the molecular envelope volume is also probably fairly accurate, but the uncertainty in the cavity volume is clearly on the order of the changes we observe. It is clear that there is some compression, and we must leave it at that.

Table 5.1: Cavity and molecular envelope volumes for the void created in the L99A mutant lysozyme, calculated using the AlphaShapes package[140, 141]. All volumes in Å^3.

dataset	nominal P (kbar)	L99A V_{cav}	Average V_{cav}	V_{mol}	Average V_{mol}
mt0k1	0	183.6	185.3	21118	21088
mt0ka		188.0		21092	
mt0kb		184.2		21052	
mt1k6	1	178.1	181.4	20971	21000
mt1k7		184.9		21006	
mt1k9		181.3		21024	
mt2k1	2	174.8	179.5	20854	20894
mt2k3		183.6		20922	
mt2k8		180.0		20905	

5.3.3 Radii of gyration

The radii of gyration are defined in terms of the masses m_i and coordinates of atoms in the molecule:

$$R_g = \left| \frac{\sum_i \sum_{\alpha=1,2,3} m_i r_{i,\alpha}^2}{\sum_i m_i} \right|^{1/2} \tag{5.4}$$

Since proteins are not in fact spherical, we would also like to know their moments of inertia. The inertia tensor is

$$\mathsf{I}_{\alpha\beta} = \sum_i m_i \left[-r_{i,\alpha} r_{i,\beta} + \delta_{\alpha\beta} \sum_{\alpha'} r_{i,\alpha'}^2 \right]. \tag{5.5}$$

It has eigenvalues I_γ where the γ are the principal axes of the inertia. The radius of gyration R_γ taken along axis γ is related to the I_γ by

$$R_\gamma^2 = M_{tot}^{-1} I_\gamma, \tag{5.6}$$

where M_{tot} is the total mass of the molecule, and $R_g^2 = (1/2) \sum_\gamma R_\gamma^2$.

Table 5.2 lists the radii of gyration for both WT* and L99A T4 Lysozymes, calculated only from the average structures at each pressure. (The differences between structures are, as we'll see below, so small that the errors in these averaged parameters should be quite small.) The principal axes are shown in Figure 5.3. The largest changes are perpendicular to the axes between the C- and N-terminal domain centers. This could indicate relative motion of the two domains.

Table 5.2: Radii of gyration in Å for L99A and WT*. Pressures are nominal and moments calculated based on all-atom aligned and averaged structures at each pressure.

Pressure	R_g	R_1	R_2	R_3
L99A				
1 bar	16.50	15.95	13.53	10.34
1 kbar	16.46	15.93	13.49	10.29
2 kbar	16.43	15.91	13.45	10.28
WT*				
1 bar	16.46	15.92	13.50	10.31
1 kbar	16.44	15.91	13.48	10.28
2 kbar	16.40	15.89	13.43	10.26

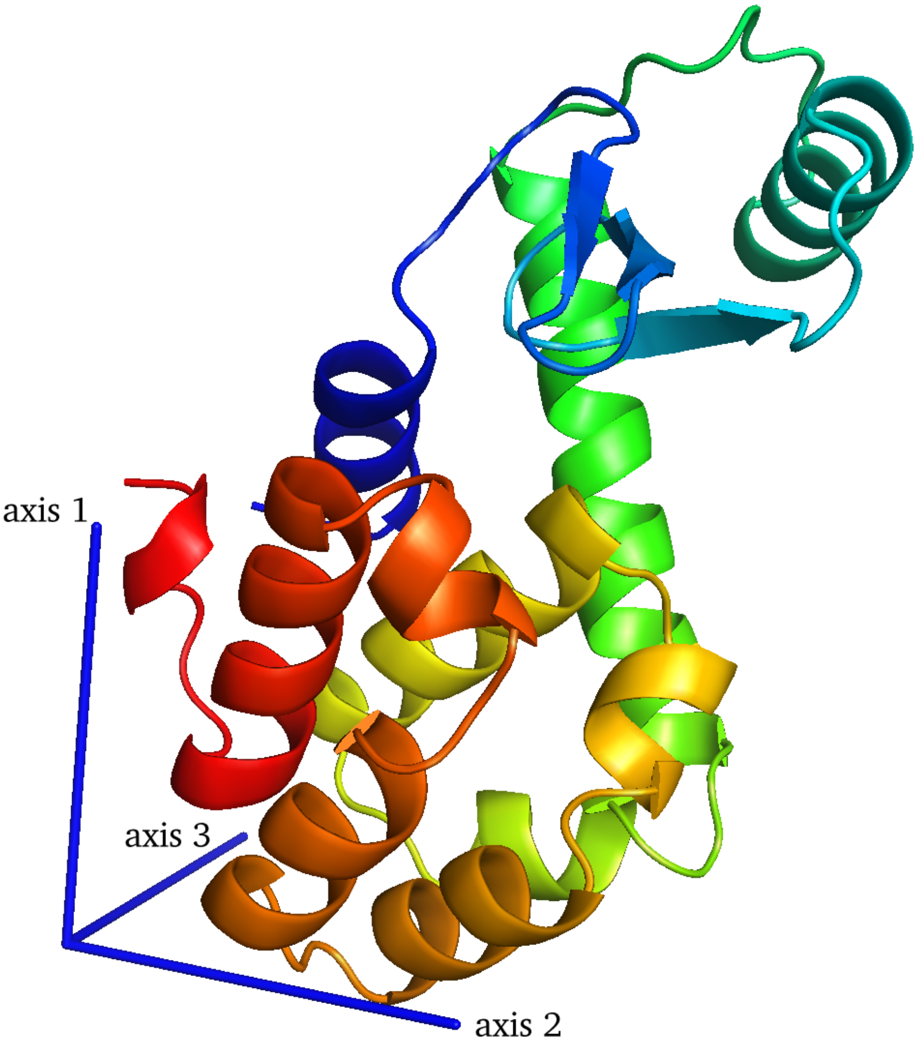

Figure 5.3: Principal axes of the inertia tensor. The first principal axis points roughly from the center of the C-terminal domain to the center of the N-terminal domain. Colors indicate the various structural elements. Helix A is in dark blue, helix C is green, helix E is yellow, and helix J (the final long helix) is red. These colors will be used for orientation throughout this chapter.

5.3.4 Differences in temperature factors

One comparison not discussed above is the difference, between high and low pressure, of the temperature factors $B = 8\pi < x^2 >$. "Colder" regions (of smaller B) have smaller fluctuations about their average positions. We must be careful not to associate this parameter with a measure of positional uncertainty. Rather is is a measure of each atom's freedom to move about in the molecule.

The cavity region of the L99A mutant is curiously the "coldest" part of the molecule (see Figure 5.4), both at ambient and high pressure. The effect is even more startling for the side chains (which for clarity are not shown) where residues actually bordering the cavity are apparently the most constrained in the entire molecule. This may be further evidence that fluctuations of side chains are primarily collective as discussed by Mulder *et al.*[95]. If this were the case we might expect that one mutation would have little effect on the conformational flexibility of side chains. The main effect of pressure on temperature factors is a decrease of the temperature factors of the molecule by about 1-5 $Å^2$ as the pressure is raised from ambient to 2 kbar.

Plotting the main-chain residue averaged B-factors against residue number (Figures 5.5 and 5.6) demonstrates the same overall trend with pressure. We also see excellent reproducibility of the general features of the B-factors, increasing our confidence in the results. One point of possible concern is that the decrease of B-factors with temperature seems to "stall-out" at 1 kbar in the L99A mutant, while they decrease continuously in the WT* protein. It is not clear why this should be. A number of factors, including crystal mosaicity and disorder, and uncertainties from data collection, contribute to the "crystallographic B-factors".

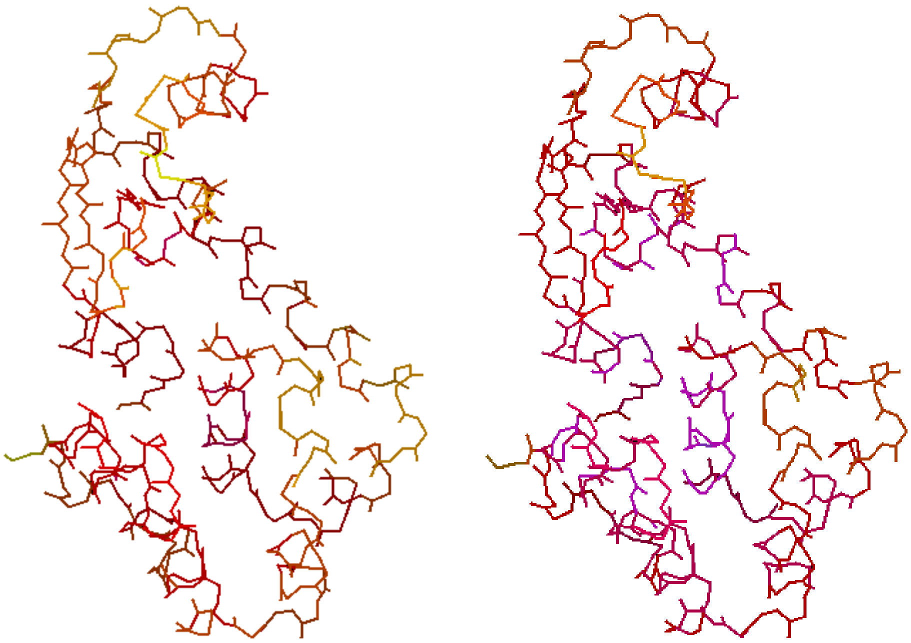

Figure 5.4: Main chain B factors of the L99A mutant. Left, ambient pressure, right, 2 kbar. Yellow corresponds to values of 30 Å^2 and above, red to values to values near 20 Å^2, and purple to values of 10 Å^2 and below.

Thus they include more than simple Debye-Waller factors, and are presumably subject to somewhat more error. Recent evidence suggests that pressure tends to improve the mosaicity of protein crystals[24], though how much of our observed trends can be explained this way is unclear. Moreover, while it is possible to construct an empirical error for the B-factors by comparing many different datasets, that uncertainty is large ($\sim$5 Å^2 or more) and varies a great deal from crystal to crystal and between datasets of different pressures. I cannot at this point assign any meaningful uncertainty to these parameters. In the absence of substantially more data, I am reluctant to say any more than this.

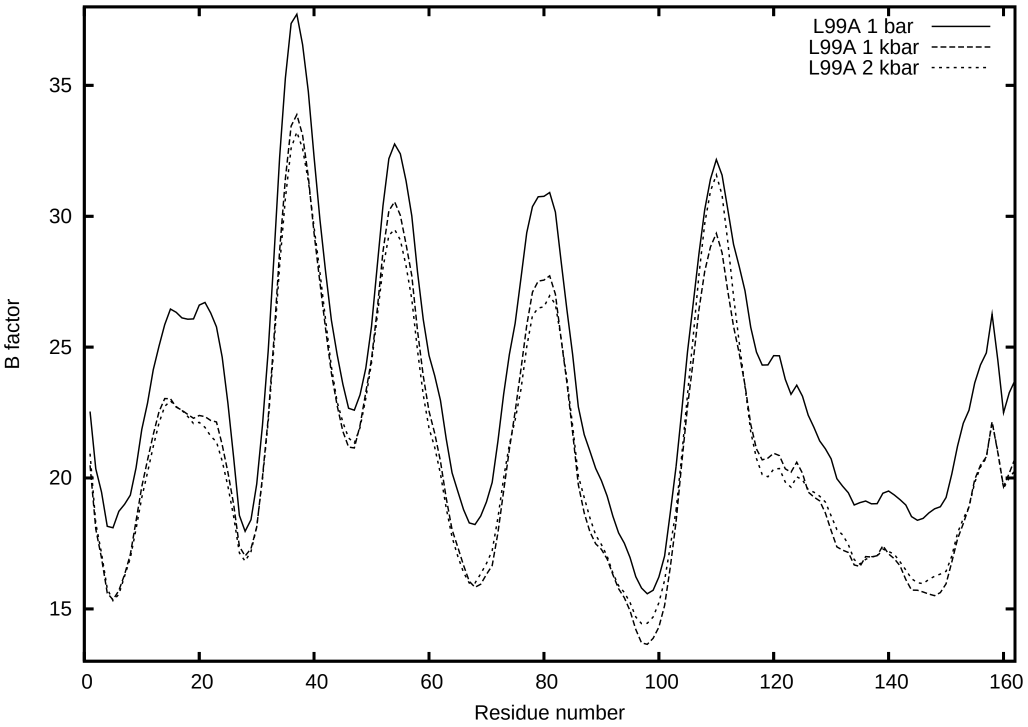

Figure 5.5: Main chain B factors of the L99A mutant at three pressures. Values are averaged over datasets at each pressure. *B*-factors in Å^2. Uncertainties have not been determined for the *B*-factors, see text.

5.3.5 Empirical uncertainties

Table 5.3 shows the root mean square displacement (rmsd) of atoms for all structures at one given pressure, for alignments based on several different subsets of atoms. (Since there is only one WT* dataset at ambient pressure, it is omitted from the table). Generally speaking the values are very low, and bracket the standard uncertainties derived from maximum likelihood refinement (see the previous chapter). The differences between the rmsd values for the main chain (MC) and the whole molecule (MC+SC) decrease as pressure is increased. Based on the table, we can generally trust that any pressure-dependent aggregate main chain displacement larger than 0.1 Å is real. By examining differences between several published

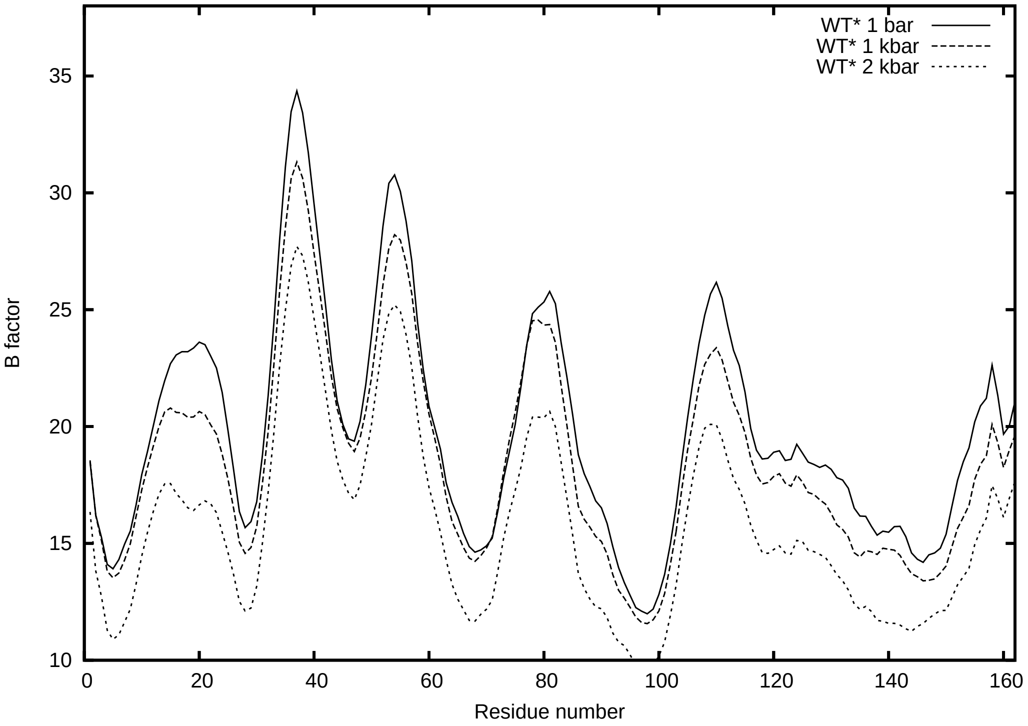

Figure 5.6: Main chain B factors of the WT* protein at three pressures. Values are averaged over datasets at each pressure. B-factors in Å^2. Uncertainties have not been determined for the B-factors, see text.

myoglobin structures, Urayama[9] also concluded that 0.1 Å displacements were detectable.

We may still be suspect of individual atomic displacements, as has been discussed. An examination of these uncertainties, derived during the alignment and averaging procedures, shows that there are not regions of particularly large or small uncertainty. For the most part, the uncertainties are uniform. There are small peaks in uncertainty near residues 40, 80 and 110. The last of these corresponds to a region of poor electron density, as discussed at the end of the last chapter.

Table 5.3: Root mean square deviations, in Å, between models at identical pressures. Alignments based on either the peptide chain (MC) or all atoms (MC+SC), either on all residues (no label) or the C-terminal domain (CT) from residue 82 to residue 162. Pressures listed are nominal; exact values and other pertinent information about the data sets can be found in Table 3.1.

P (kbar)	MC	MC+SC	MC CT	MC+SC CT	Models
0	0.088	0.185	0.096	0.195	mt0k1, mt0ka, mt0kb
1	0.088	0.210	0.092	0.239	mt1k6, mt1k7, mt1k9
2	0.066	0.152	0.065	0.135	mt2k1, mt2k3, mt2k8
1	0.068	0.160	0.079	0.180	wt1k6, wt1k7, wt1ka
2	0.045	0.134	0.044	0.099	wt2k1, wt2k2

5.3.6 Observed displacements at pressure

After alignment, structures at a common pressure are averaged, and these structures may be compared for differences due to pressure.

Table 5.4 lists the rms values calculated for alignment of the averaged main chain structures based on the C-terminal domain only, the N-terminal domain only, or for the whole molecule. The C- and N- terminal domains yield values which, as we would expect, are in all cases are smaller than the the whole molecule. The rmsd values to the ambient pressure reference structure do increase as a function of pressure. Since the rmsd values over each domain are quite similar, and the whole molecule values somewhat larger in most cases, I interpret this as a sign of domain realignment as pressure increases.

Figure 5.7 on page 137 shows the magnitude of displacements (in Å) between the averaged L99A or WT* models at 2 kbar and the corresponding models at

Table 5.4: RMS displacements in Å between low and high pressure structures. For each pair of pressures, the left number is the rms for the C-terminal domain only, the right number for the N-terminal domain, and the number in parenthesis the rms is for the whole molecule.

L99A	1 kbar	2 kbar
0 kbar	0.103/0.105 (0.119)	0.133/0.137 (0.171)
1 kbar		0.069/0.075 (0.084)

WT*	1 kbar	2 kbar
0 kbar	0.079/0.079 (0.093)	0.124/0.134 (0.164)
1 kbar		0.069/0.079 (0.089)

ambient pressure when the molecules are aligned on the C-terminal domain. Figure 5.8 shows the same displacements when the molecules are aligned using the main chain atoms of the N-terminal domain (residues 13-58.) Figure 5.9 shows the displacements for L99A at three different pressures. Each of these is averaged over a 5-residue wide window.

A number of potentially interesting features are visible. Most noticeable are the large displacements of the N-terminal residues when the models are aligned using the C-terminal main chain (Figure 5.7). This again indicates a significant domain realignment, the exact nature of which is not yet clear. The dominant feature in all three plots is a large displacement of the C-terminal end of helix C (residues 70-80 or so).

Many smaller features are visible in both the C and N terminal domains. We are especially interested in changes around the cavity of the L99A mutant, buried in the C-terminal domain. Here we can see two, possibly three interesting if small

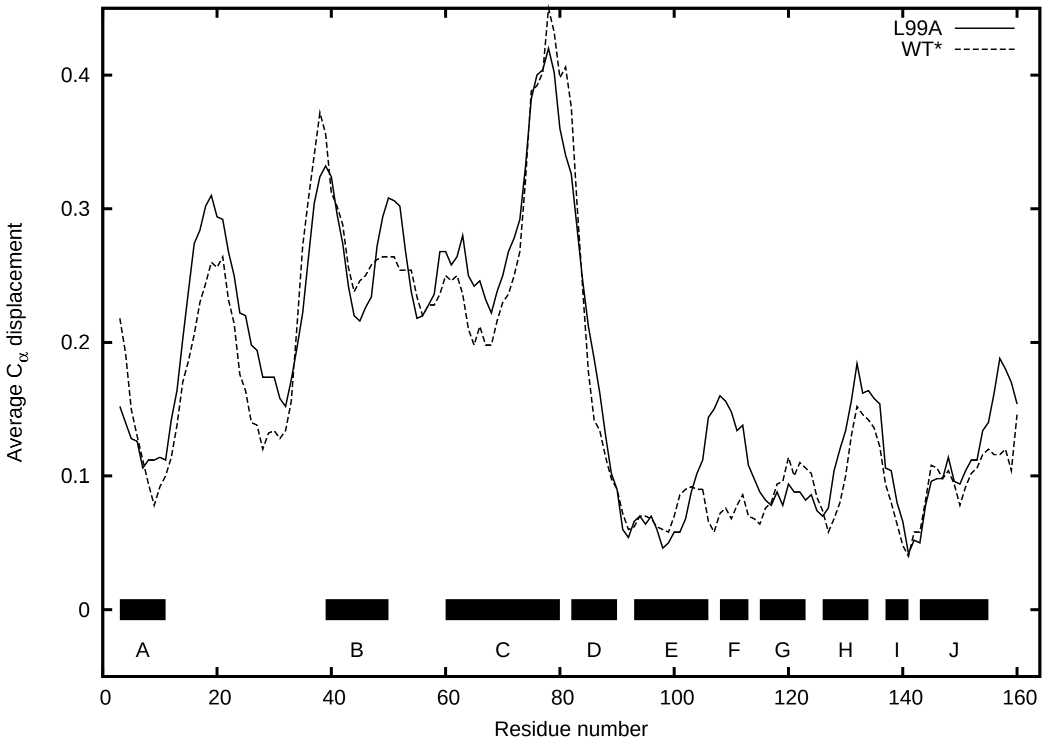

Figure 5.7: Magnitude of C_α displacements in Å at 2 kbar. Here the molecules have been aligned using only the main chain atoms of the C-terminal domain. Bars and letters indicate positions of the ten α-helices. Uncertainties are discussed in Section 5.3.5.

features, one around residues 130-140 (roughly helices H and I), which is conserved in both WT* and L99A, and one centered around residue 107 (between helix E and F) and visible primarily in L99A. A third feature between these two is smaller but perhaps interesting. The C-terminus is also displaced, but this region is poorly defined in the electron density maps, and so I do not put much stock in this observation.

Features in the N-terminal domain become much more noisy when the models are aligned on that domain. This may be a further indication that the changes in the N-terminal domain are best thought of as relative to the C-terminal domain.

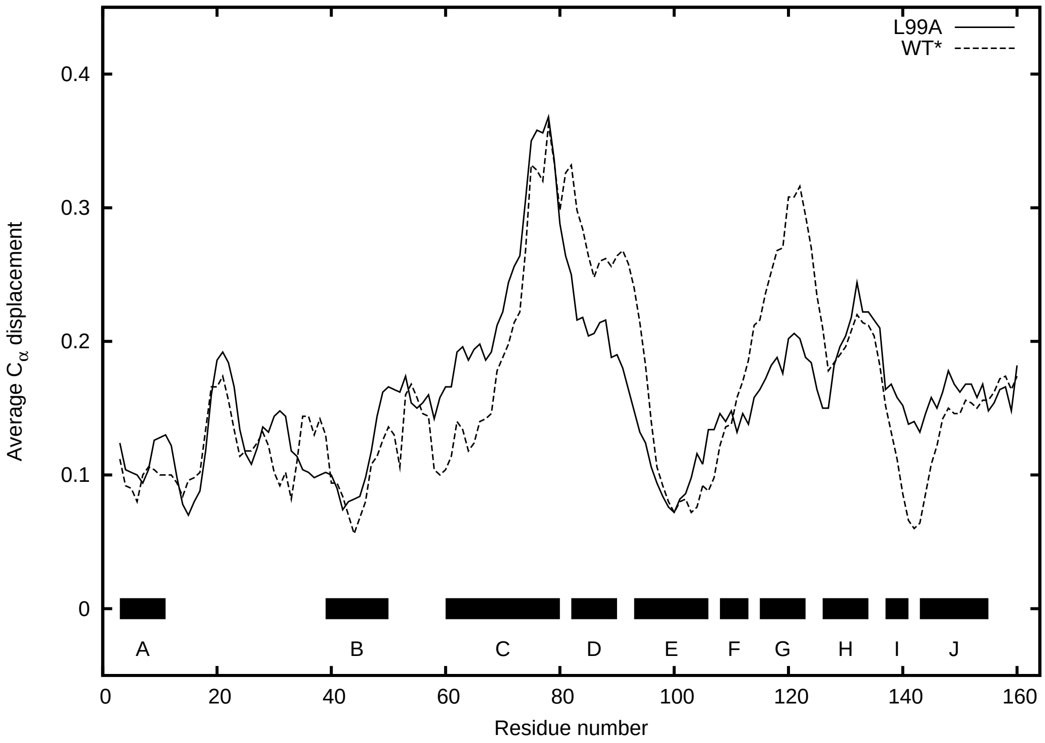

Figure 5.8: Magnitude of C_α displacements in Å at 2 kbar. Here the molecules have been aligned using only the main chain atoms of the N-terminal domain. Bars and letters indicate positions of the ten α-helices. Uncertainties are discussed in Section 5.3.5.

In contrast, displacements of the C-terminal domain and C-helix are quite robust.

In contrast to the B factors, the atomic displacements are essentially monotonic with pressure. As is visible in Figure 5.9, much of the displacement visible at 2 kbar has occured at lower pressures, but displacements continue to increase with applied pressure. Parts of the molecule which show no change above 1 kbar are also locations where there is very little change at all. From this figure, we can identify several soft regions. Again, the C-terminal end of helix C appears to be the softest part of the protein, followed by parts of the N-terminal domain. Displacements in the C-terminal domain decrease less as a function of pressure than those in the

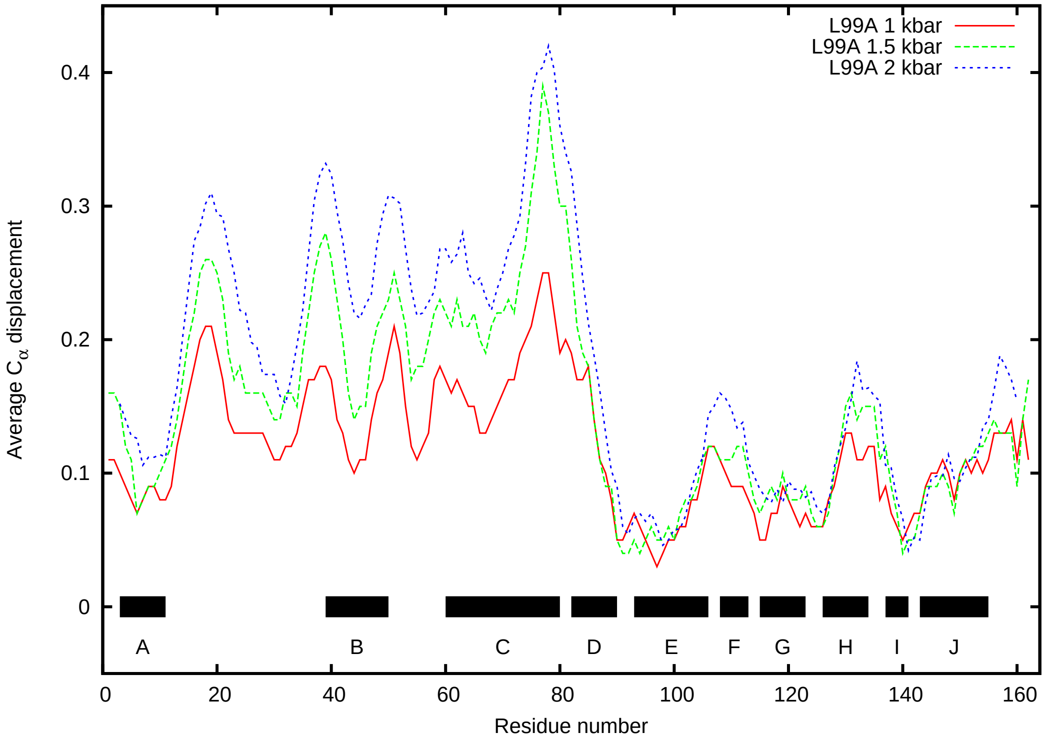

Figure 5.9: Magnitude of C_α displacements in Å for L99A at three pressures. Alignment was based on the main chain atoms of the C-terminal domain. Bars and letters indicate positions of the ten α-helices. Uncertainties are discussed in Section 5.3.5.

N-terminal domain.

5.3.7 Domain realignment

The largest concerted displacement of the L99A or WT* molecules is that of the N-terminal domain relative to the C-terminal domain. From Figure 5.7 we can see that there are at least three parts to this displacement, a peak near residue 20, another peak near residue 37, and smaller features from residue 43 to 60, where the domain ends. In this case it turns out that the minimum displacements are the most interesting.

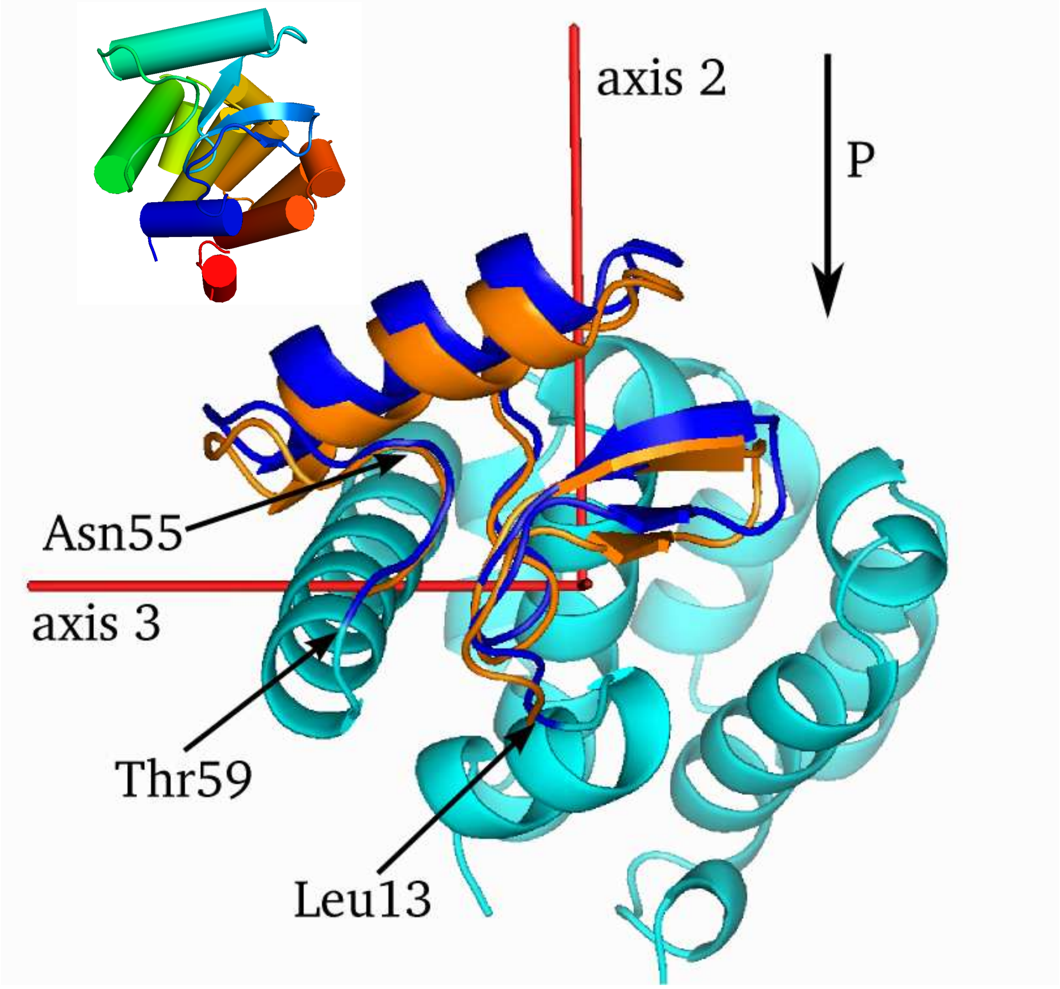

Figure 5.10: A cartoon view of N-terminal domain displacements in the L99A mutant along the first principal axis of inertia. The N-terminal domain is shown in dark blue at ambient pressure. The displacement at 2 kbar (orange) is shown magnified 5 times. The remainder of the ambient pressure structure is shown in light blue. The arrow labelled P indicates the direction of pressure-induced displacement of the N-terminal domain. The WT* N-terminal domain has essentially the same response to pressure. Inset shows relative orientation of the molecule. Refer to Figure 5.3 for colors.

Domain movement as a whole

Figure 5.10 shows that the N-terminal domain primarily translates relative to the C-terminal domain by about 0.25 Å over 2 kbar. This motion is primarily along the second principal axis of the inertia tensor (shown in the figure; the cosine between these two vectors is 0.96), and is more or less perpendicular to the line between the C- and N-terminal domain centers. The largest concerted displacement lays in the B-helix, which is displaced about 0.3 Å on average at 2 kbar. The helix moves as a unit, and does not appear to deform substantially.

Key points for the displacement of this domain are in the loop after helix A, particularly Leu13 and Arg14, and residues 55-59, which are surprisingly rigid. Probably there is little holding most of the domain in place, while helix A interacts strongly with helix E (making two hydrogen bonds and a number of close interatomic contacts) and residues 49-59 makes numerous contacts with the C-helix, including two hydrogen bonds between Glu62 and Arg52. This end of the C helix (see below) is coupled to the N-terminal domain displacements, and is probably largely responsible for resisting those displacements as pressure increases.

A curious minimum

Near residue 30 in both WT* and L99A there is a strong minimum in displacement (Figure 5.7. The most notable interaction near this feature is a hydrogen bond between Asp70 and His30. There are no other obvious contacts pinning this region together, although at higher pressure the side chain of Leu66 blocks the motion of His30 in the direction in which the rest of the domain moves. Leu66 is also implicated in the displacement structure of helix C.

5.3.8 Changes in the C-helix

Figure 5.11 illustrates the rather complex changes in the C-helix. From the displacement amplitudes seen above, we expect the two ends of the helix to behave very differently, and in fact this is what I observe. Both ends of the helix displace under pressure, but the displacements are different in amplitude and direction.

The C-terminal end residues moves as much as 0.5 Å towards the cavity (perpendicular to the axis of helix E). As seen in the figure, there is more displacement between ambient pressure and 1 kbar than between 1 kbar and 2 kbar. The motion is largely a pivot about a point near residue 67. Residues 60-70 do not move substantially perpendicular to helix E.

In another view (bottom panel of Figure 5.11, we can see that the N-terminal end of helix C does move, but primarily along the axis of helix E. Again, this appears to be a pivot about residue 67's main chain atoms. The displacement is less substantial, about 0.25 Å at the extreme N-terminal end.

There are number of interatomic contacts in the region of the C-helix kink. Asp70 makes a hydrogen bond to sheet His31, but this bond appears to deform under pressure. It appears that the kink may result from a complicated set of atomic contacts between Phe4 and Ile7 on the A helix, Phe67 and Val71 on helix C, Ile29 in the β-sheet, and Phe104 on the E helix. This will be considered further in Chapter 7.

The fact that the C-terminal end of helix C is so free to move is intriguing. We might expect this to be due to the truncation of residue Leu99 to Ala99, since a number of atoms on or near this end of the helix line the cavity ($C_{\gamma 1}, C_{\gamma 2}, C_{\delta 1}$ of Ile78, and $C_{\alpha}, C_{\beta}, C_{\delta 1}, C_{\delta 2}$ and the main chain oxygen of Val84.) However, from

Figure 5.7 and Figure 5.12 on page 145, it is quite clear that the large changes in the C helix are conserved in both the L99A and WT* mutants. In fact, it appears that two hydrogen bonds (between Glu108 and Asn81 and between Leu84 and Asn81) stabilize a somewhat open atomic configuration at ambient pressure. As pressure increases, these bonds deform, and the C-terminal end of helix C collapses inward, regardless of the presence of an enlarged cavity.

5.3.9 Changes in the C-terminal domain

From Figure 5.7 it was clear that there were a number of interesting changes in the C-terminal domain. These lie in the D helix (the shoulder of the large peak that included the C helix), a peak centered on residue 108 in helix F, a smaller feature in helix G centered on residue 120, and another significant feature centered around residue 133.

Helix E is most remarkable as it changes almost not at all. Of all the residues in the molecule, it is those immediately surrounding residue 99 on the E helix that displace the least, a fact that can been seen in each of the figures in Section 5.3.6.

Helix D

In Figure 5.13 we can see that helix D, particularly residues 82-85, move in towards the cavity by as much as 0.25 Å. As noted before, this appears to be due not to the cavity, but rather to the loosely packed quality of this region of the protein. The motion is conserved in the WT* molecule.

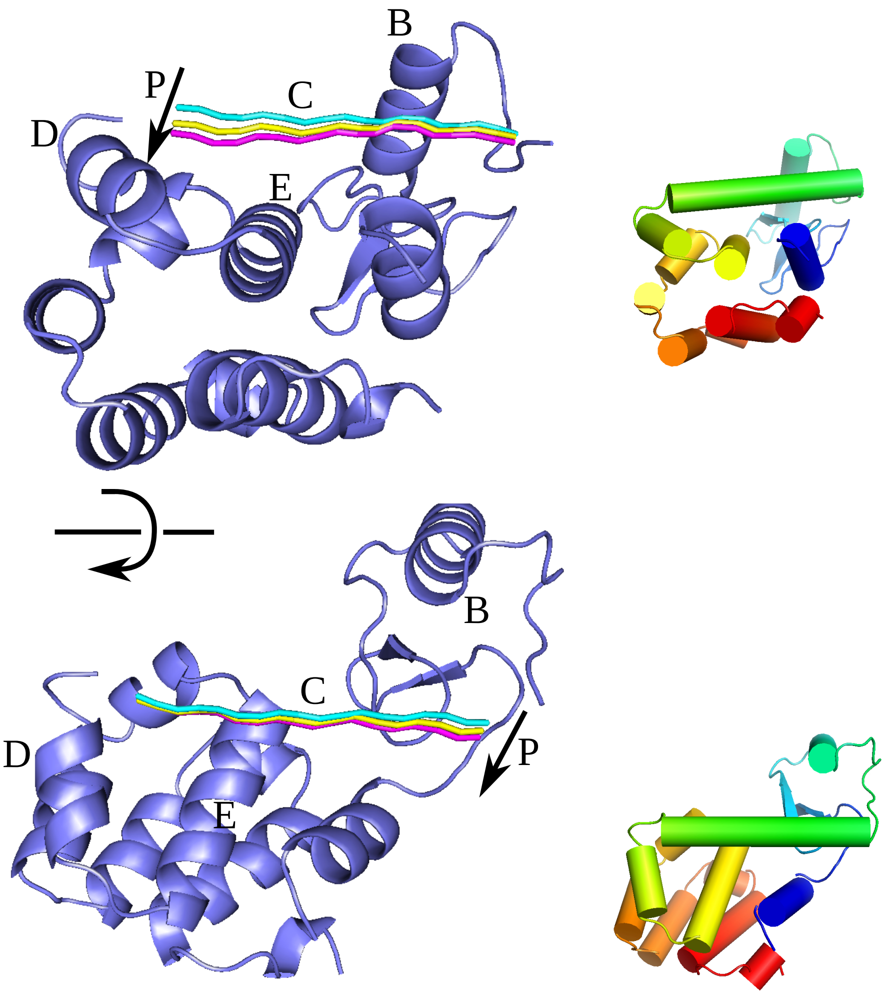

Figure 5.11: Two views of the C-helix axis in T4 mutant L99A. A cartoon of the remainder of the molecule is shown for visual orientation. Displacements magnified by 5. Colors: (cyan) ambient pressure, (yellow) 1 kbar, (magenta) 2 kbar. (Top) A view along the axis of the E helix (shown center), with the B helix in the upper right background. (Bottom) A view perpendicular to the E helix, rotated ≈ 90 degrees relative to the upper panel as shown. Letters indicate helix names. Arrows labelled P indicate the direction of pressure-induced displacements. Insets: orientation cartoons for each panel, showing helix B in light blue, helix C in green, and helix E in yellow.

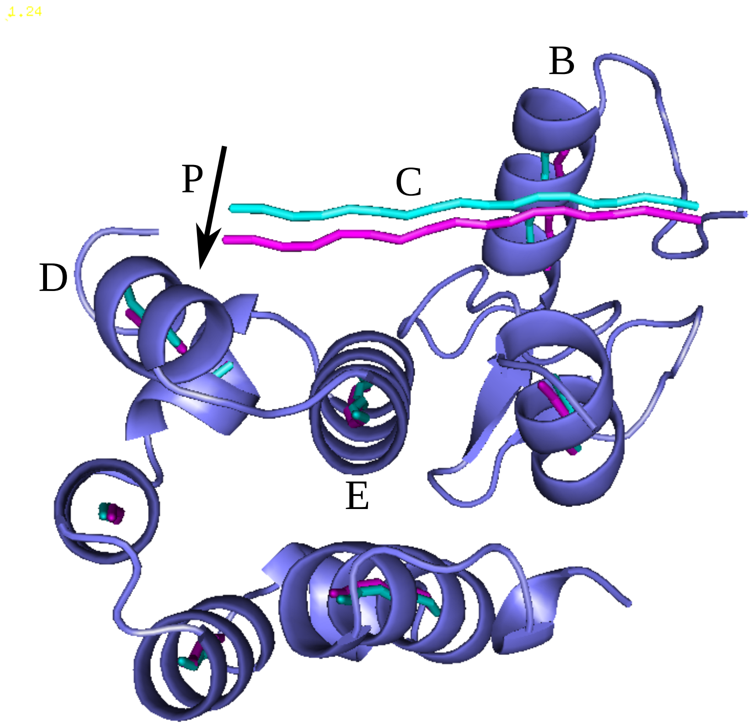

Figure 5.12: Deformation of the C-terminal end of Helix C in T4 WT*. Displacements magnified by 5. The view and colors are the same as the top panel of Figure 5.11.

Helices F and J

Helix F is shown at the top of Figure 5.14. A slight rotation is visible in the figure, in which one end (N-terminal) moves opposite to the other end. The displacements are small, on the order of 0.1 to 0.15 Å over 2 kbar. The helix rotates about 2 degrees, and this motion seems to be coupled to a very slight movement of helix G along its own axis and towards its C-terminal end. While the WT* molecule shares this displacement of helix G, the displacements in helix F are much less convincing, and look more like noise than anything real. It is somewhat difficult to attribute these differences to the enlarged cavity of L99A: residues along helices F and G

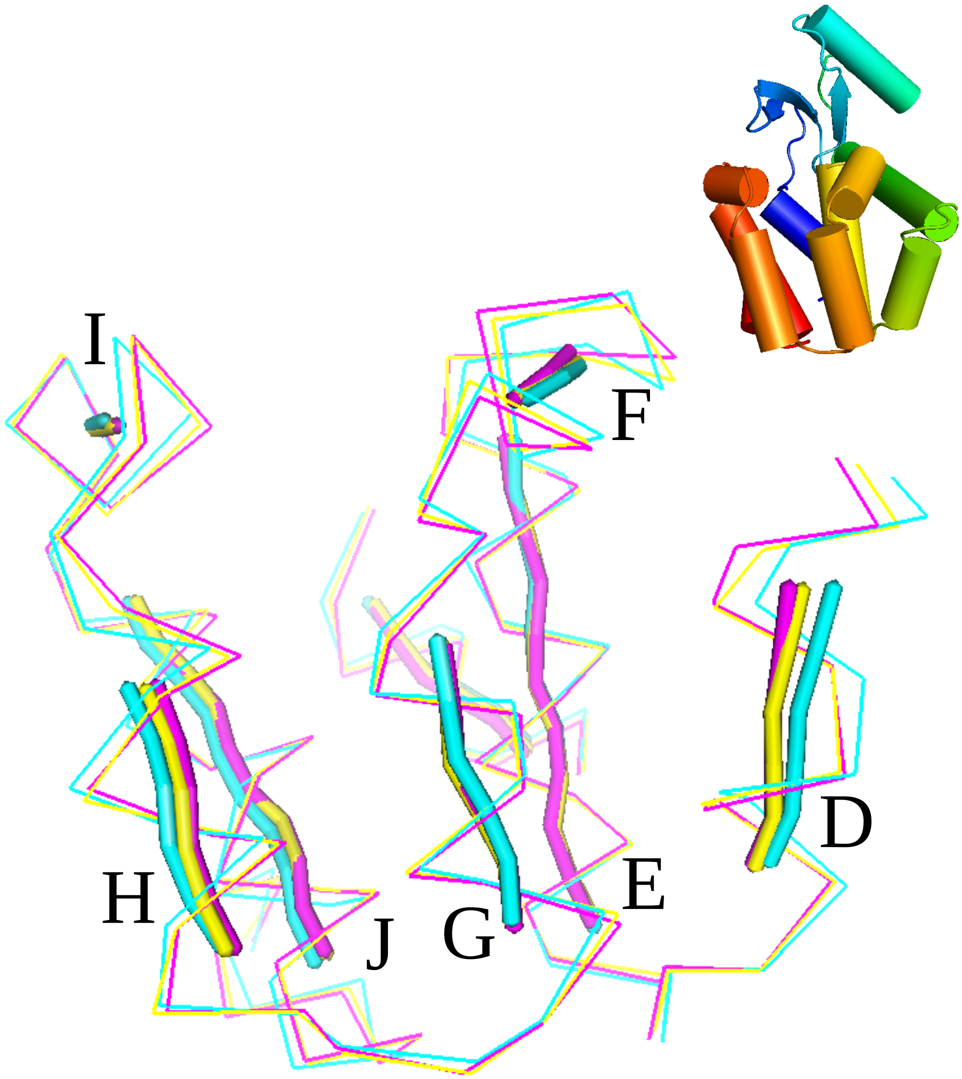

Figure 5.13: A view of the C-terminal domain, looking through helix G into the cavity, showing a C_α trace for L99A at ambient pressure, 1 and 2 kbar, and the helical axes. Colors are the same as the top panel of Figure 5.11. Displacements magnified by 5. In the orientation inset, helix D is yellow-green and helix E (yellow) is just visible behind helix G (orange).

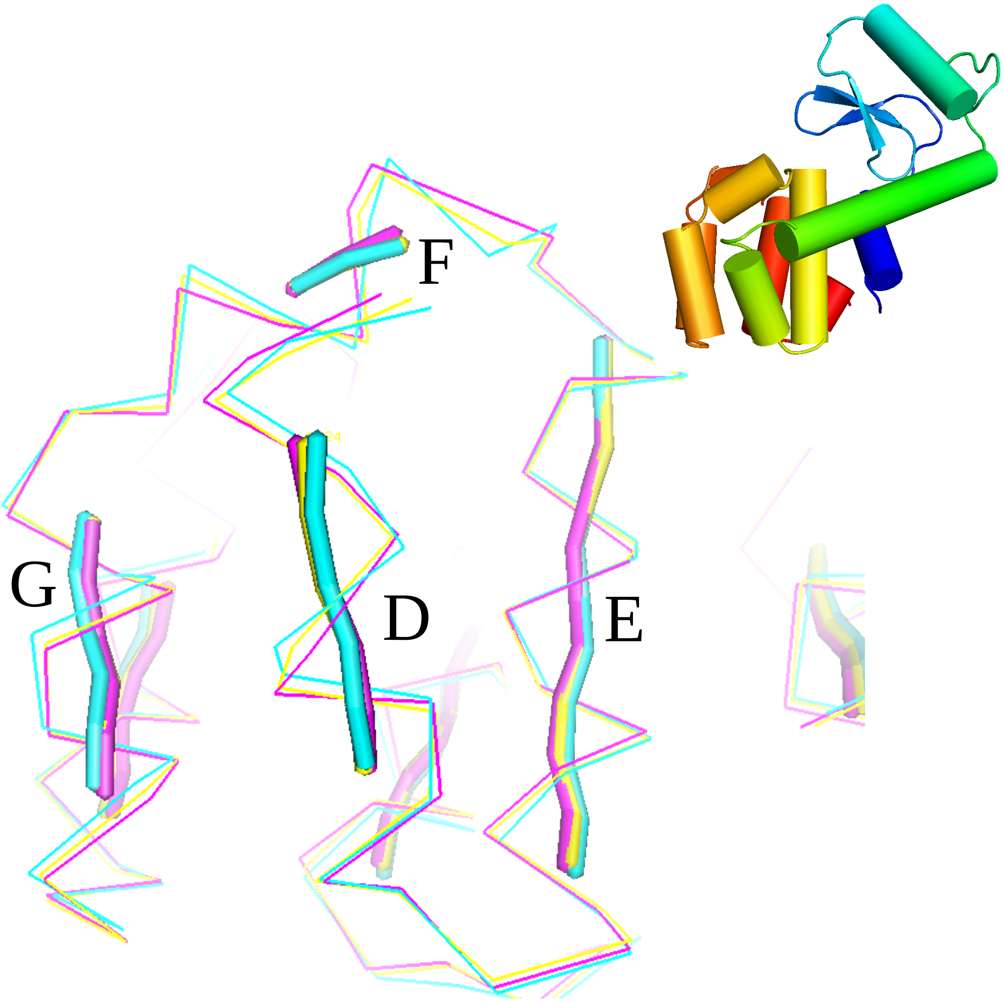

Figure 5.14: A view of the C-terminal domain looking through helix D into the cavity. Legend is as for Figure 5.14. Displacements magnified by 5. In the orientation inset, helix C is green, helix D is yellow-green, helix E is yellow, and helices F and G are orange.

border the cavity of the WT* molecule as well. At first glance it appears that the side chain of Val111 or Phe114 might contact Leu99 in the WT* protein, but again this turns out to be false. This particular set of displacements will remain a mystery for the moment.

Helix H

Helix H is visible at the extreme left of Figure 5.13. As is evident, the N-terminal end displaces very little, while the C-terminal end displaces by as much as 0.28Å into the cavity. (Again, the residues which move the most, Leu133 and Ala134, line both the L99A and WT* cavities, and equal displacements are seen in both cases.) It is interesting to note that Phe114 makes contact with Leu133, and that their displacements seem to be concerted. This still does not resolve the different changes in helix F between WT* and L99A, since Leu133 does not contact Leu99 in the WT* protein.

Helices I and J

Helix I is so short that deformation is hard to define, but one end (Trp138) moves about 0.15 Å towards the WT* cavity center. This, and a similar motion of Ala146 are related to the motion of Leu133 by a string of atomic contacts. Helix J (which includes Ala146) primarily translates about 0.1 Å towards the cavity.

5.4 Remarks

It is remarkable that the introduction of a large cavity in the core of the C-terminal domain does not result in large structural changes at ambient pressure. It is yet more remarkable that the cavity mutant and the psuedo wild type protein respond in the same way to high pressures. While there are small differences between the two mutants, I find it difficult to connect those differences to the L99A mutation. The connection may be through collective motions of side chains lining the cavity which are only detectable in certain places. Whatever the underlying interactions governing the pressure response of these molecules, it appears that the core of the molecule is not particularly important. It remains to be seen what is most important.

A remarkable set of interactions exists in the region of the C helix kink. Strong interactions and contacts pin helix A to helix E. Two residues on helix A constrain the motion of helix C near the kink, and many contacts, including three hydrogen bonds, connect the kink region to the N-terminal domain at points of minimum displacement. It seems fair to say that this region forms the structural core of the molecule.

Another notable feature is that the pressure induced displacements are larger between ambient pressure and 1 kbar than between 1 and 2 kbar. This could be an indication that the molecule is finally packing densely. Or it might be an indication of some other relaxation mode occuring. As we'll see in the next chapter, water does enter the cavity at the highest pressures in this experiment. What connection can be made between that water and the structural relaxation of the protein must wait until we have explored where and how much water is present.

Chapter 6

Observation and simulation of water in cavities at high pressure

In the previous chapter I examined only the features of the atomic models of the WT* and L99A T4 Lysozymes refined from high-pressure x-ray diffraction data. I did not consider whether the models were themselves complete. In particular, does anything fill the cavity as pressure is increased? This chapter describes the methods used to address this question, and presents the evidence suggesting that water fills the L99A cavity in a cooperative fashion as pressure is increased.

6.1 Construction of experimental electron density maps

6.1.1 Review

Electron density is the most fundamental quantity we can derive from x-ray diffraction. In this chapter we will rely almost exclusively on electron density maps to understand changes in the cavity as pressure increases. It is useful to review a few key concepts before mentioning some of the practical challenges and extensions that have been used here.

X-rays scatter overwhelmingly from the electron density in a system. In crystallography we seek to associate this electron density with an atomic model, but our model is often incomplete. To check this, we compare electron densities from experimental data and models. From a perfect knowledge of the scattered amplitude and phase for all scattering vectors $\boldsymbol{q}$, we calculate an electron density by computing the Fourier transform of the scattering factors:

$$\rho(\boldsymbol{r}) = \int \mathrm{d}^3q \; F(\boldsymbol{q}) \mathrm{e}^{i\alpha(\boldsymbol{q})} \mathrm{e}^{-\boldsymbol{q}\cdot\boldsymbol{r}}. \tag{6.1}$$

In a crystal, Equation 6.1 becomes a sum over reciprocal lattice vectors. Our experiment does not measure the phase of each Bragg reflection, so we must construct a set of phases (generally from an atomic model) and obtain

$$\rho(\boldsymbol{r}) = \sum_{hkl} F_{hkl} e^{i\alpha_{c,hkl}} e^{-\boldsymbol{q}_{hkl}\cdot\boldsymbol{r}}, \tag{6.2}$$

where the sum is over the Miller indices h, k, l. Using this equation, we calculate electron density from our data plus a model of phases or completely from an atomic model. We have one final problem, namely that we are missing the most important reflection, $h, k, l = 0, 0, 0$: we blocked this unscattered beam to protect the x-ray detector. The $h, k, l = 0, 0, 0$ reflection gives the total number of electrons in the system, and therefore an absolute scale for all other reflections. In the model refinement stage the data may be scaled to the calculated model structure factor amplitudes $F_{c,hkl}$, using methods similar to those for scaling diffraction images to each other. The calculated and (scaled) observed structures factors are then used to make electron density "difference maps", which are compared to note any changes needed in the model.

6.1.2 Observed difference electron density maps

Here we seek to observe pressure-dependent changes in proteins, in the least model-dependent fashion possible. The least model dependent object we can construct is the *observed* difference electron density

$$\rho(p_2, \boldsymbol{r}) - \rho(p_1, \boldsymbol{r}) = \sum_{\mathsf{h}} (F_{\mathsf{h}}(p_2) - F_{\mathsf{h}}(p_1)) e^{i\alpha_{c,\mathsf{h}}} e^{-\boldsymbol{q}_{\mathsf{h}}\cdot\boldsymbol{r}}, \tag{6.3}$$

where h now stands for the Miller indices $\{hkl\}$, and the p_i are the pressures of the datasets being compared.

Several problems remain. To determine absolute electron densities we must scale the data as before, but now it is not clear to what they should be scaled. We may choose either the high or low pressure *calculated* structure factors. This choice is not particularly satisfying, as it forces some model dependence on us, but the calculated phases already include such dependence. In any case we have no other choice unless we are able to accurately measure the intensity of the forward scattered amplitude F_{000}. We must also choose a set of phases (high or low pressure); the choice is similarly arbitrary. We are encouraged by the fact that the small changes in the molecule catalogued in Chapter 5 should result only in small changes in the phases and amplitudes, especially at lower resolutions. Since all comparisons are made to the ambient pressure data, we will use the ambient pressure model phases for consistency.

Finally, the maps must be multiplied by a factor of two to compensate for uncertainties in the phases[143].

If the model structures and unit cells are not identical, another problem arises. To compare two sets of structure factor amplitudes $F(p_1), F(p_2)$, the atomic models must (technically) be identical. Otherwise the phases are not identical, and the map will contain artifacts. In our case, the molecules themselves change so little that Equation 6.3 is a very good approximation as long as the unit cell also does not change. But the unit cell does change substantially (by up to 1 Å in the c-axis). While the amplitudes we measure have the same *indices*, they do not sample the same positions in reciprocal space.

Recall that the scattering of x-rays from a crystal can be thought of in two parts, the scattering amplitude of the molecule itself, and a modulation due to the crystal lattice. The crystal is a convolution of the molecule and a lattice, and the Fourier transform of a convolution $\mathcal{F}(A \circ B)$ (where $(A \circ B)$ denotes the convolution of functions A and B) is the product of Fourier transforms $\mathcal{F}(A) \times \mathcal{F}(B)$. Thus we may think of the crystal lattice sampling the intrinsic scattering of the molecule at precise positions in reciprocal space. If the unit cells of two crystals differ, then the sampling will be different as well.

We here make a somewhat arbitrary truncation at four times the maximum difference in unit cell parameters, δ. Here $4\delta = 4\,\text{Å}$. This "rule of 4" is not particularly sophisticated. Imagine a 1 Å change in the length of a lattice vector, in a unit cell approximately 100 Å across. Direct comparison of the reflections at $d = 2\,\text{Å}$ will put an atom properly at a crest of the electron density at a trough. On the other hand, for reflections of 4Å or lower resolution, points on the crest are shifted at most 1/4 wavelength.

One might argue that we do not need to use the same set of phases, and that we do not need to reference the reflections to the same unit cell. In the former case, we have simply chosen to reduce model dependence as much as possible. The latter is an option not currently available in the crystallography software used for this work.

6.1.3 Integrating electron density in the cavity

After scaling the high and ambient pressure data sets to a common absolute level, and constructing the difference map, we wish to determine the total number of electrons in the cavity. The key problem here is in determining whether a given point in the map is in the cavity or not.

This determination and integration were performed using *VOIDOO*[139] and *MAPMAN*[135], from the Uppsala Software Factory (http://xray.bme.uu.se/usf). *VOIDOO*, described previously, outputs an electron density "mask" in the form of a standard map whose values are either one if the point is in the cavity and zero otherwise. Using *MAPMAN*, a package of electron density map utilities, this map may be multiplied with the difference map described above, and the values in the map summed. Since the values near the edge of the cavity are at the noise level of the map (that is, at one standard deviation), this method should be as accurate as is reasonably achievable.

It is important to note that for consistency I have chosen to use a 4 Å cutoff resolution in the construction of all of the maps. If the 2 kbar-ambient difference maps are to be accurately compared to the 1 kbar-ambient maps, they should use the same resolution. In principle this can affect the final number of electrons calculated to be in the cavity; tests on the 1 kbar data show that the effect is on the order of ±1 electron.

6.1.4 Including water in the atomic model

We might attempt to determine the number and distribution of water molecules in the cavity from the standard electron density difference maps between model calculated and observed structure factors. Such determinations are problematic. Among the most incipient problems is that the distribution need not be limited to only one configuration or number of water molecules. As we shall see, simulation suggests that the water samples a quite broad range of configurations and occupanies. At present, we have no *experimental* data which indicates a favored cavity occupancy number or configuration. Thus a proper crystallographic refinement model would include a large number of configurations somehow weighted. Techniques exist for such modelling, but we are again limited by the number of available data, namely the observed Bragg reflections.

Future experimental work must address this issue, but for now I have used *arp_waters* as discussed in Section 4.4.1 to locate water molecules. The use of this program to locate cavity water molecules is somewhat secondary to its use in improving the solvent model around the protein. As we will see, it does a poor job of modelling the cavity water.

6.1.5 Water or something else?

If we observe electron density in the cavity, we should not immediately assume that it is due to water. Our conclusion that the observed increase in electron density is due to water will be supported by Gerhard Hummer's molecular dynamics simulations. Other experimental clues help us to arrive at this conclusion.

The principal chemicals in solution are potassium and sodium phosphates, and the additive β-mercaptoethanol (BME), which includes sulfur and oxygen atoms. Water has ten electrons per molecule, fewer than potassium ions (18), but the same as sodium ions (10). Despite our eventual conclusion that the protein core must make strong electrostatic interactions with buried water, we still assume that bare charges (each ion is +1) are substantially disfavored in a low-polarity environment. Phosphate ions have 44 electrons, making them more visible in electron density than water. Also, their charge of -3 should make them highly unfavorable in the cavity. The BME molecule has 36 electrons, but similar electron density to water. It has a molecular surface volume of $\sim 78\,\text{Å}^3$, and a solvent accessible volume of $\sim 257\,\text{Å}^3$. Compounds similar to BME do not bind in the cavity at ambient pressure[50]. Ethanols have a dipole moment of approximately 1.7 D, compared to 1.9 D for water. Due to its size, BME will not be able to enter the cavity in groups which can hydrogen bond with each other, and presumably it would lose some conformational entropy upon entering the cavity. Even if BME interactions are very similar to those of water, its concentration is about a thousand times smaller than water (50 millimolar versus 55 molar), so that almost everything filling the cavity should be water.

This analysis suggests that the most likely candidate for cavity filling is water, but it cannot rule out other possibilities. Complete filling of the cavity at higher pressures would help remove any uncertainty. Even with the data we have, the MD simulations will help us feel more certain that the cavity is filled with water, not some other substance.

6.2 Molecular dynamics simulation

Instead of trying to empirically determine the cavity water distribution, we can momentarily sidestep the data and simulate the cavity to predict the water distribution. Comparisons can then be made with the observed occupancies (integrated electron density) and shape of the electron density distribution. If these comparisons are favorable, then we can use the simulation to interpret the experimental results. I have relied on Gerhard Hummer of the National Institutes of Health to implement this part of the study, but to understand the results it is important to describe the principles of protein simulation by molecular dynamics.

6.2.1 Principles of molecular dynamics

The basic principle of molecular dynamics simulation is easy: apply Newton's second law to all atoms in a system. Of course there are many caveats, since we cannot analytically solve the resulting equations for anything more complex than two-body interactions, much less the underlying quantum mechanical interactions. We are left to solve the problem numerically. For the moment, we will assume that we have solved the (much more difficult) problem of determining the forces on any atom (or equivalently their energies). The details of this will be considered briefly in the next section.

The simulations described in this chapter use the AMBER 6.0 molecular dynamics package[144, 145]. It uses a so-called "leap-frog" variation on the Verlet algorithm[146]. All such algorithms are based on simple Taylor series expansions of the atomic positions r_i:

$$\boldsymbol{r}_i(t+\Delta t) = \boldsymbol{r}_i(t) + \boldsymbol{v}_i(t)\Delta t + \frac{1}{2m_i}\boldsymbol{f}_i(t)\Delta t^2 + \frac{1}{6}\dddot{\boldsymbol{r}}_i + \mathcal{O}(\Delta t^4). \tag{6.4}$$

$\boldsymbol{v}(t)$ and $\boldsymbol{f}(t)/m$ are the first derivative (velocity) and second derivative (force divided by the mass of the relevant atom) of $\boldsymbol{r}_i$ with respect to time. These can be approximated numerically at each point in the simulation.

Equation 6.4 could be used on its own, but it turns out that this formulation is not in fact time reversible, and therefore does not conserve energy. Summing Equation 6.4 with the equivalent expansion of $r_i(t-\Delta t)$, and rearranging we obtain the Verlet algorithm

$$\boldsymbol{r}_i(t+\Delta t) \approx 2\boldsymbol{r}_i(t) - \boldsymbol{r}_i(t-\Delta t) + \frac{\boldsymbol{f}_i(t)}{m_i}\Delta t^2, \tag{6.5}$$

which is accurate to $\mathcal{O}(\Delta t^4)$ and is less susceptible to long time scale energy drift[146]. Dropping subscripts, velocities can be calculated at each time step as

$$\boldsymbol{v}(t) = \frac{\boldsymbol{r}(t+\Delta t) - \boldsymbol{r}(t-\Delta t)}{2\Delta t}. \tag{6.6}$$

AMBER 6.0 uses the *leap-frog* algorithm, a slight variation on the Verlet scheme. Velocities are now calculated at half-integer time steps:

$$\boldsymbol{v}(t+\Delta t/2) = \frac{\boldsymbol{r}(t+\Delta t) - \boldsymbol{r}(t)}{\Delta t}, \tag{6.7}$$

and similarly for $\boldsymbol{v}(t-\Delta t/2)$. Then the positions are updated as

$$\boldsymbol{r}(t+\Delta t) \approx \boldsymbol{r}(t) + \boldsymbol{v}(t+\Delta t/2)\Delta t. \tag{6.8}$$

This yields identical trajectories to the Verlet scheme Equation 6.5, but the velocities are not calculated at the same time points.

6.2.2 The AMBER force field

The simulations presented in this chapter use the ff94 or parm94 force field of the AMBER package[144, 147, 148] available from the University of California, San Francisco or from the Scripps Research Institute (URL: http://amber.scripps.edu). The form of the force field is[148]

$$\begin{aligned}\mathcal{V} = &\sum_{\text{bonds}} K_l(l-l_{eq})^2 + \sum_{\text{angles}} K_\theta(\theta-\theta_{eq})^2 \\ &\sum_{\text{dihedrals}} \frac{K_n}{2}[1+cos(n\phi_d-\gamma)] + \sum_{i<j}\left[\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} + \frac{q_iq_j}{\epsilon r_{ij}}\right].\end{aligned} \tag{6.9}$$

The first term is a sum over bonds, restricting bond lengths to l_{eq} with a force constant K_l which varies with bonding type. The sum over angles similarly restrains bond angles to equilibrium values θ_{eq}. The equilbrium values are derived as for the stereochemical restraints used in macromolecular structure refinement (see Chapter 4), and force constants are derived from vibrational frequency data on small molecules. The dihedral, or torsion, angle term limits rotations ϕ_d about bonds and is a Fourier sum where each component n has a corresponding strength K_n and the preferred angles can be offset from zero by γ. For instance, C-C bonds generally include only a 3-fold term (that is, $n = 3$), but may include a 2-fold term if there is some reason to favor one rotamer over another, or in the event of sp^2 bonding (as in aromatic rings).

The electrostatic term is a combination of a Coulomb term (with dielectric constant ϵ) and a Lennard-Jones term. Each atom has a fixed partial charge q_i at its atomic center, derived from least-squares fits to electrostatic potentials from quantum mechanical Hartree-Fock wavefunction calculations using a 6-31G* basis set[147, 148]. These charges sum to zero so that the sum of electrostatic potentials will converge. Some form of Ewald summation must be used to handle the long ranged Coulomb interactions, since we will only simulate a finite volume. Handling these interactions is non-trivial and great care must be used, but the topic is beyond the scope of this work. We use the particle mesh method[146]. The Van der Waals parameters A and B for carbon atoms are determined by simulating appropriate liquids (*e. g.* C_3H_8 or C_4H_{10} for aliphatic carbons or benzene for aromatic carbons). Many other parameters were determined from the OPLS (Optimized Parameters for Liquid Simulation) model[148, 149].

Notably there is no explicit hydrogen bonding term. Instead this is accounted for in the electrostatics of particular typical hydrogen bonding groups, by varying the Lennard-Jones parameters of hydrogens depending on their bonding and second-bonding partners. For instance, the Lennard-Jones radius for hydrogen in O-C-H will be smaller than that in C-C-H.

The philosophy behind the parm94 parameterization is to over-charge (and over-polarize) atoms in order to compensate for the lack of polarizability in the model. This is also done to match the water model used in the simulations (typically something like the TIP3P model[148, 150], which also lacks polarizability and is over-charged). This can lead to some problems. Among other things, in this charging scheme, buried partial charges tend to be underdetermined, and conformational effects are neglected[147]. Some effort has been made to compensate for this by averaging over many conformations[147, 148], but errors may still result.

Fortunately, many properties of proteins are fairly insensitive to the exact details of the fixed atomic charge distribution, at least if the simulation is begun from a known native state(*e. g.* [151]). On the other hand, ligand binding or other behavior dependent on long ranged interactions may be sensitive to such parameters[147]. It remains to be seen how much of an effect this would have on the simulations of water in the hydrophobic cavity of T4 mutant L99A. Given that polarizability probably plays an important role in water-nonpolar residue interactions, this is an avenue for future simulation work. As of 2003 at least, most simulations of proteins did not include explicit electronic polarizabilities. As with all theoretical approximations, experiment is the final arbiter of success, and we will see below that the simulations do reasonably well. Whether this is simply luck must remain a question for another day.

6.2.3 Thermodynamics and implementation

For thermodynamic studies, we use time averaged simulation data to determine the free energy of various water-filled cavity states. An excellent survey of this sort of grand-canonical averaging in molecular dynamics (MD) can be found in Daan

Frankel and Berend Smit's book *Understanding Molecular Simulation*[146]. The structure (pdb code 1L90) is simulated with 4964 TIP3P water molecules[150], 18 sodium ions and 27 chloride ions. The experimental structure is not at an energy minimum in the MD force field, so a brief energy minimization is necessary. The "production" simulations are then performed at constant temperature (300K) and pressure (1 bar, 1 kbar, or 2 kbar) and with a 1 fs time step. We use particle mesh Ewald summation and periodic boundary conditions to handle long ranged charge interactions.

After 54 ps of simulation with the cavity held empty, water molecules are individually removed from bulk and placed into the cavity to simulate states of occupancy $N = 0, 1, 2, 3, 4, 5$. These configurations are further equilibrated for 50 to 300 ps, after which the system is simulated for 1 to 1.7 ns and structures are saved for analysis every 0.5 ps. In each of the 18 simulations, the resulting structure is averaged over the final 0.25 ns for comparison to the original structure. The α-carbon RMSD is less than 0.7 Å for the whole molecule, and by less than 0.55 Å for the N and C terminal domains separately.

The simulations themselves only allow us to determine the internal interaction energies of each system at one occupany number N and pressure p. The relative free energies of each state are determined by a grand canonical partition function for the water molecules in the cavity using histogram matching[72, 73, 146]. We occasionally insert or remove a water molecule from the cavity to determine the internal energy change, and use this to derive the free energy differences by averaging over the simulation time. Below is a summary of the method.

The grand partition function for water in the cavity is (*e.g.* [73])

$$\Xi = \sum_{N=0}^{\infty} \frac{z^N}{N!} \int d^N \mathbf{r} \mathrm{e}^{-\beta U_N}. \tag{6.10}$$

The U_N are the interaction energies of N water molecules in the cavity with each other and their (protein) environment. In equilibrium with bulk water the activity[1] can be written as[2] $z = \rho_{bulk} \exp(\beta \mu_{bulk}^{ex})$[152, 153], where μ_{bulk}^{ex} is the bulk excess chemical potential. From the test-particle insertion method[146], the excess chemical potential of water in the cavity is given by the canonical average over the energy of inserting an additional water molecule.

$$\exp(-\beta \mu_N^{ex}) =< \mathrm{e}^{-\beta(U_{N+1}-U_N)} >_N . \tag{6.11}$$

The probabilities of occupancy $P(N)$ are the terms of the sum in Equation 6.10. Then the ratios of those probabilities can be written as

$$\frac{P(N+1)}{P(N)} = \frac{\rho_{bulk} V}{N+1} \exp(\beta \mu_{bulk}^{ex} - \beta \mu_N^{ex}). \tag{6.12}$$

[1] $z = \exp(\mu/k_B T)/\Lambda^3$ with chemical potential $\mu = (\partial G/\partial N)|_{T,p}$ where N is the number of particles in the system, here water molecules in the cavity, and $\Lambda = (\hbar^2/2\pi m k_B T)^{1/2}$ is the de Broglie wavelength of a particle of mass m.

[2] To derive this, note that the chemical potential can be written in terms of the Helmholtz energy as $\mu \approx A(N+1, V, T) - A(N, V, T) = -k_B T \ln(Q_{N+1}/Q_N)$. The canonical partition function Q_{N+1} is

$$\begin{aligned} Q_{N+1} &= \frac{1}{(N+1)!\Lambda^{3N}} \int_V \exp(-\beta W_{N+1}) \mathrm{d}^3 x_1 \cdots \mathrm{d}^3 x_{N+1} \\ &= \frac{1}{(N+1)!\Lambda^{3N}} \int_V \exp(-\beta W_N) \exp(-\beta \phi) \mathrm{d}^3 x_1 \cdots \mathrm{d}^3 x_{N+1} \\ &= \frac{V}{(N+1)\Lambda^3} Q_N < \exp(-\beta \phi) > . \end{aligned}$$

Since $N/V = \rho_{bulk}$, then using the definition of μ above and the definition of z in Footnote 1 above, $z = \rho_{bulk} \exp(\beta \mu_{bulk}^{ex})$, where the excess chemical potential arises from the canonical average of the Boltzmann factor for a test particle inserted into a system of N particles, $< \exp(-\beta\phi) >$. This is the essence of the Widom test particle insertion method[146] where V is the volume into which the water is inserted.

To determine the canonical average in Equation 6.11, we collect the histogram of $\Delta U = U_{N+1} - U_N$ from insertions of a water molecule into a cavity with N water molecules already present, $p_{ins}(\Delta U)$, and removals of a water molecule from a cavity with $N+1$ water molecules, $p_{rem}(\Delta U)$. The ratio of the two distributions is[146]

$$\frac{p_{ins}}{p_{rem}} = \frac{q_{rem}}{q_{ins}} \exp^{-\beta \Delta U}, \tag{6.13}$$

where q_{rem}/q_{ins} is the ratio of the *canonical* partition functions of the $N+1$ and N states, and therefore the average we seek in Equation 6.11. The probabilites $P(N)$ can then be calculated and normalized straightforwardly.

6.3 Observed water in the cavity

Figure 6.1 shows the electron density inside the L99A T4 lysozyme cavity at 2 kbar. It also shows the positions of water molecules in the atomic model, determined automatically by the refinement software. The electron densities are each contoured at the same absolute level, $0.1\,e/\text{Å}^3$. (The average electron density of one water molecule is roughly 0.3 to 0.5 $e/\text{Å}^3$.) Integrated densities are listed in table 6.1 on the next page.

6.3.1 Electron Density Maps

The noise in the maps can be disconcerting. In Figure 6.1, noise is visible in each density map, and appears to be strongest in the 2 kbar-ambient map. However, the noise in the averaged maps is small compared to the density in the cavity. While there may be significant noise in the individual difference maps, that noise averages

Table 6.1: Integrated electron densities from observed electron density difference maps (Section 6.1.2. Comparison is by data set, units are electrons. Numbers in bold represent the average over all comparisons between high and ambient pressure. Datasets are as listed in 3.1 on page 71.

Dataset	mt0k1	mt0ka	mt0kb	average
mt1k6	10.7	7.0	4.5	7.4
mt1k7	9.0	5.8	2.1	5.6
mt1k9	5.9	4.1	0	3.3
1k average	8.5	5.6	2.2	**5.4**
mt1.5k1	17.4	14.6	11.4	**14.5**
mt2k1	20.7	17.6	15.8	18.0
mt2k3	22.6	20.6	18.5	20.6
mt2k8	24.4	22.2	20.2	22.3
2k average	22.6	20.1	18.1	**20.3**

out almost everywhere but the cavity. This again demonstrates the importance of collecting multiple datasets in high pressure crystallography experiments.

Contoured at the same absolute level, the 2 kbar - 1 bar maps indicate the same water distribution and magnitude. Similarly, the 1 kbar - 1 bar maps all show the same features. However, the noise in the maps can be quite different. The mt2k8-mt0k1 map has a considerably larger noise level than the mt2k3-mt0k1 map (0.08 compared to $0.05\,e/\text{Å}^3$). For this reason, I have chosen to average the maps. The noise level decreases, but the principal features remain the same.

Besides the clear presence of strong electron density at 2 kbar, a couple of points are worth note. The shape of the electron density indicates three or more distinct

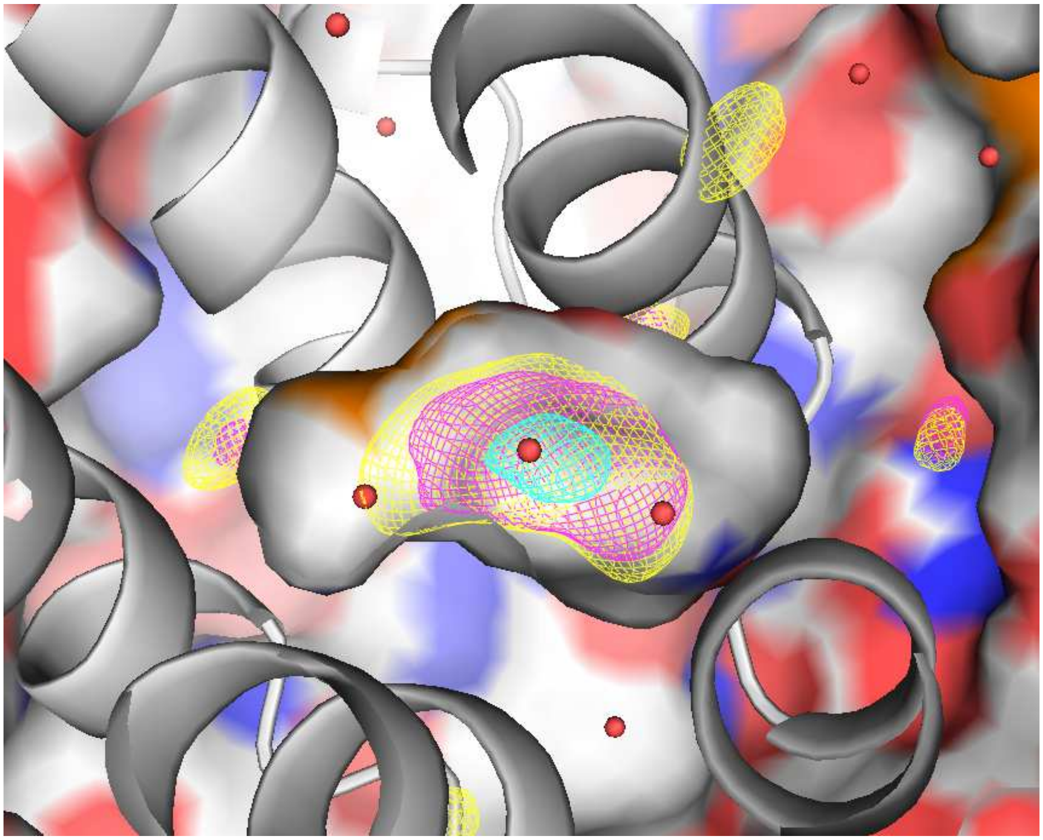

Figure 6.1: Electron density in the cavity of T4 lysozyme mutant L99A. To construct this map, the individual difference maps at any one pressure were averaged in real space. The cavity wall is shown with colors corresponding to the atoms lining the cavity: Carbon, grey; nitrogen, blue; oxygen, red; sulfur, orange. Absolute electron densities at 1 kbar (blue), 1.5 kbar (purple) and 2 kbar (yellow) are contoured at $0.1\,e/\text{Å}^3$. Uncertainties are discussed in Section 6.5.1. Small red spheres are oxygen atoms from model water molecules, identified or verified by the program arp_waters described in Sections 4.4.1 and 6.1.4. Three water molecules from the (x-ray derived) atomic model are visible in the cavity. Schematically they represent the water distribution acceptably, but they model the occupancy and electron density poorly. The view is from the F helix into the cavity.

sites for the water molecules. The 1.5 kbar-1 bar density has the same shape as the density at 2 kbar, and is only reduced in amplitude. This contrasts the case at 1 kbar, where only one peak is clearly visible. An attempt to contour the 1 kbar map at a lower level obscures the cavity water distribution with noise, so it is not clear whether this small peak is part of a broader but weak distribution, or is in fact an isolated peak indicating a more favored 1-water molecule binding site.

While no data are available between ambient pressure and 1 kbar, it is clear that there is a marked change in behavior around this pressure. The electron density in the cavity increases twice as much between 1 and 2 kbar as it does between 0 and 1 kbar. Each water molecule corresponds to 10 electrons, so while at ambient pressure there is no visible electron density, at 1 kbar we see a total of 0.5 water molecules on average, at 1.5 kbar about 1.5 water molecules, and roughly 2.0 water molecules at 2 kbar.

Examining Table 6.1, it is not entirely clear what will happen at higher pressures. It may be that the occupancy will increase very little or that it is still only at the midpoint of filling the cavity. In either case, a plot of the average cavity occupancy does seem to indicate that the cavity occupancy will continue to grow somewhat as pressure is further increased.

6.3.2 Atomic model

The atomic model shown in Figure 6.1 shows three water molecules (as small red spheres) distributed evenly across the cavity at 2 kbar. Unfortunately it is not clear that this is accurate. The total cavity water occupancy at 2 kbar is slightly more than 2 water molecules. Two water molecules alone cannot be responsible for the observed distribution without large temperature factors. Even three water molecules require large temperature factors. The B-factors for the refined water oxygens range from 40 to 70 Å^2, with a mean of about 54 Å^2. Moreover, the number of water molecules found by the refinement software is highly sensitive to various refinement parameters. For these reasons, and those mentioned above, I have little faith in these atomic models.

With those caveats, it is still interesting to note that the central water molecule in Figure 6.1 is conserved across all of the 2 kbar models, and always found within 0.1 Å of the central electron density peak observed at 1 kbar. This position may be more energetically favorable than any other. To better quantify the distribution of water in the cavity, we now turn to computer simulation.

6.4 Molecular Dynamics results

The simulations performed by Gerhard Hummer yield essentially the same result as our experiment, namely that the cavity collapses little as pressure increases and that around 1.5 kbar the cavity begins to fill with water. The occupancies derived from simulation are compared to the experiment (assuming 1 water molecule = 10 electrons and no water in the cavity at ambient pressure) in Figure 6.2, along with a perturbation theory extension to higher pressures. With a shift in the absolute chemical potential of water by $0.4\,k_BT$ (determined by a visual fit to the experimental data), the simulation matches the data fairly well. It is important to note that this shift is an *absolute* shift in the chemical potential of water, and should not be compared to the change in chemical potential as a function of pressure.

In Figure 6.3 we see that all of the occupancy probabilities P(N) increase. $P(N = 5)$ changes most, by more than a factor of 100. However, it is so strongly disfavored at 1 bar that it still contributes little at 2 kbar. At pressures greater than 1 kbar, the $N = 4$ state dominates the average cavity occupancy $< N >= \sum NP(N)$. This effect only becomes stronger as pressure is increased to 2 kbar.

There are a couple of immediate lessons from Figures 6.2 and 6.3. One lesson is that the process is apparently cooperative, so that clusters of water molecules are favored over individual water molecules in the cavity. This is reflected in the steepness of the filling curve at the filling midpoint, which is unfortunately poorly constrained by the current experimental data. This cooperativity implies interactions between the water molecules in the cavity (presumably hydrogen bonding). The x-ray data seem to agree. More, and higher-pressure data, are clearly needed to establish this observation conclusively.

Assuming that the cooperativity is real, it appears that water molecules need to each make about 1 hydrogen bond per water molecule to be stable in the cavity. Three water molecules can make at most an average of 2/3 of a hydrogen bond per water molecule. Four water molecules can make between 3/4 and one hydrogen bond per water molecule, depending on the configuration. Both linearly arranged "wire" states and cyclic "square" states are observed in the simulation, in roughly equal populations, notably similar to gas phase water[72, 154]. In contrast, water in the liquid makes nearly 4 hydrogen bonds in a nearly tetrahedral configuration at any given time (most recently discussed in reference [155] and references therein), so that there are on average almost 2 hydrogen bonds per water molecule in the liquid. Cooperativity alone is insufficient to stabilize water in the cavity; there must be significant interactions between the water and the protein, on the order of one hydrogen bond, which is about 13 kJ/mol (about $5\,k_BT$) for a water dimer, and increases with the cooperativity of hydrogen bonding[154].

We might also guess from the simulation that entropy is playing a role in the transition. Five water molecules should fit in the cavity (the average volume of water in the liquid is about 30 Å^2, and the cavity volume is roughly 160-180 Å^2 depending on how its measured). However, five water molecules are clearly strongly disfavored, as seen in Figure 6.3. One water molecule is seen to escape the cavity in the N=5 simulations as well. It seems that more configurational space, and thus entropy, is required for the cavity to be filled. Experiments and simulations using temperature as a variable would confirm this hypothesis and make it quantitative. For the moment it is mostly speculative.

Analysis of simulation results yields somewhat more quantitative determination of the interaction energies. The average potential energy of water in the cavity changes very little with pressure, as we'd expect given the small changes in the structure observed by x-ray crystallography. Instead the water filling of the cavity is due to changes in the bulk activity of water. The average interaction energy of each water molecule in the cavity decreases as the number of water molecules is increased, from -23 kJ/mol for $N = 1$ to -60 kJ/mol for the $N = 4$ state. Roughly half of this interaction is due to hydrogen bonding, while Van der Waals and charged or dipole interactions with the surrounding protein account for the remainder.

Unfortunately, it remains difficult to separate out more specifically the contributions to the interaction energies. This is the most time consuming part of the simulations, and is also the most susceptible to error. Due to the way in which the potential is parameterized (see Section 6.2.2), only the total energies are well characterized, and polarization responses cannot easily be separated from interactions between permanent dipoles. Considerably more effort would be required to obtain more detailed breakdown of the interaction energies. Still, we have learned a great deal, and the simulation has suggested some important experiments to follow those described in this thesis.

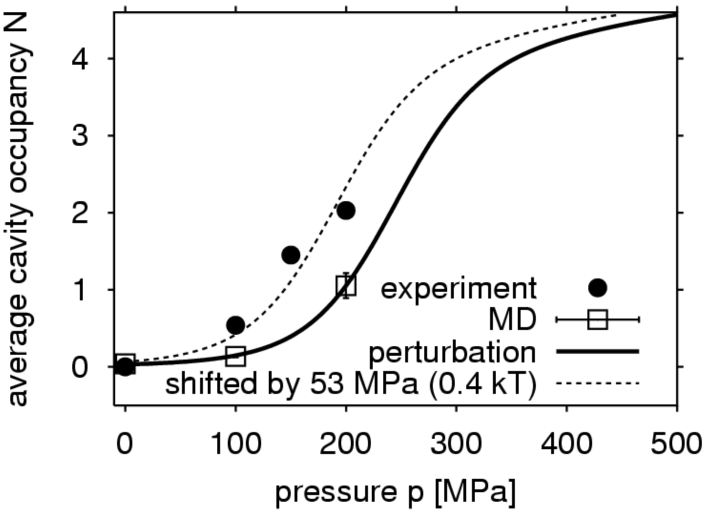

Figure 6.2: Water occupancies determined from molecular dynamics simulation. Filled circles are the experimental data, unfilled squares are determined by simulation. Experimental uncertainties are estimated at $\lesssim 0.5$ water molecules (see Section 6.5.1). The solid line is a perturbative extrapolation from the simulation, The dashed line is the same curve shifted 53 MPa, determined by a visual fit to the experimental data.

One of the more surprising findings from simulation was the escape of water molecules from the cavity. In one instance in an $N = 1$ simulation, and a separate instance in an $N = 5$ simulation, one water molecule escaped the cavity. First, the Phe114 side chain rotated, providing a clear path from the cavity to the outside bulk solvent between the side chains of Phe114, Ser117, Asn132 and Leu133. The water molecule migrates through this opening to a site observed to have bound water in the original L99A structure (pdb code 1L90, WAT196). Such a pathway corroborates earlier NMR data[35] which suggested that there might be a pathway

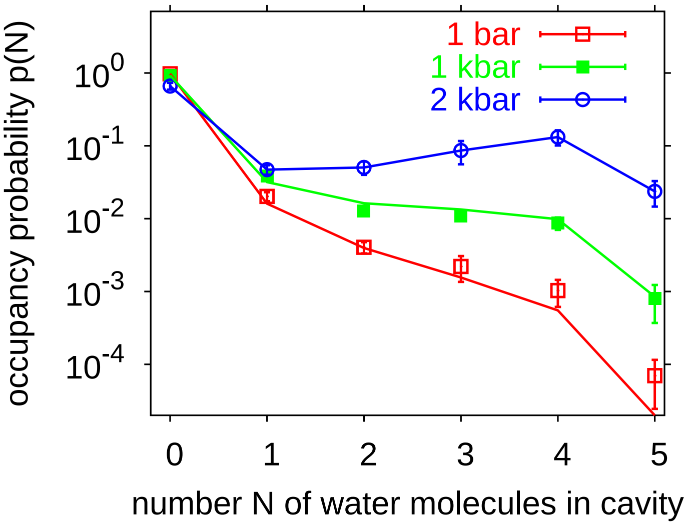

Figure 6.3: Occupancy probabilites by occupancy number N. Note that most favorable state at high pressure is 4 water molecules.

involving the F and G helices.

6.5 Other cavities

6.5.1 WT* Lysozyme

The situation is quite different in the WT* T4 lysozyme, whose main cavity is adjacent to the Leu99 side chain (and which is part of the large cavity in the L99A mutant.) In principle one water molecule could fit in this cavity, but I observe no such change at any pressure. Constructing difference maps for the wild type data as I did for the mutant, and integrating the electron density as before, I find a total cavity occupancy on the order of 1-2 electrons, a number which upon inspection of the density maps is clearly at the noise level. Moreover, this number appears to be independent of pressure. No oxygen or nitrogen atoms line the wild type cavity, making it unlikely that hydrogen bonding could take place.

The WT* protein thus gives us a nice measure of our uncertainty. The WT* cavity has a volume approximately 5 times smaller than the L99A cavity[47]. If this is random noise, the noise level in the integrated electron density of the L99A cavity should be $1.5 \times \sqrt{5} \approx 3.5$ electrons, less than one half of one water molecule. If instead this is taken to be systematic background, then the error would be on the order of one half of one water molecule. Given that the noise cancels well in the observed difference maps, I am inclined to believe that the errors are random and not systematic.

6.5.2 Other cavities in L99A Lysozyme

There are other cavities in this molecule, including a small cavity near the C_β atom of Phe153, and another cavity between the N-terminal end of helix C and the β-sheet. This latter cavity is hydrated in the ambient pressure WT* and L99A mutant structures, and there is no observable increase in electron density associated with these two water molecules. The former shows no increased electron density in the 2 kbar-ambient difference electron density maps for either the wild type or L99A mutant data.

6.6 Remarks

At pressures above 1 kbar, appreciable water can be detected in the main cavity near the L99A mutation of T4 Lysozyme. The experimental data are ambiguous as to the exact distribution of this water, but most likely indicate 3 or more water molecules having $P(N)$ substantially less than one. The molecular dynamics simulations indicate that the $N = 4$ state dominates, especially at higher pressures.

Most importantly the experiment and the simulation agree that the interactions of the water with the cavity change almost not at all as pressure increases. The experiment indicates this because the structure changes so little. The simulation is able to measure these interactions directly, showing that $\exp(-\beta\mu_N^{ex})$ defined in Equation 6.11 changes very little as a function of pressure. Thus we have isolated the interactions of the protein, and can study those interactions by changing the chemical potential of the surrounding water. This chemical potential can be determined by integrating the equation of state of water as a function of pressure: $\Delta\mu_{wat} = \int_{p_0}^{p_1} dp/\rho_{bulk}$. At 300 K and from 1 bar to 2 kbar, $\Delta\mu_{wat} = 1.4k_BT$ based on experimental data[156] and simulation. This change is essentially linear with pressure at least up to this pressure.

That this small change of μ_{wat} results in such a dramatic filling of the cavity is remarkable, especially given that the free energy of inserting a water molecule into oil can be as much as $7\,k_BT$[79]. Unlike the liquid oil we begin with an open cavity, but the primary contribution to this large energy is that due to lost hydrogen bonds of the now lone water molecule. The simulation gives us some hints as to how water in the cavity compensates for this loss. First the water molecules themselves are hydrogen bonded to each other. This accounts for the cooperative nature of the transition observed in the simulation, an important feature we must bear in mind. Previous simulations have shown that this alone is not enough until the cavity is substantially larger, where it is possible for the water molecules to make many more hydrogen bonds [72]. The simulations presented here indicate that the protein interacts strongly with the water molecules. Both of these simulation findings suggest that further experiments, particularly higher pressure experiments, must be done to complete our understanding of this system. Ideally the experimental pressure range would be extended to at least 5 kbar, where the MD simulations indicate the cavity should be completely full with water.

Chapter 7

Strongly Interacting Protein Interiors

7.1 New thoughts on protein structure and folding

In the Introduction we saw how modern hydrophobic models can be constructed, and I introduced the liquid hydrocarbon model of protein folding. While this model did quite well on some points, it failed utterly on others. At this point, we would do well to ask what we really believe about protein structure. Specifically, do the free energies of transferring nonpolar amino acids from oil to water have anything to do with protein structure? Despite our complaints, the answer still must be yes. It is no accident that protein structure prediction, using potentials based on the transfer of amino acid side chains from oils to water, has been so succesful in recent years. Similarly, it is clear from cavity studies[47–49, 69] that part of the change in stability due to the cavity-forming mutation comes from a difference in these transfer free energies.

How are we to reconcile the difficulties of the hydrophobic model with its successes?

The answer to which much of the work points is that while the protein does fold to sequester "hydrophobic" side chains from bulk water, once folded the picture changes considerably.

7.1.1 Unfolding under pressure

Using the information theory model and simulations described above, Gerhard Hummer *et al.* calculated the pressure effects on the potential of mean force (pmf) between hydrophobic solutes in water. They found that pressure favored the solvent separated minimum, which led them to suggest that pressure unfolding was due to extensive internal hydration of a protein. That is, instead of completely solvating each hydrophobic group with water, comparitively small numbers of water a forced into a protein. The volumes of transfer of hydrophobic species from oil to water would no longer apply directly to proteins, and the hydrophobic model of folding could be reconciled with the pressure-unfolding data.

A number of problems remain. The apparent principal result of Hummer's paper is not in fact new. They calculate that pressure favors separation of methane dimers or trimers and intercalation of one water molecule. The volume associated with dissociation is therefore negative, in agreement with transfer of hydrophobic solutes from their neat liquids to water (see, for instance, Harvey Kliman's thesis[157]). Kauzmann's paradox is not that the sign of the volume change of protein unfolding disagrees with that of hydrophobic transfer, it is that the magnitude is wrong. The problem is that at low pressures, the magnitude of hydrophobic transfer volumes for small molecules is comparable to that of whole proteins.

But Hummer's results have a hidden answer to this problem. The volume change upon intercalating water between methanes works out to be small, on the order 1-2 ml/mol, and is smaller for trimers of methane than for dimers. The second part of the answer is in the understated existence of a solvent separated minimum in the pmf. It is not at all clear *a priori* that any such minimum should exist, but it turns out to be central to the prediction that the pressure-unfolded state should involve intercalated water in the protein rather than full solvent exposure of hydrophobic residues.

Experimentally, there has been no direct evidence for such a solvent-separated state, and so we have not be able to verify that the creation and water-filling of cavities is the mechanism of protein unfolding under pressure. The situation in a protein has clear differences from the original calculations of pmfs between methane molecules in water. Will the solvent-separated minimum persist in a protein where intercalating water cannot make contact with the bulk? We have shown that such a state can exist, and we can predict at what pressure it would appear. The following argument is necessarily incomplete. The reader should take it as a suggestion of how one might proceed rather than as a real hypothesis. The calculation also suggests what experiments might be done next. In the next section, we'll encounter a protein, bacteriorhodopsin, well suited to such experiments.

Let us presume that pressure unfolding involves the formation and subsequent water-filling of cavities within a protein. We want to calculate the free energy of creating a cavity under pressure and filling it with water at some density. We'll begin from three pieces of information. We know that most small proteins (order 100 residues) have about 1 % cavity volume, totalling about 250 Å^3 in roughly 7 separate cavities. Cathy Royer has pointed out a correllation between total cavity volume and unfolding volumes of proteins[7], suggesting that these cavities may fill with water. However, most will be too small to contain even one water molecule, and certainly not the three or four water molecules that seem to make this filling favorable. In bacteriorhodopsin, water is hypothesized to be necessary for proton pumping[158], but again there is no single cavity which can contain a water molecule. Simulation and crystallographic evidence suggests that cavities merge via conformational changes and side chain motions[159], producing a single larger cavity which can be filled with water. Thus in all likelihood a cavity of the necessary size for filling is created not by opening up an entirely new cavity, but by connecting and expanding pre-existing cavities.

We seek a final cavity volume V_{cav}. I will denote the fraction of *new* cavity volume by f, so we need to create new cavity volume totalling fV_{cav}. There will be a loss of "hydrophobic contact" energy, estimated from Eriksson *et al.*[47, 48] as about 90 J/mol·Å^3. This probably does not include increases in side chain entropy, since it appears that the side chains lining the cavity do not have substantially more conformational freedom than in the wild-type protein. Probably this is not the case for cavities which appear in a protein as it unfolds, so we probably overestimate the cost of creating a cavity in the protein. The cost of opening up a new volume fV_{cav} in each copy of the protein under pressure p is found from integrating $v\mathrm{d}p$ over the pressure range of interest; If we consider the difference in free energy of the closed ($V = (1 - f)V_{cav}$) and open (V_{cav}) states, and assume their volumes are fixed as a function of pressure, this yields a molar free energy difference of pN_AfV_{cav}, where N_A is Avagadro's number. Thus the free energy of opening a cavity is

$$\Delta G_{cav} = (90\,\mathrm{J/mol}\cdot\text{Å}^3 + pN_A)fV_{cav}. \tag{7.1}$$

Next we must consider the free energy difference between the empty and filled cavity states, which I'll write as the difference in excess chemical potentials of water,

$$\Delta\mu^{ex}_{wat} = \mu^{ex}_{wat,cav}(N, V_{cav}) - \mu^{ex}_{wat,bulk}(p). \tag{7.2}$$

The chemical potential in the cavity is not a function of pressure directly, but instead a function of the number N of water molecules present and the cavity volume. The chemical potential of bulk water is here given only pressure dependence. Now we will have to make a substantial approximation. We do not know the volume dependence of $\mu^{ex}_{wat,cav}$, and we barely know its dependence on N. Thus, for the time being, we will approximate that

$$\Delta\mu^{ex}_{wat} = \left.\frac{\partial\mu^{ex}_{wat,bulk}}{\partial p}\right|_{P=2\,kbar}(2\,\mathrm{kbar} - p) \tag{7.3}$$

which roughly models our experimental data. The pressure derivative of $\mu^{ex}_{wat,bulk}$ is roughly 1.8 kJ/mol·kbar. (This can be derived from pressure-volume data in the Steam Tables[156]). Finally, we will approximate that each water needs 40 Å^3 to be in its most favorable state, based on our cavity volume and the molecular dynamics result that a water tetramer is the most likely occupant of our cavity at 2 kbar. Then we can write $V_{cav} = 40\,\text{Å}^3 N$. Note that this is a rough approximation at best, but it is meant to suggest what can be learned from experiments like ours.

Putting together all of the equations above, we arrive at the following condition for the midpoint of cavity formation and filling (or unfolding):

$$0 \approx N\left[3.6\,\mathrm{kJ/mol}(1+f) + p(2.4\,\mathrm{kJ/mol}\cdot\mathrm{kbar}\times f - 1.8\,\mathrm{kJ/mol}\cdot\mathrm{kbar})\right]. \tag{7.4}$$

This equation has a number of interesting features. First of all, the unfolding pressure does not depend on N (or V_{cav}). This is probably incorrect, and is a consequence of the approximation in Equation 7.3. However, if one value N dominates for each V_{cav} then this approximation is not unreasonable.

Most interestingly, the protein cannot unfold at all unless $f \lesssim 0.75$. In fact, it appears that f must be less than about 0.52, or the unfolding pressure will exceed 10 kbar, at which point water will freeze. Therefore substantial void volume in a protein is combined into larger cavities upon unfolding. We can then estimate the pressure of unfolding for different values of f; it is plotted in Figure 7.1.

As we can see from the figure, the estimates of unfolding pressures are quite high. This can be traced to the approximation in Equation 7.3 and possibly also to the fact that the cavity entropy is ignored. Loss of native contacts upon opening

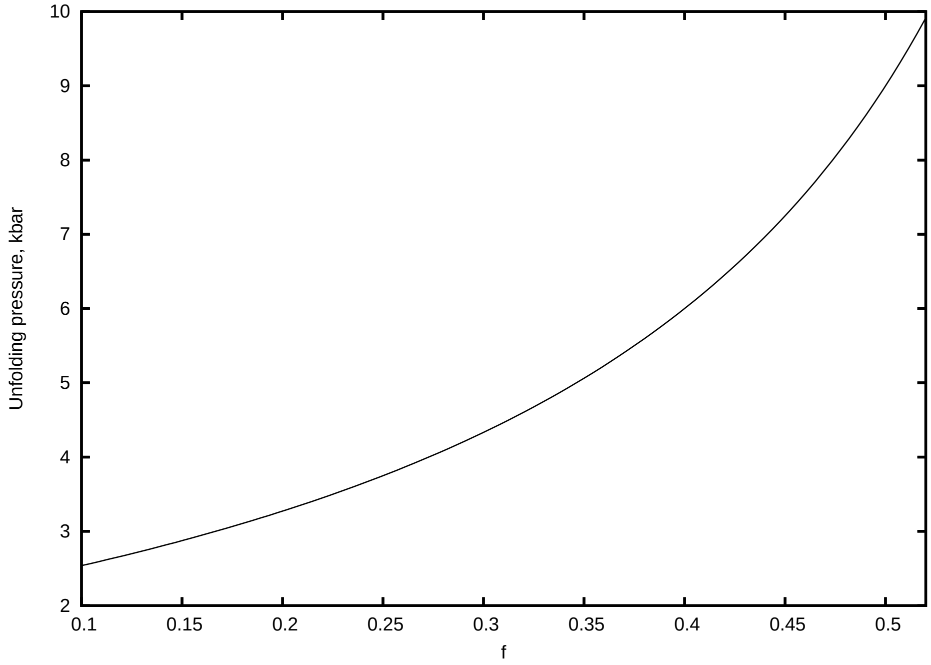

Figure 7.1: Unfolding pressures calculated from a simple model. f is the fraction of new cavity volume which must be opened up to create water-fillable cavities. Note that above 10 kbar (1 GPa) water freezes at ambient temperature. The model is described in the text.

up wholly new cavity volume should increase the side chain entropy. (Though it should be noted that in a folded protein, the presence of a cavity need not liberate its neighboring side chains. This fact will be discussed in Section 7.1.3.)

A reasonable estimate of the side-chain conformational entropy lost on folding from a random coil state is about 0.5 kcal/mol·K[160], or 0.6 kJ/mol at 300 K. Fourteen amino acids line the L99A cavity. In all likelihood the entropy change is not so large for cavity creation as it is for unfolding. Nonetheless, at $f = 0.5$, every increase of side chain entropy by 0.6 kJ/mol decreases the unfolding pressure by 1 kbar, according to Equation 7.4.

Another weakness of this calculation is that it does not include any estimate of the costs of creating new cavity volume beyond the loss of van der Waal's interactions. Displacements and rotations of intact helices may allow for such cavity formation without large energetic costs[159], and could be entropically favorable.

In spite of these weaknesses, the calculation is quite successful. For a small protein with a volume $\approx$ 20,000 Å^3, and having 1 % cavity volume already present, its resting cavity volume is *larger* than the unfolding volumes of most small proteins[7]. Since we therefore do not need to create much new cavity volume (if any) in the unfolding process, small values of the parameter f are not entirely unreasonable, and we predict unfolding pressures on the order of a few kilobar. The suggestion that the small unfolding volumes of proteins primarily reflects preexisting cavity volume is not new[7], and we see here that it is consistent with Hummer's predictions discussed above.

This model highlights the need for further experiments at higher pressures, over a range of cavity sizes, and the need for better measurements of side-chain and water entropy. Experiments like those detailed in Chapters 5 and 6, performed on other cavity mutants should be able to provide more information that would improve the model. Particularly interesting would be high-pressure experiments on cavity-mutants of T4 lysozyme where the structure partially relaxed in response to the mutation. Here we might hope to reopen the cavities, directly demonstrating that water could force its way into a protein structure and in the process rearrange the available cavity volume.

7.1.2 Buried water and protein function

One of the lessons of the previous section is that hydrophobicity is not always what it seems. As we learned from Chapter 6, the protein interior is not nearly as unfavorable to water as oily liquids. The reason for this appears to be van der Waal's interactions with the cavity walls and dipole interactions between water and the peptide electrical dipole, which can be quite strong.

This may seem to be a purely academic excercise, but water is in many cases central to the biological function of proteins, and has been proposed to be present in hydrophobic cavities of important metabolic proteins[5, 90, 161]. We'll consider two proteins in this class, cytochrome-*c* oxidase, and bacteriorhodopsin.

Cytochrome-*c* oxidase is an essential part of the metabolic pathway of most aerobic organisms[162]. It transfers electrons from cytochrome-*c* to one of its two heme groups and produces water in a redox reaction as well as pumping multiple protons across the membrane. The resulting electrostatic gradient is used to generate ATP for use in the cell[161].

With all proton pumping proteins, the central questions are along what path the protons will travel, and how the reaction is gated or switched. In the case of cytochrome oxidases, the problem is yet more complicated, as several successive electron and proton transfer reactions must take place for the protein to function properly[161, 162]. As it happens there are two proton conduction pathways, dubbed the D and K pathways for key residues along the two routes. One of these, D, begins at a Aspartic acid on the cytoplasmic side of the membrane, and proceeds along a now well-understood water and polar-residue path in a cavity, ending just short of the two heme groups at a glutamic acid group and a hydrophobic cavity[161, 162]. Where it should go from there is not immediately clear.

One suggestion has been that water fills the hydrophobic cavity[90, 161]. In the most recent iteration[161], a detailed model is proposed which accounts for water formation by dioxygen reduction and explains the pump mechanism as well.

Amazingly, the cavity water acts as a wire for protons as well as electrons, and is actively switched between two important pathways connecting the cytoplasmic side to the periplasmic side and to the "binuclear" heme a3.

It should not be necessary to consider the details of the model here, but rather we should ask how our experiment reflects on this proposed mechanism. A number of important questions are raised which we can answer reasonably well with our data.

Before now it has been experimentally unclear whether such a model made any sense at all. The cavity is almost completely hydrophobic, with only one oxygen (Gly 286 O in PDB code 1M56) lining the cavity, in this sense identical to the cavity of L99A T4 lysozyme. It is considerably smaller in volume, and much flatter than the L99A T4 lysozyme cavity, yet 3-4 water molecules are proposed to be inside[161]. Even by Buckle's criteria, which suggest that most cavities thought of as hydrophobic actually have hydrogen bonding sites for each water molecule present[69], it still would seem unlikely that 4 four water molecules could be present.

Our data show, in agreement with computer simulations[161], that it is quite possible for water to be present in such a cavity, provided that the environment is sufficiently polar to interact with water dipoles. Furthermore, our data indicate a cooperative effect, so that it is easier for multiple water molecules to enter a hydrophobic cavity together than one at a time. This cooperativity dovetails nicely with the concept of water chains as proton "wires". The implicit inter-water hydrogen bonding provides a pathway for conducting protons.

A key question then is how the water-carried proton pathway is switched. For cytochrome-*c* to function properly, it must first conduct an electron from heme a to heme a3, conduct a proton from the cytoplasmic side to the δ-propionate of heme a3, and finally conduct a proton from the cytoplasmic side to the binuclear Fe-Cu center of heme a3. The cooperativity is well suited to producing conducting wires, but something else must switch the conduction state.

From the simulation and observation of water in the T4 lysozyme cavity, it appears that hydrogen bonding between water molecules is not enough to stabilize water in a hydrophobic cavity. Electrostatic interactions with the protein are necessary to make water filling favorable. In cytochrome-*c* oxidase, this will also be the case, but the charge environment continues to change as the molecule moves through its functional cycle. That changing electrostatic environment around the water may be what switches the orientation of buried water, leading to the observed proton-pathway switching behavior, as has been suggested previously[161].

It would, however, be a stretch to say that our experiment confirms this model of proton-pumping, oxygen reduction, and switching behavior in the D-pathway of cytochrome-*c* oxidase. The model of proton-transfer in the cytochromes is already validated by the same sort of molecular dynamics simulation we have used to understand the energetics of cavity hydration. Those simulations and this thesis suggest a series of experiments to determine the effects of changing the electrostatic environment outside of a cavity.

The bacterial photosynthetic protein bacteriorhodopsin (bR) is, like the cytochrome oxidases, a proton pump which generates a proton gradient for use in the synthesis of chemical energy for the cell[163]. The proton pathway is complex. A central chromophore, retinal, sits between seven transmembrane helices (Figure 7.2). The retinal is bound to the protein by a Schiff base linkage to Lys216, and reisomerizes upon absorbing light. A proton is released to Asp85 on the extracellular side, after which the Schiff base accessibility switches to the cytoplasmic (CP) side and is reprotonated from Asp96. Electrostatically this is far from trivial. The Asp96 has a high pK_a relative to the Schiff base, allowing it to hold onto its proton long into the photocycle, due to its highly hydrophobic environment ([158, 159, 164] and see Figure 7.3). Luecke *et al.*[164] suggested transient intercalation of one or more water molecules into this otherwise hydrophobic environment lowers the pKa of Asp96, allowing it to reprotonate the Schiff base. They also suggest[158] that this water stabilizes a non-proline kink in helix G which is important in the reisomerization of retinal. Buried water in a hydrophobic environment is also likely to be necessary to lower the free energy barrier for proton transfer to Asp96, believed to come from Asp38, some 10 Å away. The tunnel between these two residues is almost entirely lined by hydrophobic residues, and in several crystal structures is closed off from outside solvent[159].

We have already seen that intercalating water into buried, hydrophobic cavities is not especially costly. With the charged interactions that would be present during the photocycle of bR and the presence of some polar residues in the hydrophobic proton passageway, it is now not hard to imagine that water can favorably enter this protein during some parts of its photocycle. Two other points are worth note in this unusual system.

The hydrophobic environment surrounding Asp96 may provide a number of potentially useful characteristics for buried water. Since water will not strongly bond to anything in the cavity (except, perhaps, the Aspartic acid) the nonpolar residues provide insulation and specificity to the proton conduction. If the water is not attracted to the cavity walls, it should be rotationally flexible, and on average will find itself in the center of the cavity. Thus a *lack* of specific interactions

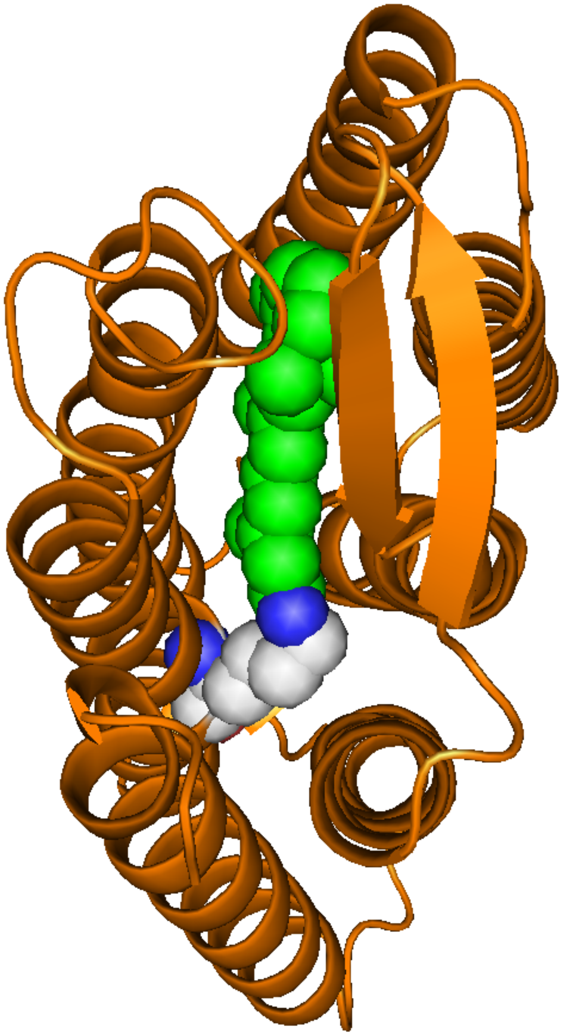

Figure 7.2: Structure of bacteriorhodopsin. View of the extracellular side.

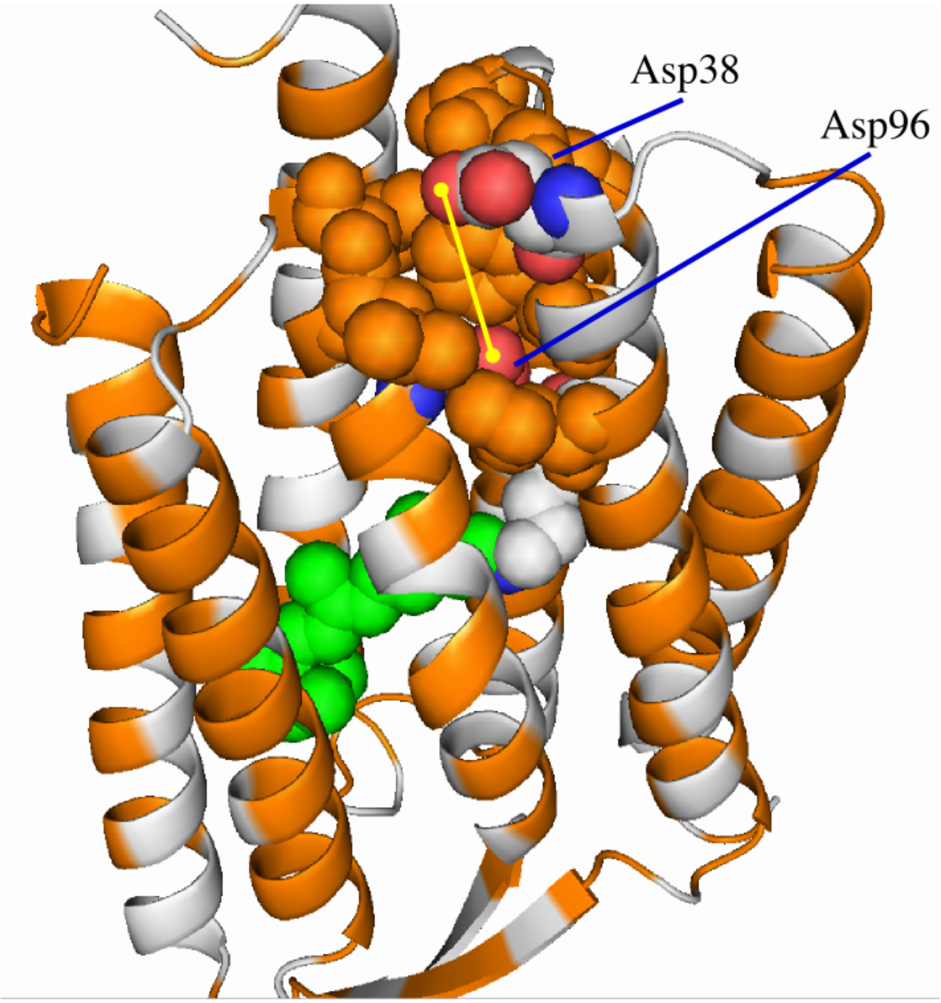

Figure 7.3: Cytoplasmic face of bR. Key charged residues and the retinal chromophore are shown as van der Waals spheres or sticks, and hydrophobic residues are colored orange.

can influence structure. If several water molecules are involved, their attraction for each other, and effective repulsion from cavity walls, will hold them in the correct orientation to act effectively as proton wires, much as they do in carbon nanotubes[73].

Even more interesting is that the direct path between Asp38 and Asp96 (shown in Figure 7.3 as a yellow line) is too narrow for water to be permanently in place[159]. In order for water to be present in the cytoplasmic channel of bR, conformational changes must expand or merge existing cavity volume. High-pressure spectroscopic measurements of bR photocycle kinetics do indicate positive activation volumes for each of the steps of the photocycle[5]. This may be consistent with water entering a hydrophobic cavity in the cytoplasmic side of bR.

The high-pressure kinetic observations, coupled with bR's need for a proton conduction pathway, and our understanding of water in hydrophobic spaces suggest an obvious experiment. If the model of bR is correct, when subjected to pressure, the protein should undergoe conformational changes which allow it to accomodate water in buried, mostly hydrophobic cavities. Such an experiment would provide a host of interesting data. Is such rearragement to accomodate water possible? Answering this question should yield more insight into the nature of pressure unfolding. What is the free energy cost of these structural rearrangements? We can perhaps learn more about the photocycle of bR through thermodynamic experiments. Answering these questions will help better understand what role buried water may play this protein, which remains a subject of debate[165].

The final protein to consider here is Staphylococcal nuclease. As mentioned above, measurements of pK_a for buried titratable residues indicated unexepectedly high dielectric constants. It has been unclear from the original evidence whether this was due to water buried in the protein[83]. Our data are unambiguous on this matter. Water should be easily buried even in a hydrophobic environment, especially in the presence of a titratable acidic residue.

7.1.3 Side chain flexibility

A major focus of debate about the hydrocarbon model of the protein interior[66] has been the balance between conformational flexibility and rigidity of proteins. Jie Liang and Ken Dill put it best: "Is a protein packed more like a liquid or a solid?[93]" F.M. Richards was first to point out that the average density of proteins was nearly that of amino-acid crystals[70]. Cyrus Chothia's work later indicated that the packing density in the core might be higher than that of crystals[166] (although this certainly makes clear that when we say "crystals", we are not speaking of hexagonally close packed lattices and the optimal packing densities of spheres.) Thus we might be led to believe that the core of a protein is solid.

On the other hand, proteins are frequently tolerant to an amazing array of mutations[93, 167, 168], with some caveats[168], suggesting that proteins are at least plastic in their folded forms.

More recent analytical techniques (see Section 5.2.3) and the vast increase in crystallographically resolved protein structures since Richards' original work have allowed better studies of amino acid side chain packing in proteins. The general conclusion that packing densities are high remains unchanged. The new result is that even with this high packing density, the distribution of void volume in proteins more closely resembles that of liquids[93].

Furthermore, the distribution of cavity sizes closely resembles that of random spheres near percolation thresholds[93]. This finding leads the authors to conclude that proteins are more like random spheres than a “jigsaw puzzle” of a well packed solid.

Another curious result of this recent study is that protein unfolding enthalpies per residue are virtually independent of the packing density of proteins[93]. This seems at odds with many of the cavity-mutation studies which indicate a significant role for non-specific van der Waals interactions between buried nonpolar residues (*e. g.* [47, 48, 69, 168]). The authors of the recent study[93] seem to feel that the packing density of the protein core is more an accident than anything truly important.

One issue we raised previously with the hydrophobic model is the apparent negative correlation between buried hydrophobic surface area and the enthalpy of unfolding. That is, as the protein interior becomes more polar, the folded state becomes more stabilized, apparently contrary to the hydrophobic model. One suggested explanation [67] was that this reflected the near close-packing of the protein interior, which would maximize van der Waals interactions, relative to the unfolded, solvated protein. Such an explanation implies that the polar interactions “stitch up” the interior to pull atoms closer together[67]. However even that explanation leaves something to be desired, since large amounts of buried polar surface area seem to increase conformational flexibility in at least some cases[169].

So which is it? Liquid, solid, or something else? And does this in fact tell us anything about protein stability or function?

The answer appears to be “something else”. The protein interior is (by crystal standards) disordered, but the atoms are still not free to diffuse throughout the interior. If we return to the discussion, in Chapter 5, of the comparative structural changes between the WT* T4 lysozyme and large-cavity containing mutant L99A, we see that the cavity makes almost no difference to the pressure response of the protein. Only one small region shows substantial differences, and that region sits in the least constrained electron density of the entire protein!

One might expect[94, 160] that the introduction of a cavity into the non-specifically stabilized protein core would increase fluctuations of side chains. With increased fluctuations, we expect increased compressibility (see Equation 1.4). If nothing else, for an anisotropic solid presumably stabilized by non-directional interactions, pressure ought to somehow deform a cavity like that in L99A T4 lysozyme. The total possible work energy of collapsing this cavity volume at 200 MPa is

$$2 \times 10^8\,\mathrm{Pa} \times 180 \times 10^{-30}\,\mathrm{m}^3 \times 6.023 \times 10^{23}\,\mathrm{mol}^{-1} \approx 22\,\mathrm{kJ/mol},$$

or about $8\,k_BT$! Under no circumstance can we call this a liquid. But does the protein's rigid response to compression tell us that it is like a crystalline solid?

Additionally the temperature factors of residues immediately lining the cavity are lower than anywhere else in the molecule (see Figure 5.4). This may provide a clue into the surprising rigidity of this region.

We can make no definitive statements about the origins of this rigidity, but we can speculate somewhat. To begin, let's ask what makes a structure rigid? The simplest incompressible structure consists of four identical hard spheres tetrahedrally packed (that is, three form an equilateral triangle, and the other is stacked in the interstitial dimple between the first three). No hydrostatic compression can reduce the volume of this system.

Another rigid structure without such well defined packing is a glass. Window glass is essentially random, and may contain large bubbles of air but be very resistant to shear forces. In a random packing of spheres, cavity collapse must be resisted by strong interactions between atoms (as those between silicon and oxygen atoms in the glass) which resist the inevitable anisotropic forces of compression along the cavity wall.

So two features which yield rigidity are large packing density and strong, directional interactions between atoms.

In a putatively hydrocarbon interior, there are no strong, directional forces, only van der Waals forces. A large cavity in a protein as that described by the picture of Liang and Dill[93] should most certainly collapse under pressure. There are two possibilities that may explain the observed rigidity. Probably the real answer to the puzzle is a combination of these two. They are summarized in Figure 7.4.

The first possibility is that the side chains have nothing to do with the observed rigidity. The cavity is contained in an entirely α-helical domain, and buried deeply. Helices are known to form cooperatively[67] and are stabilized by intrahelix backbone hydrogen bonds[25] so that they are fairly rigid even in solution. In addition the side chains generally lose flexibility upon helix formation, even for isolated helices[160]. In such a view we can picture helices as rigid cylinders with a perhaps somewhat soft outer coating. A bundle of these helices can only compress so far, regardless of whether we pack them together poorly.

Relaxing the constraint of rigid helices, another possibility is that the side chains themselves are interlocked and do not fluctuate individually, akin to suggestions by Mulder *et al.*[95]. Side chains are held in rough orientation merely by the topology of the surrounding helix backbone (see black and grey dots representing connections to the backbone in Figure 7.4). In this picture, cavity lining side chains may have some conformational flexibility, but cannot make subtantial

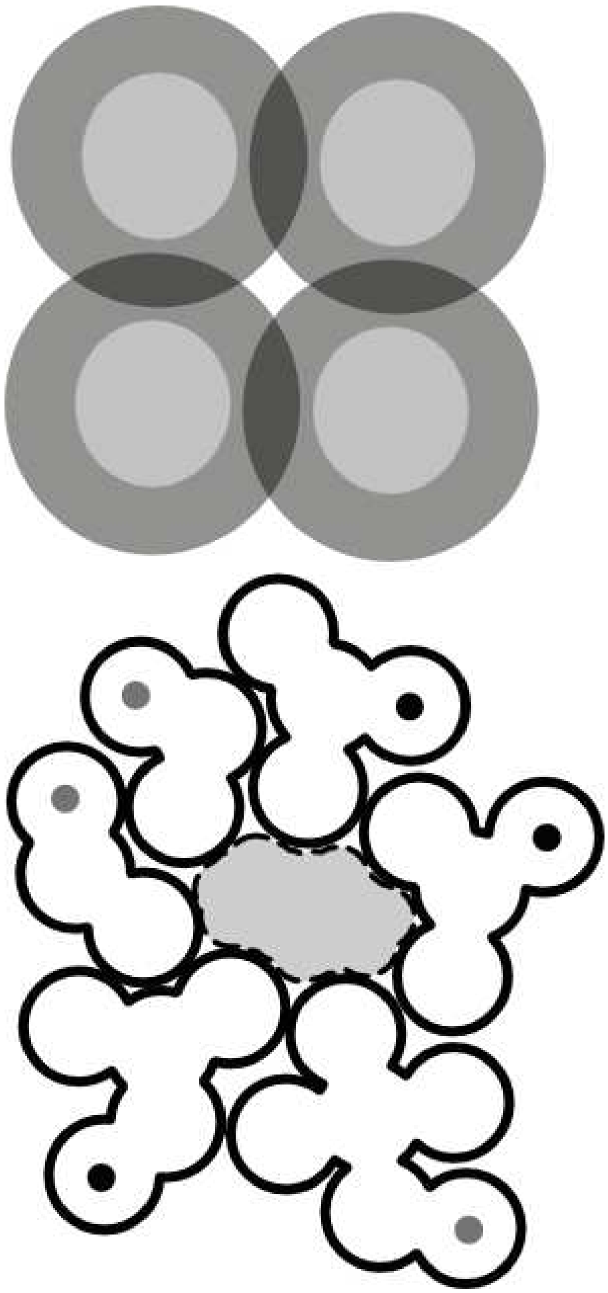

Figure 7.4: Structural basis for rigidity. (Top) Rigid cylinders, even with soft boundaries, can approach only to a minimum separation. Light grey: rigid core (helix backbone), medium grey: partially liquid side chains, dark grey: overlap regions resulting in excluded volume repulsion. (Bottom) Interlocked, densely packed side chains cannot move past each other if they are constrained (grey and black dots) not to rotate.

rearrangements without unfolding the protein. Here rigidity arises as a result of interlocked surfaces of atoms. The topology of the protein is important: without some constraints, the side chains in Figure 7.4 could collapse. Thus rigidity is a cooperative property between the surface roughness of each helix, and the rough topology of the backbone, but it need not depend on the rigidity of the backbone itself.

The obvious complaint with such a model as that in Figure 7.4, the cavity is *always* inaccessible to water. This is true, if proteins were two-dimensional. On the other hand, in three dimensions, pathways into cavities' rigid, interlocked structures are completely permissible. Consider C_{60}, fullerene. We can remove many of the carbon atoms, or even whole hexagons or pentagons of carbon, without affecting the overall rigidity of the molecule. Alternatively, consider a stone turret. The turret can of course have windows which open and close without affecting its overall rigidity.

A still somewhat unresolved problem in protein folding is whether secondary structure forms first (as a result, for instance, of the helical propensity for certain amino-acid sequences) or whether helix and sheet formation is a result of tertiary contacts[67, 170]. If helices form first, before they associate into tertiary domains, we might loosely expect that the first of our two possibilities for rigidity dominates. In this case, however, the crystallographic temperature factors of the side chains ought to be higher than we observe. Moreover, in the rigid helix picture, there is still room for the helices to slide past each other or rotate with respect to each other, something we observe in only one location (the D helix).

The best evidence for interlocked amino acid side chains giving rise to cavity rigidity may actually come from other parts of the protein. At least one helix is seen to bend substantially (the cavity end of helix C) in both the WT* and L99A proteins. As noted in Chapter 5, an impressive set of interatomic contacts appears to lock the middle of helix C in place, but its less sterically constrained ends are quite flexible. A helix's rigidity must involve side-chain steric constraints from *residues not part of that helix.*

Thus our data seem to indicate, albeit weakly, that interlocked side chains are partially responsible for this rigidity. This *may* be a sign of secondary structure formation stabilized by tertiary contacts made before the protein is fully folded. This is akin to the nucleation-condensation picture of protein folding[171] and the idea of constrained hydrophobic collapse[67]. In both pictures, hydrophobic contacts form a core around which helices or sheets form, driven by their intrinsic propensities and hydrogen bonding.

Unfortunately our data are weak on this subject. Higher pressures, where the deformations are more evident, may help tease out the nature of rigidity in this protein. NMR methods may be able to directly measure the dynamics of side chains and better determine the effects of the cavity mutation. For now, we can only speculate. Destabilizing cavity lining helices, and introducing further cavity-forming mutations into this region may help to separate out the two hypotheses of rigidity mentioned above. Such studies may help us to understand what drives the formation of helices and tertiary structure in this lysozyme.

7.2 The Janus face of water

The most robust conclusion that arises out of this thesis is that the behavior of nonpolar residues in the thermally unfolded state is different from that in the folded or even pressure unfolded state. This two-faced behavior is, once again, really a property of water and not so much of the protein itself.

There is a very close connection to the formation of clathrates in water. These highly ordered solvent cages around nonpolar solutes such as methane are not stable at ambient pressure. But with only mild pressures ($\lesssim 120\,\mathrm{bar}$), these clathrate-hydrate structures will form around atoms such as Krypton[58, 172]. The keys to this formation are the van der Waals interactions between water and the guest nonpolar molecule or atom. This interaction compensates for the loss of entropy upon forming the cage-like structure of water.

In the case of a cavity-containing protein, we again see that it is these same weak van der Waals interactions which ultimately stabilize water in "nonpolar" environments within the protein. Pressure perturbs the bulk water just enough to tip the balance. Both effects are predicted from the solvent-separated minimum in the potential of mean force between nonpolar solutes in water, as discussed earlier in this chapter.

There are some differences, of course. The clathrate-hydrate of a nonpolar guest retains its hydrogen bonded structure, and probably becomes even more hydrogen bonded, further lowering its enthalpy. Water in the L99A cavity cannot do this. Increased entropy, and possibly stronger van der Waals interactions, must compensate for this enthalpic loss. This loss of hydrogen bonding is also the reason that our cavity hydrates at pressures well above those at which clathrates form around small nonpolar solutes.

What is really new in our experiment is the demonstration that the context in which the water finds itself can have dramatic impacts on how these unusual features of water will manifest themselves. I cannot help but wonder whether a hydrophobic environment is in fact the best environment in which to harness the unusual electrostatic properties of water.

I have also shown that water is *constantly* moving in and out of proteins. If the presence of water in the L99A lysozyme at 2 kbar is favorable, and water can continue to fill the cavity up to this pressure, then the free energy penalty of water filling the cavity at ambient pressure is not large, and the cavity should be accessible at ambient pressure. Perhaps the kinetics of water passing in and out of the cavity change, but water is finding its way in and out of proteins, even into extremely hydrophobic regions. Is there in fact any reason to expect that there is only one pathway? We only observed one pathway in the simulations, but those simulations are only nanoseconds long, while we know that for compounds like benzene the timescale for entry into the cavity is millisecond[95]. It seems more likely that water is penetrating into many parts of the protein at all times. I have already examined some consequences of the presence of equilibrium water in proteins, but the consequences of transient water populations in proteins are potentially much farther reaching.

Until very recently, the explicit role of water in hydrophobic macromolecular assembly has been downplayed, often being treated through continuum electrostatic methods or by burying it in contact potentials between hydrophobic groups[23, 67, 78]. But it is increasingly clear that the specific, molecular level properties of water are important to protein structure and function, as evidenced by many recent theoretical studies[19, 21, 23, 59–61, 67, 72] This thesis provides an experimental verification that water behaves very differently depending on its context and environment, and suggests a way forward to quantify this behavior.

7.3 Future directions

By now it should be clear that pressure has allowed us to learn a great deal about proteins. As always, many questions remain open. What is the volume dependence of cavity filling by water? What is the entropy of filling a cavity with water? These questions can be answered straightforwardly, building on the work described in this thesis.

Measurement of the dielectric constant of protein interiors has remained challenging, but now we have a potential new method to determine this important parameter. Isosteric charged residue substitutions or changes in pH should lead to changing electrostatic environments in proteins, and in cavities inside the protein. This should, if our experiments are correct, change the cavity hydration behavior, but in a way which depends on the charge screening behavior of the material between the cavity and the charged substitution. High pressure cavity-filling experiments with other polar solvents, such as alcohols, or with deuterated water may also provide insight into the interactions between water and protein interiors.

Pressure unfolding theories need a next test, and that is to see whether pressure can force a cavity to grow by flooding it with water. Other mutants of T4 lysozyme whose cavities partially collapse from the WT* structure are excellent candidates for such studies, as is bacteriorhodopsin.

Finally, the amazing rigidity of T4 Lysozyme under hydrostatic pressure remains something of a mystery. Further experiments on the wide variety of cavity mutants of this protein would establish whether this is a general phenomena or a special case. Mutations along the cavity wall, and in the helices of the C-terminal domain, of the L99A mutant may also help to understand the interactions between side chains in the folded protein and what role they play in rigidity. Such experiments should help us to understand protein folding processes better. Theoretical methods, based on rigidity percolation and normal mode analysis, may also be useful.

Appendix A

Refinement scripts

A.1 Master refinement script

This script, `refine`, is called by higher level shell scripts which input specific parameters (such as the unit cell dimensions) for a given data set.

```
#!/usr/bin/perl -w

use File::Basename;
use strict;

# Refine generates then runs CCP4 tools to automatically
    refine protein
# protein structures from *.mtz data.

# Default parameters (ncyc, initial values, etc...)
my $sym = "'p3221'";
my $arp_sym = 'P3221';
my $freeRFrac = '0.05';
my $ncyc = 5; #number of cycles for initial refinement.

my $home;
my $tmp = "/tmp/marcus/";
if (-e "/home/marcus") {
     $home = "/home/marcus";
     print "It␣appears␣that␣you␣are␣using␣a␣Linux-like␣
         system\n\n";
} elsif (-e "/Users/Marcus") {
     $home = "/Users/marcus";
     print "It␣appears␣that␣you␣are␣using␣OSX.\n\n";
} else {
     print "I␣can't␣tell␣what␣file␣structure␣you␣are␣using
         ...␣quitting.\n";
     exit;
}

# Flag to skip rigid body refinement
my $skipRBR = 'no';

# First check for the "convert sca to mtz only" flag
```

```
my $convert_only = 'no';
if ($ARGV[0] eq '-so') {
  shift @ARGV;
  $convert_only = 'yes';
}

my $freeRfile = '${home}/rcsb/freeR.mtz';
if ($ARGV[0] eq '-fr') {
  shift @ARGV;
  $freeRfile = shift @ARGV;
}
print STDOUT "Using␣file␣$freeRfile␣for␣test␣set␣generation
   ␣(if␣converting␣from␣.sca)\n";

# Arguments in: location of the mtz file.
my ($fullFileName,$cellparams,$res,$pdbfile) = @ARGV
   [0,1,2,3];
my @res = split "␣", $res;
die "File␣does␣not␣exist:␣$fullFileName"
    unless -e $fullFileName;
my ($file,$workingDir,$type)
    = File::Basename::fileparse($fullFileName,qr{\.\w+});
#print "$file, $workingDir, $type \n";
my $ImportFlag = 'no';
if ($type eq '.sca') {
  $ImportFlag = 'yes';
} elsif ($type eq ".mtz") {
  print STDOUT "Recognized␣mtz␣file␣extension.\n";
} else {
  die "Cannot␣read␣file␣type␣$type!\n";
}


# Open any necessary files (note log files are opened
   implicitly later)
open RIG, ">␣${workingDir}rigid.com";
chmod 0777, $workingDir.'rigid.com';

open SUM, ">>␣${file}.sum";

# First import the data if necessary, and write an
   appropriate .mtz
if ($ImportFlag eq 'yes') {
    open COM, ”>␣${workingDir}tmp.com”;
    print COM ”#!/bin/bash␣\n”;
    print COM ”
scalepack2mtz␣HKLIN␣$fullFileName␣HKLOUT␣$tmp${file}.mtz.
   tmp␣<<eor

symmetry␣$sym
cell␣$cellparams␣90.0␣90.0␣120.0
resolution␣$res
scale␣1.0
anomalous␣–
␣␣␣␣NO
PNAME␣$file
DNAME␣$file
end
eor

truncate␣HKLIN␣$tmp${file}.mtz.tmp␣HKLOUT␣$tmp${file}.mtz.
   trn␣<<eor
truncate␣–
␣␣␣␣YES
anomalous␣–
␣␣␣␣NO
nresidue␣1
plot␣–
␣␣␣␣OFF
labout␣IMEAN=IMEAN_$file␣SIGIMEAN=SIGIMEAN_$file␣F=F_$file␣
   SIGF=SIGF_$file
falloff␣–
␣␣␣␣yes
PNAME␣$file
DNAME␣$file
RSIZE␣80
end
eor

mtzutils␣HKLIN␣$tmp${file}.mtz.trn␣HKLOUT␣$tmp${file}.mtz.
   inc␣<<eor
include␣F_$file␣SIGF_$file
end
eor\n\n”;
    unless (−e ”${freeRfile}”) {
        print STDERR "Generating new unique reflection set 
           and FreeR flags.\n\n";
        print COM "
unique HKLOUT ${home}/rcsb/unique.mtz <<eor
CELL $cellparams 90.0000 90.0000 120.0000
SYMMETRY 'P 32 2 1'
LABOUT F=FUNI SIGF=SIGFUNI
RESOLUTION $res[1]
end
eor

freerflag HKLIN ${home}/rcsb/unique.mtz HKLOUT ${home}/rcsb
   /freeR.mtz <<eor-free
FREERFRAC $freeRFrac
end
eor-free\n\n";
      }
      print COM "
cad HKLIN2 ${home}/rcsb/freeR.mtz HKLIN1 $tmp${file}.mtz.
   inc \\
HKLOUT $tmp${file}.mtz.cad <<eor-cad
LABI FILE 2 E1=FreeR_flag
LABI FILE 1 ALLIN
END
eor-cad

freerflag HKLIN $tmp${file}.mtz.cad HKLOUT $workingDir${
   file}.mtz <<eor-final
COMPLETE FREE=FreeR_flag
END
eor-final
";
     chmod 0777, $workingDir.'tmp.com';
     system "${workingDir}/tmp.com >> ${file}.imp.log";
}

die "Finished sca->mtz conversion and exiting.\n" if
   $convert_only eq 'yes';

print "Continuing on to refinement...\n";
print "Rigid body refinement using ${workingDir}rigid.com\n
   ";
```

```
print RIG "
#!/bin/bash

set -e

refmac5 HKLIN ${workingDir}${file}.mtz HKLOUT ${workingDir}
    ${file}.rig.mtz \\
XYZIN $pdbfile XYZOUT $workingDir${file}.rig.pdb << eor

LABIN FP=F_$file SIGFP=SIGF_$file FREE=FreeR_flag
LABOUT FC=FC PHIC=PHIC FWT=2FOFCWT DELFWT=FOFCWT

REFI TYPE RIGI RESI MLKF
REFI BREF OVER METH CGMAT
FREE 0

SCAL TYPE BULK
SCAL LSSC ANISOT
SCAL LSSC FIXB BBUL 150\n";
    print STDOUT "Warning: fixing BBUL at 150...\n";
print RIG "
MONI MEDI
RIGI NCYC 15
END
eor\n";
close RIG;

if ($skipRBR eq 'no') {
  system "${workingDir}rigid.com > rigid.log";

  print SUM "Results of rigid body refinement follow: \n";
  my $header = "\t Ncyc\tR\tRfree\tFOM\trmsBOND\trmsANGLE\
     trmsCHIRAL\n\n";
  print SUM $header;
  print $header;
  open IN, "< rigid.log";
  my $R;
  my $Rmax = 0.3;
  while(<IN>) {
    if (/^\s+(\d+)\s+(0.\d{3})\s+0.\d+/) {
      print SUM;
      print;
      $R = $2;
    }
  }
  close IN;
  if ($R > $Rmax) {
    print "R-value␣$R␣is␣higher␣than␣$Rmax.␣␣Reindexing.\n"
       ;
    open REIND, ">␣${workingDir}reind.com";
    chmod 0777, "${workingDir}reind.com";
    rename "$workingDir${file}.mtz", "$workingDir${file}
       _bad.mtz";
    print REIND "
#!/bin/bash
reindex␣HKLIN␣$workingDir${file}_bad.mtz␣HKLOUT␣
   $workingDir${file}.mtz␣<<eor
reindex␣HKL␣-h,-k,l
end
eor
";
    close REIND;

    system "${workingDir}reind.com";


# That isn't everything... need to re-refine with rigid
   body refinement...
    system "${workingDir}rigid.com␣>␣${workingDir}rigid2.
       log";
    open IN, "<␣${workingDir}rigid2.log";
    my $R;
    my $RMax = 0.3;
    print SUM "Values␣for␣reindexed␣data...\n";
    while(<IN>) {
      if (/^\s+(\d+)\s+(0.\d{3})\s+0.\d+/) {
        print SUM;
        print;
        $R = $2;
      }
    }
    close IN;
    if ($R > $Rmax) {
      print STDOUT "Unable␣to␣refine␣satisfactorily!␣
         Exiting␣NOW!\n";
      exit;
      }
    }

    print SUM "Check these carefully for convergence!\n";
}

# We can now begin full out restrained refinement.  Setup
    various important parameters
# and filenames first.

my $weight = 0.3;
my $ARPXYZIN = "${workingDir}${file}.rig.pdb";
my $ARPXYZOUT = "${tmp}water.pdb";
my $RM5XYZOUT = "${workingDir}${file}.pdb";
my $HKLIN = "${workingDir}${file}.rig.mtz";
my $RM5HKLOUT = "${workingDir}${file}.res.mtz";

# These parameters are used by ARP_waters to find water,
    and we want to
# display maps later in O using the same cutoffs, in order
    to evaluate their
# rationality and the success of ARP in finding water.

my $arp_cycles = 10;
my $cutFIND = 3.0;
my $cutREMOVE = 1.0;
my $cutN = 20;
my $findN = 20;


foreach my $cycle (1..$arp_cycles) {
    print STDOUT "Refining at weight $weight... Cycle: $cycle
        Set: $file\n";
    open RESTR, "> ${workingDir}restrain${cycle}.com";
    chmod 0777, "${workingDir}restrain${cycle}.com";

    print RESTR "
#!/bin/bash

# Maps for ARP_waters, which we'll run on the output of
    rigid body refinement first.

fft HKLIN $HKLIN MAPOUT ${tmp}diff.map.tmp <<eor
```

```
scale F1 1.0
scale F2 1.0
labin -
   F1=F_$file SIG1=SIGF_$file F2=FC PHI=PHIC
end
eor

mapmask MAPIN ${tmp}diff.map.tmp MAPOUT ${tmp}${file}.df.
    map  << eor
XYZLIM 0.0 1.0 0.0 1.0 0.0 0.1667
end
eor

fft HKLIN $HKLIN MAPOUT ${tmp}map.tmp <<eor
scale F1 2.0
scale F2 1.0
labin -
   F1=F_$file SIG1=SIGF_$file F2=FC PHI=PHIC
end
eor

mapmask MAPIN ${tmp}map.tmp MAPOUT ${tmp}${file}.map  << 
    eor
XYZLIM 0.0 1.0 0.0 1.0 0.0 0.1667
end
eor

arp_waters XYZIN $ARPXYZIN XYZOUT $ARPXYZOUT \\
           MAPIN1 ${tmp}${file}.map MAPIN2 ${tmp}${file}.df
    .map <<eor
MODE update waters
SYMM $arp_sym
RESO $res
FIND atoms $findN chain W cutsigma $cutFIND
REMOVE atoms $cutN cutsigma $cutREMOVE merge 2.2
REFI waters
end
eor

refmac5 HKLIN $workingDir${file}.mtz HKLOUT $RM5HKLOUT \\
XYZIN $ARPXYZOUT XYZOUT $RM5XYZOUT <<eor

LABIN FP=F_$file SIGFP=SIGF_$file FREE=FreeR_flag
```

```
LABO␣FC=FC␣PHIC=PHIC␣FWT=2FOFCWT␣DELFWT=FOFCWT

FREE=0

WEIG␣MATR␣$weight

REFI␣TYPE␣REST␣RESI␣MLKF
REFI␣BREF␣ISOT␣METH␣CGMAT

SOLV␣YES

SCAL␣TYPE␣BULK
SCAL␣LSSC␣ANIS
SCAL␣LSSC␣FIXB␣BBUL␣200

MAKE␣HYDR␣N
MONI␣MEDI
NCYC␣$ncyc
END
eor


”;
  close RESTR;

  system ”${workingDir}restrain${cycle}.com␣>␣${workingDir}
     restrain${cycle}.log”;
  open IN, ”<␣${workingDir}restrain${cycle}.log”;
  my ($n, $curR, $curRfree) = (0,1,1);
  print SUM ”Restrained␣refinement␣results␣for␣weight␣
     $weight␣cycle␣$cycle\n”;
  print ”Results␣for␣weight␣$weight␣\n”;
  while(<IN>) {
    if (/^\s+(\d+)\s+(0.\d{3})\s+(0.\d+)/) {
      print SUM;
      print;
      if ($2 < $curR){
        $n = $1;
        $curR = $2;
        $curRfree = $3;
      }
    }
  }
  close IN;

  # change the input files for fft/mapmask/arp_waters
  $ARPXYZIN = "$RM5XYZOUT";
  $HKLIN = "$RM5HKLOUT"

}

# It turns out that the output of Refmac 5 doesn't produce
   water molecules that O can
# read easily.  It is unclear why this should be, since
   Refmac 5 can read them just fine.
# (At least, it does refine the new water molecules'
   positions...)  In any case, we'll use
# pdbset to fix the problem.

open PDBSET, ">␣${tmp}${file}set.com";
chmod 0777, "${tmp}${file}set.com";
print PDBSET "#!/bin/bash

pdbset␣XYZIN␣$RM5XYZOUT␣XYZOUT␣${tmp}${file}tmp.pdb␣<<eor

replace␣-
␣␣␣␣residue␣HOH␣by␣WAT
replace␣-
␣␣␣␣atom␣\"OW0\"␣by␣\"␣O\"␣in␣WAT
replace␣-
␣␣␣␣atom␣\"OW0\"␣by␣\"␣O\"␣in␣HOH
end
eor

mv␣${tmp}${file}tmp.pdb␣${workingDir}${file}.pdb";

close PDBSET;
system "$tmp${file}set.com";

# Finally, make some maps, cut any junk out of the pdb file
    if it isn't actually in
# density, using ARP.
open MAP, ">␣${workingDir}map.com";
print MAP "
#!/bin/bash
```

```
fft HKLIN $RM5HKLOUT MAPOUT ${tmp}diff.map.tmp <<eor

scale F1 1.0
scale F2 1.0
labin -
   F1=F_$file SIG1=SIGF_$file F2=FC PHI=PHIC
end
eor

mapmask MAPIN ${tmp}diff.map.tmp MAPOUT ${workingDir}${file
    }.df.map  \\
XYZIN ${workingDir}${file}.pdb << eor
BORDER 5
end
eor

fft HKLIN $RM5HKLOUT MAPOUT ${tmp}map.tmp <<eor

scale F1 2.0
scale F2 1.0
labin -
   F1=F_$file SIG1=SIGF_$file F2=FC PHI=PHIC
end
eor

mapmask MAPIN ${tmp}map.tmp MAPOUT ${workingDir}${file}.map
     \\
XYZIN ${workingDir}${file}.pdb << eor
BORDER 5
end
eor";
chmod 0777, $workingDir.'map.com';
system "${workingDir}map.com > ${workingDir}map.log";

# And how about an .omac script to load important files?
# Note that fm_f has two extra "no" statements—these are
    dummies
# to avoid having to put in symmetry files. O just goes on
    , saying,
# "well, dork, I don't know what 'no' means here."
my $molname = $file;
$molname =~ s/[t_]//g;
open OMAC, "> $workingDir${file}.omac";
```

```
print OMAC "fm_z diff\n";
print OMAC "fm_z norm\n";
print OMAC "pdb_r $workingDir${file}.pdb $molname y y\n";
print OMAC "fm_f $workingDir${file}.df.map diff ;\n";
print OMAC "fm_s diff 15.0 solid 2 -$cutFIND red $cutFIND green\n";
print OMAC "fm_d diff\n";
print OMAC "fm_f $workingDir${file}.map norm ;\n";
print OMAC "fm_s norm 15.0 solid 1 $cutREMOVE white\n";
print OMAC "fm_d norm\n";
print OMAC "ce_at A99 CB\n";

close OMAC;
print "Thanks for refining with us!\n";
```

A.2 Calling scripts

This script calls the master refinement script for the L99A mutant datasets. A similar script exists for WT* data.

```
#!/bin/bash

MUTAMB=/Users/marcus/rcsb/1L90.pdb
RCSB2k=/Users/marcus/HPt4/final/mt2kstart.pdb
RCSB1k=/Users/marcus/HPt4/final/mt1kstart.pdb
RCSB0k=/Users/marcus/HPt4/final/mt0kstart.pdb
BASE=/Users/marcus/HPt4

echo 'Refining Mutants at 2, 1.5, 1, and 0 kbar'
refine $BASE/mt2k_8/mt2k_8.mtz "60.594 60.594 95.646" "30.0
   2.1" $RCSB2k
refine $BASE/mt2k_1/mt2k_1.mtz "60.604 60.604 95.708" "30.0
   2.1" $RCSB2k
refine $BASE/mt2k_3/mt2k_3.mtz "60.627 60.627 95.600" "30.0
   2.2" $RCSB2k

refine $BASE/mt1_5k_1/mt1_5k_1.mtz "60.680 60.680 95.805" "
   30.0 2.1" $MUTAMB

refine $BASE/mt1k_9/mt1k_9.mtz "60.755 60.755 96.052" "30.0
   2.1" $RCSB1k
refine $BASE/mt1k_6/mt1k_6.mtz "60.760 60.760 96.070" "30.0
   2.1" $RCSB1k
refine $BASE/mt1k_7/mt1k_7.mtz "60.768 60.768 96.077" "30.0
   2.15" $RCSB1k

refine $BASE/mt0k_1/mt0k_1.mtz "60.967 60.967 96.581" "30.0
   2.4" $RCSB0k
refine $BASE/mt0k_a/mt0k_a.mtz "60.954 60.954 96.607" "30.0
   2.4" $RCSB0k
refine $BASE/mt0k_b/mt0k_b.mtz "60.956 60.956 96.597" "30.0
   2.4" $RCSB0k
```

Appendix B

Structure analysis scripts

This chapter lists the code used to compare structures determined from refinement. It relies heavily on several Perl modules available from `www.cpan.org`, in particular the Perl Data Language. The first two files listed are Perl *modules*, which are simply dynamically loaded libraries of useful functions.

All of this code is self documented in the Perl POD format. If the script is in the current path, typing `perldoc <script>` at the command line will print documentation on the file.

This set of tools, and in particular the libraries, can be put together in many ways to achieve many otherwise difficult goals. I have written dozens of scripts based on these, more than space here will allow. However, these scripts are the core of the analysis in Chapter 5 and should provide a clear example of how to perform many other tasks. One such example is the final script in this appendix, which generated many of the figures used in the text.

Note: In order to typeset the code in a reasonable fashion, some lines have been split. This may result in errors, though in most cases it should not. Should you copy this code verbatim from this manuscript, be aware that there may be some difficulty in running it.

B.1 pdbtools.pm

This module contains many useful routines for reading, writing, and manipulating Protein Data Bank formatted files. The code can be hard to follow, and requires a good working knowledge of reference handling in Perl. Perl novices are advised not to modify this code, especially the function `vet_pdb`.

```
#!/usr/bin/perl -w
```

```
=pod DWARF Protein Structure comparison utilities: pdbtools
   .pm

=head1 Overview

pdbtools.pm is a package for reading, writing, sorting, and
    vetting PDB files.
It also has a number of interface functions which help it
   deal with PDL data
(aka "piddles" or "pdls"). These functions are documented
   below.

Marcus D. Collins Sept. 2004, marcus@bigbro.biophys.cornell
   .edu

Known oddities: read_pdb does not read in water molecules,
   nor beta sheets as secondary structure

Known bugs: sorting will not sort correctly (i.e. by
   residue first) if a numeral precedes an atom
type. This is almost never a problem, except when
   assigning hydrogen atoms, where certain programs
list the H atoms as, e.g., 1HE, 2HE, instead of HE1, HE2.
   This makes sense, as it is possible to
have, e.g. CD1 and CD2. Then you want to be able to write
   1HD1 and 1HD2.

The reason is that the atom code is <RESID><RESN><ATOMNAME
   >, so it can't␣distinguish
LEU,7,1HD1␣and␣LEU,71,HD1.

=cut

package␣pdbtools;

use␣PDL;
use␣Exporter;
@ISA␣=␣('Exporter');
@EXPORT␣=␣('read_pdb','write_pdb','vet_pdb','
   write_new_coords','write_field','get_coords');
@EXPORT_OK␣=␣('read_pdb','write_pdb','vet_pdb','
   insert_field','select_chain',
              'write_new_coords', 'by_atom','
    get_values_as_pdl','get_coords','write_field');

use strict;

=head1 Subroutines

=head2 read_pdb (I<input_filename>)

read_pdb takes one argument, which must be formatted 
    according to the PDB rules.
It returns a hash reference which I refer to as a pdb 
    reference.  The keys of
the hash %$pdb_ref are the following field names:

=over 4

=item

I<atom_num>, I<atom_type>  the atom number and type (e.g. 
    CG1) from the original PDB file

=item

I<res_num>, I<res_type>  the residue number and type

=item

I<chain_id>  the chain identifier

=item

I<x>,I<y>,I<z>  the coordinates of the atom from the PDB 
    model.

=item

I<B>, I<occup> the B factor and occupancy

=item

I<sort_keys>  a concatenation of chaid_id, residue number 
    and atom_type, this is used for sorting
```

```
the␣record.

=item

I<second>␣stores␣the␣second␣structure␣records.␣␣As␣of␣20␣
    DEC␣2004,␣only␣helices␣are␣actually␣stored.
Other␣records␣will␣be␣stored␣in␣the␣future.␣␣The␣helix␣
    identifier␣(e.g.␣H4),␣the␣beginning␣and␣ending
residue␣numbers␣are␣recorded.␣␣E.g.␣C<$pdb_ref->{"second
    "}[2][1]>␣is␣the␣beginning␣residue␣number
for␣the␣third␣helix␣listed␣in␣the␣original␣pdb␣file.

=back

With␣the␣exception␣of␣I<sort_keys>␣the␣values␣of␣%$pdb_ref␣
    are␣array␣references;␣%$pdb_ref{"sort_keys"}
is␣a␣hash␣reference.␣␣The␣arrays␣contain␣the␣values␣of␣each
    ␣field␣in␣the␣order␣they␣were␣read␣from␣the
file.␣␣The␣values␣of␣%{$pdb_ref->{"sort_keys"}}␣are␣
    sequential␣numbers␣starting␣at␣0,␣recording␣the
original␣read␣order␣of␣the␣lines␣in␣the␣pdb␣file.␣␣These␣
    are␣used␣later␣in␣vet_pdb␣for␣sorting␣purposes.

=head3␣Note␣on␣what␣is␣read␣from␣the␣file.

As␣of␣this␣version,␣only␣lines␣beginning␣with␣ATOM␣are␣read
    ␣in.␣␣It␣would␣be␣straightforward␣to␣read␣in
HETATM␣records␣in␣a␣similar␣fashion.␣␣read_pdb␣also␣does␣
    not␣read␣in␣chain␣identifiers.␣␣This␣could
cause␣serious␣problems␣with␣multiple␣chain␣proteins.␣␣Be␣
    careful,␣or␣modify␣this␣code␣and␣send␣your␣new
version␣to␣me!

=cut

sub␣read_pdb␣{
␣␣my␣($file)␣=␣@_;
␣␣my␣$debug␣=␣0;

␣␣open␣STRUCT,␣"<␣$file"␣or␣die␣"$file␣cannot␣be␣opened!\n
    ";
␣␣#␣Establish␣arrays␣that␣will␣be␣used␣in␣constructing␣the␣
    matrices␣and␣other␣stuff
```

```
  my (@x, @y, @z, @atom_type , @res_type , @chain_id , 
    @res_num , @B, @atom_num , @occ , @atonelet ) ;
  my %sort_keys ;
  my @chain_ids ; #This array will contain the chain 
    identifiers ,

  # The following entries are the beginning of an attempt 
    to keep more information
  # from the pdb record .  We will start with helices .
  # These will be accessible from the array $pdb_ref->{"
    second"} , whose value will be
  # the reference to the array @secondary , which itself is 
    an array of references to
  # arrays .  E.g. to get helix information ,
  #
  # $pdb_ref = read_pdb ($filename) ;
  # $pdb_ref->{"second"}[n][0 ,1 ,2];
  #
  # where n is the number of the secondary structure 
    element (starting from zero , read
  # in order of the pdb file ) , and 0 ,1 ,2 correspond to the 
    element name (e.g. H3) , the
  # first residue number of the element , and the last 
    number of the element (possibly more
  # for B-sheets ) .


  my (@secondary ) ;
  #$debug = 1;
  print "Reading PDB file now...\n" if $debug ;
  my $i = 0;
 READPDB: while (<STRUCT>) {
   if (/^HELIX\s+\d+\s+H?(\d+)\s\w{3}\s[A-Z]?\s+(\d+)\s+\w
    {3}\s[A-Z]?\s+(\d+)/) {
     my $ident = "H".$1;
     push @secondary , [$ident , $2 , $3 ];
     print "READ_PDB: $  \nSecondary : $1 , $2 , $3\n" if 
    $debug ;
     #$debug = 0;
   }

   #            at_num  at_name   res_name chain_id? 
    res_num coords ...............................   
    occupancy␣␣␣␣␣␣␣B
 ␣␣␣if␣(/^ATOM\s+(\d+)\s+(\w{1,4})\s+(\w{3})\s(\w?)\s+(\d+)\
    s+(-?\d+.\d+)\s+(-?\d+.\d+)\s+(-?\d+.\d+)\s+(\d+.\d\d)\s
    ?\s?(\d+.\d\d)/)␣{
     ␣␣␣␣␣next␣READPDB␣if␣(($3␣eq␣'WAT')␣||␣($3␣eq␣'HOH'));␣#␣We
    ␣do␣not␣like␣water!
     ␣␣␣␣␣my␣@line␣=␣split␣/\s+/,␣$_;
     ␣␣␣␣␣$atonelet[$i]␣=␣pop␣@line;

     ␣␣␣␣␣$atom_num[$i]␣␣=␣$1;
     ␣␣␣␣␣$atom_type[$i]␣=␣$2;
     ␣␣␣␣␣$res_type[$i]␣␣=␣$3;
     ␣␣␣␣␣$chain_id[$i]␣␣=␣$4;
     ␣␣␣␣␣$res_num[$i]␣␣␣=␣$5;
     ␣␣␣␣␣$x[$i]␣␣␣␣␣␣␣␣␣=␣$6;
     ␣␣␣␣␣$y[$i]␣␣␣␣␣␣␣␣␣=␣$7;
     ␣␣␣␣␣$z[$i]␣␣␣␣␣␣␣␣␣=␣$8;
     ␣␣␣␣␣$occ[$i]␣␣␣␣␣␣␣=␣$9;
     ␣␣␣␣␣$B[$i]␣␣␣␣␣␣␣␣␣=␣$10;
     ␣␣␣␣␣$sort_keys{$4.$5.$2}␣=␣$i;␣␣␣␣#This␣will␣serve␣as␣a␣
    lookup␣table␣for␣sorting␣and␣selection␣routines.
     ␣␣␣␣␣$i␣+=␣1;

     ␣␣␣␣␣#Put␣the␣chain␣identifiers␣in␣a␣list...

     ␣␣␣␣␣push␣@chain_ids,␣$4␣if␣($4␣&&␣!((defined␣$chain_ids
    [-1])␣&&␣($chain_ids[-1]␣eq␣$4)));
     ␣␣␣}
     ␣}
  ␣␣print␣"READPDB:␣Chain␣identifiers:␣@chain_ids\n";
  ␣␣warn␣"\nWARNING:␣PDB␣file␣$file␣appears␣to␣be␣empty!!\n\n
    "␣if␣$i==0;

  ␣␣my␣$pdb_hash_ref␣=␣{␣"atom_num"␣=>␣\@atom_num,
  ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"atom_type"␣=>␣\@atom_type,
  ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"res_type"␣=>␣\@res_type,
  ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"chain_id"␣=>␣\@chain_id,
  ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"res_num"␣=>␣\@res_num,
  ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"x"␣=>␣\@x,␣"y"␣=>␣\@y,
  ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"z"␣=>␣\@z,␣"B"␣=>␣\@B,␣"occup"␣=>␣\
    @occ,
  ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"sort_keys"␣=>␣\%sort_keys,␣"second"
    ␣=>␣\@secondary,
                      "chains" => \@chain_ids, "atonelet" 
    => \@atonelet };

  close STRUCT;
  bless $pdb_hash_ref, "pdbtools";
  return $pdb_hash_ref;
}

=head2 write_pdb (I<pdb_reference, output_file>)

write_pdb writes a pdb format file based on the structure 
    defined in read_pdb.

=cut

sub write_pdb {
    my ($strc_ref, $pdb_file, $model) = @_;
    print "Writing PDB format file: ${pdb_file}...\n";

    warn "WARNING: write_pdb requires a reference to a 
    structure object (hash): see pdbtools.pm\n"
        unless (ref($strc_ref) eq "pdbtools");
    if ($model) {
      open NEW, ">> $pdb_file" or die "Cannot open output 
    pdb file: $pdb_file";
      print NEW "REMARK\tThis pdb file was generated by the
     DWARF utilities.\n";
      print NEW "MODEL $model\n";
    } else {
      open NEW, "> $pdb_file" or die "Cannot open output 
    pdb file: $pdb_file";
      print NEW "REMARK\tThis pdb file was generated by the
     DWARF utilities.\n";
    }
    select NEW;

    print "REMARK\tThis pdb file was generated by the DWARF
     utilities.\n";

    my $num = 1;
    foreach (sort by_num values %{$strc_ref->{"sort_keys
    "}}) {
        print "ATOM";
        if ($$strc_ref{"atom_type"}[$_] =~ /^\d/) {
          #printf("%7d %-5s", $$strc_ref{"atom_num"}[$_],
    $$strc_ref{"atom_type"}[$_]);
          printf("%7d %-5s",$num, $$strc_ref{"atom_type"}[$_
    ]);
          $num++;
        } else {
          #printf("%7d  %-4s", $$strc_ref{"atom_num"}[$_],
    $$strc_ref{"atom_type"}[$_]);
          printf("%7d  %-4s",$num, $$strc_ref{"atom_type"}[
    $_]);
          $num++;
        }
        printf("%3s%6d", $$strc_ref{"res_type"}[$_],
    $$strc_ref{"res_num"}[$_]);
        printf("%12.3f%8.3f%8.3f", $$strc_ref{"x"}[$_],
    $$strc_ref{"y"}[$_], $$strc_ref{"z"}[$_]);
        printf("%6.2f%6.2f%8s", $$strc_ref{"occup"}[$_],
    $$strc_ref{"B"}[$_], $$strc_ref{"atonelet"}[$_]);
        print "\n";
    }

    print "ENDMDL\n";
    select STDOUT;
    close NEW;
    return 1;
}

=head2 insert_field (I<pdb_reference, labels_reference, 
    values_rref>)

insert_field is the mechanism by which to write into an 
    existing pdb
reference.  In principle it could be used to generate a 
    blank reference,
but this is probably ill-advised.  It is not exported.

I<labels_reference> is a reference to an array of labels, 
    which should match the labels
defined in read_pdb.

I<values_rref> is a reference to an array of references, 
    each of which points to an array
```

```
of values corresponding to each label in @$label_reference.

=cut

sub insert_field {
  my $debug = 0;
  my ($structure_ref, $labels_ref, $values_rref) = @_;
  my @labels = @$labels_ref;
  print "This is insert field.  Labels are: @labels\n" if 
    $debug;
  my @val_refs = @$values_rref;

  foreach my $label (@labels) {
    my $val_ref = shift @val_refs;
    print join " ", @$val_ref, "\n" if $debug;
    @{$structure_ref->{$label}} = @$val_ref;  # @{pdb field
     reference} = @{ref to new vals}
  }
  return $structure_ref;
}


# write_new_coords is what is usually called to interface 
    to insert_field
# It expects a dwarf structure reference (see read_pdb), 
    and a 3xN pdl.

=head2 write_new_coords (B<3xN pdl> I<coordinates, 
    pdb_reference>)

write_new_coords is an exported function which uses 
    insert_field to write
into an existing pdb_reference.  The coordinates must be a 
    3 rows x N columns
pdl.

=cut

sub write_new_coords {
    my ($coords, $structure_ref) = @_;
    my @x = $coords->slice(":,0")->list();
    my @y = $coords->slice(":,1")->list();
    my @z = $coords->slice(":,2")->list();
```

```
␣␣␣␣my␣@coords_ref␣=␣(\@x,\@y,\@z);
␣␣␣␣my␣@labels␣=␣qw/x␣y␣z/;
␣␣␣␣return␣&insert_field($structure_ref,\@labels,\
    @coords_ref);
}

=head2␣write_field␣(B<1xN␣pdl>␣I<data,␣field␣name,␣
    pdb_reference>)

This␣exportable␣function␣allows␣the␣user␣to␣write␣into␣any␣
    one␣field␣(e.g.␣occupancy,
B␣factor,␣atom␣name)␣one␣at␣a␣time.␣␣It␣is␣mostly␣an␣
    external␣wrapper␣for␣insert_field
that␣allows␣the␣user␣not␣to␣have␣to␣construct␣references␣
    needed␣for␣insert_field.

=cut

#Okay,␣you've decided to look under the hood, haven't␣you?␣
    ␣Well,␣yes,␣this␣one␣is
#wierd.␣␣It␣creates␣a␣lot␣of␣seemingly␣unnecessary␣
    references.␣␣But␣this␣is␣for␣compatibility
#with␣insert_field,␣which␣I␣decided␣long␣ago␣would␣be␣the␣
    one␣and␣only␣path␣for␣writing␣into␣a
#pdb␣reference.␣␣That␣way␣if␣the␣structure␣of␣a␣
    pdb_reference␣changes,␣I'll only have to fix it
#there, leaving the user interface alone.  vet_pdb will be
    enough work if I have to change things.
#insert_pdb has the multiple levels of references to handle
     the possibility of writing an
#arbitrary number of fields at once, as in write_new_coords
    .

sub write_field {
  my $debug = 0;
  my ($data,$label,$pdb_ref) = @_;
   print $data if $debug;
   print "This␣is␣write_field\n" if $debug;
  my @labels;
  my @dataref;
   push @labels, $label;
  my @data = $data->list();
   print "Write␣field:␣@data\n" if $debug;
  push @dataref, \@data;
  return &insert_field($pdb_ref,\@labels, \@dataref);
}

=head2 _get_values_as_pdl (I<pdb_reference, column name
   1,..., column name m>)

An internal function which returns a single mxN pdl
   containing each specified column of
data in a row (the PDL is row centric). Using this
   function outside of this module
is discouraged, as it assumes a format for the
   pdb_reference. It mainly wraps some
somewhat confusing reference passing into a convenient
   function.

=cut

sub _get_values_as_pdl {
    my ($structure_ref, @column_names) = @_;
    my $pdl_string = 'pdl␣['; #This will be what we feed
        to pdl in an eval statement...
     foreach my $column (@column_names) {
         $pdl_string .= "\[\@\{\$structure_ref->\{\"$column
            \"\}\}\],";
     }
     $pdl_string .= ']';
     return my $pdl = eval($pdl_string);
}

=head2 get_coords (I<pdb_reference>)

get_coords is an outside interface to _get_values_as_pdl.
   Please use this
unless you need something other than the coordinates.

=cut

sub get_coords {
  my $structure_ref = shift @_;
  return &_get_values_as_pdl($structure_ref,'x','y','z');
}
```

```
=head2 get_helices (I<pdb_reference>)

This routine is simple: it returns an array of strings,
   useable by vet_pdb,
to aid in the selection of helices.  The array can be
   manipulated as necessary
to recover whichever helices you like; use the ”CA” flag in
    vet_pdb if you
want only the backbone.

Incidentally, if you call it in scalar context, it will
   return all the strings
concatenated together as one.  Just a warning...

=cut

sub get_helices {
  my $structure_ref = shift @_;

  my @vet_strings;

  foreach my $helix (@{$structure_ref->{”second”}}) {
    if ($helix->[0] =~ /^H/i) {
      push @vet_strings, join ”-”, $helix->[1], $helix
          ->[2];
    }
  }
  wantarray ? return @vet_strings : return join ”␣”,
     @vet_strings;
}


=head2 vet_pdb (I<pdb_ref1, pdb_ref2, [things to keep]>)

vet_pdb is the most complicated part of this entire
   distribution, and is at
the cornerstone of its ability to compare potentially very
   different molecules.

B<Syntax:> vet_pdb expects two pdb references as defined in
    read_pdb.  After this
it expects an array of strings of one of two formats:
```

```
=over 4

=item

I<residue ranges> specified as (first residue number)-(last
    residue number).  vet_pdb will keep
all residues from the first to last residue specified.

=item

I<atom types> specified by their respective PDB codes.  The
    comparison to what's␣in␣the␣actual
structure␣is␣done␣with␣regex,␣but␣oddly␣something␣like␣C.*␣
   won't work.

=back

If nothing is specified, vet_pdb will return all atoms that
    match in both structures.  You may
specify as many restrictions as you like.

vet_pdb will then return a new pdb reference which contains
    only the specified atoms.

=head3 Algorithm

In an effort to head off unwary attempts at modifying this
   beast, some description of the algorithm
is in order.  Here is how it works:

The pdb_refs are held internally in s_ref1 and s_ref2,
   while the array to "things␣to␣keep" is called
@keep.

=cut

sub vet_pdb {
  my ($s_ref1,$s_ref2, @keep) = @_;

  my @res_nums;
  my @atom_types;

  my $new_ref1;
  my $new_ref2;
  my @keys_to_pick;
  my ($res_pattern, $atom_pattern);
  my @chains1;
  my @chains2;

# Check that the chain identifier lists are the same; if
    not don't exit
# but warn user

  @chains1 = @{$s_ref1->{"chains"}};
  @chains2 = @{$s_ref2->{"chains"}};
  print join "␣", "Chains␣for␣prot␣1␣(in␣order␣read):",
     @chains1, "\n";
  print join "␣", "Chains␣for␣prot␣2␣(in␣order␣read):",
     @chains2, "\n";
  if (@chains1 != @chains2) {
    print STDERR "WARNING\t\t\tWARNING\t\t\tWARNING\nNumber
       ␣of␣chain␣IDs␣in␣your␣pdb␣files␣do␣not␣match!\n";
    print STDERR join "␣", "1:", scalar @chains1, "\t2:",
       scalar @chains2, "\n";
    print STDERR "Program␣may␣produce␣erroneous␣results␣or␣
       crash!\nWARNING\t\t\tWARNING\t\t\tWARNING\n\n\n";
  } else {
    foreach (0..@chains1-1) {
      if ($chains1[$_] ne $chains2[$_]) {
        print STDERR "WARNING:␣The␣#$_␣␣chain␣identifiers␣of
           ␣your␣pdb␣files␣do␣not␣match!\n";
        print STDERR "WARNING:␣Program␣may␣crash␣or␣produce
           ␣erroneous␣results!\n";
      }
    }
  }


#=pod
#
#First two arrays are constructed, one listing every
    residue to be kept, and one which
#lists each atom type to be kept.  This is done with
    regular expression matching: if
#something matches the pattern /(\d+)-(\d+)/ it gets used
    for residue numbers, otherwise
```

```
#it is pushed onto the atom stack.
#
#=cut

  foreach (@keep) {
    if (/(\d+)-(\d+)/) {
      push @res_nums, $1..$2;
    } elsif (/(\d+)/) {
      push @res_nums, $1;
    } else {                                #if (/^[CNOS]\w*$
       /) {
      push @atom_types, $_;
           }
  }


=pod

Then vet_pdb constructs an array @keys_to_pick: the atom
   identifier codes that will be
matched to the ”sort_keys” described above.  If no residues
    or atom types are specified,
@keys_to_pick is set to C<keys %{$s_ref1->{”sort_keys”}}>.
    Otherwise, a regex
string is generated for the residues and for the atom_types
   , and matched against the
corresponding fields in (pdb_ref) $s_ref1.  If the array is
    undefined, then vet_pdb
uses ”.*” as the regex.  (It is thus somewhat redundant to
   check whether they both defined
earlier, but it speeds things up in certain situations, and
    is more robust).

The code ”SC” will override all other atom selections and
   stands for a pre-defined set of side chain atoms.
”MC” will represent main chain atoms, and takes precedence
   over SC.

The code SR means that any code that follows is a complete
   atom identifier.  This code takes precedence
over all others.

=cut
```

```
if (!@atom_types && !@res_nums){
  @keys_to_pick = keys %{$s_ref1->{"sort_keys"}};
} elsif ((join "␣",@atom_types) =~ m/SR/) {
  #print STDERR "Entering specific atom selection mode\n
     ";
  #print STDERR join " ", @atom_types, "\n";
  @keys_to_pick = @atom_types;
} else {
  (@res_nums) ? ($res_pattern = "(" . join(')|(',
     @res_nums) . ")" )
      : ($res_pattern = ".*");
  (@atom_types) ? ($atom_pattern = "(" . join('|',
     @atom_types) . ")" )
      : ($atom_pattern = ".*");

  if ($atom_pattern =~ m/MC/) {
    $atom_pattern = "(C)|(CA)|(N)|(O)";
  }
  if ($atom_pattern =~ m/SC/) {
    $atom_pattern = "(C[B-Z]\\d?)|(O.+)|(N.+)|(S.*)";
  }

  my $i = 0;

  foreach (keys %{$s_ref1->{"sort_keys"}}) {
    if ( ($s_ref1->{"res_num"}->[$s_ref1->{"sort_keys"
        }->{"$_"}] =~ m/^($res_pattern)$/) &&
          ($s_ref1->{"atom_type"}->[$s_ref1->{"sort_keys"
             }->{"$_"}] =~ m/^($atom_pattern)$/) ) {
      #print "Match: $_\n";
      $keys_to_pick[$i] = $_;
      $i += 1;
    }
  }
}



# This retrieves references to the "sort_keys" hash of
   the pdb records written
# by read_pdb (above). The keys to %sort_keys are a
   concatenation of the residue
  # number and the atom code for every ATOM record in the
      original pdb file.
  # The actual value is the atom serial number.
  my $key_ref1 = $s_ref1->{"sort_keys"};
  my $key_ref2 = $s_ref2->{"sort_keys"};


=pod

Next comes the real meat of the routine.  For each C<
    @keys_to_pick>, vet_pdb determines whether
that key exists in the second structure (by checking C<
    defined $s_ref2->{"sort_keys"}->{"$atom_identifier"}>.)
If it is defined, then both structures have that atom and
    it has passed the vetting restrictions
specified by the user.  Then the routine C<&_copy_pdb_line
    ()> is called to write a line of a new pdb_ref.

=cut

# For some reason, Emacs doesn't like pod comments...  I
    can't tell what is wrong,
# but the code works fine.

  my $i = 0;
  #print STDERR join " ",@keys_to_pick,"\n";
  foreach my $atom_identifier (sort by_atom @keys_to_pick)
     {
    if (defined $$key_ref2{"$atom_identifier"}) {
      my $index1 = $s_ref1->{"sort_keys"}->{"
          $atom_identifier"};
      my $index2 = $s_ref2->{"sort_keys"}->{"
          $atom_identifier"};
      $new_ref1 = &_copy_pdb_line($s_ref1,$new_ref1,$index1
          ,$i,$atom_identifier);
      $new_ref2 = &_copy_pdb_line($s_ref2,$new_ref2,$index2
          ,$i,$atom_identifier);
      $i += 1;
    }
  }
  #my @CH = qw/A/;
  #$new_ref1->{"chains"} = \@CH;
  #$new_ref2->{"chains"} = \@CH;
  bless $new_ref1, "pdbtools";
  bless $new_ref2, "pdbtools";
  return ($new_ref1, $new_ref2);
}

=head2 select_chain (I<pdb_ref, chain_string>)

Returns (as a pdb_ref) those atoms in I<pdb_ref> whose
   chain identifiers match those in
    I<chain_string>.  It is assumed that a chain identifier
         is only one letter; the
user may specify as many as they like.

=cut

sub select_chain {
    my ($pdb_ref, $chain) = @_;
    my $i = 0;
    my $new_pdb_ref; # a new pdb reference to be structured
         as def'd in read_pdb.
    foreach my $atom_identifier (keys %{$pdb_ref->{"
       sort_keys"}}) {
        if ($atom_identifier =~ m/^[$chain]/) {
            my $index = $pdb_ref->{"sort_keys"}->{"
               $atom_identifier"};
            $new_pdb_ref = &_copy_pdb_line($pdb_ref,
               $new_pdb_ref,$index,$i,$atom_identifier);
            $i += 1;
        }
    }
    return $new_pdb_ref;
}

=head2 I<internal> _copy_pdb_line

This is the routine that actually copies lines of
   pdb_references across.  See
code for signature and algorithm.  It is fairly simple, but
    should not be used
outside of pdbtools.pm.
```

```
_copy_pdb_line also acts as a de facto constructor of
   pdb_refs, although
it cannot do so without an existing pdb_ref.

=cut

sub _copy_pdb_line {
    my ($old_pdb_ref,$new_pdb_ref,$old_line_num,
       $new_line_num,$atom_id) = @_;
    #print @_,"\n";
    foreach my $pdb_type (keys %$old_pdb_ref) {
      if ($pdb_type eq "chains") {
        $new_pdb_ref->{"chains"} = $old_pdb_ref->{"chains"
           };
      } elsif (ref($old_pdb_ref->{"$pdb_type"}) eq "ARRAY")
          {
        $new_pdb_ref->{"$pdb_type"}->[$new_line_num] =
           $old_pdb_ref->{"$pdb_type"}->[$old_line_num];
      } elsif ($pdb_type eq "sort_keys") {
        $new_pdb_ref->{"sort_keys"}->{"$atom_id"} =
           $new_line_num;
      }
    }
    return $new_pdb_ref;
  }

=head1 Auxilliary functions

mass_table is useful in calculating various weightings, and
    centering
molecules.

=cut


sub mass_table {
    my ($elements) = @_;
    if (ref $elements eq "pdbtools") {
      $elements = $elements->{"atom_type"};
    }
    print "$elements\n";
    my %mass_hash = ("C" => 12.011, "N" =>14.007, "O" =>
       15.999, "S" => 32.064, "H" => 1.008);
    my $i=0;
    my @masses;
    foreach my $atom (@$elements) {
        $atom =~ m/^(\w)(\w*)$/;
        $masses[$i] = $mass_hash{"$1"};
        $i += 1;
        warn "Unknown␣atom␣type:␣$atom\n" unless ($atom =~
           m/^[CNOSH]/);
    }
    return \@masses;
}


#
# by_atom is a sorting routine used by vet_pdb.  See the
   Perl builtin "sort" for
# details of how sorting routines are to be structured...
#


=head1 Sorting algorithms

There are two internal sorting routines.  B<by_num> sorts
   explicitly
by numerical comparison.  B<by_atom> is more complicated,
   and allows
for sorting the sort_keys of a pdb reference first by
   residue number
and then later by atom type.  This prevents asciibetical
   sorting in
the output pdb file.

=cut

sub by_atom {
    $a =~ m/^(\D)?(\d+)(\w+)$/;
    my $res_a = $2;
    my $atom_a = $3;
    my $chain_a = ($1 or "␣");

    $b =~ m/^(\D)?(\d+)(\w+)$/;
    my $res_b = $2;
    my $atom_b = $3;
    my $chain_b = ($1 or ”␣”);

    return (($chain_a cmp $chain_b) or ($res_a <=> $res_b)
       or ($atom_a cmp $atom_b));
}


sub by_num {
    $a <=> $b;
}
```

B.2 linear.pm

This module contains several useful linear algebra routines which are not built into the PDL.

```
#!/usr/bin/perl -w

########################################################################
#

=head1 linear.pm overview

linear.pm is a Perl language module that has various
subroutines and sub-subroutines for performing linear
transformations minimizing the rms misfit between two
sets of points.  It was written originally for mapping
protein structures onto each other.

=head1 Dependencies

Uses the PDL modules.

=head1 General notes

It may at first confuse the user how one should use many
of these functions.  For instance, how should they
generate the 3xN pdls used by most (if not all) of these
functions.  The answer is that you shouldn't;␣you␣should
let␣the␣routines␣in␣pdbtools.pm␣do␣this␣for␣you␣by␣reading
in␣pdb␣files␣and␣retrieving␣coordinates␣from␣the␣resulting
structures.␣␣See␣pdbtools.html␣for␣more␣information,␣or␣
    type
I<perldoc␣pdbtools.pm>␣at␣the␣command␣line␣in␣the␣directory
containing␣these␣modules.

=head2␣Oddities

Due␣to␣the␣matrix␣implementation␣in␣the␣PDL,␣which␣is␣row␣
    centric,
the␣indexing␣may␣seem␣a␣bit␣odd␣at␣first,␣should␣you␣dig␣
    into
the␣code.␣␣(Note␣that␣we␣do␣not␣use␣PDL::Matrix␣here).␣␣The
first␣index␣is␣always␣a␣column␣index,␣and␣the␣second␣index␣
    is
```

```
always a row.

=cut

#
# Functions (all subroutines in this revision.
#
# rms: calculates a weighted rms between two sets of three-
    vectors,
# returns a scalar number
#
# weighted_outer_product: takes two sets of 3-vectors, 
    calculates
# the outer product of each pair (it assumes they are 
    sorted properly),
# and calculates a weighted sum of those, returning a 3x3 
    matrix
# reference (useable by Math::MatrixReal
#
# weighted_inner_product: pretty much the same, returns a 1
    x1
# matrix reference.
#
# sqrtm takes a matrix squareroot.  It assumes a 3x3 matrix
    , though
# this could easily be generalized.  Returns a reference to
     a 3x3 matrix.
#
# move: takes a structure (3-vectors) and a length 3 array
# $x, $y, $z; translates the structure by the values $x, $y
    , $z.
# Assumes that the input structure vectors are listed as x,
    y,z
# triples.
#
# general_linear, find_rotation: these two subroutines take
# references to two matrices 3 col x n rows, and a 1xn 
    weighting
# matrix.  They return references to 3x3 transformation 
    matrices.
#
# find_translation: takes the same input arguments as
# find_rotation.  Returns a reference to a translated 
    structure
# (3xn matrix) and an array ($dx,$dy,$dz) containing the
# translation.
#
# scale: finds a scalar (not 1x1 matrix) value scale factor
     between
# two structures.
#
# transform: (Added 7 Sept 2004, MDC) Perform a general 
    linear
# transformation on the input molecule.  Modified 29 Sept 
    to allow
# for translations before and after the linear transform.


package linear;

use PDL;
use Exporter;
@ISA = ('Exporter');
@EXPORT = ('rms','move','general_linear', 'find_rotation','
    find_translation','scale','transform');
@EXPORT_OK = ('weight_out_prod','weight_inn_prod','sqrtm','
    rms','move',
              'general_linear', 'find_rotation','
    find_translation','scale',
              'transform','diff_1d');

use strict;

=head1 General mathematical routines

=head2 rms (I<(3xN pdl) coords1, (3xN pdl) coords2, (1xN 
    pdl) $weights>)

Returns the weighted root-mean-square difference between 
    two sets of coordinates.

=cut

sub rms {
    my ($vectors1, $vectors2, $weights) = @_;
    my $diffs = (&diff_1d($vectors1, $vectors2));
    return wantarray ? $diffs->list() : sqrt(sum($weights*(
        $diffs**2))/sum($weights));

 }

=head2 diff_1d(I<(3xN pdl) coords1, (3xN pdl) coords2>)

Returns a 1xN pdl of the lengths between atoms in two sets 
    of coordinates.

=cut

sub diff_1d {
    my ($coords_in, $coords_tg) = @_;

    # Calculate differences between the two structures
    my $diff_3d = $coords_in - $coords_tg;
    my $diff_1d = (($diff_3d->slice(':,0'))**2
                    + ($diff_3d->slice(':,1'))**2
                    + ($diff_3d->slice(':,2'))**2)**(1/2);

    return $diff_1d;

}

=head2 weight_out_prod (I<(3xN pdl) coords1, (3xN pdl) 
    coords2>, (1xN pdl) $weights>)

Returns the weighted sum of outer products of pairs of 3-
    vectors as a 3x3 pdl.

=cut

# Curiously, it's faster to do it in loops than with the
# matrix multiplication routines. &...(x,y,w) returns
# \Sigma_a w_a x_a^T y_a (if you read TeX).

sub weight_out_prod {
    my ($vectors1, $vectors2, $weights) = @_;
    my $atoms = $vectors1->getdim(0);
    my $wop = 0;
    # Note: PLD::slice returns a ROW vector if you use the
       slice(”($i),0”) syntax
    # rather than slice(”$i,0”) even if the slice is in
       principle a column...

    for (my $i=0;$i<$atoms;$i++) {
        my $temp = ($vectors1->slice(”($i),:”)->transpose()
            x $vectors2->slice(”($i),:”)) * $weights->slice
           (”($i),(0)”);
        $wop += $temp;
    }

    return $wop*(1/sum($weights));
    #Normalized, $vectors2 (column vectors) X $vectors1 (
       row vectors)

}


=head2 weight_inn_prod (I<(3xN pdl) coords1, (3xN pdl)
   coords2>, (1xN pdl) $weights>)

Returns the weighted sum of inner products of pairs of 3-
   vectors as a 1x1 pdl.

=cut
# And then, a weighted INNER product (aka dot product,
   projection, etc...)

sub weight_inn_prod {
    my ($vectors1, $vectors2, $weights) = @_;
    my $atoms = $vectors1->getdim(0);
    my $wip = 0;

    for my $i (0..$atoms-1) {
        $wip += ($vectors1->slice(”($i),:”) x $vectors2->
           slice(”($i),:”)->transpose())*$weights->slice(”(
           $i),(0)”);
    }
     return $wip*(1/sum($weights));
    #Normalized, $vectors2 (column vectors) X $vectors1 (
       row vectors)
}
```

```
# What follows is how we take square roots of matrices.
# This is how we do things downtown, after we kicked
# Matlab in the shins.  If A has eigenvaluesD (organized
# in a diagonal matrix) and eigenvectors S,
# then A^1/2 = S*D^1/2*S'

=head2 sqrtm (I<(NxN pdl) matrix>)

Matrix square root.  See code for algorithm.

=cut

sub sqrtm {
    my ($matrix) = @_;
    my ($eig_vecs, $eig_vals) = $matrix->eigens();

    # ->stretcher() puts a vector on a diagonal of the
       appropriate size.
    return ($eig_vecs x stretcher(sqrt($eig_vals)) x
       $eig_vecs->transpose());
  }

=head1 Applying linear transformation and translations.

=head2 transform (I<(MxN pdl) coordinates, (LxM pdl) matrix
   , (1xM pdl) translation 1, (1xM) translation 2>)

Performs an arbitrary matrix transformation on the
   coordinates.  It will function with matrices
of arbitrary size, but in this distribution generally L=M
   =3.  Translation 1 is applied before the
matrix transformation, and translation 2 is applied
   afterwords.

Returns an LxN pdl.

=cut

# The coordinates are kept in a pdl where the matrix is 3 x
    N (that is, 3 rows...)

sub transform {
    my ($coords, $transformation,$move1, $move2) = @_; #
       These must be PDL references (aka piddles, pdls).

    my $c1 = $coords;
    $c1 = &move($c1,$move1) if (defined $move1);
    my $c2 = $transformation x $c1;
    $c2 = &move($c2,$move2) if (defined $move2);
    return $c2;
}

#
# move does just that: move $molecule1 by $x,$y,$z.
# It is used in several places, so it made sense
# to make it a subroutine.
#

=head2 move (I<(MxN pdl) coordinates, (1xM pdl) translation
   >)

Returns an MxN pdl set of coordinates translated by I<
   translation>.  Here M=3
for practical purposes, but the algorithm is not limited to
    any value of M.

=cut

sub move {
    my ($molecule1,$trans) = @_;
    return $molecule1 = $molecule1 + $trans->transpose() x
       ones($molecule1->getdim(0));
}

####################################################
#
# Below are the actual functions that find the
# transformations
#
###################################################

=head1 Finding rms minimum linear transformations between
   two sets of coordinates
```

```
=head2 general_linear (I<(3xN pdl) coords1, (3xN pdl)
   coords2, (1xN pdl) $weights>)

Returns a 3x3 pdl general linear transformation that
   minimizes the weighted rms difference between
the (transformed) input structure and the target structure.
     In principle this is a
convolution of rotation, shearing, and scaling.  It is
   therefore somewhat dangerous
to use, for instance if we were looking for shearing
   effects in the actual molecule.
If that is the case, you are better to use the rotation and
    scaling functions noted
below in series.

=cut

sub general_linear {
    # Note that x = $struct1 and y = $struct2 are
       references to matrices generated
    # in &get_data.  The form of the transform is:
    # Rij = (\Sigma_a x_a^i y_a^j)*inv(\Sigma_a x_a^i x_a^j
       ), i is the column index,
    # j is the row index.  I hope I have that right!  If
       not, it's just
    # a transposition to fix it!
    my ($struct, $target,$weight) = @_;

    return &weight_out_prod($target, $struct,$weight) x (&
       weight_out_prod($struct,$struct,$weight)->inv({"s"})
       );
 }

# Isotropic scaling...

=head2 scale (I<(3xN pdl) coords1, (3xN pdl) coords2, (1xN
   pdl) $weights>)

Returns a 1x1 pdl scale factor minimizing the weighted rms
   misfit between the two sets
of coordinates.

=cut
```

```
sub scale {
    my ($struct, $target,$weight) = @_;
    return &weight_inn_prod($struct,$target,$weight) x (&
       weight_inn_prod($struct, $struct,$weight)->inv({”s”
       }));
}

# Rotation
=head2 find_rotation (I<(3xN pdl) coords1, (3xN pdl)
   coords2, (1xN pdl) $weights>)

Returns the 3x3 antisymmetric rotation matrix minimizing
   the weighted rms misfit
betwen the two sets of coordinates.

=cut

sub find_rotation {
    # Notation: $xty refers to \vec x transposed times \vec
        y, and so on.
    my ($struct, $target,$weight) = @_;
    my $ytx = (&weight_out_prod($target,$struct,$weight));
    my $rotation = &sqrtm($ytx x $ytx->transpose()) x ($ytx
       ->transpose()->inv({”s”}));
    return ($rotation);
}

# Translation, now in PDL!

=head2 find_translation (I<(3xN pdl) coords1, (3xN pdl)
   coords2, (1xN pdl) $weights>)

Returns the 3x1 pdl translation vector minimizing the
   weighted rms difference
between the two sets of coordinates. E.g. is the weights
   are masses of atoms,
this will return the distance between their centers of mass
   .

=cut

sub find_translation {
    my ($struct, $target, $weight) = @_;
    my $diff = $target - $struct;

    my $sum_w = sum($weight);
    my $trans  = pdl ( sum($weight*$diff->slice(':,(0)'))/
       $sum_w,     # X
                       sum($weight*$diff->slice(':,(1)'))/
                          $sum_w,     # Y
                       sum($weight*$diff->slice(':,(2)'))/
                          $sum_w  ); # Z

  #  print "$trans \n";
    return $trans;
}
```

B.3 superpose

This script aligns two protein structures in PDB format, subject to various options and constraints. Its use of `vet_pdb` allows it to compare structures which may have potentially very different sequences. It determines the transformations (usually a rotation plus a translation) based on a user-specified subset of the structures, but applies the transform to all atoms in the original PDB files.

```
#!/usr/bin/perl

=head1 SUPERPOSE Overview

SUPERPOSE is a Perl script whose job it is to transform one
 set of protein structure in such as way as to minimize the
    rms
 misfit between those transformed coordinates and a target
 protein structure.

The procedure is

=over 4

=item

Read in and "vet" the two pdb files using routines from I<
    pdbtools.pm>

=item

Move both sets of coordinates so that their centers of mass
     are at the
 origin.

=item

 Calculate the requested transformation types (see options
    below).

=item

Move both molecules back to the target molecule's␣center␣of
    ␣mass.

=item
```

```
Write␣a␣new␣set␣of␣transformed␣coordinates␣to␣a␣user␣
    specified␣file.

=back

More␣details␣may␣be␣found␣below.

Marcus␣D.␣Collins␣Sept.␣2004␣␣marcus@bigbro.biophys.cornell
    .edu

=head2␣Dependencies

superpose␣requires␣a␣”header”␣file␣called␣dwarf_req.pm␣␣See
    ␣dwarf_req.html
for␣more␣information.

=head2␣Output

superpose␣appends␣a␣log␣file␣”align.log”␣containing␣a␣
    datestamp,␣information
about␣how␣superpose␣was␣called,␣some␣diagnostic␣information
    ,␣the␣transformations
found,␣and␣the␣translations␣from␣the␣center␣of␣mass␣of␣the␣
    two␣molecules␣to␣the
origin.␣␣(That␣is,␣the␣center␣of␣mass␣in␣the␣original␣
    coordinates).

=head2␣Odditites

Currently,␣superpose␣runs␣with␣no␣error␣messages␣if␣you␣
    send␣it␣something␣really␣odd␣(e.g.
you␣do␣not␣specify␣any␣pdb␣files␣to␣use).␣␣This␣may␣or␣may␣
    not␣ever␣get␣fixed...

Due␣to␣the␣matrix␣implementation␣in␣the␣PDL,␣which␣is␣row␣
    centric,␣the␣indexing␣may␣seem
a␣bit␣odd␣at␣first,␣should␣you␣dig␣into␣the␣code.␣␣(Note␣
    that␣we␣do␣not␣use␣PDL::Matrix
here).␣␣The␣first␣index␣is␣always␣a␣column␣index,␣and␣the␣
    second␣index␣is␣always␣a␣row.

=head1␣Usage
```

```
superpose␣−1I<input␣pdb>␣−2I<target␣pdb>␣[−TI<transform␣
   flag␣1>,␣−TI<transform␣flag␣2>,␣...]␣[−WI<m␣or␣b>]
␣␣␣␣␣␣␣␣␣␣[−vI<vetting␣flag␣1>,␣−vI<vetting␣flag␣2>,␣...]␣
   [−PI<output␣pdb>]␣[−dI<filename>]

All␣command␣line␣flags␣are␣optional,␣but␣be␣warned␣that␣the
   ␣output␣may␣not␣be␣meaningful␣for␣all
possible␣combinations.␣␣Think␣carefully␣about␣what␣you're
   doing.

If two pdb files are loaded successfully, superpose will
   move the input structure's␣COM␣to␣the
target␣structure's COM, and output the translations to the
   log file superpose.log.

Further options are specified below

=head2 Linear transformations

=head3 General notes
Transformations are performed in the order specified on the
    command line, e.g.

superpose ... −Tr −Tc

Will first calculate a rotation to minimize the differences
    between the two molecules,
then calculate an overall isotropic scale factor.  The
   independent rotation matrix,
scale factor and overall transformation matrix will be
   output to superpose.log.

Note that with the exception of an overall scale factor,
   these operations do not
necessarily commute, so that the order does matter.

=head3 Command line options

−Tg: puts molecules COMs on origin, performs the most
   general linear transformation,
and translates back to #2's original center of mass.
```

```
-Tr: puts molecules COMs on origin, rotates 1 to 2,
and then translates both back to #2's original center of
   mass

-Tt: translation only.  Vestigal, and therefore not
   recommended.

-Tc: scaling.  Note that this is not particularly useful on
    its own, and is potentially
mathematically unstable.  While this is unlikely in
   structures we are likely to compare,
it is worth noting.  See dwarf.html for more information.

=head2 Weighting options

The following options allow the rms misfit to be weighted
   on a per atom basis.  While
you can specify this as many times as you like, only the
   last entry on the command line
will be read in.

-Wb use averaged inverse B-factor weighting.  The geometric
    average of the two B-factors is
used, to favor well localized atoms.

-Wm use mass weighting.  This is used implicitly in the COM
    calculations.

In the future occupancy based weighting may be included.

=head2 Atomic "Vetting" options

=head3 What is this?

The routine pdbtools::vet_pdb is at the core of this
   distribution's␣ability␣to␣compare
potentially␣very␣different␣structures.␣␣For␣instance,␣how␣
   are␣we␣to␣compare␣two␣mutant
structures␣to␣each␣other?␣␣Can␣we␣make␣a␣fair␣comparison?

The␣solution␣here␣is␣to␣throw␣away␣any␣atoms␣which␣are␣not␣
   the␣same,␣based␣on␣residue␣number
```

```
and␣atom␣type.␣␣It␣ignores␣residue␣type.␣␣Any␣atoms␣not␣
    common␣to␣both␣structures␣are
automatically␣removed,␣and␣the␣following␣options␣allow␣
    further␣restrictions.␣␣Currently,
atom␣type␣selections␣are␣global,␣so␣you␣cannot␣select␣CA␣
    atoms␣on␣one␣residue␣range␣and␣C␣or␣N
atoms␣on␣another.␣␣E.g.

-v10-50␣-vCA␣-vCB␣-v60-70␣-vN

will␣select␣CA,␣CB,␣and␣N␣atoms␣for␣residues␣10␣through␣50␣
    and␣60␣through␣70.␣␣You␣may
specify␣as␣many␣or␣as␣few␣options␣as␣you␣like.␣␣The␣default
    ␣(no␣specification)␣is␣to␣use␣all
atoms␣common␣to␣both␣molecules.␣␣Things␣like␣C.*␣(all␣
    carbon␣atoms)␣do␣work,␣since␣the␣flags
are␣all␣eventually␣passed␣to␣a␣regular␣expression␣matcher.␣
    ␣Be␣careful␣with␣these:␣regex␣can
be␣very␣confusing.

In␣the␣most␣recent␣version,␣support␣for␣chain␣identifiers␣(
    A-Z)␣is␣available.␣␣Use␣the␣-c
option␣to␣select␣chains.␣␣Note␣however␣that␣to␣compare␣two␣
    chains␣to␣each␣other,␣they
must␣have␣the␣same␣identifier.␣␣This␣may␣be␣fixed␣in␣future
    ␣versions␣by␣allowing␣the␣user␣to
change␣chain␣identifiers.

The␣routine␣underlying␣this␣is␣actually␣quite␣complicated,␣
    and␣not␣at␣all␣transparent.␣␣See
pdbtools.html␣for␣more␣information.

=head3␣Command␣line␣options

-v[I<residue␣range>␣or␣I<atom␣type>]␣␣Residue␣ranges␣must␣
    be␣specified␣as␣I<number1-number2>
and␣I<number1>␣must␣be␣smaller␣than␣I<number2>␣or␣Perl␣will
    ␣get␣confused.␣␣E.g.␣C<-v10-40>␣is
acceptable␣but␣C<-v10␣40>␣or␣C<-v40-10>␣are␣not.␣␣Atom␣
    types␣must␣be␣specified␣as␣with␣a␣C<-v>
for␣each␣type.␣␣For␣instance,␣where␣you␣to␣use␣only␣C<-
    vCACB>,␣Perl␣would␣generate␣an␣error
```

```
indicating (essentially) that no coordinates were loaded, 
    since no atom type "CACB" exists in
the PDB format.

-c[A-Z]  Select any specified chains.  This calls the 
    routine pdbtools::select_chain and returns
a pdb_ref containing only atoms with chain identifiers 
    specified.  E.g. -cAC will tell superpose
to compare only chains A and C.  Watch out!  Not all PDB 
    files have chain identifiers, and some
have very odd chain identifiers (such as those written by 
    Refmac5).

=head3 Oddities

If you call vet_pdb implicitly by using -v in the call to 
    superpose, something must be specified.
A single blank -v is actually interpreted to mean "load no 
    coordinates".  It is unclear what would
happen if you used C<-v -vCA> or somesuch.  Better not to 
    try it.

=head2 Input and Output Options
No spaces are allowed between flags and their values!

-1 or -i Specifies the input pdb file (the structure to be 
    transformed).

-2 or -t  Specifies the target structure (to which the 
    input structure is mapped.

-dI<filename> Specifies the filename to which superpose 
    will write a distance difference
map, and is the implicit flag to calculate said map.  The 
    file will be written as a PNG
I<filename>.png in the current directory.

This can be computationally intensive.  Choose your vetting
     options well, or wait a while.
The map itself can be calculated very quickly, but drawing 
    it takes some time.
```

```
−PI<filename>␣Output␣a␣pdb␣file␣I<filename>.pdb␣␣(If␣not␣
    specified,␣default.pdb␣is␣output).

=head1␣Algorithms␣and␣Procedures

See␣the␣code␣itself␣for␣detailed␣discussions␣of␣methods␣and
    ␣see␣module␣documentation
and␣code␣for␣detailed␣explanations␣of␣underlying␣
    subroutines.

=cut

#␣Pragmas␣and␣modules
use␣PDL;
use␣strict;

#use␣lib␣qw(/Users/marcus/DWARF);␣␣␣#␣Current␣location␣of␣
    modules.
#use␣dwarf_req;␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣#␣Central␣file␣that␣
    keeps␣track␣of␣DWARF's dependencies
use pdbtools;
use linear;

my ($input_pdb,$output_pdb, $target_pdb, $transform_flags,
    $weight_flag, $diff_map);
my $unity_mat = pdl [[1,0,0],[0,1,0],[0,0,1]];
my $transform = $unity_mat; #Inititally defined as the
    identity matrix.
my ($structure_ref, $target_ref, $coords_in, $coords_trgt,
    $coords_out);
my @keep_what;
my $chains_to_keep;

#Defaults
$output_pdb = "default.pdb";
my $scale = 0.7;

# Prepare a log file.  Appending a generic file minimizes
    user input.
open LOG, ">>␣align.log" or die "Cannot␣open␣logfile␣align.
    log";
my $datestamp = ‘date‘;
```

```
# Get files off command line, and command arguments
foreach (@ARGV) {
    if (/^-1(.*)$/ || /^-i(.*)$/) {
        $input_pdb = $1;
        $structure_ref = &read_pdb($1);  #Ref. to hash of
           refs.  See pdbtools.pm
        next;
    } elsif (/^-2(.*)$/ || /^-t(.*)$/) {
        $target_pdb = $1;
        $target_ref = &read_pdb($1);
        next;
    } elsif (/^-T(.*)$/) {
        $transform_flags .= $1;
        next;
    } elsif (/^-W(.*)$/) {
        $weight_flag = $1;
    } elsif (/^-D(.*)$/i) {
        $diff_map = $1;
    } elsif (/^-P(.*)$/) {
        $output_pdb = $1;
    } elsif (/^-v(.*)$/) {
        push @keep_what, $1;
    } elsif (/^-c(.*)$/) {
        $chains_to_keep .= $1;
    } elsif (/^-s(.*)$/i) {
        $scale = $1;
    } else {
        print LOG "Option␣not␣supported:␣$_\n";
    }
}


print LOG "\n".("*"x80)."\n\nSUPERPOSE␣run␣parameters:\n\
   n$datestamp";
print LOG "$transform_flags\nMapping␣$input_pdb␣onto␣
   $target_pdb,␣weight:␣$weight_flag\n";
print LOG "Vetting␣flags:␣@keep_what,␣Chains␣kept:␣
   $chains_to_keep␣\n";
print LOG "Dist.␣Diff␣Map␣scale:␣$scale␣\n";
#print STDOUT "$structure_ref \t$target_ref\n";
```

```
print "Vet_pdb␣sorting␣and␣vetting...␣see␣log␣for␣included␣
   residues/atom␣types.\n";
($structure_ref,$target_ref) = &vet_pdb($structure_ref,
   $target_ref, @keep_what);
if ($chains_to_keep) {
    $structure_ref = pdbtools::select_chain($structure_ref,
        $chains_to_keep);
    $target_ref = pdbtools::select_chain($target_ref,
       $chains_to_keep);
}
print LOG "Vet_pdb␣finished:␣" . `date` ."\n";
#print STDOUT "$structure_ref \t$target_ref\n";

# TEST STATEMENTS FOR VET_PDB'S RETURN VALUES
#print ref($structure_ref->{"sort_keys"});
#my @array = @{$structure_ref->{"atom_type"}};
#print "Array: @array\n";
#my @array = keys %$structure_ref;
#print "Array of keys: @array\n";

# Put stuff into relevant matrices... &get_coords returns
   a 3xN pdl
# NB: X is row 1, Y is row 2, Z is row 3... EVERYTHING
   else is predicated on that...
$coords_in = &get_coords($structure_ref);
$coords_trgt = &get_coords($target_ref);

print STDOUT "Coordinates␣loaded...\n";
#print "Input coordinates: $coords_in\n";
#print "Output coordinates: $coords_trgt\n";

my $masses = pdl ( @{&mass_table($structure_ref->{"
   atom_type"})} );
my $weights = (0 * $masses) + 1; #PDL does this operation
   on every element...
my $B1 = pdl [@{$structure_ref->{"B"}}];
my $B2 = pdl [@{$target_ref->{"B"}} ];

print STDOUT "Weights␣loaded...\n";

# Check that the structures are the same size.
die "Structures␣are␣not␣the␣same␣size,␣cannot␣continue!"
```

```
    unless ($coords_in->getdim(0) == $coords_trgt->getdim
        (0));

# Weighting schemes.
if ($weight_flag =~ m/B/i) {
    $weights = 1/((1/$B1) + (1/$B2));
} elsif ($weight_flag =~ m/M/i) {
    $weights = $masses;
}


# Mass table is used only above; takes a REFERENCE to an
    array, and returns a Reference
sub mass_table {
    my ($elements) = @_;
    my %mass_hash = ("C" => 12.011, "N" =>14.007, "O" =>
        15.999, "S" => 32.064, "H" => 1.008);
    my $i=0;
    my @masses;
    foreach my $atom (@$elements) {
        $atom =~ m/^(\w)(\w*)$/;
        $masses[$i] = $mass_hash{"$1"};
        $i += 1;
        warn "Unknown␣atom␣type:␣$atom\n" unless ($atom =~
            m/^[CNOSH]/);
    }
    return \@masses;
}

# Move both structures to the origin.  Always use masses as
    weights...
print LOG "CENTERING␣MOLECULES...␣␣USING␣MASSES␣FOR␣
    WEIGHTING\n";
print STDOUT "CENTERING␣MOLECULES...␣␣USING␣MASSES␣FOR␣
    WEIGHTING\n";

my $trans1 = &find_translation($coords_in, $coords_in->
    zeroes(),$masses);
my $trans2 = &find_translation($coords_trgt, $coords_trgt->
    zeroes(),$masses);
my $mol1_cent = &move($coords_in,$trans1);
my $mol2_cent = &move($coords_trgt,$trans2);
```

```
print LOG "Translations␣to␣origin:\nMolecule␣1:\n$trans1\
    nMolecule2:\n$trans2\n\n";
print LOG "NOTE:␣These␣are␣the␣vectors␣from␣the␣original␣
    centers␣of␣mass␣to␣the␣origin!\n";

print LOG "Beginning␣transformations...\n";

########
#
# Here's the loop to find (and do) the transforms.  It
    cycles through all options looking
# For any that are transforms.
#
while ($transform_flags =~ s/^([gcrth])(\w*)$/-$2/i) {
     my $option = $1;
     if ($option eq 'g') {
          print LOG "Performing␣linear␣transformation!\n";
          my $gen_trans = &general_linear($transform x
              $mol1_cent, $mol2_cent, $weights);
          $transform = $gen_trans x $transform;
          print LOG "Linear␣transformation␣matrix:\
              n$gen_trans";
     } elsif ($option eq 'c') {
          print STDOUT "Scaling␣molecules␣now...\n";
          my $scale = (&scale($transform x $mol1_cent,
              $mol2_cent, $weights)->slice('(0),(0)'))*
              $unity_mat;
          $transform = $scale x $transform;
          print LOG "\nScale␣factor␣=␣$scale\n\n";
     } elsif ($option eq 'r') {
          print STDOUT "Rotating␣now!\n";
          my $rotation = &find_rotation($mol1_cent,
              $mol2_cent, $weights);
          $transform = $rotation x $transform;
          print LOG "Rotation␣matrix␣:\n$rotation";
     } elsif ($option eq 't') {
          print STDOUT "Translating␣now!\n";
          print STDOUT "WARNING:␣Translation␣function␣is␣
              vestigal,␣and␣may␣produce␣misleading␣results!\n"
              ;
          my $trans = &find_translation($coords_in,
              $coords_trgt, $weights);
          print LOG "Translation␣vector␣=\n␣$trans\n";
    } elsif ($option eq 'h') { &shear();}
}

print LOG "Overall␣transformation␣matrix:␣$transform\n";

#Now actually make the transformation to get the output
    coordinates
#If $transform wasn't changed above, it will be a 3x3
    identity matrix.
#$coords_out = &transform($coords_in,$transform,$trans1,-
    $trans2);

# The coordinates are in the PDL format, whereas write_pdb
    requires a special format defined in
# read_pdb (see pdbtools.pm). &write_new_coords handles
    copying the new coordinates into the
# old structure record.

#NOTE WELL: This will write out all atoms that were in the
    original pdb files!
#All of them will be tranformed according to the
    transformation determined above.
my $pdb_ref = pdbtools::read_pdb($input_pdb);
my $coords = pdbtools::get_coords($pdb_ref);
$coords_out = &transform($coords,$transform,$trans1,-
    $trans2);
print join "␣", $coords_out->dims(), "\n";
$pdb_ref = &write_new_coords($coords_out, $pdb_ref);
&write_pdb($pdb_ref,$output_pdb);

#print LOG $coords_out;
$coords_out = &transform($coords_in,$transform,$trans1,-
    $trans2);
print LOG "RMS␣misfit␣(angstroms):␣";
my $rms = &rms($coords_out,$coords_trgt,$weights);
print LOG "$rms\n";

#Write the atom-by-atom differences to a file
open DIFFS, ">␣align.diffs";
my @diffs = &rms($coords_out,$coords_trgt,ones($weights));
my $atom = 1;
foreach (@diffs) {
  print DIFFS "$atom\t$_\n";
  $atom ++;
}
close DIFFS;

# Clean up a little bit...
undef $weights;
undef $masses;

undef $mol1_cent;
undef $mol2_cent;

print LOG `date`;

####
#
# Next, calculate anything else the user wanted...

# Difference maps
my $drawmap = 'yes';    #This flag exists to allow more
    flexible use of ddiff
                        # and draw_mat, while here calling
                            only ddiff.
&ddiff($coords_out,$coords_trgt,$diff_map,$scale,$drawmap)
    if ($diff_map);

print LOG "Returned␣from␣ddiff:␣" if ($diff_map);
print LOG `date`;
```

B.4 multipose

This script aligns multiple structures at once. Like `superpose` it can be invoked to only align on part of the input structures, but will output complete structures based on those alignments.

```
#!/usr/bin/perl -w

use PDL;
use PDL::Opt::Simplex;
use PDL::NiceSlice;

use linear;
use pdbtools;

use strict;

# This program is designed to simultaneously minimize
# the pairwise differences between multiple protein
   structures.
#
# A great deal of the preparation work is done similarly to
    my other
# program, superpose.  However, due to the nature of this
   problem, a
# numerical optimization method is implemented here.
   Currently, the
# PDL::Opt::Simplex routine is used.  If this appears to be
    inefficient or
# frustrated, then a simulated annealing method may be
   implemented later.
#
# Vetting is handled differently here than in superpose.
   See below.
#
# The current implementation also uses rotations and
   translations only.
# Weighting can be by mass, B-factor, or nothing at all.

# Declarations
my @coord_pdls; #An array to hold the pdl references of
   coordinates
my @strc_refs; #This array holds references to the pdb
   structs def'd in pdbtools.pm
```

```
my @weight_pdls;

my @molecules; #An array of references to hashes which
    contain pdl refs to useful information.
my @trans; #This will hold the translations of the centers
    of mass to use later.

my $rms;

# Read in the command line.  NOTE: Unlike superpose, it
    requires
# some vetting flags to be set (even if it's just an empty
    string) and
# it does not handle chains (though this could be
    implemented using
# pdbtools::select_chain.)

die "Multipose:␣\"<vetting flags>\"␣\" <weighting>\"␣pdb1.
   pdb␣pdb2.pdb␣...␣"
     if (scalar @ARGV < 4);

my @vetFlags = split(/\s+/, shift @ARGV);
my $weighting = shift @ARGV;
my @PDBs = @ARGV;

print "Now␣reading␣pdb␣files...";

# Now load the files
foreach my $pdb (@PDBs) {
  push @strc_refs, read_pdb($pdb);
}

# Vet the files pairwise.  This is horrifyingly inefficient
    ...
my $nStruct = @strc_refs;
print "Number␣of␣structures␣loaded:␣$nStruct␣\n";
foreach my $i (0..$nStruct-1) {
  foreach my $j ($i+1..$nStruct-1) {
    ($strc_refs[$i],$strc_refs[$j]) = vet_pdb($strc_refs[$i
        ],$strc_refs[$j],@vetFlags);
  }
}
```

```
#print "Vetting options used: mm@{vetFlags}mm\n";

# Now get coordinates and set the appropriate weighting
foreach my $ref (@strc_refs) {
  push @coord_pdls, get_coords($ref);
  #print "Coords: ". &get_coords($ref);
  if ($weighting eq 'mass') {
    push @weight_pdls, pdl(@{&mass_table($ref->{"atom_type"
       })});
  } elsif ($weighting eq 'B') {
    push @weight_pdls, (pdl(@{$ref->{"B"}}))**(-1);
  } elsif ($weighting eq 'none') {
    push @weight_pdls, 1 + (0 * pdl(@{$ref->{"B"}}));
  }

  #Eventually, some error checking here would be good!
  # However, vet_pdb and the other routines in
  # pdbtools.pm have become very robust in developing
      superpose...
}

# Construct the array of hashes to efficiently pass
    everything...
for my $i (0..$nStruct-1) {
  push @molecules, {"coords" => $coord_pdls[$i], "weights"
      => $weight_pdls[$i], "pdbData" => $strc_refs[$i]};
}

# Move all molecules COW to the origin...

foreach my $molecule (@molecules) {
  #print "$molecule \n";
  #print $molecule->{"coords"};
  #print $molecule->{"weights"};
  my $trans = find_translation($molecule->{"coords"},0*(
      $molecule->{"coords"}),$molecule->{"weights"});
  push @trans, $trans;
  print "Translation␣to␣origin␣for␣$molecule:␣$trans␣\n";
  $molecule->{"centCoords"} = move($molecule->{"coords"},
      $trans);
  print "Translation␣complete.\n";
  #print $molecule->{"centCoords"}->transpose();
}
```

```
# Devious testing idioms...
#$molecules[0]->{"centCoords"} =
#     &GenRot(1,0.45,0.05) x $molecules[0]->{"centCoords"};

# Now it comes down to a minimization problem.
# The residual is defined as the sum of all
# pairwise differences.

#Initial parameters to try.  Euler angles in radians...
my $initRot = PDL->zeroes(3*($nStruct-1));

# this is used by PDL::...::simplex to generate
# the initial simplex it is essentially the "size"
#of the simplex.  May need to be adjusted.
my $initSize = 3.14159/36;

#convergence criterion in terms of stepsize of the simplex.
my $convValue = 0.001;

#Don't do more iterations than this.
my $maxIter = 1000;

print "Beginning␣optimization...\n";
my ($optimumRotations, $ssize)
    = PDL::Opt::Simplex::simplex($initRot,$initSize,
        $convValue,$maxIter,\&main::rotRMS,undef);
print "Optimum␣Rotations:␣$optimumRotations␣\nStep␣Size:␣
   $ssize\n";

my ($finalrms,@newmols) = rotRMS($optimumRotations);
print "Final␣RMS:␣$finalrms\n";

# Added 25 August 2005
# Seems like a nice idea to just average
# the structures (not written out)
# and calculate the rms for each atom!

my $avcoords = $newmols[0]->{"centCoords"}->zeroes();
my $nn=0;
foreach my $newmol (@newmols) {
  $avcoords += $newmol->{"centCoords"}; #this is a pdl.
  $nn++;
}
#print $avcoords;
$avcoords /= $nn;
#print STDERR "$nn\n";
my $diffsq = $avcoords(:,(0))->zeroes();
foreach (@newmols) {
  my $vecdiff = $_->{"centCoords"} - $avcoords;
  $diffsq += (($vecdiff->transpose() x $vecdiff)->diagonal
      (0,1));
}
$diffsq /= ($nn);
#print $diffsq;
my @sigma_i = (sqrt($diffsq))->list();
open DAT, ">␣multipose.dat";
foreach (@sigma_i) {
  print DAT "$_\n";
}

# Need to write out the entire pdb file...
# This is super kluge-y, but we'll just reread the original
     pdb files,
# write the coordinates to the right place, and rerun
   rotRMS...
#

#Free up some variables and arrays.
undef(@molecules);
undef(@strc_refs);
die "WARNING:␣molecules␣are␣not␣undefined." if @molecules;

# Reload pdb files, get coordinates, and translate the
    molecules
#Don't need to set any weighting...
print "Commencing␣full␣pdb␣transformations...\n";
foreach my $pdb (@PDBs) {
  my $ref = &read_pdb($pdb);
  push @strc_refs, $ref;
  my $tmpCoords = &get_coords($ref);
  my $centCoords = &move($tmpCoords,shift @trans);
  push @molecules, {"centCoords" => $centCoords, "weights"
      => 1 + (0 * pdl(@{$ref->{"B"}}))};
}
```

```
# Now the molecules should be loaded in, with no vetting.
   Apply rotations
my (undef,@finalCoords) = rotRMS($optimumRotations);

#Write the pdb files...
foreach my $i (0..@strc_refs-1) {
  write_new_coords($finalCoords[$i]->{”centCoords”},
     $strc_refs[$i]);
  write_pdb($strc_refs[$i],”mOut${i}.pdb”);
}



############################################################
# Subroutines follow....

sub rotRMS {

  #print ”In rotRMS”;
  my ($rotations) = @_;
  my $nd = $rotations->getdim(0);
  my $nPoints = $rotations->getdim(1);
  my $sums = zeroes($nPoints);
  my $tmpSum;

  my $debug = 0;
  print ”\nrotRMS\tdims:␣$nd␣points:␣$nPoints\n” if $debug;
  print $rotations if $debug;

  my @rotMols;

  for (my $i=0; $i<$nPoints; $i++) {
    my @angles = list($rotations->slice(”:,($i)”));
    my @rotatedMols;
    print ”\nCycle:␣$i\t” if $debug;
    print ”\nAngles:␣@angles␣\n” if $debug>1;
    # Rotations should be a PDL, with 3 rot. angles
    # for each of N-1 structures. It is easiest
    # for what follows just to set the last of the
    # rotations to the identity matrix. Note that
    # molecules is a global variable from the top
    # level of the script. So much for strictness.
```

```
#print "@molecules \n";

#Construct individual rotation matrices, and append
# each to the appropriate molecule.
foreach my $mol (@molecules) {
  if (@angles) {
    my $a = shift @angles; my $b = shift @angles; my $g
        = shift @angles;
    $mol->{"currentRot"} = &GenRot($a,$b,$g);
  } else {
    $mol->{"currentRot"} = &GenRot(0,0,0); #The
        identity matrix
  }
}
print "Molecules␣rotated...\t" if $debug;

# Now, we don't want to screw with the original
# structures, so we need to copy them over
# into a new spot.  Be careful!

foreach my $mol (@molecules) {
  my $tmpWt = $mol->{"weights"}->copy();
  my $tmpRotCoord = ($mol->{"currentRot"}) x ($mol->{"
     centCoords"});
  push @rotatedMols, {"weights" => $tmpWt, "centCoords"
      => $tmpRotCoord};
}

print "Molecules␣copied...\t" if $debug;

# And now...
# Behold the stupid indexing tricks!  PDL works on
# references.  $tmpSum is a ref to an entry in $sums
# so changing the value of $tmpSum changes the
# corresponding element in $sums.
$tmpSum = $sums->slice("($i)");
$tmpSum .= sqrt((sum &mult_diff(@rotatedMols))/(
    $rotatedMols[0]->{"centCoords"}->getdim(0)));

# This next line deserves some note, since it took
# me a while to get it right.  I want to average
# the ms values between all (ordered, but matched)
# atoms.  That means I need to divide through by the
```

```
    # number of comparisons made.  Consider a difference
    # matrix D_ij: the diagonal is zero, and the uht and
    # lht are transposes of each other.  So, the number of
    # unique non−zero elements is N(N−1)/2  (note that the
    # redundancy of lht and uht is taken care of in &
       mult_diff.
    $tmpSum /= sqrt((scalar @rotatedMols) * ((scalar
       @rotatedMols) − 1)/2);
    print "sum␣calculated.\n" if $debug;
    @rotMols = @rotatedMols;
  }
  print $sums if $debug;
  print "\nReturning␣to␣PDL::Opt::Simplex::simplex.\n" if
     $debug;
  wantarray ? return ($sums, @rotMols) : return $sums;
}

sub GenRot {
  my ($a,$b,$g) = @_;
  my $Ra = pdl[[cos($a), sin($a), 0],[−sin($a),cos($a)
     ,0],[0,0,1]];
  my $Rb = pdl[[cos($b), 0, −sin($b)],[0,1,0],[sin($b),0,
     cos($b)]];
  my $Rg = pdl[[cos($g), sin($g), 0],[−sin($g),cos($g)
     ,0],[0,0,1]];
  my $R = $Rg x $Rb x $Ra;
}

sub mult_diff {
  my @mols = @_;
  #print join " ", @mols, "\n";


  my $residual = 0*($mols[0]−>{"centCoords"}−>slice(':,(0)
     '));
  my $sumWeights; #Summed weights for a given atom... so we
      can normalize.

  foreach my $i (@mols) {
    foreach my $j (@mols) {
      #print "Structures: $i, $j \n";
      my ($curCoords1, $curCoords2) = ($i−>{"centCoords"},
         $j−>{"centCoords"});
      my ($curWeight1, $curWeight2) = ($i->{”weights”},$j
         ->{”weights”});

      #print $curCoords1->transpose(), $curCoords2->
         transpose();

      # Note what is below is PDL syntax...
      my $uw = (($curCoords1-$curCoords2)->transpose()) x (
         $curCoords1-$curCoords2);

      #Don't double count: divide by 2!
      $residual += ($uw->diagonal(0,1))*sqrt($curWeight1*
         $curWeight2)/2;
    }
  }

  $residual;
}

sub mass_table {
  my ($elements) = @_;
  my %mass_hash = (”C” => 12.011, ”N” =>14.007, ”O” =>
     15.999, ”S” => 32.064, ”H” => 1.008);
  my $i=0;
  my @masses;
  foreach my $atom (@$elements) {
    $atom =˜ m/ˆ(\w)(\w*)$/;
    $masses[$i] = $mass_hash{”$1”};
    $i += 1;
    warn ”Unknown␣atom␣type:␣$atom\n” unless ($atom =˜ m/ˆ[
       CNOSH]/);
  }
  return \@masses;
}
```

B.5 avstruct

Averages input structures in real space.

```
#!/usr/bin/perl -w

use PDL;

use linear;
use pdbtools qw/read_pdb write_pdb get_coords
   write_new_coords write_field _get_values_as_pdl/;

use strict;

if (@ARGV == 0) {
  print "avstruct␣<output␣pdb␣file>␣<input␣pdb␣files>\n";
  exit;
}
my $pdb_out = shift @ARGV;
my @coords_pdl;
my $struct;
my @B;


foreach my $file (@ARGV) {
  $struct = read_pdb($file);
  push @coords_pdl, get_coords($struct);
  my $tmp = _get_values_as_pdl($struct, 'B');
  push @B, $tmp;
  print join "␣", $tmp->dims(), "\n";
}

my $sum = $coords_pdl[0]->zeroes;
my $B = $B[0]->zeroes;

foreach (@coords_pdl) {
  $sum += $_;
}
foreach (@B) {
  $B += $_;
}

$sum = $sum/(scalar @coords_pdl);
$B /= scalar @B;
```

```
#print $sum;
#print $struct, ”\n”;

#print $B;

write_field($B,'B',$struct);
#my $test = _get_values_as_pdl($struct,'B');
#print $test;
#print join ” ”, $test->dims(), ”\n”;
#print join ” ”, $B->dims(), ”\n”;


write_new_coords($sum, $struct); # Obnoxious overwrite—but
    I have no blank pdb constructor...
write_pdb($struct,$pdb_out);
```

B.6 helixcalc

This script calculates helix geometry. See the code itself for references.

```
#!/usr/bin/perl -w

##########################################################
#
# helixcalc: ./helixcalc input.pdb
#
# MDC 18 January 2005
#
# Calculates helix geometry of an input pdb file using
# the method of Sugeta and Miyazawa, 1967
# Biopolymers 5: 673-679
#
# inspired by Kumar and Bansal, Biophysical J.,
# 71:1574-1586, 1996 who implemented the method in
# Fortran.
#
# helixcalc [-h <helixpdb>] in.pdb
#
# helixcalc will calculate orientations of helices
# of in.pdb optionally using the secondary structure
# definitions in <helixpdb>
#
# The script will finish by writing a table of
# helical parameters to STDOUT and a pdb file drawing
# each helical axis (which may be curved or kinked).
#
###########################################################

use strict;
use PDL;
use PDL::NiceSlice;
use File::Basename;
use pdbtools;
use linear;

my $debug = 0;

my $useh;
my $helixpdb = undef;
if ($ARGV[0] eq "-h") {
  shift @ARGV;
  $helixpdb = shift @ARGV;
  $useh = 1;
}

my @files = @ARGV;

foreach my $file (@files) {
  $helixpdb = $file unless $useh;
  print "Using $helixpdb for helix definitions.\n\n";

  my $pdb_ref = read_pdb($file);
  my ($shortname, $dir, $type) = File::Basename::fileparse(
      $file,qr/\.pdb/);
  open HELIX, "> ${shortname}_hx.pdb";
  open TABLE, "> ${shortname}_hx.tab";

#Get the helices
  my $hpdb_ref = read_pdb($helixpdb);
  my @vetstrings = pdbtools::get_helices($hpdb_ref);

#print "$hpdb_ref, $pdb_ref\n";
#print join " ", "VET: ", @vetstrings, "\n";

# Print out the residue ranges for each helix.
  foreach (@vetstrings) {
    print STDERR "$_\n" if $debug;
  }

# Now go helix by helix and construct the geometry.
  my $label = 'A';
  my $i = 1;
  my %helixTable; #This hash will store helical parameters.

  foreach my $vetstring (@vetstrings) {
    print STDERR "VET: $vetstring\n" if $debug;
    # Define some local arrays.
    # Note that lengths of vectors in axes will be the
        local helical pitch
    my $avg_theta = pdl[0];
    my $avg_pitch = pdl[0];
    my $avg_axis = pdl[0,0,0];
```

```
#Useful for proper numbering of helical segments later
    on
my @startend = split /−/, $vetstring;
my @range = $startend[0]..$startend[1];

my $helix = pdbtools::vet_pdb($pdb_ref,$pdb_ref,”CA”,
    $vetstring);
# Coordinates of CA atoms in helix.
my $helixC = pdbtools::get_coords($helix);

my $helixN = $helixC−>getdim(0); #number of residues in
     the helix.
 print STDERR ”Helix␣$label\n” if $debug;
# Current starting residue.
# $n + 3 must be less than or equal to $helixN.
my $n = 0;
my $pos;

my $AT_LET = ’A’; #Lettering of ”atoms” in pdb file;


 foreach my $nCur ($n..$helixN−4) {

   # displacement vectors for the four atoms in question
       .
   my $B12 = $helixC(($nCur+1),:) − $helixC(($nCur),:);
   my $B23 = $helixC(($nCur+2),:) − $helixC(($nCur+1),:)
       ;
   my $B34 = $helixC(($nCur+3),:) − $helixC(($nCur+2),:)
       ;

   # Difference vectors used to define local helical
       axis
   my $C13 = $B12 − $B23;
   my $C24 = $B23 − $B34;

   # helical angles
   my $theta = acos(($C13 x $C24−>transpose())/(sqrt(sum
       ($C13*$C13)) * sqrt(sum($C24*$C24))));
   $avg_theta += $theta;

   # helical axis: cross product of difference
   # vectors, normalized.  d = distance
```

```
        # along axis between atoms 2 and 3.
        my $axis = (cross($C13,$C24));
        $axis = $axis->norm();
        my $pitch = $B23 x $axis->transpose();

# For purposes of plotting, the axis must be of length
    pitch.
        $axis *= $pitch;

# These aren't really averages, they are sums.
# They'll be normed later.
        $avg_axis += $axis;
        $avg_pitch += $pitch;

        (print $B12, "\t",$B12->getdim(0),"\t",$theta,$pitch,
            "\n") if $debug > 1;

        #if ($nCur == 0) {
        $pos = $helixC(($nCur),:) + $helixC(($nCur+1),:) +
            $helixC(($nCur+2),:) + $helixC(($nCur+3),:);
        $pos /= 4;
        #}
        if ($nCur == 0) {
          print HELIX "REMARK␣Helix␣$label\n";
          printf HELIX "HET␣␣␣␣HEL%7d%8d␣␣␣␣HELIX-$i␣\n",
              100*$i, $helixN-2;
          #$pos = $helixC((1),:) - (0.5)*$C13/(1 - cos($theta
              ));
          my @pos = $pos->list();
          printf HELIX "ATOM␣␣%5d␣␣C$AT_LET␣␣HEL%6d%12.3f%8.3
              f%8.3f␣␣1.00␣10.00\n", 100*$i,100*$i,@pos;
          $AT_LET++;
          # Then for the table
          print TABLE "Table␣of␣helix␣parameters␣for␣Helix␣
              $label,␣$file\n";
          print TABLE "NUMBER␣␣␣␣␣PITCH␣␣␣␣␣ANGLE␣␣AXIS\n";
        }

        #$pos += $axis;
        my @pos = $pos->list();
        printf HELIX "ATOM␣␣%5d␣␣C$AT_LET␣␣HEL%6d%12.3f%8.3f
            %8.3f␣␣1.00␣10.00\n", 100*$i+$nCur+1,100*$i,@pos;
        my @axis = $axis->norm()->list();
        printf TABLE (”%6d%10.2f%10.2f␣␣%7.5f%9.5f%9.5f\n”,
            $range[$nCur],
                        $pitch->at(0,0),($theta->at(0,0))
                            *180/3.14159, @axis);
        $AT_LET++;
      }

    # Print out average pitch, angle, and axis
      my $avaxis = $avg_axis->norm();
      $avg_pitch /= $helixN - 3;
      $avg_theta /= $helixN - 3;
      print TABLE ”————————————————————————\n”;
      printf TABLE (”AVER␣␣%10.2f%10.2f␣␣%7.5f%9.5f%9.5f\n\n”
          , $avg_pitch->at(0,0),
                      ($avg_theta->at(0,0))*180/3.14159,
                        $avaxis->list());


      $label++; $i++;

    # By now the graphical representations are calculated.
    # While cute, these are not particularly useful.  They
    # do provide a good check that things have not wandered
    # way off into never-never land.
    }
  }

  # Vector cross product, only in 3D.  It would be nice in
  # the long run to make a christoffel symbol generator.
  sub cross {
    my ($vec1,$vec2) = @_;
    my $matrix = pdl[ [0,              -$vec2->at(2), $vec2->
        at(1)],
                      [$vec2->at(2),  0,              -$vec2->
                        at(0)],
                      [-$vec2->at(1), $vec2->at(0),   0  ]];
    return $vec1 x $matrix;
  }
```

B.7 magdiff

A convenient function for magnifying small structural changes.

```
#!/usr/bin/perl -w

use strict;
use pdbtools;
use PDL;
use PDL::NiceSlice;

die "Usage: magdiff vetflags factor reference.pdb different
    .pdb\n" unless @ARGV==4;

my $pdb2 = pop @ARGV;
my $pdb1 = pop @ARGV;
my $mag = pop @ARGV;
my @vet = split /\s+/,$ARGV[0];

my $pdbRef1 = read_pdb($pdb1);
my $pdbRef2 = read_pdb($pdb2);

my ($vRef1, $vRef2) = vet_pdb($pdbRef1,$pdbRef2,@vet);
my $coords1 = get_coords($vRef1);
my $coords2 = get_coords($vRef2);

my $vectorDiff = $coords2 - $coords1;
my $newcoords = $coords1 + $mag*$vectorDiff;

#print $newcoords;

write_new_coords($newcoords,$vRef2);

write_pdb($vRef2,'magdiff.pdb');
```

B.8 Master figure generation script

This script calls many of the Perl scripts listed above, and should provide a useful example of how to integrate this scripts with other commonly available crystallographic and plotting software.

```
#!/bin/bash

# This is the master script used to do all alignment and
   figure making, except
# for figures made with pymol (although this script
   generates all necessary
# objects (maps, pdb files, magnified difference pdb files)
    needed for PyMOL images
# this script will also do all of the integration of
# electron densities in the cavities.

# Things missing: cavity volumes, side chain alignments and
    rmsd values.

# Link to all necessary files (don't make more copies!!!)
# Not currently in use (i.e. I'll make copies and burn
   memory)

# Wierd quirk: there's only one wt 0k structure, so, make a
    copy and average
# it with itself!
cp wt0k_a.pdb wt0k_fake.pdb;

# Align and average comparable structures

multipose ”MC” ”none” mt0k_*.pdb; avstruct mt0kav.pdb mOut
   *.pdb; rm mOut*.pdb
#multipose ”MC” ”none” mt1k_*.pdb; avstruct mt1kav.pdb mOut
   *.pdb; rm mOut*.pdb
multipose ”MC” ”none” mt2k_*.pdb; avstruct mt2kav.pdb mOut
   *.pdb; rm mOut*.pdb
multipose ”MC” ”none” wt0k_*.pdb; avstruct wt0kav.pdb mOut
   *.pdb; rm mOut*.pdb
#multipose ”MC” ”none” wt1k_*.pdb; avstruct wt1kav.pdb mOut
   *.pdb; rm mOut*.pdb
multipose ”MC” ”none” wt2k_*.pdb; avstruct wt2kav.pdb mOut
   *.pdb; rm mOut*.pdb
```

```
multipose "MC␣82-162" "none" mt0k_*.pdb; avstruct mt0kavc.
   pdb mOut*.pdb; rm mOut*.pdb
#multipose "MC 82-162" "none" mt1k_*.pdb; avstruct mt1kavc.
   pdb mOut*.pdb; rm mOut*.pdb
multipose "MC␣82-162" "none" mt2k_*.pdb; avstruct mt2kavc.
   pdb mOut*.pdb; rm mOut*.pdb
multipose "MC␣82-162" "none" wt0k_*.pdb; avstruct wt0kavc.
   pdb mOut*.pdb; rm mOut*.pdb
#multipose "MC 82-162" "none" wt1k_*.pdb; avstruct wt1kavc.
   pdb mOut*.pdb; rm mOut*.pdb
multipose "MC␣82-162" "none" wt2k_*.pdb; avstruct wt2kavc.
   pdb mOut*.pdb; rm mOut*.pdb

# Don't need the fake anymore
rm wt0k_fake.pdb

# Align the high P structures onto the ambient structures,
   reuse names
superpose -Tr -vMC -v1-162 -1mt2kav.pdb -2mt0kav.pdb; mv
   default.pdb mt2kav.pdb
superpose -Tr -vMC -v1-162 -1mt1kav.pdb -2mt0kav.pdb; mv
   default.pdb mt1kav.pdb
superpose -Tr -vMC -v1-162 -1wt2kav.pdb -2wt0kav.pdb; mv
   default.pdb wt2kav.pdb
superpose -Tr -vMC -v1-162 -1wt1kav.pdb -2wt0kav.pdb; mv
   default.pdb wt1kav.pdb

superpose -Tr -vMC -v82-162 -1mt2kavc.pdb -2mt0kavc.pdb; mv
     default.pdb mt2kavc.pdb
superpose -Tr -vMC -v82-162 -1mt1kavc.pdb -2mt0kavc.pdb; mv
     default.pdb mt1kavc.pdb
superpose -Tr -vMC -v82-162 -1wt2kavc.pdb -2wt0kavc.pdb; mv
     default.pdb wt2kavc.pdb
superpose -Tr -vMC -v82-162 -1wt1kavc.pdb -2wt0kavc.pdb; mv
     default.pdb wt1kavc.pdb

# Calculate helix geometry using helixcalc: blah.pdb ->
   blah_hx.pdb and blah_hx.tab
#helixcalc -h ~/rcsb/1L63.pdb mt0kavc.pdb
#helixcalc -h ~/rcsb/1L63.pdb mt1kavc.pdb
#helixcalc -h ~/rcsb/1L63.pdb mt2kavc.pdb
#mkdir helix
#mv *_hx.pdb helix/
```

```
#Don't really need the tables currently
#rm *.tab

# Use magdiff to generate magnified difference structures.
magdiff "1-162" 5 mt0kavc.pdb mt2kavc.pdb; mv magdiff.pdb
   mt2kavc.m5.pdb
#magdiff "1-162" 5 mt0kavc.pdb mt1kavc.pdb; mv magdiff.pdb
   mt1kavc.m5.pdb
magdiff "1-162" 5 mt0kavc.pdb mt1kavc.pdb; mv magdiff.pdb
   mt1kavc.m5.pdb
mkdir mag
mv *.m5.pdb mag/

#Store plots in another directory
mkdir plots

# To avoid confusion, add current directory to the title
# Will need to be removed in final version.
curdir=`pwd`

#Make the 1-162 MC rmsd plot
rmsd "CA" mt0kav.pdb mt2kav.pdb; mv rmsd.dat tmp.mt.dat
rmsd "CA" wt0kav.pdb wt2kav.pdb; mv rmsd.dat  tmp.wt.dat
smooth 1 5 7 tmp.mt.dat > mt.sm.dat #Col 1: res #, Col 5 is
    displacement, box window 7
smooth 1 5 7 tmp.wt.dat > wt.sm.dat

gnuplot <<EOF
set title "$curdir"
set key top right
set terminal postscript landscape eps enhanced lw 2 "
   Helvetica" 14
set output "diffs.eps"


set xlabel "Residue␣number"
set ylabel "Average␣C␣{/Symbol=12␣a}␣displacement"

plot 'mt.sm.dat' u 1:2 t "L99A" w line, 'wt.sm.dat' u 1:2 t
    "WT*" w line

set terminal x11
```

```
EOF

# For some reason, gnuplot messes up the .eps files in the
   current directory...
#so move the output
mv diffs.eps plots/

#Make the 82-162 MC rmsd plot
rmsd "CA" mt0kavc.pdb mt2kavc.pdb; mv rmsd.dat tmp.mt.dat
rmsd "CA" wt0kavc.pdb wt2kavc.pdb; mv rmsd.dat tmp.wt.dat
smooth 1 5 7 tmp.mt.dat > mt.sm.dat #Col 1: res #, Col 5 is
    displacement, box window 7
smooth 1 5 7 tmp.wt.dat > wt.sm.dat

gnuplot <<EOF
#set title "$curdir"
set key top right
set terminal postscript landscape eps enhanced lw 2 "
   Helvetica" 14
set output "ctdiffs.eps"

set xlabel "Residue␣number"
set ylabel "Average␣C_{/Symbol=12␣a}␣displacement"
set yrange [-0.05:0.45]
set xrange [0:164]
set ytics 0,0.1,0.4

set label "A" at 7,-0.03 center
set label "B" at 44.5,-0.03 center
set label "C" at 70,-0.03 center
set label "D" at 86,-0.03 center
set label "E" at 99.5,-0.03 center
set label "F" at 110.5,-0.03 center
set label "G" at 119,-0.03 center
set label "H" at 130,-0.03 center
set label "I" at 139,-0.03 center
set label "J" at 149,-0.03 center

plot 'mt.sm.dat' u 1:2 t "L99A" w line, 'wt.sm.dat' u 1:2 t
    "WT*" w line,\
'SS.dat' u 1:11 notitle w l lt -1 lw 7, 'SS.dat' u 2:11
   notitle w l lt -1 lw 7,\
'SS.dat' u 3:11 notitle w l lt -1 lw 7, 'SS.dat' u 4:11
   notitle w l lt -1 lw 7,\
'SS.dat' u 5:11 notitle w l lt -1 lw 7, 'SS.dat' u 6:11
   notitle w l lt -1 lw 7,\
'SS.dat' u 7:11 notitle w l lt -1 lw 7, 'SS.dat' u 8:11
   notitle w l lt -1 lw 7,\
'SS.dat' u 9:11 notitle w l lt -1 lw 7, 'SS.dat' u 10:11
   notitle w l lt -1 lw 7

set terminal x11
EOF

mv ctdiffs.eps plots/
rm *.dat

echo "Generating maps now."

# Generate difference maps and average them.
# All mtz files and pdb files need to be in the local
   directory for this to work.

diff.com . mt2k_8 mt0k_1
diff.com . mt2k_8 mt0k_a
diff.com . mt2k_8 mt0k_b
diff.com . mt2k_3 mt0k_1
diff.com . mt2k_3 mt0k_a
diff.com . mt2k_3 mt0k_b
diff.com . mt2k_1 mt0k_1
diff.com . mt2k_1 mt0k_a
diff.com . mt2k_1 mt0k_b

diff.com . mt1_5k_1 mt0k_1
diff.com . mt1_5k_1 mt0k_a
diff.com . mt1_5k_1 mt0k_b

diff.com . mt1k_6 mt0k_1
diff.com . mt1k_6 mt0k_a
diff.com . mt1k_6 mt0k_b
diff.com . mt1k_7 mt0k_1
diff.com . mt1k_7 mt0k_a
diff.com . mt1k_7 mt0k_b
diff.com . mt1k_9 mt0k_1
diff.com . mt1k_9 mt0k_a
diff.com . mt1k_9 mt0k_b

# This does the map averaging for the Fo-Fo maps.
# It requires MAPMAN, available from the Uppsala Software
   Factory

osx_mapman <<EOF

re map1 mt1k_6-mt0k_1-P1.map ccp4
re map2 mt1k_6-mt0k_a-P1.map ccp4
op map1 + map2
del map2
re map2 mt1k_6-mt0k_b-P1.map ccp4
op map1 + map2
del map2
re map2 mt1k_7-mt0k_1-P1.map ccp4
op map1 + map2
del map2
re map2 mt1k_7-mt0k_a-P1.map ccp4
op map1 + map2
del map2
re map2 mt1k_7-mt0k_b-P1.map ccp4
op map1 + map2
del map2
re map2 mt1k_9-mt0k_1-P1.map ccp4
op map1 + map2
del map2
re map2 mt1k_9-mt0k_a-P1.map ccp4
op map1 + map2
del map2
re map2 mt1k_9-mt0k_b-P1.map ccp4
op map1 + map2
del map2
div map1 9
wr map1 mt1k-0kav.map ccp4
re map2 m2k8_1.ezd ezd
op map2 * map1
in map2 102 167 -7 72 -21 57

del *
re map1 mt2k_8-mt0k_1-P1.map ccp4
re map2 mt2k_8-mt0k_a-P1.map ccp4
op map1 + map2
```

```
del map2
re map2 mt2k_8-mt0k_b-P1.map ccp4
op map1 + map2
del map2
re map2 mt2k_3-mt0k_1-P1.map ccp4
op map1 + map2
del map2
re map2 mt2k_3-mt0k_a-P1.map ccp4
op map1 + map2
del map2
re map2 mt2k_3-mt0k_b-P1.map ccp4
op map1 + map2
del map2
re map2 mt2k_1-mt0k_1-P1.map ccp4
op map1 + map2
del map2
re map2 mt2k_1-mt0k_a-P1.map ccp4
op map1 + map2
del map2
re map2 mt2k_1-mt0k_b-P1.map ccp4
op map1 + map2
del map2
div map1 9
wr map1 mt2k-0kav.map ccp4
re map2 m2k8_1.ezd ezd
op map2 * map1
in map2 102 167 -7 72 -21 57

del *
re map1 mt1_5k_1-mt0k_1-P1.map ccp4
re map2 mt1_5k_1-mt0k_a-P1.map ccp4
op map1 + map2
del map2
re map2 mt1_5k_1-mt0k_b-P1.map ccp4
op map1 + map2
del map2
div map1 3
wr map1 mt15k-0kav.map ccp4
re map2 m2k8_1.ezd ezd
op map2 * map1
in map2 102 167 -7 72 -21 57
```

EOF

```
# For neatness, move the P1 maps to another folder
mkdir P1
mv *P1.map P1/.
```

REFERENCES

1. M. Gross and R. Jaenicke. Proteins under pressure: the influence of high hydrostatic pressure on structure, function, and assembly of proteins and protein complexes. *European Journal of Biochemistry*, **221**, 617–630 (1994).

2. R. Winter and W. Dzwolak. Exploring the temperature-pressure configurational landscape of biomolecules: from lipid membranes to proteins. *Philosophical Transactions of the Royal Society A*, **363**, 537–563 (2005).

3. P. W. Bridgman. The coagulation of albumin by pressure. *Journal of Biological Chemistry*, **19**, 511–512 (1914).

4. V. V. Mozhaev, K. Heremans, J. Frank, P. Masson and C. Balny. High pressure effects on protein structure and function. *Proteins: Structure, Function and Genetics*, **24**, 81–91 (1996).

5. G. Varo and J. K. Lanyi. Effects of hydrostatic pressure on the kinetics reveal a volume increase during the bacteriorhodopsin photocycle. *Biochemistry*, **34**, 12161–12169 (1995).

6. E. B. Smith, F. Bowser-Riley, A. Daniels, I. T. Dunbar, H. C. B. and W. D. Paton. Species variation and the mechanism of pressure anaesthetic interactions. *Nature*, **311**, 56–57 (1984).

7. C. A. Royer. Revisiting volume changes in pressure-induced unfolding. *Biochimica et Biophysica Acta*, **1595**, 201–209 (2002).

8. H. Frauenfelder, N. A. Alberding, A. Ansari, D. Braunstein, B. R. Cowen, M. K. Hong, I. E. Iben, J. B. Johnson, S. Luck, M. C. Marden, J. R. Mourant,

P. Ormos, L. Reinisch, R. Scholl, A. Schulte, E. Shyamsunder, L. B. Sorensen, P. J. Steinbach, A. Xie, R. D. Young and K. T. Yue. Proteins and pressure. *Journal of Physical Chemistry*, **94**, 1024–1037 (1990).

9. P. K. Urayama. *Techniques for High Pressure Macromolecular Crystallography and the Effects of Pressure on the Structure of Sperm Whale Myoglobin*. Ph.D. thesis, Princeton University (2001).

10. H. Frauenfelder and B. H. McMahon. Energy landscape and fluctuations in proteins. *Annals of Physics*, **9**, 665–667 (2000).

11. C. E. Kundrot and F. M. Richards. Collection and processing of x-ray diffraction data from protein crystals at high pressure. *Journal of Applied Crystallography*, **19**, 208–213 (1986).

12. C. E. Kundrot and F. M. Richards. Crystal structure of hen egg-white lysozyme at a hydrostatic pressure of 1000 atmospheres. *Journal of Molecular Biology*, **193**, 157–170 (1987).

13. M. Refaee, T. Tezuka, K. Akasaka and M. P. Williamson. Pressure-dependent changes in the solution structure of hen egg-white lysozyme. *Journal of Molecular Biology*, **327**, 857–865 (2003).

14. P. Urayama, G. N. Phillips and S. M. Gruner. Probing substates in sperm whale myoglobin using high-pressure crystallography. *Structure*, **10**, 51–60 (2002).

15. G. Mann and J. Hermans. Modeling protein-small molecule interactions: structure and thermodynamics of noble gases binding in a cavity in mutant phage T4 lysozyme L99A. *J Mol Biol*, **302**, 979–989 (2000).

16. K. A. Dill. Dominant forces in protein folding. *Biochemistry*, **29**, 7133–7155 (1990).

17. A. Zipp and W. Kauzmann. Pressure denaturation of metmyoglobin. *Biochemistry*, **12**, 4217–4228 (1973).

18. W. Kauzmann. Thermodynamics of unfolding. *Nature*, **325**, 763–764 (1987).

19. M. S. Cheung, A. E. Garcia and J. N. Onuchic. Protein folding mediated by solvation: water expulsion and formation of the hydrophobic core after collapse. *Proceedings of the National Academy of Sciences of the USA*, **99**, 685–690 (2002).

20. J. Woenckhaus, R. Kohling, P. Thiyagarajan, K. C. Littrell, S. Seifert, C. A. Royer and R. Winter. Pressure-jump small-angle x-ray scattering detected kinetics of staphylococcal nuclease folding. *Biophysical Journal*, **80**, 1518–1523 (2001).

21. R. Zhou, X. Huang, C. J. Margulis and B. J. Berne. Hydrophobic collapse in multidomain protein folding. *Science*, **305**, 1605–1609 (2004).

22. D. Harries, D. C. Rau and V. A. Parsegian. Solutes probe hydration in specific association of cyclodextrin and adamantane. *Journal of the American Chemical Society*, **127**, 2184–2190 (2005).

23. G. A. Papoian, J. Ulander, M. P. Eastwood, Z. Luthey-Schulten and P. G. Wolynes. Water in protein structure prediction. *Proc Natl Acad Sci U S A*, **101**, 3352–3357 (2004).

24. C. U. Kim, R. Kapfer and S. M. Gruner. High-pressure cooling of protein crystals without cryoprotectants. *Acta Crystallographica*, **D61**, 881–890 (2005).

25. K. A. Dill. Polymer principles and protein folding. *Protein Science*, **8**, 1166–1180 (1999).

26. W. Kauzmann, A. Bodanszky and J. Rasper. Volume changes in protein reactions. ii. comparison of ionization reactions in proteins and small molecules. *Journal of the American Chemical Society*, **84**, 1777–1788 (1962).

27. L.-N. Lin, J. F. Brandts, J. M. Brandts and V. Plotnikov. Determination of the volumetric properties of proteins and other solutes using pressure perturbation calorimetry. *Analytical Biochemistry*, **302**, 144–160 (2002).

28. Y. O. Kamatari, R. Kitahara, H. Yamada, S. Yokoyama and K. Akasaka. High-pressure NMR spectroscopy for characterizing folding intermediates and denatured states of proteins. *Methods*, **34**, 133–143 (2004).

29. M. P. Williamson, K. Akasaka and M. Refaee. The solution structure of bovine pancreatic trypsin inhibitor at high pressure. *Protein Sci*, **12**, 1971–1979 (2003).

30. G. Panick, R. Malessa, R. Winter, G. Rapp, K. J. Frye and C. A. Royer. Structural characterization of the pressure-denatured state and unfolding/refolding kinetics of staphylococcal nuclease by synchrotron small-angle X-ray scattering and Fourier-transform infrared spectroscopy. *J Mol Biol*, **275**, 389–402 (1998).

31. A. Paliwal, D. Asthagiri, D. P. Bossev and M. E. Paulaitis. Pressure denat-

uration of staphylococcal nuclease studied by neutron small-angle scattering and molecular simulation. *Biophysical Journal*, **87**, 3479–3492 (2004).

32. G. M. Clore and C. D. Schwieters. Theoretical and computational advances in biomolecular NMR spectroscopy. *Current Opinion in Structural Biology*, **12**, 146–153 (2002).

33. L. E. Kay. NMR studies of protein structure and dynamics. *Journal of Magnetic Resonance*, **173**, 193–207 (2005).

34. T. K. Hitchens and R. G. Bryant. Pressure dependence of amide hydrogen-deuterium exchange rates for individual sites in T4 lysozyme. *Biochemistry*, **37**, 5878–5887 (1998).

35. F. A. A. Mulder, A. Mittermaier, B. Hon, F. W. Dahlquist and L. E. Kay. Studying excited states of proteins by NMR spectroscopy. *Nature Structural Biology*, **8**, 932–935 (2001). See also comments in the same issue: p. 909, pp. 912-914.

36. R. Kitahara, S. Yokoyama and K. Akasaka. NMR snapshots of a fluctuating protein structure: ubiquitin at 30 bar–3 kbar. *Journal of Molecular Biology*, **347**, 277–285 (2005).

37. C. R. Cantor and P. R. Schimmel. *Biophysical Chemistry* (W. H. Freeman and Company, 1980).

38. P. I. Haris and D. Chapman. The conformational analysis of peptides using Fourier transform IR spectroscopy. *Biopolymers*, **37**, 251–263 (1995).

39. M. L. Quillin, W. A. Breyer, I. J. Griswold and B. W. Matthews. Size *versus* polarizability in protein-ligand interactions: binding of noble gases within engineered cavities in phage t4 lysozyme. *Journal of Molecular Biology*, **302**, 955–977 (2000).

40. B. W. Matthews and S. J. Remington. The three dimensional structure of the lysozyme from bacteriophage T4. *Proc Natl Acad Sci U S A*, **71**, 4178–4182 (1974).

41. W. F. Anderson, M. G. Grutter, S. J. Remington, L. H. Weaver and B. W. Matthews. Crystallographic determination of the mode of binding of oligosaccharides to T4 bacteriophage lysozyme: implications for the mechanism of catalysis. *Journal of Molecular Biology*, **147**, 523–543 (1981).

42. M. G. Grutter and B. W. Matthews. Amino acid substitutions far from the active site of bacteriophage T4 lysozyme reduce catalytic activity and suggest that the C-terminal lobe of the enzyme participates in substrate binding. *Journal of Molecular Biology*, **154**, 525–535 (1982).

43. M. G. Grutter, T. M. Gray, L. H. Weaver, T. A. Wilson and B. W. Matthews. Structural studies of mutants of the lysozyme of bacteriophage T4. The temperature-sensitive mutant protein Thr157→Ile. *Journal of Molecular Biology*, **197**, 315–329 (1987).

44. T. Alber, J. A. Bell, D. P. Sun, H. Nicholson, J. A. Wozniak, S. Cook and B. W. Matthews. Replacements of Pro86 in phage T4 lysozyme extend an alpha-helix but do not alter protein stability. *Science*, **239**, 631–635 (1988).

45. M. Matsumura, W. J. Becktel and B. W. Matthews. Hydrophobic stabilization in T4 lysozyme determined directly by multiple substitutions of Ile 3. *Nature*, **334**, 406–410 (1988).

46. H. Nicholson, W. J. Becktel and B. W. Matthews. Enhanced protein thermostability from designed mutations that interact with alpha-helix dipoles. *Nature*, **336**, 651–656 (1988).

47. A. E. Eriksson, W. A. Baase, X.-J. Zhang, M. Blaber, E. P. Baldwin and B. W. Matthews. Response of a protein structure to cavity-creating mutations and its relation to the hydrophobic effect. *Science*, **255**, 178–183 (1992).

48. A. E. Eriksson, W. A. Baase and B. W. Matthews. Similar hydrophobic replacements of Leu99 and Phe153 within the core of T4 lysozyme have different structural and thermodynamic consequences. *Journal of Molecular Biology*, **229**, 747–769 (1993).

49. J. Xu, W. A. Baase, E. Baldwin and B. W. Matthews. The response of t4 lysozyme to large-to-small substitutions within the core and its relation to the hydrophobic effect. *Protein Science*, **7**, 158–177 (1998).

50. A. Morton, W. A. Baase and B. W. Matthews. Energetic origins of specificity of ligand binding in an interior nonpolar cavity of T4 lysozyme. *Biochemistry*, **34**, 8564–8575 (1995).

51. W. Kauzmann. Some factors in the interpretation of protein denaturation. *Adv. Protein Chem.*, **14**, 1–63 (1959).

52. L. Zhang and J. Hermans. Hydrophilicity of cavities in proteins. *Proteins*, **24**, 433–438 (1996).

53. G. Nemethy and H. A. Scheraga. Structure of water and hydrophobic bonding in proteins. I. A model for the thermodynamic properties of liquid water. *Journal of Chemical Physics*, **36**, 3382–3400 (1962).

54. G. Nemethy and H. A. Scheraga. Structure of water and hydrophobic bonding in proteins. II Model for the thermodynamic properties of aqueous solutions of hydrocarbons. *Journal of Chemical Physics*, **36**, 3401–3417 (1962).

55. J. H. Griffith and H. A. Scheraga. Statistical thermodynamics of aqueous solutions. I. Water structure, solutions with non-polar solutes, and hydrophobic interactions. *Journal of Molecular Structure*, **682**, 97–113 (2004).

56. G. Hummer, S. Garde, A. E. Garcia, M. E. Paulaitis and L. R. Pratt. Hydrophobic effects on a molecular scale. *Journal of Physical Chemistry B*, **102**, 10469–10482 (1998).

57. G. Hura, D. Russo, R. M. Glaeser, T. Head-Gordon, M. Krack and M. Parrinello. Water structure as a function of temperature from X-ray scattering experiments and *ab initio* molecular dynamics. *Physical Chemistry Chemical Physics*, **5**, 1981–1991 (2003).

58. H. S. Ashbaugh, D. Asthagiri, L. R. Pratt and S. B. Rempe. Hydration of krypton and consideration of clathrate models of hydrophobic effects from the perspective of quasi-chemical theory. *Biophysical Chemsitry*, **105**, 323–338 (2003).

59. G. Hummer, S. Garde, A. E. Garcia, M. E. Paulaitis and L. R. Pratt. The pressure dependence of hydrophobic interactions is consistent with the ob-

served pressure denaturation of proteins. *Proceedings of the National Academy of Sciences of the USA*, **95**, 1552–1555 (1998).

60. L. Maibaum, A. R. Dinner and D. Chandler. Micelle formation and the hydrophobic effect. *Journal of Physical Chemistry B*, **108**, 6778–6781 (2004).

61. K. Lum, D. Chandler and J. D. Weeks. Hydrophobicity at small and large length scales. *Journal of Physical Chemistry*, **103**, 4570–4577 (1999).

62. L. R. Pratt. Molecular theory of hydrophobic effects: "She is too mean to have her name repeated.". *Annual Reviews of Physical Chemistry*, **53**, 409–436 (2002).

63. L. R. Pratt and A. Pohorille. Theory of hydrophobicity: Transient cavities in molecular liquids. *Proceedings of the National Academy of Sciences of the USA*, **89**, 2995–2999 (1992).

64. L. R. Pratt and D. Chandler. Theory of the hydrophobic effect. *Journal of Chemical Physics*, **67**, 3683–3704 (1977).

65. R. L. Baldwin and N. Muller. Relation between the convergence temperatures T_h^* and T_s^* in protein unfolding. *Proceedings of the National Academy of Sciences of the USA*, **89**, 7110–7113 (1992).

66. R. L. Baldwin. Temperature dependence of the hydrophobic interaction in protein folding. *Proceedings of the National Academy of Sciences of the USA*, **83**, 8069–8072 (1986).

67. B. Honig. Protein folding: from the Levinthal paradox to structure prediction. *Journal of Molecular Biology*, **293**, 283–293 (1999).

68. F. Xu and T. A. Cross. Water: Foldase activity in catalyzing polypeptide conformational rearrangements. *Proceedings of the National Academy of Sciences of the USA*, **96**, 9057–9061 (1999).

69. A. M. Buckle, P. Cramer and A. R. Fersht. Structural and energetic responses to cavity-creating mutations in hydrophobic cores: observation of a buries water molecule and the hydrophilic nature of such hydrophobic cavities. *Biochemistry*, **35**, 4298–4305 (1996).

70. F. M. Richards. The interpretation of protein structures: Total volume, group volume distributions and packing density. *Journal of Molecular Biology*, **82**, 1–14 (1974).

71. D. Paschek and A. E. Garcia. Reversible temperature and pressure denaturation of a protein fragment: A replica exchange molecular dynamics study. *Physical Review Letters*, **93** (2004).

72. S. Vaitheeswaran, H. Yin, J. C. Rasaiah and G. Hummer. Water clusters in nonpolar cavities. *Proceedings of the National Academy of Sciences of the USA*, **101**, 17002–17005 (2004).

73. S. Vaitheeswaran, J. C. Rasaiah and G. Hummer. Electric field and temperature effects on water in the narrow nonpolar pores of carbon nanotubes. *Journal of Chemical Physics*, **121**, 7955–7965 (2004).

74. G. J. Vidugiris, J. L. Markley and C. A. Royer. Evidence for a molten globule-like transition state in protein folding form determination of activation volumes. *Biochemistry*, **34**, 4909–4912 (1995).

75. G. Desai, G. Panick, M. Zein, R. Winter and C. A. Royer. Pressure-jump studies on the folding/unfolding of trp repressor. *Journal of Molecular Biology*, **288**, 461–75 (1999).

76. R. Kitahara, C. Royer, H. Yamada, M. Boyer, J.-L. Saldana, K. Akasaka and C. Roumestand. Equilibrium and pressure-jump relaxation studies of the conformational transitions of P13MTCP1. *J Mol Biol*, **320**, 609–628 (2002).

77. H. Herberhold and R. Winter. Temperature- and pressure-induced unfolding and refolding of ubiquitin: a static and kinetic Fourier transform infrared spectroscopy study. *Biochemistry*, **41**, 2396–2401 (2002).

78. G. A. Papoian, J. Ulander and P. G. Wolynes. Role of water mediated interactions in protein-protein recognition landscapes. *J Am Chem Soc*, **125**, 9170–9178 (2003).

79. C. Amovilli and F. M. Floris. Solubility of water in liquid hydrocarbons: a bridge between the polarizable continuum model and the mobile order theory. *Physical Chemistry Chemical Physics*, **5**, 363–368 (2003).

80. B. Garcia-Moreno, J. J. Dwyer, A. G. Gittis, E. E. Lattman, D. S. Spencer and W. E. Stites. Experimental measurement of the effective dielectric in the hydrophobic core of a protein. *Biophysical Chemistry*, **64**, 211–224 (1997).

81. C. A. Fitch, D. A. Karp, K. K. Lee, W. E. Stites, E. E. Lattman and B. Garcia-Moreno. Experimental pK_a values of buried residues: Analysis with continuum methods and role of water penetration. *Biophysical Journal*, **82**, 3289–3304 (2002).

82. J. J. Dwyer, A. G. Gittis, A. A. Karp, E. E. Lattman, D. S. Spencer, W. E. Stites and B. Garcia-Moreno. High apparent dielectric constants in the interior of a protein reflect water penetration. *Biophysical Journal*, **79**, 1610–1620 (2000).

83. V. P. Denisov, J. L. Schlessman, B. Garcia-Moreno E. and B. Halle. Stabilization of internal charges in a protein: water penetration or conformational change? *Biophysical Journal*, **87**, 3982–3994 (2004).

84. A. A. Rashin, M. Iofin and B. Honig. Internal cavities and buried waters in globular proteins. *Biochemistry*, **25**, 3619–3625 (1986).

85. J. A. Ernst, R. T. Clubb, H. X. Zhou, A. M. Gronenborn and G. M. Clore. Demonstration of positionally disordered water within a protein hydro phobic cavity by NMR. *Science*, **267**, 1813–1817 (1995).

86. B. W. Matthews, A. G. Morton and F. W. Dahlquist. Use of NMR to detect water within nonpolar protein cavities. *Science*, **270**, 1847–1849 (1995). Comment.

87. B. Yu, M. Blaber, A. M. Gronenborn, G. M. Clore and D. L. Caspar. Disordered water within a hydrophobic cavity visualized by x-ray crystallography. *Proceedings of the National Academy of Sciences of the USA*, **96**, 103–108 (1999).

88. K. Takano, Y. Yamagata and K. Yutani. Buried water molecules contribute to the conformational stability of a protein. *Protein Engineeering*, **16**, 5–9 (2003).

89. M. Prevost. Dynamics of water molecules buried in cavities of apolipoprotein e studied by molecular dynamics simulations and continuum electrostatic calculations. *Biopolymers*, **75**, 196–207 (2004).

90. M. Wikstrom. Proton translocation by bacteriorhodopsin and heme-copper oxidases. *Current Opinion in Structural Biology*, **8**, 480–488 (1998).

91. B. Schobert, L. S. Brown and J. K. Lanyi. Crystallographic intermediates of structures of the m and n bacteriorhodopsin assembly of a hydrogen-bonded chain of water molecules between asp-96 and the retinal schiff base. *J. Mol. Biol.*, **330**, 553–570 (2003).

92. I. Schlichting, J. Berendzen, K. Chu, A. M. Stock, S. A. Maves, D. E. Benson, R. M. Sweet, D. Ringe, G. A. Petsko and S. G. Sligar. The catalytic pathway of cytochrome p450cam at atomic resolution. *Science*, **287**, 1615–1622 (2000).

93. J. Liang and K. A. Dill. Are proteins well-packed? *Biophysical Journal*, **81**, 751–766 (2001).

94. P. Cioni, E. de Waal, G. W. Canters and G. B. Strambini. Effects of cavity-forming mutations on the internal dynamics of azurin. *Biophysical Journal*, **86**, 1149–1159 (2004).

95. F. A. Mulder, B. Hon, A. Mitternaier, F. W. Dahlquist and L. E. Kay. Slow internal dynamics in proteins: application of NMR relaxation dispersion spectroscopy to mehtyl groups in a cavity mutant of T4 lysozyme. *Journal of the American Chemical Society*, **124**, 1443–1451 (2002).

96. F. H. O. Osterberg. *Induced changes in the diffuse x-ray scattering background from protein crystals*. Ph.D. thesis, Princeton University (1996).

97. P. T. C. So. *High pressure effects on the mesophases of lipid-water systems*. Ph.D. thesis, Princeton University (1992).

98. I. L. Spain and J. Paauwe, eds. *High Pressure Technology Volume 1: Equipment Design, Materials and Properties* (Dekker, New York, 1977).

99. R. Fourme, I. Ascone, R. Kahn, M. Mezouar, P. Bouvier, E. Girard, T. Lin and J. E. Johnson. Opening the high-pressure domain beyond 2 kbar to protein and virus crystallography–technical advance. *Structure (Camb)*, **10**, 1409–1414 (2002).

100. E. Girard, R. Kahn, I. Ascone, M. Mezouar, A.-C. Dhaussy, T. Lin, J. E. Johnson and R. Fourme. A new dimension in structural biology: fully fledged high-pressure macromolecular crystallography. *High Pressure Research*, **24**, 173–182 (2004).

101. E. Girard, R. Kahn, M. Mezouar, A. C. Dhaussy, T. Lin, J. E. Johnson and R. Fourme. The first crystal structure of a macromolecular assembly under high pressure: CpMV at 330 MPa. *Biophys J*, **88**, 3562–3571 (2005).

102. U. F. Thomanek, F. Parak, R. L. Mössbauer, H. Formanek, P. Schwager and W. Hoppe. Freesing of myoglobin crystals at high pressure. *Acta Crystallographica A*, **29**, 263–265 (1973).

103. M. Chai, J. M. Brown and L. J. Slutsky. The elastic constants of an aluminous orthopyroxene to 12.5 GPa. *Journal of Geophysical Research*, **102**, 12333–12340 (1997).

104. E. H. Abramson, L. J. Slutsky, M. D. Harrell and J. M. Brown. Speed of

sound and equation of state for fluid oxygen to 10 GPa. *Journal of Chemical Physics*, **110**, 10493 (1999).

105. J. Drenth. *Principles of Protein X-ray Crystallography* (Springer, 1999), 2nd edition.

106. W. J. Becktel and W. A. Baase. Thermal denaturation of bacteriophage t4 lysozyme at neutral ph. *Biopolymers*, **26**, 619–623 (1987).

107. Certified Scientific Software website: www.certif.com.

108. J. Als-Nielsen and D. McMorrow. *Elements of Modern X-ray Physics* (John Wiley & Sons, Ltd., 2001).

109. J. J. Sakurai. *Advanced Quantum Mechanics* (Addison-Wesley, 1976).

110. Z. Sturm. Dynamic structure factor: an introduction. *Zeitschrift fur Naturforsch*, **48a**, 233–242 (1993).

111. B. W. Batterman and H. Cole. Dynamical diffraction of X rays by perfect crystals. *Reviews of Modern Physics*, **36**, 681–717 (1964).

112. P. Abbamonte, K. D. Finkelstein, M. D. Collins and S. M. Gruner. Imaging density disturbances in water with a 41.4-attosecond time resolution. *Physical Review Letters*, **92** (2004).

113. J. Drenth. Basic Crystallography. In M. G. Rossman and E. Arnold, eds., *International Tables for Crystallography, Vol. F. Crystallography of Biological Macromolecules*, chapter 17, pages 353–356 and 366–367 (Dordrecth: Kluwer Academic Publishers, The Netherlands, 2001).

114. Z. Otwinowski and W. Minor. Processing of x-ray diffraction data collected in oscillation mode. In C. W. Carter, Jr. and R. M. Sweet, eds., *Methods in Enzymology*, volume 276, pages 307–326 (Academic Press, 1997).

115. G. C. Fox and K. C. Holmes. An alternative method of solving the layer scaling equations of Hamilton, Rollett and Sparks. *Acta Crystallographica*, **20**, 886–891 (1966).

116. V. Y. Lunin and T. P. Skovoroda. *R*-free Likelihood-Based Estimates of Errors for Phases Calculated from Atomic Models. *Acta Crystallographica*, **A51**, 880–887 (1995).

117. M. Falcioni and M. W. Deem. A biased monte carlo scheme for zeolite structure solution. *Journal of Chemical Physics*, **110**, 1754–1766 (1999).

118. G. N. Murshudov, A. A. Vagin and E. J. Dodson. Refinement of macromolecular structures by the maximum-likelihood method. *Acta Crystallographica*, **D53**, 240–255 (1997).

119. T. A. Jones, J.-Y. Zou and S. W. Cowan. Improved methods for building protein models in electron density maps and the location of errors in these models. *Acta Crystallographica*, **A47**, 110–119 (1991).

120. R. J. Read. Structure-factor probabilities for related structures. *Acta Crystallographica*, **A46**, 900–912 (1990).

121. N. S. Pannu and R. J. Read. Improved structure refinement through maximum likelihood. *Acta Crystallographica*, **A52**, 659–668 (1996).

122. R. Srinivasan and G. N. Ramachandran. Probability distribution connected with structure amplitudes of two related crystals. V. The effect of errors in the atomic coordinates on the distribution of observed and calculated structure factors. *Acta Crystallographica*, **19**, 1008–1014 (1965).

123. R. J. Read. Improved fourier coefficients for maps using phases from partial structures with errors. *Acta Crystallographica*, **A42**, 140–149 (1986).

124. D. W. J. Cruickshank. Remarks about protein structure precision. *Acta Crystallographica*, **D55**, 583–601 (1999).

125. W. H. Press, B. P. Flannery, S. A. Teukolsky and W. T. Vetterling. *Numerical Recipes: the Art of Scientific Computing* (Cambridge University Press, 1986).

126. E. W. Weisstein. ”nonlinear least squares fitting” from mathworld–a wolfram web resource.
mathworld.wolfram.com/NonlinearLeastSquaresFitting.htm.

127. D. W. J. Cruickshank. The required precision of intensity measurements for single-crystal analysis. *Acta Crystallographica*, **13**, 774–777 (1960).

128. R. A. Engh and R. Huber. Accurate bond and angle parameters for x-ray protein structure. *Acta Crystallographica*, **A47**, 392–400 (1991).

129. M. A. DePristo, P. I. de Bakker and T. L. Blundell. Heterogeneity and inaccuracy in protein structures solved by x-ray crystallography. *Structure*, **12**, 831–838 (2004).

130. K. Lindorff-Larsen, R. B. Best, M. A. Depristo, C. M. Dobson and M. Vendruscolo. Simultaneous determination of protein structure and dy-

namics. *Nature*, **433**, 128–132 (2005). Supplementary information at: www.nature.com/nature/journal/v433/n7022/suppinfo/nature03199.html.

131. A.-N. Bondar, S. Fischer, J. C. Smith, M. Elstner and S. Suhai. Key role of electrostatic interactions in bacteriorhodopsin proton transfer. *Journal of the American Chemical Society*, **126**, 14668–14677 (2004).

132. C.-I. Brändén and T. A. Jones. Between objectivity and subjectivity. *Nature*, **343**, 687–689 (1990).

133. A. T. Brunger. Free *R* value: a novel statistical quantity for assessing the accuracy of crystal structures. *Nature*, **355**, 472–475 (1992).

134. G. J. Kleywegt and T. A. Jones. Phi/Psi-chology: Ramachandran revisited. *Structure*, **4**, 1395–1400 (1996).

135. G. J. Kleywegt, J. Y. Zou, M. Kjeldgaard and T. A. Jones. Around O. In M. G. Rossman and E. Arnold, eds., *International Tables for Crystallography, Vol. F. Crystallography of Biological Macromolecules*, chapter 17, pages 353–356 and 366–367 (Dordrecth: Kluwer Academic Publishers, The Netherlands, 2001).

136. V. A. Feher, E. P. Baldwin and F. W. Dahlquist. Access of ligands to cavities within the core of a protein is rapid. *Nature Structural Biology*, **3**, 516–521 (1996).

137. S. Kumar and M. Bansal. Geometrical and sequence characteristics of alpha-helices in globular proteins. *Biophys J*, **75**, 1935–1944 (1998).

138. M. L. Connolly. Solvent-accessible surfaces of proteins and nucleic acids. *Science*, **221**, 709–713 (1983).

139. G. J. Kleywegt and T. A. Jones. Detection, delineation, measurement and display of cavities in macromolecular structures. *Acta Crystallographica*, **D50**, 178–185 (1994).

140. J. Liang, H. Edelsbrunner, P. Fu and P. V. Sudhakar. Analytical shape computation of macromolecules: I. molecular area and volume through alpha shape. *Proteins: Structure, Function and Genetics*, **33**, 1–17 (1998).

141. J. Liang, H. Edelsbrunner, P. Fu and P. V. Sudhakar. Analytical shape computation of macromolecules: Ii. inaccessible cavities in proteins. *Proteins: Structure, Function and Genetics*, **33**, 18–29 (1998).

142. M. F. Sanner, A. J. Olson and J.-C. Spehner. Fast and robust computation of molecular surfaces. In *Proceedings of the eleventh annual ACM symposium on Computational Geometry* (1995).

143. V. Luzatti. Resolution d'une structure cristalline lorsque les positions d'une partie des atomes sont connues: traitement statistique. *Acta Crystallographica*, **6**, 142–152 (1953).

144. D. A. Pearlman, D. A. Case, J. W. Caldwell, W. S. Ross, T. E. Cheatham III, S. DeBolt, D. Ferguson, G. Seibel and P. Kollman. AMBER, a package of computer programs for applying molecular mechanics, normal mode analysis, molecular dynamics and free energy calculations to simulate the structural and energetic properties of molecules. *Computer Physics Communications*, **91**, 1–41 (1995).

145. D. A. Case, D. A. Pearlman, J. W. Caldwell, T. E. I. Cheatham, W. S. Ross, C. L. Simmerling, T. A. Darden, K. M. Merz, R. B. Stanton, A. L. Cheng, J. J. Vincent, M. Crowley, T. B., R. J. Radmer, Y. Duan, P. J., I. Massova, G. L. Seibel, U. C. Singh, P. K. Weiner and P. A. Kollman. *AMBER 6*. University of California, San Francisco (1999).

146. D. Frankel and B. Smit. *Understanding molecular simulation: From algorithms to applications* (Academic Press, 2002), second edition.

147. J. W. Ponder and D. A. Case. Force fields for protein simulations. *Advances in Protein Chemistry*, **66**, 27–85 (2003).

148. W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz, D. M. Fergurson, D. C. Spellmeyer, T. Fox, J. W. Caldwell and P. A. Kollman. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *Journal of the American Chemical Society*, **117**, 5179–5197 (1995).

149. W. L. Jorgensen and J. Tirado-Rives. The OPLS potential functions for proteins. Energy minimizations for crystals of cyclic peptides and crambin. *Journal of the American Chemical Society*, **110**, 6474–6487 (1988).

150. W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey and M. L. Klein. Comparison of simple potential functions for simulating liquid water. *Journal of Chemical Physics*, **79**, 926–935 (1983).

151. D. J. Price and C. L. Brooks. Modern protein force fields behave comparably in molecular dynamics simulations. *Journal of Computational Chemistry*, **23**, 1045–1057 (2002).

152. B. Widom. Some topics in the theory of fluids. *Journal of Chemical Physics*, **39**, 2808–2812 (1963).

153. D. S. Corti. Alternative derivation of Widom's test particle insertion method using the small system grand canonical ensemble. *Molecular Physics*, **93**, 417–420 (1998).

154. F. N. Keutsch and R. J. Saykally. Water clusters: Untangling the mysteries of the liquid, one molecule at a time. *Proceedings of the National Academy of Sciences of the USA*, **98**, 10533–10540 (2001).

155. J. D. Eaves, J. J. Loparo, C. J. Fecko, S. T. Roberts, A. Tokmakoff and P. L. Geissler. Hydrogen bonds in liquid water are broken only fleetingly. *Proceedings of the National Academy of Sciences of the USA*, **102**, 13019–22 (2005).

156. J. H. Keenan. *Steam tables: thermodynamic properties of water including vapor, liquid and solid phases* (Krieger Publishing Company, 1992).

157. H. L. Kliman. *The solubility of 4-octanone in water, a model compound study of hydrophobic interactions at high pressure*. Ph.D. thesis, Princeton University (1970).

158. H. Luecke. Atomic resolution structures of bacteriorhodopsin photocycle intermediates: the role of discrete water molecules in the function of this light-driven ion pump. *Biochimica et Biophysica Acta*, **1460**, 133–156 (2000).

159. R. Friedman, E. Nachliel and M. Gutman. The role of small intraprotein cavities in the catalytic cycle of bacteriorhodopsin. *Biophysical Journal*, **85**, 886–896 (2003).

160. G. Nemethy, S. J. Leach and H. A. Scheraga. The influence of amino acid side chains on the free energy of helix-coil transitions. *Journal of Physical Chemistry*, **70**, 998–1004 (1966).

161. M. Wikstrom, M. I. Verkhovsky and G. Hummer. Water-gated mechanism of proton translocation by cytochrome c oxidase. *Biochimica et Biophysica Acta*, **1604**, 61–5 (2003).

162. C. Ostermeier, S. Iwata and H. Michel. Cytochrome c oxidase. *Current Opinion in Structural Biology*, **6**, 460–466 (1996).

163. D. Oesterhelt. The structure and mechanism of the family of retinal proteins from halophilic archaea. *Current Opinion in Structural Biology*, **8**, 489–500 (1998).

164. H. Luecke, B. Schobert, J. P. Cartailler, H. T. Richter, A. Rosenfarth, R. Needleman and L. J. K. Coupling photoisomerization of retinal to directional transport in bacteriorhodopsin. *Journal of Molecular Biology*, **300**, 1237–1255 (2000).

165. U. Lehnert, V. Reat, G. Zaccai and D. Oesterhelt. Proton channel hydration and dynamics of a bacteriorhodopsin triple mutant with an M-state-like conformation. *European Biophysics Journal*, **34**, 344–352 (2005).

166. Y. Harpaz, M. Gerstein and C. Chothia. Volume changes on protein folding. *Structure*, **2**, 641–649 (1994).

167. D. Shortle, W. Stites and A. Meeker. Contributions of the large hydrophobic amino acids to the stability or staphyloccocal nuclease. *Biochemistry*, **29**, 8033–8041 (1990).

168. B. W. Matthews. Studies on protein stability with T4 Lysozyme. *Advances in Protein Chemistry*, **46**, 249–278 (1995).

169. D. L. Minor, Y.-F. Lin, B. C. Mobley, A. Avelar, Y. N. Jan, L. Y. Jan and J. W. Berger. The polar T1 interface is linked to conformational changes that open the voltage-gated potassium channel. *Cell*, **102**, 657–670 (2000).

170. D. L. Minor and P. S. Kim. Context-dependent secondary structure formation of a designed protein sequence. *Nature*, **380**, 730–734 (1996).

171. A. R. Fersht. Nucleation mechanisms in protein folding. *Current Opinion in Structural Biology*, **7**, 3–9 (1997).

172. D. T. Bowron, A. Filipponi, M. A. Roberts and J. L. Finney. Hydrophobic hydration and the formation of a clathrate hydrate. *Physical Review Letters*, **81**, 4164–4167 (1998).