Improvements to the Pseudospectral Electronic Structure Method and Experimental Protein Model Initiation

Burnham H. Greeley

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

under the Executive Committee

of the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2006

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

_____

Richard A. Friesner, Principal Advisor
Department of Chemistry

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

_____

Bruce J. Berne
Department of Chemistry

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

_____

David R. Reichman
Department of Chemistry

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

_____

John F. Hunt
Department of Biological Sciences

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

_____

Ruhong Zhou
IBM Thomas J. Watson Research Center

ABSTRACT

Improvements to the Pseudospectral Electronic Structure Method and Experimental Protein Model Initiation

Burnham H. Greeley

This dissertation comprises two works in the field of molecular modeling. First, it presents improvements and a significant reformulation of Friesner's pseudospectral method for Hartree-Fock electronic structure calculations into a hybrid method using both analytic and grid-based integration schemes. The improvements are applicable to other *ab initio* electronic structure methodologies as well. It describes the use of and improvements to a new approach to generating integrals required for the core grid-based method. Additionally, a complex reformulation relying on fast analytic methods and an efficient selection and control structure allows the inclusion of selected analytic correction terms, greatly reducing the required grid density. Absolute energies agree with conventional basis set codes to within 0.25 kcal/mole, and relative energies agree to better than 0.1 kcal/mole for a wide variety of test molecules. Accelerations of CPU times of as large as a factor of 6.5 are obtained as compared to GAUSSIAN 92, with the actual timing

advantage increasing for larger basis sets and larger molecules. The method is shown to be highly reliable and capable of handling extended basis sets.

Second, it presents an investigation into a novel method of initial model building through fragment placement. Despite continuing efforts, protein structure determination from X-ray crystallography data at lower resolutions usually requires manual intervention. Placement of atoms in the partial structure at the start of model building can be particularly critical since errors can be self-reinforcing in ensuing work. The approach uses a coarse six-dimensional real-space search followed by a constrained minimization. Results are given in comparison to a tool based on an exhaustive six-dimensional search alone from a popular crystallography package using ten sets of experimental data. Placement of a standard set of fragments shows equal or often significantly improved agreement with final independently solved models.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENTS

Foremost, I would like to thank my advisor Rich Friesner. While few would complete a doctoral program without support, the circumstances here are perhaps particularly unusual. A break in studies often becomes permanent. Rich's circumspection and encouragement were critical factors in avoiding that end.

In a similar vein, I owe a great deal to the generosity of Matt Jacobson. His advice and support have been invaluable, and have been offered consistently without hesitation.

On such large undertakings, the line blurs between inspiration, collaboration, helpful discussions and the value of simple comradery (one might even say solidarity). I am grateful in particular to Murco Ringnalda. He was the senior student on the electronic structure project when I joined, and has always been a model of hard work and dedication, not to mention a valuable source of Tom Petty lyrics. Much of this work was done in conjunction with Tom Russo. A list of other influences and sources of support, no doubt incomplete, would include Jean-Marc Langlois, Daniel Mainz, Bryan Marten, Vageli Coutsias, Suely M. Black, Julie Wright, Steve Stuart, Tom Ingledue, Paul Stankus, Michelle Craig, Cal Lobel, Betty Cusack, Katarzyna Bernacki and Chakrapani "CK" Kalyanaraman. My thanks to Liang Tong for supplying the experimental data used in the crystallography work. Finally, I would like to thank my family for a myriad of forms of

support.

Some molecular graphics images were produced using the UCSF Chimera package from the Resource for Biocomputing, Visualization, and Informatics at the University of California, San Francisco (supported by NIH P41 RR-01081).[1]

To my family

1

INTRODUCTION

With the enormous success of quantum mechanics and statistical physics in the early 1900s there came the perception that the fundamental structure was in place that would allow, at least in theory, a detailed understanding of cellular processes at a molecular level. This is among the most exciting of human scientific endeavors; the possibility of understanding how we ourselves function in the most basic terms.

In this work we address advances in two areas applicable to this overarching theme. We present significant improvements to the Pseudospectral (PS) method for solving the Hartree-Fock (HF) equations for the electronic structure of molecules. These advances are shown to achieve superior performance and scaling compared to conventional electronic structure codes, allowing the *ab initio* modeling of larger molecular systems than would otherwise be possible. This work has already found application in areas such as the development of new polarizable force-fields for use in protein chemistry, studies of relative peptide energies, protein solvation and mixed quantum mechanics/molecular mechanics (QM/MM) simulations.[2–9] We also present an investigation into an improved method for building initial models from protein X-ray crystallography data. Proteins carry

out the bulk of the critical processes in the cellular machine. Great strides have been made in the past few decades in the ability to obtain structural models experimentally, but the model building stage can still be arduous and often requires prolonged and close attention by a highly skilled crystallographer.

1.1 Large Scale Electronic Structure Calculations

Researchers made significant advances in the calculation of integrals fundamental to the Hartree-Fock and related methods, based on the application of recurrence relationships. In chapter 2 we present two important applications to the improvement of the PS method. We show that modifications to a nuclear attraction integral (NAI) method increases the efficiency of the core one-electron integrals at the heart of the numerical integration terms of the PS approach. We also present an algorithmic strategy which substantially reduces the number of grid points per atom required to achieve accurate relative energies and total energies for calculations on large molecules. This method employs a larger number of analytical integrals in the assembly of the Coulomb and exchange operators than previously used. The development of efficient recurrence schemes for the evaluation of two-electron repulsion integrals (ERIs) has significantly reduced the computational effort required per integral, in part by reducing operation counts, in part by achieving better performance from reorganization of data structures. As we utilize only two-center and a selected set of three-center ERIs, the number of integrals to be evaluated in our formalism is orders of magnitude less than in conventional electronic structure codes. At the same time, this small subset of integrals includes the largest (by at least an order of magnitude) terms in the electrostatic energy; we are therefore able to substantially reduce the number of grid points that we

employ, as the precision required of the numerical integration scheme to achieve equivalent accuracy decreases accordingly. The overall scheme for applying two-electron corrections to the pseudospectral calculations will be referred to as the two-electron correction (TEC) algorithm in what follows.

We show the results of the accuracy and efficiency on a large number of molecules and molecular conformations. These improvements are described in the context of HF theory, but similar results have been obtained for higher order correlation methods. Formally PS methods scale computationally at one order of magnitude less than those of conventional solutions, e.g. N^3 versus N^4 for HF, where N is the system size. Careful implementation of cutoffs reduces the actual scaling even further. This reduction allows treatment of systems much larger than conventional codes, while maintaining chemical accuracy.

The chapter is organized as follows. Section 2.2 presents the general formalism for the HF solution to the Schrödinger equation. Section 2.3 describes the method that became the basis for most common conventional solutions. Section 2.4 gives a brief discussion of gradient calculations in the interests of motivating the inclusion of first derivative terms in the TEC algorithm. Section 2.5 discusses the computational issues faced by the conventional solutions.

Section 2.6 provides an overview of the development of the PS formalism. The full PS method is a complex, hybrid approach. We present an overview with concentration on the structure necessary to understand the significance of the work described. Details of other aspects of the PS method can be found in previous papers.[10–16] In section 2.7, we present the formalism associated with the TEC algorithm. Section 2.8 introduces recurrence formalism and discusses some common features, while the following section describes some basic common definitions

and notation.

The complete formalism and our modifications to the Obara-Saika recurrence techniques for NAI-type terms[17] and a detailed analysis of the computational considerations are given in section 2.10. Gill, Head-Gordon and Pople described an algorithm for the efficient generation of general two-electron integrals and their derivatives.[18] Our implementation differs markedly from theirs. Section 2.11 follows the derivation of the subset of relations relevant to the TEC algorithm, and describes the implementation in detail. Section 2.12 presents accuracy and timing tests, comparing our results with those obtained from GAUSSIAN 92.

In previous work, we have emphasized agreement with analytical methods of the total energy, typically achieving a 0.1 kcal/mole level of agreement. However, in reality, the only relevant quantities are total energy differences. By relaxing the constraint on such close agreement of the total energy (particularly for large molecules) but insisting on maintaining agreement for relative energies (easily tested by studying a series of molecular conformations), we are able to make significant reductions in our computational effort. Within the new framework, we developed a parameter set which display very small total energy deviations (less than $0.25\,\mathrm{kcal/mole}$) from GAUSSIAN 92 for the small and medium-size molecules we report here (the largest is porphine, with 38 atoms).

The methods implemented here form major parts of the commercially available PS-GVB/Jaguar package. Jaguar is widely used by both academic and industrial institutions, further proving the stability, accuracy and practical benefits of the PS method.

1.2 Automated Feature Detection in Protein Crystallography

Despite a large amount of active research, methods for substantially deriving protein models from X-ray data automatically still do not work well enough at lower resolutions (~ 2.7Å and worse). Pushing the limits of resolutions at which tools can reliably solve structures automatically past the 3Å mark would be a tremendous boon to the field. In chapter 3 we present an investigation into a novel approach to beginning model building through feature recognition. We present an algorithm that performs a six-dimensional search through an electron density map (EDM), combined with functional minimization, capable of positing accurate placement of fragments of any arrangement of atoms. We show that this method produces substantially better initial fragment placements than current methods, using a much coarser sampling of trial poses.

The chapter is organized as follows. Section 3.1 gives a brief overview motivating the study of X-ray crystallography techniques, followed by section 3.2 which details the theoretical framework of crystallography and problems presented by it. In section 3.3 we summarize the major efforts for performing automated protein structure assignment to date. Section 3.4 introduces the specifics of our study and lays the mathematical groundwork. Section 3.5 describes in detail the algorithm we developed, while the following section further discusses key considerations. We present our results and compare them to a similar approach, FFFEAR, part of the popular CCP4 crystallographic toolkit, in section 3.7. We show that our approach achieves a more robust result when tested against a set of ten X-ray data sets, albeit at the cost of substantially more computer time. Section 3.8 summarizes our conclusions, and suggests a direction for future work.

2

PSEUDOSPECTRAL ELECTRONIC STRUCTURE CALCULATIONS

2.1 Overview

In the following pages, we describe our advances in electronic structure calculations the context of Hartree-Fock (HF) theory. The HF approximation is both useful in its own right, and serves as a starting point for more accurate methods that include higher-order electron correlation effects. The fact that these technologies apply to many correlation methods and, alternately, to the density functional approach to electronic structure calculations is an important point that should not be overlooked.[24–27] However, we focus on single point and gradient HF calculations. The extensive body of knowledge in place, the relative simplicity of the HF equations, and the fact that HF theory remains a basic probative tool makes it the best and most appropriate proving ground for our research.

As a matter of completeness, we begin in section 2.2 by tracing the now standard derivation of the Hartree-Fock equations, starting with the time-independent Schrödinger equation. We complete the section by presenting the so-called restricted closed-shell form of the equations. Section 2.3 discusses the usual method

of solution first proposed by Roothaan. We show how the introduction of a set of known basis functions converts the HF equations into a set of algebraic equations. We include a brief discussion of energy gradients in the Roothaan formulation next. These equations are the foundation of the purely spectral approaches. Section 2.5 analyzes the formal computational scaling of spectral methods based on the Roothaan equations. These first sections follow the description of Szabo and Ostlund closely.[28]

Section 2.6 presents an alternate method for solving the Roothaan equations that uses both the usual basis sets and a numerical grid, hereafter referred to as the pseudospectral (PS) technique. Pseudospectral methods have been developed and applied previously in solving other non-linear systems, particularly in fluid mechanics.[29–31] We describe the PS formalism as it relates to the HF problem, with special emphasis on the portions of the theory that have been advanced by the research described in this work. We discuss the formal scaling of this newer method and contrast it with that of the spectral approach.

Sections 2.8 and 2.9 lay a bit of groundwork for the analysis to follow by describing general considerations regarding recurrence techniques and defining some common terms. Sections 2.10 and 2.11 cover the details of the advances presented here in depth. Section 2.12 presents the results of an extensive set of tests and comparisons. Section 2.13 concludes the chapter.

In presenting the background theory and new developments, our primary goal is to make clear the impediments to creating a computationally efficient general purpose program for solving the HF and related equations. The emphasis will tend toward practicality on modern day computer architectures.

2.2 The Hartree-Fock Equations

2.2.1 The Schrödinger Equation

We take as a starting point the time-independent Schrödinger equation (TISE) for an isolated molecular system. The TISE is an operator eigenvalue equation which states

$$\mathcal{H}|\,\psi\,\rangle = \mathrm{E}|\,\psi\,\rangle$$

where $\mathcal{H}$ is the Hamiltonian operator. Neglecting relativistic effects, and written in atomic units, the Hamiltonian operator for a system with N electrons and M nuclei is

$$\mathcal{H} = -\sum_{a=1}^{N}\frac{1}{2}\nabla_a^2 - \sum_{A=1}^{M}\frac{1}{2M_A}\nabla_A^2 - \sum_{a=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{aA}} + \sum_{a=1}^{N}\sum_{b>a}^{N}\frac{1}{r_{ab}} + \sum_{A=1}^{M}\sum_{B>A}^{M}\frac{Z_A Z_B}{r_{AB}}$$

Here, uppercase labels refer to nuclei, while lowercase refer to electrons. Z represents the atomic number of the nuclei. M_A refers to nuclear masses. Denominators r refer to the distances between two objects, e.g. $r_{aA} = |\mathbf{r}_{aA}|$ is the distance between electron a and nucleus A. See Figure 2.1 for an illustration of the definitions. With the exception of a few universal constants, the TISE is a purely mathematical statement that describes the nature of a quantum system. It is a multi-variable partial differential equation, solvable in closed form for only a few special cases. Various approximation techniques can be employed to produce something computationally tractable. *Ab initio* techniques refer to the class of methods derived from the TISE without reference to empirically obtained information. In a scientific sense, this not only improves our ability to assess the validity of such a method, it also places it in formal relationship to the exact theory. Though this

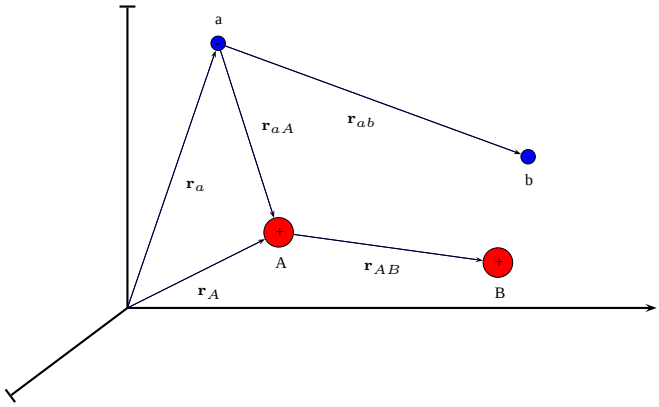


Figure 2.1: Diagram illustrating the molecular coordinate system definitions used. Uppercase labels refer to nuclei (shown in red), lowercase labels refer to electrons (shown in blue)

may not always guarantee that an approach can be systematically improved, it is a powerful benefit in this regard. With this in mind, we outline the approximations necessary to arrive at the HF equations, the most commonly used *ab initio* method.

2.2.2 Born-Oppenheimer Approximation

The TISE contains both electronic and nuclear terms. Nuclei are sufficiently massive compared to electrons that, in chemical systems, to a good approximation electrons can be viewed as moving in a field of fixed nuclei. In this, the Born-Oppenheimer approximation, the nuclear kinetic energy can be neglected. The nuclear-nuclear repulsion energy becomes a constant and so has no effect on the wavefunction. Dropping those terms from the Hamiltonian, we obtain a new equation for what we will call the electronic wavefunction

$$\mathcal{H}_e|\,\psi_e\,\rangle = \mathrm{E_e}|\,\psi_e\,\rangle \tag{2.1}$$

where

$$\mathcal{H}_e = -\sum_{a=1}^{N} \frac{1}{2}\nabla_a^2 - \sum_{a=1}^{N}\sum_{A=1}^{M} \frac{Z_A}{r_{aA}} + \sum_{a=1}^{N}\sum_{b>a}^{N} \frac{1}{r_{ab}} \tag{2.2}$$

The electronic wavefunction still depends on the nuclear coordinates, but in a parametric fashion. The effects of finite nuclear mass can be reintroduced after solving this equation. However, for most applications it can simply be ignored. For our purposes then, $|\,\psi_e\,\rangle$ becomes the item of interest. From here on, we will drop the subscript, and refer to the electronic wavefunction as $|\,\psi\,\rangle$.

2.2.3 The Functional Variation Technique

Eq. (2.1), while simpler, is still intractable. Hartree originally derived the precursors to the HF equations by making an ansatz at this stage. We will arrive at the same results using a general technique known as functional variation. Rather than solving Eq. (2.1) explicitly for $|\,\psi\,\rangle$, we introduce a trial wavefunction $|\,\tilde{\psi}\,\rangle$. The expectation value $E\big[\,\tilde{\psi}\,\big]$ of the Hamiltonian for this function is

$$E\big[\,\tilde{\psi}\,\big] = \langle\,\tilde{\psi}\,|\mathcal{H}|\,\tilde{\psi}\,\rangle \tag{2.3}$$

i.e. $E\big[\,\tilde{\psi}\,\big]$ is a functional of $|\,\tilde{\psi}\,\rangle$. It can be easily shown that the energy E of the true wavefunction is a lower bound on the energy $E\big[\,\tilde{\psi}\,\big]$, given the normalization constraint $\langle\,\tilde{\psi}\,|\,\tilde{\psi}\rangle = 1$. Hence, within the spectrum of normalized functions representable by the trial function, the one with the lowest energy best approximates the exact wavefunction. Formally, to enforce the normalization constraint, we use the method of Lagrange's undetermined multipliers, and define the new functional

$$\mathcal{L}\big[\,\tilde{\psi}\,\big] = \langle\,\tilde{\psi}\,|\mathcal{H}|\,\tilde{\psi}\,\rangle - \varepsilon(\langle\,\tilde{\psi}\,|\,\tilde{\psi}\rangle - 1)$$

The condition that the energy $E[\tilde{\psi}]$ be minimum is equivalent to requiring that the first order variation of $\mathcal{L}[\tilde{\psi}]$ with respect to infinitesimal changes in $|\tilde{\psi}\rangle$ equal 0. (Technically this just guarantees that E is stationary, but normally it will be a minimum); i.e. we require that

$$\frac{\delta \mathcal{L}}{\delta \tilde{\psi}} = 0 \tag{2.4}$$

Before we can carry out the variation, we must choose a form for the trial function. This choice differentiates HF from other theories.

2.2.4 The Hartree-Fock Approximation - Slater Determinants

Hartree originally proposed using a trial function consisting of a product of single electron spin orbitals.[32] This form is the simplest possible. It includes no electron-electron correlation effects and is equivalent to taking each electron as moving in the average field of all others. The wavefunction in the Hartree approximation lacks a crucial property, however. The wavefunction has no specific symmetry, and doesn't (automatically) satisfy the Pauli exclusion principle. To overcome this, we use an anti-symmetrized product of spin orbitals. Certain properties of this trial function are most easily understood when it is written as a Slater determinant. We write

$$\tilde{\psi}\left(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\right) = (N!)^{-\frac{1}{2}} \begin{vmatrix} \chi_a(\mathbf{x}_1) & \chi_b(\mathbf{x}_1) & \cdots & \chi_c(\mathbf{x}_1) \\ \chi_a(\mathbf{x}_2) & \chi_b(\mathbf{x}_2) & \cdots & \chi_c(\mathbf{x}_2) \\ \vdots & \vdots & & \vdots \\ \chi_a(\mathbf{x}_N) & \chi_b(\mathbf{x}_N) & \cdots & \chi_c(\mathbf{x}_N) \end{vmatrix} \tag{2.5}$$

where the vertical lines symbolize the usual notation for determinants. Here χ denotes a spin orbital. N is the number of electrons. Note that exchanging any two electrons corresponds to the interchange of two rows of orbitals, which changes the sign of the determinant, and, by extension, of the wavefunction. Thus any trial function constructed from a linear combination of Slater determinants will be antisymmetric with respect to the exchange of two electrons. The Hartree-Fock approximation equates to choosing a single Slater determinant for the form of the trial wavefunction. For brevity we denote this function as $|\,\chi_1\chi_2\cdots\chi_N\,\rangle$.

2.2.5 Orthonormality Constraints

Using the usual properties of determinants, it becomes clear that we may require the spin orbitals to be orthogonal without loss of generality. This restriction simplifies the expression for the expectation value $E\big[\,\tilde{\psi}\,\big]$ greatly by eliminating overlap terms. With a single Slater determinant, $\mathcal{L}$ becomes a functional of the set of spin orbitals $\big\{\chi_a\big\}$. To enforce orthonormality, we must use a set of Lagrange multipliers. For an arbitrary choice, we have in general

$$\mathcal{L}\big[\,\big\{\chi_a\big\}\,\big] = \langle\,\chi_1\chi_2\cdots\chi_N\,|\mathcal{H}|\,\chi_1\chi_2\cdots\chi_N\,\rangle - \sum_{a=1}^{N}\sum_{b=1}^{N}\varepsilon_{ab}(\langle\,\chi_a\,|\,\chi_b\rangle - \delta_{ab})$$

Because $\mathcal{L}$ is real, the multipliers ε_{ab} form a Hermitian matrix. Therefore there exists a unitary transformation $\mathbf{U}$ which diagonalizes this matrix. The expectation value $E\big[\,\tilde{\psi}\,\big]$ is invariant under a unitary transformation of the spin orbitals, so we are free to restrict ourselves further to a set for which

$$\mathcal{L}\big[\,\big\{\chi_a\big\}\,\big] = \langle\,\chi_1\chi_2\cdots\chi_N\,|\mathcal{H}|\,\chi_1\chi_2\cdots\chi_N\,\rangle - \sum_{a=1}^{N}\varepsilon_a(\langle\,\chi_a\,|\,\chi_a\rangle - 1) \qquad (2.6)$$

2.2.6 Derivation of the Hartree-Fock Equations

Having chosen the form of the trial wavefunction, we must now evaluate the expectation value of the Hamiltonian. This can be done easily by noting that the Hamiltonian is built from two types of operators, ones involving the coordinates of a single electron and those dependent on a pair of electrons. Using $h(1)$ to denote a general single electron portion of the Hamiltonian due to electron one and $v(1,2)$ the portion involving the electron pair one and two, the electronic Hamiltonian (Eq. (2.2)) can be written as

$$\mathcal{H} = \sum_{i=1}^{N} h(i) + \sum_{i=1}^{N}\sum_{j>i}^{N} v(i,j)$$

The expectation values are then

$$\langle\, \chi_1\chi_2\cdots\chi_N \,|\sum_{i=1}^{N} h(i)|\, \chi_1\chi_2\cdots\chi_N \,\rangle = \sum_{a=1}^{N}\langle\, \chi_a \,|h|\, \chi_a \,\rangle$$

and

$$\begin{aligned}\langle\, \chi_1\chi_2\cdots\chi_N \,|\sum_{i=1}^{N}\sum_{j>i}^{N} v(i,j)|\, \chi_1\chi_2\cdots\chi_N \,\rangle& \\ = \sum_{a=1}^{N}\sum_{b>a}^{N}\big(\langle\, \chi_a\chi_b \,|v|\, \chi_a\chi_b \,\rangle - \langle\, \chi_a\chi_b \,|v|\, \chi_b\chi_a \,\rangle\big)& \\ = \frac{1}{2}\sum_{a=1}^{N}\sum_{b=1}^{N}\big(\langle\, \chi_a\chi_b \,|v|\, \chi_a\chi_b \,\rangle - \langle\, \chi_a\chi_b \,|v|\, \chi_b\chi_a \,\rangle\big)&\end{aligned}$$

All other terms are eliminated due to the orthogonality of the spin orbitals. The first term in the second equation comes from what one might call the normally

ordered product of spin orbitals. The second term represents a purely quantum mechanical effect due to the antisymmetry of the wavefunction. It comes from the Slater term with the coordinates of electrons one and two exchanged.

With these expressions in hand, we can immediately write down the expectation for the energy of $|\,\chi_1\chi_2\cdots\chi_N\,\rangle$:

$$E\big[\,\{\chi_a\}\,\big] = \sum_{a=1}^{N}\langle\,\chi_a\,|h|\,\chi_a\,\rangle + \frac{1}{2}\sum_{a=1}^{N}\sum_{b=1}^{N}\big(\langle\,\chi_a\chi_b\,|v|\,\chi_a\chi_b\,\rangle - \langle\,\chi_a\chi_b\,|v|\,\chi_b\chi_a\,\rangle\big)$$

Referring to Eq. (2.2) we have

$$h(1) = -\frac{1}{2}\nabla_1^2 - \sum_{A=1}^{M}\frac{Z_A}{r_{1A}} \tag{2.7}$$

and

$$v(1,2) = \frac{1}{r_{12}} \tag{2.8}$$

Applying the requirement that $\mathcal{L}$ remain stationary with respect to arbitrary small variations in the functions $\{\chi_a\}$ then gives a set of N equations

$$\big[h(1) + \sum_{b=1}^{N}\mathcal{J}_b(1) - \mathcal{K}_b(1)\big]\chi_a(\mathbf{x}_1) = \varepsilon_a\chi_a(\mathbf{x}_1) \tag{2.9}$$

where we define the Coulomb operator $\mathcal{J}_b(1)$ as

$$\mathcal{J}_b(1)\chi_a(\mathbf{x}_1) = \big[\int d\mathbf{x}_2\,\chi_b^*(\mathbf{x}_2)r_{12}^{-1}\chi_b(\mathbf{x}_2)\big]\chi_a(\mathbf{x}_1)$$

and the exchange operator $\mathcal{K}_b(1)$ as

$$\mathcal{K}_b(1)\chi_a(\mathbf{x}_1) = \Big[\int d\mathbf{x}_2\, \chi_b^*(\mathbf{x}_2) r_{12}^{-1} \chi_a(\mathbf{x}_2)\Big]\chi_b(\mathbf{x}_1)$$

These are the Hartree-Fock equations. The Fock operator is defined as

$$f(1) = h(1) + \sum_{b=1}^{N} \mathcal{J}_b(1) - \mathcal{K}_b(1)$$

Thus Eq. (2.9) is an eigenvalue equation for the Fock operator.

2.2.7 Restricted Closed-Shell Hartree-Fock

To this point we have used a generalized spin orbital χ. We now make the transition to spatial orbitals. For simplicity, we will confine ourselves to a closed-shell system, i.e. one with an even number of electrons. The spin orbitals can be written in terms of spatial orbitals and two spin functions. We define

$$\chi_a(\mathbf{x}) = \begin{cases} \psi_a(\mathbf{r})\alpha(\omega) \\ \psi_a(\mathbf{r})\beta(\omega) \end{cases}$$

with α and β representing orthonormal spin states. We impose the restriction that each spatial orbital be doubly occupied. They form a set of $N/2$ orthonormal functions $\{\psi_a\}$. Substituting these definitions into Eq. (2.9) and integrating over the spin variable ω, we obtain the closed-shell spatial Hartree-Fock equation

$$f(1)\psi_a(\mathbf{r}_1) = \varepsilon_a \psi_a(\mathbf{r}_1) \tag{2.10}$$

where

$$f(1) = h(1) + \sum_{b=1}^{N/2} 2J_b(1) - K_b(1)$$
$$J_b(1)\psi_a(\mathbf{r}_1) = \Big[\int d\mathbf{r}_2\, \psi_b^*(\mathbf{r}_2) r_{12}^{-1} \psi_b(\mathbf{r}_2)\Big]\psi_a(\mathbf{r}_1)$$
$$K_b(1)\psi_a(\mathbf{r}_1) = \Big[\int d\mathbf{r}_2\, \psi_b^*(\mathbf{r}_2) r_{12}^{-1} \psi_a(\mathbf{r}_2)\Big]\psi_b(\mathbf{r}_1)$$

are the spatial Fock, Coulomb and exchange operators, respectively. The physical interpretation here is intuitive. Each electron has a Coulombic interaction with all the others, but has an exchange interaction with only half of the rest by virtue of the orthogonality of the spin functions. This accounts for the factor of 2 multiplying the J operator that's missing from the K operator in the spatial Fock operator.

2.3 The Roothaan Equations

Roothaan suggested a method by which the integro-differential equations for the spacial orbitals can be turned into a set of matrix equations. To do this, the unknown spatial orbitals $\{\psi_a\}$ are approximated by an expansion in terms of a set of known basis functions:

$$\psi_a = \sum_{\nu=1}^{M} C_{\nu a}\phi_\nu \tag{2.11}$$

(Here M is the number of functions in the expansion). The solution would be exact in the limit that the set $\{\phi_\nu\}$ becomes complete. Substituting Eq. (2.11)

into Eq. (2.10), multiplying by ϕ_μ on the left and integrating, we have

$$\sum_{\nu=1}^{M} F_{\mu\nu} C_{\nu a} = \varepsilon_a \sum_{\nu=1}^{M} S_{\mu\nu} C_{\nu a} \qquad a = 1, 2, \ldots, N/2$$

where we define the Fock matrix $\mathbf{F}$ as

$$F_{\mu\nu} = \int d\mathbf{r}_1 \, \phi_\mu^*(\mathbf{r}_1) f(1) \phi_\nu(\mathbf{r}_1)$$

Since the basis functions are not typically orthogonal, we also define the overlap matrix $\mathbf{S}$.

$$S_{\mu\nu} = \int d\mathbf{r}_1 \, \phi_\mu^*(\mathbf{r}_1) \phi_\nu(\mathbf{r}_1)$$

The problem can now be written shorthand in matrix notation as

$$\mathbf{FC} = \mathbf{SC}\boldsymbol{\varepsilon} \tag{2.12}$$

These are the Roothaan equations. Through this technique, solving the Hartree-Fock equations has become a matter of solving an eigenvalue equation for the Fock matrix $\mathbf{F}$ to obtain the coefficients $C_{\nu a}$ and eigenvalues ε_a.

The Coulomb and exchange terms in the Fock matrix become

$$\begin{aligned} 2J_{\mu\nu} - K_{\mu\nu} &= \sum_{a=1}^{N/2} \int d\mathbf{r}_1 \, \phi_\mu^*(\mathbf{r}_1) \big[2J_a(1) - K_a(1) \big] \phi_\nu(\mathbf{r}_1) \\ &= \sum_{a=1}^{N/2} \sum_{\lambda\sigma} C_{\lambda a} C_{\sigma a}^* \big[2\big(\mu\nu|\lambda\sigma\big) - \big(\mu\sigma|\lambda\nu\big) \big] \\ &= \sum_{\lambda\sigma} \rho_{\lambda\sigma} \big[\big(\mu\nu|\lambda\sigma\big) - \frac{1}{2}\big(\mu\sigma|\lambda\nu\big) \big] \end{aligned} \tag{2.13}$$

where we define the the density matrix $\boldsymbol{\rho}$ by

$$\rho_{\lambda\sigma} = 2\sum_{a=1}^{N/2} C_{\lambda a} C^*_{\sigma a}$$

and

$$\big(\mu\nu|\lambda\sigma\big) = \int d\mathbf{r}_1\, d\mathbf{r}_2 \phi^*_\mu(\mathbf{r}_1)\phi_\nu(\mathbf{r}_1) r_{12}^{-1} \phi^*_\lambda(\mathbf{r}_2)\phi_\sigma(\mathbf{r}_2)$$

denotes the two-electron repulsion integrals (ERIs).

For completeness, we define the "core" matrix elements

$$H^{core}_{\mu\nu} = \int d\mathbf{r}_1\, \phi^*_\mu(\mathbf{r}_1) h(1) \phi_\nu(\mathbf{r}_1)$$

where $h(1)$ is given in Eq. (2.7).

In terms of these quantities

$$\mathrm{E_e} = \sum_{\mu\nu} \rho_{\mu\nu} H^{core}_{\mu\nu} + \frac{1}{2}\sum_{\mu\nu\lambda\sigma} \rho_{\mu\nu}\rho_{\lambda\sigma}\big[\big(\mu\nu|\lambda\sigma\big) - \frac{1}{2}\big(\mu\sigma|\lambda\nu\big)\big]$$

is the electronic energy of the system.

2.4 Gradient Calculations

For gradient calculations, we no longer assume the nuclei are fixed. At most temperatures of interest, the momenta of the nuclei are still negligible, but changes in the nuclear-nuclear repulsion terms are not. The Hartree-Fock energy is

$$\mathrm{E} = \mathrm{E_e} + V_{nuc}$$

where $\mathrm{E_e}$ is the electronic energy given in the previous section and

$$V_{nuc} = \sum_A \sum_{B>A} \frac{Z_A Z_B}{\mathbf{R}_{AB}}$$

is the nuclear repulsion potential energy.

In Roothaan's method the energy depends on the positions both explicitly and implicitly through the coefficients $C_{\mu a}$. The total derivative with respect to a particular coordinate, then, of the energy is

$$\frac{d\mathrm{E}}{dX_A} = \frac{\partial \mathrm{E}}{\partial X_A} + \sum_\mu \sum_a \frac{\partial \mathrm{E}}{\partial C_{\mu a}} \frac{\partial C_{\mu a}}{\partial X_A}$$

where X_A represents any of the three Cartesian coordinates of atom A.

We may rewrite a portion of the right hand side in a useful way by considering the following. In the Roothaan formulation, the variational condition Eq. (2.4) becomes equivalent to the condition that $\mathcal{L}$ be invariant to first order with respect to changes in the coefficient $C_{\mu a}$. From Eq. (2.6) we then have

$$\begin{aligned} \frac{\partial \mathrm{E}}{\partial C_{\mu a}} &= \frac{\partial}{\partial C_{\mu a}} \sum_b \varepsilon_b \sum_{\lambda\nu} C_{\lambda b} C_{\nu b} S_{\lambda\nu} \\ &= 2\varepsilon_a \sum_\nu C_{\nu a} S_{\mu\nu} \end{aligned} \tag{2.14}$$

The orthonormality condition is

$$\sum_{\mu\nu} C_{\mu a} S_{\mu\nu} C_{\nu b} = \delta_{ab}$$

from which we obtain the expression

$$2\sum_{\mu\nu}\frac{\partial C_{\mu a}}{\partial X_A}S_{\mu\nu}C_{\nu b} = -\sum_{\mu\nu}C_{\mu a}C_{\nu b}\frac{\partial S_{\mu\nu}}{\partial X_A}$$

Combining everything gives

$$\begin{aligned}\frac{d\mathrm{E}}{dX_A} =& \sum_{\mu\nu}\rho_{\mu\nu}\frac{\partial H^{core}_{\mu\nu}}{\partial X_A} + \frac{1}{2}\sum_{\mu\nu\lambda\sigma}\rho_{\mu\nu}\rho_{\lambda\sigma}\Big[\frac{\partial\big(\mu\nu|\lambda\sigma\big)}{\partial X_A} - \frac{1}{2}\frac{\partial\big(\mu\sigma|\lambda\nu\big)}{\partial X_A}\Big] \\ & - 2\sum_{\mu\nu}W_{\mu\nu}\frac{\partial S_{\mu\nu}}{\partial X_A} + \frac{\partial V_{nuc}}{\partial X_A}\end{aligned}$$

where

$$W_{\mu\nu} = 2\sum_{a}\varepsilon_a C_{\mu a}C_{\nu a}$$

is the energy weighted density matrix. This expression has the beneficial property that it does not require calculating derivatives of the coefficients $C_{\mu\nu}$. However, it does contain first derivatives of the ERIs.

2.5 Computational Considerations of the Roothaan Equations

Because the matrix elements of $\mathbf{F}$ in the Roothaan equations depend on the expansion coefficients $C_{\mu a}$, the eigenvalue equation for $\mathbf{F}$ can not be solved directly. Instead, the usual procedure begins by making an initial guess for the density matrix $\boldsymbol{\rho}$, calculating the resultant Fock matrix, then solving Eq. (2.12) for a new set of coefficients. This new set defines a new density matrix, which in turn gives a new Fock matrix. The process repeats iteratively until the density matrix converges (meaning to some level of precision the coefficients from one iteration equal those of the previous one). Because the equations must be repeatedly solved

until the coefficients no longer change, this process is often referred to as the self-consistent field method.

We have yet to discuss the actual functional form of the basis functions. Most practical software implementations use either Slater or Gaussian orbitals (functions where the main radial character goes as e^{-r} and e^{-r^2}, respectively). Slater orbitals provide a better qualitative fit to the character of actual molecular orbitals, but they are difficult and inefficient to use in the integrals needed. Integrals over Gaussians are much easier to compute. Boys first suggested using Gaussian functions in 1950.[33] Researchers have since proposed the use of contracted sets of Gaussians (basis functions formed from a linear combination of Gaussians) to better mimic the form of molecular orbitals whilst reducing computational cost.[34–36] In particular, we use Cartesian Gaussian functions. See section 2.9 on page 33 for details and notation.

The class of approaches based on the direct evaluation of the ERIs are known as spectral methods. Note the ERIs depend on four basis functions. Formally, calculating all the ERIs scales as the number of basis functions to the fourth power. The computational effort required to form these integrals is the main roadblock limiting the size of the systems that can be solved. Pseudospectral techniques reduce the rate at which the computational effort scales.

2.6 Pseudospectral Theory

Pseudospectral methods are based upon representing a two-electron integral $\big(\mu\nu|\lambda\sigma\big)$ as a quadrature over grid points:

$$\big(\mu\nu|\lambda\sigma\big) = \sum_{\mathbf{g}} Q_\mu(\mathbf{g})\phi_\nu(\mathbf{g})A_{\lambda\sigma}(\mathbf{g}) \tag{2.15}$$

where $\phi_\nu(\mathbf{g})$ is an atomic basis function ν evaluated at the grid point location g, and

$$A_{\lambda\sigma}(\mathbf{g}) = \int d\mathbf{r}\, \frac{\phi_\lambda(\mathbf{r})\phi_\sigma(\mathbf{r})}{|\mathbf{r}-\mathbf{g}|} \tag{2.16}$$

is a three-center, one-electron integral (potential integral) representing the field at g due to the product charge distribution of basis functions ϕ_λ and ϕ_σ. This integral has the same form as the nuclear attraction integrals (NAIs) needed as part of the one-electron portion of the Fock operator.

The matrix $Q_\mu(\mathbf{g})$ is a least squares fitting operator which is designed to fit any right hand side $\phi_\nu(\mathbf{g})A_{\lambda\sigma}(\mathbf{g})$ in the region of space relevant to atomic basis function ϕ_μ. Briefly, functions on the grid are approximated by an expansion in terms of a fitting basis. The overlap integrals of the fitting and atomic bases are calculated analytically. The full mathematical details of this procedure have been given in several papers and we shall not repeat them here.[11,12,14,15] For our purposes, one can think of $Q_\mu(\mathbf{g})$ as a set of quadrature weights that are specially designed to provide accurate integration over the function ϕ_μ. Because of this design, the accuracy of the result for a given number of grid points is necessarily much better than for a generic quadrature scheme (e.g. Gaussian quadrature) unless there are instabilities in the fitting procedure. From long experience, we have been able to

control the instabilities by a variety of techniques so that the algorithm provides a robust performance for arbitrary molecules.

Because of the projection onto analytical overlap integrals, our method reduces to the analytical result in the limit that the quadrature scheme becomes exact. This can be accomplished either by making the fitting basis complete with respect to the right hand side or by making the underlying quadrature on the grid exact (in the latter case, one of course would not need to use the fitting basis). In practice a combination of the two approaches is used to make the PS results very close, but not identical to, those obtained from analytical theory.

Substituting Eq. (2.15) into Eq. (2.13) leads to the following PS expressions for J and K:

$$J_{\mu\nu} = \sum_{\mathbf{g}} Q_{\mu}(\mathbf{g}) J(\mathbf{g}) \phi_{\nu}(\mathbf{g}) \tag{2.17}$$

where the physical space Coulomb operator $J(\mathbf{g})$ is given by

$$J(\mathbf{g}) = \sum_{\lambda\sigma} \rho_{\lambda\sigma} A_{\lambda\sigma}(\mathbf{g}) \tag{2.18}$$

and

$$K_{\mu\nu} = \sum_{\mathbf{g}} Q_{\mu}(\mathbf{g}) K_{\nu}(\mathbf{g}) \tag{2.19}$$

where

$$K_{\nu}(\mathbf{g}) = \sum_{\lambda} A_{\nu\lambda}(\mathbf{g}) T_{\lambda}(\mathbf{g}) \tag{2.20}$$

is the pseudospectral physical space exchange field. The intermediate quantity $T_{\lambda}(\mathbf{g})$ is defined as

$$T_{\lambda}(\mathbf{g}) = \sum_{\sigma} \rho_{\lambda\sigma} \phi_{\sigma}(\mathbf{g}) \tag{2.21}$$

These are the equations used in our electronic structure package, PS-GVB, prior to the methods presented here. The formal scaling of these equations is N^2M (where N is the basis set size and M the grid size). Since the grid size scales linearly with the addition of new atoms, comparing systems of different size solved with the same basis set, the scaling is proportional to N^3. With the use of integral cutoffs this becomes N^2 for large systems.

2.7 Two-Electron Integral Corrections

2.7.1 Overview

From an early point in its development, PS-GVB incorporated analytical integrals for one-center Coulomb and exchange terms. These terms are small in number and their evaluation analytically is obviously trivial; hence, the computational cost of this strategy is virtually nonexistent. Accuracy of the total energy is typically increased by one to two orders of magnitude as compared to a fully pseudospectral calculation for a grid of 1000 points/atom.

The development of very fast recursive two-electron integral algorithms proceeded to the point where the calculation of additional integrals analytically became worthwhile. We will show that through a judicious ordering of the integrals into classes, we can calculate analytically only the largest (and, for PS methods, most challenging) terms. If these are a sufficiently small fraction of the total number of integrals (which can be enforced by the use of cutoffs), the CPU time required is essentially negligible and permits a 3–5-fold reduction in grid size for a comparable level of accuracy.

We consider three general types of integrals, those of the form $\left(aa'|bc\right)$, $\left(aa'|bb'\right)$

and $\big(ab|a'b'\big)$. Primed letters indicate basis functions centered on the same atom as their unprimed counterpart. For example, with integrals of the form $\big(aa'|bc\big)$, a and a' are (possibly different) basis functions on the same atom (atom A) and b and c are basis functions on atoms B and C, respectively. Besides restricting the number of integrals, these forms render the calculation of individual integrals less expensive than for the general form $\big(ab|cd\big)$, as described below.

For the Coulomb operator, a justification of this strategy is straightforward. Examination of statistics for the size of density matrix elements indicates that those in which both indices are valence functions on the same center are 100 or more times larger than those for which the indices are on different atoms or for which one index is a polarization function. This observation follows from the well-known fact that electron densities in molecules are small perturbations on those in an atom. Hence, terms of the form $\big(ab|cd\big)$ where the pairs $\big(a,b\big)$ and $\big(c,d\big)$ are both on different atoms, contribute 10 000 times less amplitude to the Coulomb energy.

For the exchange term, we have chosen to include only two-center analytical integrals in assembly of the exchange operator, on the basis of empirical experimentation with the effects of three-center terms. This approximation works quite well in practice and, in fact, can be restricted to nearest neighbor two-center terms for many of the self-consistent iterations (those where only $\sim$.01–.001 au accuracy is required).

Several technical considerations have played an important role in the development of the scheme presented below. First, the terms to be computed analytically must be subtracted from the pseudospectral operators to avoid double counting. For an arbitrary set of two-electron integrals, the subtraction procedures are non-

trivial and indeed easily could be considerably more expensive to implement than the Fock matrix assembly itself. Consequently, it is important to carry out analytical corrections on groups of terms which can be replaced with relatively little effort in the pseudospectral assembly scheme.

A second feature of the methodology is the use of overlap integrals to estimate the size of terms when considering whether to compute them analytically or numerically. Cutoff thresholds are established and terms whose estimators are below these thresholds are either done numerically or neglected entirely. This allows a considerable reduction in CPU time as compared with having to analytically evaluate the entire class of terms.

To calculate the integrals, we have implemented algorithms based on the work of Gill, Head-Gordon and Pople[18] (implemented in the well known GAUSSIAN package), but with extensive differences to optimize integrals sharing an atomic center.

2.7.2 Coulomb Corrections

For the Coulomb operator, we consider the following types of two-electron integrals for analytical corrections:

1. One-center terms of the form $\left(aa'|a''a'''\right)$,
2. Two-center terms of the form $\left(aa'|bb'\right)$,
3. Two-center terms of the form $\left(ab|a'b'\right)$,
4. Two-center terms of the form $\left(aa'|a''b\right)$,
5. Three-center terms of the form $\left(aa'|bc\right)$.

Here a, b, and c represent atomic basis functions on atoms A, B, and C respectively.

A crude way to assess the importance of each class of terms is to assign a value of 1 to density matrix elements for occupied orbitals on the same atom and δ for all other density matrix elements. In the expression for the Coulomb energy, terms 1 through 5 are multiplied by the following density matrix elements:

1. $\rho_{aa'}\rho_{a''a'''}$
2. $\rho_{aa'}\rho_{bb'}$
3. $\rho_{ab}\rho_{a'b'}$
4. $\rho_{aa'}\rho_{a''b}$
5. $\rho_{aa'}\rho_{bc}$

Assuming that all of the functions are occupied orbitals, this leads to an approximate magnitude for each class of terms as follows:

$\mathcal{O}(1)$: terms 1 and 2;
$\mathcal{O}(\delta)$: terms 4 and 5;
$\mathcal{O}(\delta^2)$: term 3.

This leads to the following strategy (which must be tested empirically). Terms of the form (1) and (2) should be done analytically for all atoms A and all pairs a and b. Notice as well that there is virtually no falloff in the size of the integrals in class (2) as a function of the A–B separation distance, hence no distance cutoffs are employed here. Terms of the form (4) and (5) are done analytically provided the overlap of the function pair lacking a center coincidence [$a''b$ in (4), bc in

(5)] is greater than a specified threshold. The threshold is adjusted empirically to yield acceptable molecular properties in a given overall pseudospectral scheme, and hence will be a function of the grid, dealiasing scheme, etc. The cutoffs are actually applied to blocks of integrals (e.g. a 2p-2p block, containing 9 different bc function pairs) and utilize pseudo-overlaps computed by averaging the absolute values of the individual overlap integrals in the block and dividing by the number of integrals.

Terms of the form (3) are likely to be important only for atoms in close physical proximity. Consequently, we employ here a simple distance cutoff in which analytical corrections for these terms are carried out provided the distance between atoms A and B is less than a specified tolerance. This may not be optimal (perhaps different functions on the atoms should be treated differently) but leads to reasonable results as shown below.

When a or a' is a polarization function in an aa' pair, the magnitude of the corresponding density matrix element is considerably diminished. For (1) and (2), the cost of computing the analytical integrals is essentially trivial and we ignore this effect. For (4) and (5), we set the cutoff threshold on the bc overlaps differently for this case than for the case where both a and a' are occupied. The two cutoffs are empirically adjusted on a set of test molecules to yield reliable energies and other properties.

Having constructed our correction scheme, we must now devise efficient algorithms for implementing the pseudospectral subtractions. We define two types of restricted Coulomb operators to be subtracted from the full operator in differing specific cases: J_A, in which sums are over functions a, a' that are both on the same atom, and J_S, in which only terms where the absolute value of the overlap integral

of the two basis functions is greater than a given threshold is included:

$$J_A(\mathbf{g}) = \sum_{aa'} \rho_{aa'} A_{aa'}(\mathbf{g}) \tag{2.22}$$

$$J_S(\mathbf{g}) = \sum_{S_{\lambda\sigma} > S_{\min}} \rho_{\lambda\sigma} A_{\lambda\sigma}(\mathbf{g}) \tag{2.23}$$

Note that the sum in Eq. (2.22) extends over all atoms in the molecule.

Two versions of J_A and J_S are constructed. For J_A, one term includes polarization functions and one does not. For J_S, two different thresholds $S_{\min}$ are defined, one of which, S_1, is to be used when a polarization function is involved and the other, S_2, which is used when no polarization function is involved (the usage of J_A and J_S are described below). The computational cost of assembling these four operators is identical to that for evaluating a single operator, as partial sums can be constructed and then added into the appropriate term.

Once the four operators are constructed, the final assembly of the spectral Coulomb matrix is carried out with the appropriate operator, i.e. one in which the terms to be computed analytically are subtracted from the pseudospectral evaluation of matrix element, so as to avoid double counting. For example, in the evaluation of a Coulomb matrix element $J_{aa'}$ between two occupied orbitals a and a', we would subtract operator J_{S_1} if a or a' is a polarization function:

$$J_{aa'} = \sum_{\mathbf{g}} Q_a(\mathbf{g}) \big(J(\mathbf{g}) - J_{S_1}(\mathbf{g}) \big) \phi_{a'}(\mathbf{g}) \tag{2.24}$$

If neither a or a' is a polarization function, Eq. (2.24) would be used with J_{S_2} replacing J_{S_1}. Similarly, if functions b and c are on different atoms B and C, we would subtract either J_{A_1}, J_{A_2}, or nothing, depending upon the size of the overlap

integral S_{bc}.

The principal reason for the use of these cutoffs is to limit the number of three-center, two-electron integrals of the form $\big(aa'|bc\big)$ that must be evaluated analytically. The same criteria involving the overlap integrals $S_{\mu\nu}$ are implemented in the two-electron integral code and only terms satisfying the criteria (e.g., for an $\big(aa'|bc\big)$ integral with a, a' not polarization functions, $S_{bc} > S_1$) are evaluated.

The above algorithm deals with all relevant terms above except for those of the form $\big(ab|a'b'\big)$. In this case, one must compute a "diatomic" Coulomb correction matrix via the equation

$$J_{ab}^{(\mathrm{D})} = \sum_{\mathbf{g}} Q_a(\mathbf{g}) J_{\mathrm{D}}(\mathbf{g}) \phi_b(\mathbf{g}) \tag{2.25}$$

where the diatomic pseudospectral Coulomb field $J_{\mathrm{D}}(\mathbf{g})$ is defined by:

$$J_{\mathrm{D}}(\mathbf{g}) = \sum_{ab} \rho_{ab} A_{ab}(\mathbf{g}) \tag{2.26}$$

the sum being restricted to functions a on atom A, b on atom B.

While $J_{\mathrm{D}}(\mathbf{g})$ itself does not involve extra computation (it can be formed as an intermediate step in ordinary Coulomb assembly), assembly of the correction matrix $J_{ab}^{(\mathrm{D})}$ in Eq. (2.25) is additional work. $J_{ab}^{(\mathrm{D})}$ is then subtracted from the usual J_{ab}. Fortunately, for most iterations in the PS scheme, corrections of this type can be restricted to atom pairs that are nearest neighbors.

2.7.3 Exchange Corrections

For exchange, an analysis of the density matrix elements analogous to that given above for the Coulomb operator yields, for the cases 1-5 defined above:

$\mathcal{O}(1)$: terms 1 and 3;
$\mathcal{O}(\delta)$: term 4;
$\mathcal{O}(\delta^2)$: terms 2 and 5.

This suggests that we treat only two-center terms analytically for exchange. Some compelling reasons for this are: (1) corrections for the three-center terms of the fifth case are quite expensive; also, there is no reason to believe that these terms are larger than many four center terms; (2) a simple distance cutoff can be used. The correction procedure involves calculation of a "diatomic" $\mathbf{K}$ matrix $K_{ab}^{(\mathrm{D})}$:

$$K_{ab}^{(\mathrm{D})} = \sum_{\mathbf{g}} Q_a(\mathbf{g}) K_b^{(\mathrm{D})}(\mathbf{g}) \tag{2.27}$$

where

$$K_b^{(\mathrm{D})}(\mathbf{g}) = \sum_{\lambda \in \{a,b\}} A_{b\lambda}(\mathbf{g}) T_\lambda(\mathbf{g}) \tag{2.28}$$

is the pseudospectral physical space exchange field for the diatomic AB pair and the intermediate quantity $T_\lambda(\mathbf{g})$ is defined as:

$$T_\lambda(\mathbf{g}) = \sum_{\sigma \in \{a,b\}} \rho_{\lambda\sigma} \phi_\sigma(\mathbf{g}) \tag{2.29}$$

The sum over σ again is restricted to functions on A or B.

Once $K_{ab}^{(\mathrm{D})}$ is computed, it can be subtracted from the usual pseudospectral $\mathbf{K}$ matrix. Again, distance cutoffs are used to restrict the exchange corrections

(which do involve extra work) to a small subset of atom pairs.

2.8 Recurrence Techniques

As can be seen from the preceding discussion, the overall speed of our algorithm is fundamentally limited by the ability to calculate NAIs quickly. In addition, an obvious advantage is gained when fast techniques exist for calculating ERIs. Standard spectral codes are themselves constrained by the speed of ERI calculation. This fact has lead over the years to a great deal of attention being focused on ERIs in particular. Although they focused primarily on ERIs, in 1986 Obara-Saika reported a significant advance in the efficient calculation of a variety of integrals over Cartesian Gaussian functions, including NAIs. Since then, others have augmented and improved the Obara-Saika approach, again mostly focusing on ERIs. At the core of all of these methods are sets of recurrence relations. These formulas share characteristics which make them particularly attractive for implementation on computers.

Before discussing particular cases, it may be instructive to consider the advantages achieved by the discovery and usage of the recurrence relations, and the common features that lie at the core of their usefulness. In section 2.10 we will derive recurrence relations applicable to nuclear attraction and related integrals. Without going into detail yet, one example is

$$\left[p_x|A(\mathbf{g})|s\right]^{(0)} = (P_x - A_x)\left[s|A(\mathbf{g})|s\right]^{(0)} + (P_x - C_x)\left[s|A(\mathbf{g})|s\right]^{(1)}$$

where p_x represents a p-type orbital (angular momentum of one) and s is an s-type orbital (angular momentum of zero). Taking the equation at face value for

the moment, we see how it allows us to write an integral containing higher angular momentum elements in terms of integrals of the same form but over functions with lower angular momenta. This demonstrates important general features of this and other recurrence equations. They provide a known path to reduce integrals of any starting function (e.g. s, p, d, f and so on) to quantities we know how to calculate. And, as a by-product, they tend to highlight common terms that can be grouped together beneficially, both within the calculation of a single integral, and in the calculation of groups of integrals. For example, many of the terms above would be identical for a p_y or p_z integral, so we gain extra efficiencies in calculating a related set simultaneously.

Although exemplified here by the relatively simple case of a nuclear attraction-type integral, workers have obtained relations having the same key attributes for other important cases as well. In particular, Pople and coworkers have developed a complete set of expressions from which any two-electron-repulsion integral and their nth-order derivatives may be constructed. In coming chapters we will discuss our use of and modifications to both the Obara-Saika relations and the GHGP techniques.

2.9 Notation and Common Relations

In this work we restrict ourselves to using Cartesian Gaussians for basis functions.

An *unnormalized primitive Cartesian Gaussian function* centered at position $\mathbf{A}$ is given by

$$\varphi_a(\mathbf{r};\alpha,\mathbf{a},\mathbf{A}) = (x-A_x)^{a_x}(y-A_y)^{a_y}(z-A_z)^{a_z}\exp[-\alpha(\mathbf{r}-\mathbf{A})^2], \quad (2.30)$$

where $\mathbf{r} = (x, y, z)$ represents the coordinates of the electron, α the orbital exponent, and $\mathbf{a}$ denotes a set of non-negative integers (a_x, a_y, a_z) known as the angular momentum vector. The angular momentum of the function is defined as $a = a_x + a_y + a_z$. Following the usual nomenclature, the functions with angular momentum 0, 1, 2,... are referred to as s, p, d,..., respectively. A group of functions with a common center, angular momentum and exponent constitute a shell. Each function in the shell is called a component of the shell, with each shell containing $(a+1)(a+2)/2$ components in all. For example, a p shell consists of a set of three functions, with momentum vectors $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$, labeled p_x, p_y, and p_z respectively. The angular momentum vector $\mathbf{1}_i$, with i a Cartesian variable, is defined by

$$\mathbf{1}_i = (\delta_{ix}, \delta_{iy}, \delta_{iz})$$

where δ represents the Kronecker delta. $N_i(\mathbf{a})$ takes the value of the i component of $\mathbf{a}$. It is useful to note that

$$N_i(\mathbf{a} + \mathbf{a}') = N_i(\mathbf{a}) + N_i(\mathbf{a}'),$$

and that

$$N_i(\mathbf{1}_j) = \delta_{ij}.$$

The normalization constant for a primitive Gaussian, obtained by requiring that the integral of the square of the function equal one, is

$$\mathcal{N}(\alpha, \mathbf{a}) = \left(\frac{2^{a_x+a_y+a_z}(2\alpha)^{a_x+a_y+a_z+\frac{3}{2}}}{(2a_x-1)!!(2a_y-1)!!(2a_z-1)!!\pi^{\frac{3}{2}}} \right)^{\frac{1}{2}}$$

This is normally left out of discussions for convenience.

Primitive Gaussians satisfy the important differential relationship

$$\frac{\partial}{\partial A_i}\varphi_a(\mathbf{r};\alpha,\mathbf{a},\mathbf{A}) = 2\alpha\varphi_a(\mathbf{r};\alpha,\mathbf{a}+\mathbf{1}_i,\mathbf{A}) - N_i(\mathbf{a})\varphi_a(\mathbf{r};\alpha,\mathbf{a}-\mathbf{1}_i,\mathbf{A}) \quad (2.31)$$

Current calculations usually employ basis functions made up of linear combinations of primitive Gaussians, called *contracted Cartesian Gaussian* functions, defined as

$$\phi_a(\mathbf{r};\mathbf{a},\mathbf{A}) = \sum_{k=1}^{K} D_k^{(a)}\varphi_{ak}(\mathbf{r};\alpha_k,\mathbf{a},\mathbf{A})$$

where $D_k^{(a)}$ is the contraction coefficient. The index k runs over all the primitives in a contracted function. We will often leave this index off where it is clearly implied by the context. In the recurrence relations to follow uncontracted terms will be denoted by square brackets, while parentheses indicate contracted terms (wherein uncontracted functions of primitive Gaussians have been multiplied by the related contraction coefficients and summed).

2.10 Core Pseudospectral $A_{ab}(\mathbf{g})$ Calculations

2.10.1 Overview

Previously we showed that calculations of integrals of the form

$$\int d\mathbf{r}\, \varphi_a(\mathbf{r};\alpha,\mathbf{a},\mathbf{A})\frac{1}{|\mathbf{r}-\mathbf{g}|}\varphi_b(\mathbf{r};\beta,\mathbf{b},\mathbf{B})$$

are central to the pseudospectral method. These integrals are identical in form to the nuclear attraction integrals needed as part of the Hartree-Fock core term, with the gridpoints replacing the role of the atomic nuclei. In their 1985 paper, Obara and Saika derived, among other things, a generalized recursion relation for nuclear attraction integrals in terms of auxiliary integrals. Their approach provides a highly efficient method of calculating nuclear attraction integrals over primitive Gaussian functions. In the following sections, we derive the standard Obara-Saika equations, present two modifications to the standard relationships that produce further reductions in computational cost, and analyze the application of these techniques in detail.

2.10.2 The Obara-Saika Equations

We define the uncontracted potential integral for a gridpoint located at $\mathbf{g}$ as

$$\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\big] = \int d\mathbf{r}\, \varphi_a(\mathbf{r};\alpha,\mathbf{a},\mathbf{A})\frac{1}{|\mathbf{r}-\mathbf{g}|}\varphi_b(\mathbf{r};\beta,\mathbf{b},\mathbf{B})$$

where we have denoted the electric potential operator as $A(\mathbf{g}) = \frac{1}{|\mathbf{r}-\mathbf{g}|}$. Using the identity

$$|\mathbf{r_1}-\mathbf{r_2}|^{-1} = \frac{2}{\pi^{\frac{1}{2}}}\int_0^\infty du\, \exp[-u^2(\mathbf{r_1}-\mathbf{r_2})^2]$$

we may rewrite this as

$$\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\big] = \frac{2}{\pi^{\frac{1}{2}}}\int_0^\infty du\big[\mathbf{a}|\mathbf{0}_g|\mathbf{b}\big] \tag{2.32}$$

where

$$\big[\mathbf{a}|\mathbf{0}_g|\mathbf{b}\big] = \int d\mathbf{r}\,\varphi_a(\mathbf{r};\alpha,\mathbf{a},\mathbf{A})\varphi_b(\mathbf{r};\beta,\mathbf{b},\mathbf{B})\exp[-u^2(\mathbf{r}-\mathbf{g})^2] \tag{2.33}$$

The exponential in Eq. (2.33) corresponds to an s-type Cartesian Gaussian centered on $\mathbf{g}$ with exponent u^2. This integral, then, is a special case of a more general integral

$$\big[\mathbf{a}|\mathbf{c}|\mathbf{b}\big] = \int d\mathbf{r}\,\varphi_a(\mathbf{r};\alpha,\mathbf{a},\mathbf{A})\varphi_c(\mathbf{r};\gamma,\mathbf{c},\mathbf{C})\varphi_b(\mathbf{r};\beta,\mathbf{b},\mathbf{B}),$$

known as a three-center overlap integral.

Three-center overlap integrals

Using Eq. (2.31), the integral $\big[\mathbf{a}+\mathbf{1}_i|\mathbf{c}|\mathbf{b}\big]$ can be rewritten as

$$\big[\mathbf{a}+\mathbf{1}_i|\mathbf{c}|\mathbf{b}\big] = \frac{1}{2\alpha}\frac{\partial}{\partial A_i}\big[\mathbf{a}|\mathbf{c}|\mathbf{b}\big] + \frac{1}{2\alpha}N_i(\mathbf{a})\big[\mathbf{a}-\mathbf{1}_i|\mathbf{c}|\mathbf{b}\big] \tag{2.34}$$

Evaluating the right hand side of this equation provides the basis for a useful recursion relation. We proceed by finding an expression for the integral $\big[\mathbf{a}|\mathbf{c}|\mathbf{b}\big]$. To do this, recall that each function has the form of an exponential multiplied by

a polynomial. We first combine the exponential factors to write

$$\alpha(\mathbf{r}-\mathbf{A})^2+\beta(\mathbf{r}-\mathbf{B})^2+\gamma(\mathbf{r}-\mathbf{C})^2 = \frac{\alpha\beta}{\alpha+\beta}(\mathbf{A}-\mathbf{B})^2+\frac{(\alpha+\beta)\gamma}{\alpha+\beta+\gamma}(\mathbf{P}-\mathbf{C})^2+ (\alpha+\beta+\gamma)(\mathbf{r}-\mathbf{G})^2 \quad (2.35)$$

$\mathbf{P}$ represents the usual product center of two Gaussians:

$$\mathbf{P}=\frac{\alpha\mathbf{A}+\beta\mathbf{B}}{\alpha+\beta} \quad (2.36)$$

$\mathbf{G}$ is the combined product center for all three Gaussians:

$$\mathbf{G}=\frac{(\alpha+\beta)\mathbf{P}+\gamma\mathbf{C}}{\alpha+\beta+\gamma}=\frac{\alpha\mathbf{A}+\beta\mathbf{B}+\gamma\mathbf{C}}{\alpha+\beta+\gamma} \quad (2.37)$$

We now have a single Gaussian centered on $\mathbf{G}$ multiplied by a constant exponential prefactor. With a bit of algebra, the exponent of the prefactor can be written in a more symmetric form:

$$\frac{(\alpha+\beta)\gamma}{\alpha+\beta+\gamma}(\mathbf{P}-\mathbf{C})^2+\frac{\alpha\beta}{\alpha+\beta}(\mathbf{A}-\mathbf{B})^2 = (\alpha+\beta+\gamma)\Big\{\frac{\alpha\mathbf{A}^2+\beta\mathbf{B}^2+\gamma\mathbf{C}^2}{\alpha+\beta+\gamma}-\mathbf{G}^2\Big\} \quad (2.38)$$

To evaluate the remaining integral over a Gaussian centered on $\mathbf{G}$, we must express the original polynomial coefficients in terms of polynomials centered on

G, using the following equation.

$$(r_i - R_i)^{n_i} = (r_i - G_i + G_i - R_i)^{n_i} = \sum_{k=0}^{n_i} \binom{n_i}{k} (r_i - G_i)^k (G_i - R_i)^{n_i - k} \quad (2.39)$$

Combining the steps above, we can write

$$\left[\mathbf{a}|\mathbf{c}|\mathbf{b}\right] = \mathcal{K}_{abc} \mathcal{I}_x(a_x, b_x, c_x) \mathcal{I}_y(a_y, b_y, c_y) \mathcal{I}_z(a_z, b_z, c_z) \quad (2.40)$$

where

$$\mathcal{K}_{abc} = \exp\left[(\alpha + \beta + \gamma)\left\{\mathbf{G}^2 - \frac{\alpha\mathbf{A}^2 + \beta\mathbf{B}^2 + \gamma\mathbf{C}^2}{\alpha + \beta + \gamma}\right\}\right] \quad (2.41)$$

and

$$\begin{aligned}
&\mathcal{I}_i(a_i, b_i, c_i) \\
&= \int dr_i \exp\left[-(\alpha + \beta + \gamma)(r_i - G_i)^2\right] \sum_{k_a=0}^{a_i} \sum_{k_b=0}^{b_i} \sum_{k_c=0}^{c_i} \binom{a_i}{k_a} \binom{b_i}{k_b} \binom{c_i}{k_c} \\
&\quad (r_i - G_i)^{k_a} (G_i - A_i)^{a_i - k_a} (r_i - G_i)^{k_b} (G_i - B_i)^{b_i - k_b} (r_i - G_i)^{k_c} (G_i - C_i)^{c_i - k_c} \\
&= \sum_{k_a=0}^{a_i} \sum_{k_b=0}^{b_i} \sum_{k_c=0}^{c_i} \binom{a_i}{k_a} \binom{b_i}{k_b} \binom{c_i}{k_c} (G_i - A_i)^{a_i - k_a} (G_i - B_i)^{b_i - k_b} (G_i - C_i)^{c_i - k_c} \\
&\quad \int dr_i \exp\left[-(\alpha + \beta + \gamma)(r_i - G_i)^2\right] (r_i - G_i)^{k_a + k_b + k_c} \\
&= \left(\frac{\pi}{\alpha + \beta + \gamma}\right)^{\frac{1}{2}} \sum_{\substack{k_a=0 \\ (k_a + k_b + k_c \text{ even})}}^{a_i} \sum_{k_b=0}^{b_i} \sum_{k_c=0}^{c_i} \binom{a_i}{k_a} \binom{b_i}{k_b} \binom{c_i}{k_c} \\
&\quad \frac{(k_a + k_b + k_c)!!}{[2(\alpha + \beta + \gamma)]^{k_a + k_b + k_c}} (G_i - A_i)^{a_i - k_a} (G_i - B_i)^{b_i - k_b} (G_i - C_i)^{c_i - k_c}
\end{aligned} \quad (2.42)$$

We may now take the derivative required in Eq. (2.34). The derivative of the product center $\mathbf{G}$ is

$$\frac{\partial}{\partial A_i} G_j = \frac{\alpha}{\alpha+\beta+\gamma}\delta_{ij} \tag{2.43}$$

The derivative of the prefactor is then

$$\frac{1}{2\alpha}\frac{\partial}{\partial A_i}\mathcal{K}_{abc} = (G_i - A_i)\mathcal{K}_{abc} \tag{2.44}$$

Looking at a simplified polynomial sum, we have

$$\begin{aligned}
&\frac{1}{2\alpha}\frac{\partial}{\partial A_i}\sum_{k_a=0}^{a_i}\binom{a_i}{k_a}(G_i - A_i)^{a_i-k_a} \\
&= \sum_{k_a=0}^{a_i}(a_i - k_a)\binom{a_i}{k_a}(G_i - A_i)^{a_i-k_i-1}(\frac{\alpha}{\alpha+\beta+\gamma} - 1) \\
&= a_i\sum_{k_a=0}^{a_i-1}\binom{a_i-1}{k_a}(G_i - A_i)^{a_i-k_i-1}(\frac{\alpha}{\alpha+\beta+\gamma} - 1)
\end{aligned} \tag{2.45}$$

Inserting in Eq. (2.42) gives

$$\begin{aligned}
\frac{1}{2\alpha}\frac{\partial}{\partial A_i}\mathcal{I}_i = a_i&\left\{\frac{1}{2(\alpha+\beta+\gamma)} - \frac{1}{2\alpha}\right\}\mathcal{I}_i(a_i-1, b_i, c_i) \\
&+ b_i\frac{1}{2(\alpha+\beta+\gamma)}\mathcal{I}_i(a_i, b_i-1, c_i) \\
&+ c_i\frac{1}{2(\alpha+\beta+\gamma)}\mathcal{I}_i(a_i, b_i, c_i-1)
\end{aligned} \tag{2.46}$$

Assembling all the derivative terms, Eq. (2.34) becomes

$$\begin{aligned}\big[\mathbf{a}+\mathbf{1}_i|\mathbf{c}|\mathbf{b}\big] &= (G_i - A_i)\big[\mathbf{a}|\mathbf{c}|\mathbf{b}\big] \\ &+ \frac{1}{2(\alpha+\beta+\gamma)} N_i(\mathbf{a})\big[\mathbf{a}-\mathbf{1}_i|\mathbf{c}|\mathbf{b}\big] \\ &+ \frac{1}{2(\alpha+\beta+\gamma)} N_i(\mathbf{b})\big[\mathbf{a}|\mathbf{c}|\mathbf{b}-\mathbf{1}_i\big] \\ &+ \frac{1}{2(\alpha+\beta+\gamma)} N_i(\mathbf{c})\big[\mathbf{a}|\mathbf{c}-\mathbf{1}_i|\mathbf{b}\big]\end{aligned} \tag{2.47}$$

Recurrence Relationships

The previous equation gives an expression for the general case of a three-center overlap integral. Returning to the particular case of interest and defining $\zeta = \alpha+\beta$ we have

$$\begin{aligned}\big[\mathbf{a}+\mathbf{1}_i|\mathbf{0}_g|\mathbf{b}\big] &= (G_i - A_i)\big[\mathbf{a}|\mathbf{0}_g|\mathbf{b}\big] \\ &+ \frac{1}{2(\zeta+u^2)} N_i(\mathbf{a})\big[\mathbf{a}-\mathbf{1}_i|\mathbf{0}_g|\mathbf{b}\big] \\ &+ \frac{1}{2(\zeta+u^2)} N_i(\mathbf{b})\big[\mathbf{a}|\mathbf{0}_g|\mathbf{b}-\mathbf{1}_i\big] \\ &= (P_i - A_i)\big[\mathbf{a}|\mathbf{0}_g|\mathbf{b}\big] - (P_i - C_i)\frac{u^2}{\zeta+u^2}\big[\mathbf{a}|\mathbf{0}_g|\mathbf{b}\big] \\ &+ \frac{1}{2\zeta} N_i(\mathbf{a})\left(1 - \frac{u^2}{\zeta+u^2}\right)\big[\mathbf{a}-\mathbf{1}_i|\mathbf{0}_g|\mathbf{b}\big] \\ &+ \frac{1}{2\zeta} N_i(\mathbf{b})\left(1 - \frac{u^2}{\zeta+u^2}\right)\big[\mathbf{a}|\mathbf{0}_g|\mathbf{b}-\mathbf{1}_i\big]\end{aligned} \tag{2.48}$$

We now define an auxiliary integral

$$\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\big]^{(m)} = \frac{2}{\pi^{\frac{1}{2}}}\int_0^\infty du \left(\frac{u^2}{\zeta+u^2}\right)^m \big[\mathbf{a}|\mathbf{0}_g|\mathbf{b}\big] \tag{2.49}$$

which reduces to Eq. (2.32) for $m = 0$. Finally, with this definition, inserting Eq. (2.48) into Eq. (2.32) we have the Obara-Saika recursion relation

$$\begin{aligned}\big[\mathbf{a}+\mathbf{1_i}|A(\mathbf{g})|\mathbf{b}\big]^{(m)} &= (P_i - A_i)\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\big]^{(m)} - (P_i - C_i)\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\big]^{(m+1)} \\ &+ \frac{1}{2\zeta}N_i(\mathbf{a})\{\big[\mathbf{a}-\mathbf{1_i}|A(\mathbf{g})|\mathbf{b}\big]^{(m)} - \big[\mathbf{a}-\mathbf{1_i}|A(\mathbf{g})|\mathbf{b}\big]^{(m+1)}\} \\ &+ \frac{1}{2\zeta}N_i(\mathbf{b})\{\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}-\mathbf{1_i}\big]^{(m)} - \big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}-\mathbf{1_i}\big]^{(m+1)}\}\end{aligned} \tag{2.50}$$

By symmetry, we also have

$$\begin{aligned}\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}+\mathbf{1_i}\big]^{(m)} &= (P_i - B_i)\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\big]^{(m)} - (P_i - C_i)\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\big]^{(m+1)} \\ &+ \frac{1}{2\zeta}N_i(\mathbf{a})\{\big[\mathbf{a}-\mathbf{1_i}|A(\mathbf{g})|\mathbf{b}\big]^{(m)} - \big[\mathbf{a}-\mathbf{1_i}|A(\mathbf{g})|\mathbf{b}\big]^{(m+1)}\} \\ &+ \frac{1}{2\zeta}N_i(\mathbf{b})\{\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}-\mathbf{1_i}\big]^{(m)} - \big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}-\mathbf{1_i}\big]^{(m+1)}\}\end{aligned} \tag{2.51}$$

By applying these relationships iteratively, we can reduce any potential integral to a combination of integrals over s functions. The problem then becomes that of evaluating three-center integrals of s functions. Starting with Eq. (2.40), we have

$$\begin{aligned}\big[\mathbf{0}_A|\mathbf{0}_C|\mathbf{0}_B\big] &= \big(\frac{\pi}{\alpha+\beta+\gamma}\big)^{\frac{3}{2}}\mathcal{K}_{abc} \\ &= \left(\frac{\alpha+\beta}{\alpha+\beta+\gamma}\right)^{\frac{3}{2}}\big[\mathbf{0}_A|\mathbf{0}_B\big]\exp\big[-\frac{(\alpha+\beta)\gamma}{\alpha+\beta+\gamma}(\mathbf{P}-\mathbf{C})^2\big]\end{aligned} \tag{2.52}$$

$\big[\mathbf{0}_A|\mathbf{0}_B\big]$ is the overlap integral between two s functions centered at $\mathbf{A}$ and $\mathbf{B}$,

which can be expressed in closed form:

$$\big[\mathbf{0}_A|\mathbf{0}_B\big] = \left(\frac{\pi}{\alpha+\beta}\right)^{\frac{3}{2}} \exp\big[-\frac{\alpha\beta}{\alpha+\beta}(\mathbf{A}-\mathbf{B})^2\big] \tag{2.53}$$

Inserting these results into the definition of the auxiliary integrals gives

$$\begin{aligned}
&\big[\mathbf{0}_A|A(\mathbf{g})|\mathbf{0}_B\big]^{(m)} \\
&= \frac{2}{\pi^{\frac{1}{2}}} \int_0^\infty du \left(\frac{u^2}{\zeta+u^2}\right)^m \left(\frac{\zeta}{\zeta+u^2}\right)^{\frac{3}{2}} \big[\mathbf{0}_A|\mathbf{0}_B\big] \exp\left[-\zeta(\mathbf{P}-\mathbf{C})^2 \frac{u^2}{\zeta+u^2}\right] \\
&= 2\left(\frac{\zeta}{\pi}\right)^{\frac{1}{2}} \big[\mathbf{0}_A|\mathbf{0}_B\big] \mathcal{F}_m(\zeta(\mathbf{P}-\mathbf{C})^2)
\end{aligned} \tag{2.54}$$

where the function

$$\mathcal{F}_m(T) = \int_0^\infty du \left(\frac{u^2}{\zeta+u^2}\right)^m \left(\frac{\zeta}{\zeta+u^2}\right)^{\frac{3}{2}} \exp\left[-T\frac{u^2}{\zeta+u^2}\right]$$

was first introduced by Boys and is found throughout the literature.[17,18,33] With a transformation of variables

$$t^2 = \frac{u^2}{\zeta+u^2}$$

we obtain the familiar alternate form

$$\mathcal{F}_m(T) = \int_0^1 dt\, t^{2m} \exp(-Tt^2)$$

See the discussion of Eq. (2.75) for further details on computing values of this function.

Using these expressions it is a straightforward matter to write, explicitly or

implicitly, any potential integral in terms of auxiliary integrals of s functions. The following examples illustrate a few pertinent cases.

$$\big[p_i|A(\mathbf{g})|s\big]^{(0)} = (P_i - A_i)\big[s|A(\mathbf{g})|s\big]^{(0)} + (P_i - C_i)\big[s|A(\mathbf{g})|s\big]^{(1)}$$

$$\begin{aligned}\big[p_i|A(\mathbf{g})|p_j\big]^{(0)} &= (P_j - B_j)\big[p_i|A(\mathbf{g})|s\big]^{(0)} + (P_j - C_j)\big[p_i|A(\mathbf{g})|s\big]^{(1)} \\ &\quad + \frac{\delta_{ij}}{2\zeta}\{\big[s|A(\mathbf{g})|s\big]^{(0)} - \big[s|A(\mathbf{g})|s\big]^{(1)}\}\end{aligned}$$

$$\begin{aligned}\big[d_{ij}|A(\mathbf{g})|s\big]^{(0)} &= (P_j - A_j)\big[p_i|A(\mathbf{g})|s\big]^{(0)} + (P_j - C_j)\big[p_i|A(\mathbf{g})|s\big]^{(1)} \\ &\quad + \frac{\delta_{ij}}{2\zeta}\{\big[s|A(\mathbf{g})|s\big]^{(0)} - \big[s|A(\mathbf{g})|s\big]^{(1)}\}\end{aligned}$$

$$\begin{aligned}\big[d_{ij}|A(\mathbf{g})|p_k\big]^{(0)} &= (P_k - B_k)\big[d_{ij}|A(\mathbf{g})|s\big]^{(0)} + (P_k - C_k)\big[d_{ij}|A(\mathbf{g})|s\big]^{(1)} \\ &\quad + \frac{\delta_{ik}}{2\zeta}\{\big[p_j|A(\mathbf{g})|s\big]^{(0)} - \big[p_j|A(\mathbf{g})|s\big]^{(1)}\} \\ &\quad + \frac{\delta_{jk}}{2\zeta}\{\big[p_i|A(\mathbf{g})|s\big]^{(0)} - \big[p_i|A(\mathbf{g})|s\big]^{(1)}\}\end{aligned}$$

$$\begin{aligned}\big[d_{ij}|A(\mathbf{g})|d_{kl}\big]^{(0)} &= (P_l - B_l)\big[d_{ij}|A(\mathbf{g})|p_k\big]^{(0)} + (P_l - C_l)\big[d_{ij}|A(\mathbf{g})|p_k\big]^{(1)} \\ &\quad + \frac{\delta_{il}}{2\zeta}\{\big[p_j|A(\mathbf{g})|p_k\big]^{(0)} - \big[p_j|A(\mathbf{g})|p_k\big]^{(1)}\} \\ &\quad + \frac{\delta_{jl}}{2\zeta}\{\big[p_i|A(\mathbf{g})|p_k\big]^{(0)} - \big[p_i|A(\mathbf{g})|p_k\big]^{(1)}\} \\ &\quad + \frac{\delta_{kl}}{2\zeta}\{\big[d_{ij}|A(\mathbf{g})|s\big]^{(0)} - \big[d_{ij}|A(\mathbf{g})|s\big]^{(1)}\}\end{aligned}$$

2.10.3 Analysis of the Computational Costs of an Algorithm Based on the Obara-Saika Equations

It is useful to study the number of mathematical operations* required to form one of the sets of integrals given previously. Such analysis does not tell the whole story of the computational efficiency of an algorithm, but usually uncovers the main character. At this point we note that the equations have been given without regard to contraction of the basis functions. This is an important factor that adds complexity to the problem. Due to the recursive nature of the equations, it becomes necessary to look at the cost of computing both contracted and uncontracted integrals, in order to know the final cost of a set of contracted terms.

For the purposes of this discussion we consider quantities such as $\left[\mathbf{0}_A|\mathbf{0}_B\right]$ and $\mathcal{F}_m$ to be fundamental. They are required by essentially all algorithms such as the one under discussion. Forming these terms can be viewed as a constant cost common to all, and therefore left out of consideration.

In constructing contracted integrals, we note the following:

- A_i and B_i are constants.
- C_i depends on the gridpoint.
- P_i and ζ depend on the contraction pair.
- $\left[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\right]^{(m)}$ depends on the contraction pair and gridpoint.

*By "operations" we mean any of the elemental mathematical operations normally available on computers, i.e. adds, subtracts, multiplies and divides.

Computational cost of $\big(s|A(\mathbf{g})|s\big)^{(m)}$

The uncontracted integral $\big[s|A(\mathbf{g})|s\big]^{(m)}$ is calculated directly from the related overlap and $\mathcal{F}_m$ terms, as given in Eq. (2.54). The contracted integral is then

$$\big(s|A(\mathbf{g})|s\big)^{(m)} = \sum_{k=1}^{K_k}\sum_{l=1}^{K_l} D_k D_l \big[s|A(\mathbf{g})|s\big]^{(m)} \tag{2.55}$$

Let n_p denote the number of contraction pairs (i.e. $K_k K_l$), and n_g the number of gridpoints. Then the computational effort to perform the summation in Eq. (2.55) requires $n_p n_g + (n_p - 1)n_g$ operations (assuming the coefficients $D_k D_l$ have previously been combined).

Computational cost of $\big(p|A(\mathbf{g})|s\big)^{(m)}$

All three instances of the $\big[p_i|A(\mathbf{g})|s\big]^{(m)}$ case have the form

$$\big[p_i|A(\mathbf{g})|s\big]^{(m)} = (P_i - A_i)\big[s|A(\mathbf{g})|s\big]^{(m)} + (P_i - C_i)\big[s|A(\mathbf{g})|s\big]^{(m+1)} \tag{2.56}$$

The three terms $(P_i - A_i)$, $i \in (x, y, z)$ require $3n_p$ operations to form. Forming all $(P_i - C_i)$ requires $3n_p n_g$. With these values, the final assembly of each term $\big[p_i|A(\mathbf{g})|s\big]^{(m)}$ in Eq. (2.56) requires $3n_p n_g$ operations, or $9n_p n_g$ for all three. The total cost for a set of uncontracted $\big[p_i|A(\mathbf{g})|s\big]^{(m)}$ integrals is then $3n_p + 3n_p n_g + 9n_p n_g = 3n_p + 12n_p n_g$. Summing to perform the contraction requires $33n_p n_g + 3(n_p - 1)n_g$ steps. The total operation count for a single group of contracted integrals $\big(p_i|A(\mathbf{g})|s\big)^{(m)}$ $i \in (x, y, z)$ is $3n_p - 3n_g + 18n_p n_g$.

Computational cost of $\big(p|A(\mathbf{g})|p\big)^{(m)}$

$\big[p_i|A(\mathbf{g})|p_i\big]^{(m)}\ \ i \in \{x, y, z\}$ have the form

$$\big[p_i|A(\mathbf{g})|p_i\big]^{(m)} = (P_i - B_i)\big[p_i|A(\mathbf{g})|s\big]^{(m)} + (P_i - C_i)\big[p_i|A(\mathbf{g})|s\big]^{(m+1)} + \frac{1}{2\zeta}\{\big[s|A(\mathbf{g})|s\big]^{(m)} - \big[s|A(\mathbf{g})|s\big]^{(m+1)}\} \tag{2.57}$$

whereas $\big[p_i|A(\mathbf{g})|p_j\big]^{(m)}\ \ i, j \in \{x, y, z\}, i \neq j$ have the form

$$\big[p_i|A(\mathbf{g})|p_j\big]^{(m)} = (P_j - B_j)\big[p_i|A(\mathbf{g})|s\big]^{(m)} + (P_j - C_j)\big[p_i|A(\mathbf{g})|s\big]^{(m+1)} \tag{2.58}$$

Here we see the first instance where we can do something better to generate the contracted case than just forming the uncontracted terms and summing. Three of the integrals have the term $\frac{1}{2\zeta}\{\big[s|A(\mathbf{g})|s\big]^{(m)} - \big[s|A(\mathbf{g})|s\big]^{(m+1)}\}$ in common. For the uncontracted case we obtain some savings by calculating this term once and adding it to all three integrals. For the contracted case we can save even more by forming *and contracting separately* this term, then adding it to the appropriate sums. Compare the implied order of operations in

$$\big(p_i|A(\mathbf{g})|p_i\big)^{(m)} = \sum_k \sum_l D_k D_l \big[(P_i - B_i)\big[p_i|A(\mathbf{g})|s\big]^{(m)} + (P_i - C_i)\big[p_i|A(\mathbf{g})|s\big]^{(m+1)} + \frac{1}{2\zeta}\{\big[s|A(\mathbf{g})|s\big]^{(m)} - \big[s|A(\mathbf{g})|s\big]^{(m+1)}\}\big] \tag{2.59}$$

versus

$$\big(p_i|A(\mathbf{g})|p_i\big)^{(m)} = \sum_k \sum_l D_k D_l \big[(P_i - B_i)\big[p_i|A(\mathbf{g})|s\big]^{(m)} + (P_i - C_i)\big[p_i|A(\mathbf{g})|s\big]^{(m+1)}\big] + \sum_k \sum_l D_k D_l \big[\frac{1}{2\zeta}\{\big[s|A(\mathbf{g})|s\big]^{(m)} - \big[s|A(\mathbf{g})|s\big]^{(m+1)}\}\big] \quad (2.60)$$

Ignoring steps common to both Eqs. (2.59) and (2.60), we will only consider the differences. The factors $D_k D_l$ can be folded in with other terms, so we ignore them too. Adding the uncontracted terms in Eq. (2.59) requires $3n_p n_g$ operations. Contracting the combined terms takes $9(n_p - 1)n_g$ operations, for a total of $12n_p n_g - 9n_g$ steps for completion. Performing the two contraction sums in Eq. (2.60) uses $9(n_p - 1)n_g$ and $(n_p - 1)n_g$ operations, respectively. The final addition of contracted terms takes $3n_g$, for a total of $10n_p n_g - 7n_g$ steps for completion. Subtracting the totals for comparison, we have a difference of $2n_p n_g - 2n_g$ fewer operations using the second method.

For $n_p = 1$, the operation counts are the same, as would be expected. For any contraction degree $n_p > 1$, the second method takes fewer operations. Here n_g acts only as a scale factor. It is a very important scale factor for the Pseudospectral method, though, because of the large number of gridpoints used. From this comparison, we can see explicitly that the computational savings comes from being able to drop three $n_p n_g$ additions of uncontracted terms for an extra $(n_p - 1)n_g$ contraction step and three n_g additions of contracted terms. By comparison to the overall cost of computing these integrals, saving $2n_p n_g - 2n_g$ is not overly significant. Following the same methodology, though, the savings grows rapidly as the

angular momenta of the functions involved increase. The reduction comes at the expense of little effort, due to the form of the recursion relations. Furthermore, we shall see that related techniques can cut the computational effort even more.

2.10.4 Improvements to the Obara-Saika Equations in the Molecular Frame

It turns out we can improve upon the Obara-Saika method even further for the contracted case. Combining Eqs. (2.50) and (2.51) results in the simple relation

$$\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}+\mathbf{1_i}\big]^{(m)} = \big[\mathbf{a}+\mathbf{1_i}|A(\mathbf{g})|\mathbf{b}\big]^{(m)} + (A_i - B_i)\big[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\big]^{(m)} \quad (2.61)$$

This allows us to write any integral in terms of integrals where one function is always an s type. Forming an uncontracted set of integrals this way is actually more expensive than the previous method, but we if we make use of the same technique of contracting the terms on the right of Eq. (2.62) before adding them together, we achieve a savings for any contraction degree greater than one. To illustrate this, using the same example of two p functions from the last section, we have

$$\big[p_i|A(\mathbf{g})|p_j\big]^{(m)} = \big[d_{ij}|A(\mathbf{g})|s\big]^{(m)} + (A_j - B_j)\big[p_i|A(\mathbf{g})|s\big]^{(m)} \quad (2.62)$$

Because the coefficients $(A_i - B_i)$ depend only on the coordinates of the atomic centers, we can immediately write

$$\big(p_i|A(\mathbf{g})|p_j\big)^{(m)} = \big(d_{ij}|A(\mathbf{g})|s\big)^{(m)} + (A_j - B_j)\big(p_i|A(\mathbf{g})|s\big)^{(m)} \quad (2.63)$$

Besides the benefit of contracting common terms, this formulation has other

advantages. Because d functions are invariant under interchange of the indices i and j, there are only six independent terms in $\left(d_{ij}|A(\mathbf{g})|s\right)^{(m)}$, not nine. Recalling Eq. (2.50) we also see that the second integral on the right hand side $\left[p_i|A(\mathbf{g})|s\right]^{(m)}$ is required to form the first, $\left[d_{ij}|A(\mathbf{g})|s\right]^{(m)}$, so we have it on hand at no additional computational expense.

The steps in computing Eqs. (2.60) and (2.63) diverge at an earlier stage than in the previous example. Examining Eq. (2.60), forming the combined uncontracted terms on the right hand side uses $29n_pn_g$ operations. The rest of the computational cost follows as given previously, for a total of $39n_pn_g - 7n_g$ steps for completion. Turning to Eq. (2.63), forming the uncontracted components of $\left[d_{ij}|A(\mathbf{g})|s\right]^{(m)}$ takes $20n_pn_g$ operations. Contracting and assembling the contracted integrals $\left(d_{ij}|A(\mathbf{g})|s\right)^{(m)}$ calls for another $7(n_p - 1)n_g + 3n_g$. Forming the contracted set $\left(p_i|A(\mathbf{g})|s\right)^{(m)}$ adds another $3(n_p - 1)n_g$. Pulling together the final contracted integrals $\left(p_i|A(\mathbf{g})|p_j\right)^{(m)}$ uses $18n_g$ operations, for a total of $11n_g + 30n_pn_g$ step for completion.

Comparing, for $n_p = 1$ using Eq. (2.60) takes $32n_g$ versus $41n_g$ steps. But for any $n_p > 1$, Eq. (2.63) takes fewer operations. The trade comes in needing to construct nine fewer uncontracted components for Eq. (2.63) in exchange for an extra $18n_g$ operations combining contracted terms. Note the invariance under the interchange of indices gives increasing gains at higher angular momenta. A $\left[d|A(\mathbf{g})|d\right]$ set has 36 unique integrals, whereas the related $\left[g|A(\mathbf{g})|s\right]$ set has only 15.

2.10.5 The Diatomic Frame

The previous section discussed the Obara-Saika relations using the general molecular frame, i.e. a frame in which the atom pairs involved have no special spatial orientation or position with regards to the coordinate system. Because the large number of gridpoints used allows us to block out operations for sets of gridpoints for efficiency, we can treat one atom pair at a time. Once an atom pair has been selected we transform the system to a new coordinate system in which the pair is aligned with the z-axis, i.e. the x and y coordinates of the atomic centers are zero. This eliminates a number of terms from the calculations described previously, enough to more than make up for the cost of transforming the grid to the new coordinate system. Looking at Eq. (2.50), we have, for $i \in (x, y)$,

$$\begin{aligned}\left[\mathbf{a}+\mathbf{1_i}|A(\mathbf{g})|\mathbf{b}\right]^{(m)} &= C_i\left[\mathbf{a}|A(\mathbf{g})|\mathbf{b}\right]^{(m+1)} \\ &+ \frac{1}{2\zeta}N_i(\mathbf{a})\{\left[\mathbf{a}-\mathbf{1_i}|A(\mathbf{g})|\mathbf{b}\right]^{(m)} - \left[\mathbf{a}-\mathbf{1_i}|A(\mathbf{g})|\mathbf{b}\right]^{(m+1)}\} \\ &+ \frac{1}{2\zeta}N_i(\mathbf{b})\{\left[\mathbf{a}|A(\mathbf{g})|\mathbf{b}-\mathbf{1_i}\right]^{(m)} - \left[\mathbf{a}|A(\mathbf{g})|\mathbf{b}-\mathbf{1_i}\right]^{(m+1)}\} \quad (2.64)\end{aligned}$$

Not only have several terms been eliminated, but the coefficient of the first term on the right side (the grid point coordinate) is now independent of the function pair. This leads to another instance where a common term can be contracted separately and added in to reduce the overall computational cost.

Eq. (2.62) reduces to something even simpler in the same case. Again, with i, $j \in (x, y)$

$$\left[p_i|A(\mathbf{g})|p_j\right]^{(m)} = \left[d_{ij}|A(\mathbf{g})|s\right]^{(m)} \quad (2.65)$$

Because $d_{xy} = d_{yx}$, this means we only need to compute three integrals, not four.

2.10.6 Superblocks

Another advantage of the Obara-Saika recursion formulas is that it is particularly easy to take advantage of superblocks. Superblocks are sets of integrals of different angular momenta that share the same exponents. Since, for example, an integral of a p function requires the construction of the related zeroth order s auxiliary function, we essentially get the s function for free as a side benefit, at only the cost of performing the contraction. In contrast, the older direct methods could only make use of some common elements that still required considerable extra manipulation to obtain results. Basis sets most often contain s and p type functions in superblocks.

2.11 Two-Electron Integral Calculation

2.11.1 Overview

The Pseudospectral method has been shown previously to be capable of attaining chemical accuracy with a computational effort that scales more favorably with system size than spectral methods. A hybrid approach goes further, limiting the number of two-electron integrals that dominate the scaling of spectral methods while reducing the number of gridpoints needed for the grid-based Pseudospectral quadrature. This hybrid method significantly reduces the computational costs relative to both non-hybrid methods without compromising accuracy.

In the following pages, we describe an efficient generalized recurrence technique for calculating two-electron integrals, based on the work by Gill, Head-Gordon and Pople.[18] The recurrence relationships used share similarities with the Obara-Saika work discussed previously, but here the emphasis will be on the somewhat complex details necessary to make a hybrid approach practical and effective. Gill *et al.* described a general algorithm for computing two-electron integrals using their recurrence relationships. Since we are concerned with a number of special cases, our implementation differs greatly. We will therefore discuss the recurrence relationships, then describe the actual computer implementation in some depth.

2.11.2 Notation

Gill *et al.* defined a notation to aid in manipulating ERIs and their derivatives. First, from previous definitions, we write a *primitive ERI* as the integral

$$\big[\mathbf{a}_k\mathbf{b}_\ell|\mathbf{c}_m\mathbf{d}_n\big] = \int\int d\mathbf{r}_1 d\mathbf{r}_2\, \varphi_{ak}(\mathbf{r}_1)\varphi_{bl}(\mathbf{r}_1) r_{12}^{-1} \varphi_{cm}(\mathbf{r}_2)\varphi_{dn}(\mathbf{r}_2)$$

Following the usual route we then define a *contracted ERI* as

$$\big(\mathbf{ab}|\mathbf{cd}\big) = \sum_{k=1}^{K_a}\sum_{\ell=1}^{K_b}\sum_{m=1}^{K_c}\sum_{n=1}^{K_d} D_k^{(a)} D_\ell^{(b)} D_m^{(c)} D_n^{(d)} \big[\mathbf{a}_k\mathbf{b}_\ell|\mathbf{c}_m\mathbf{d}_n\big]$$

We refer to the set of all integrals $\big(\mathbf{ab}|\mathbf{cd}\big)$ formed from a quartet of shells as a *class*. For example, a class consisting of a quartet of d, s and two p functions would have $6 \cdot 1 \cdot 3 \cdot 3 = 54$ unique ERIs.

Recalling Eq. (2.30), we rewrite the pairs of one-electron functions as

$$\big[\mathbf{ab}\big| = \exp\big[-\alpha(\mathbf{r}_1 - \mathbf{A})^2 - \beta(\mathbf{r}_1 - \mathbf{B})^2\big] \prod_{i=x,y,z} (i_1 - A_i)^{a_i} (i_1 - B_i)^{b_i}$$

and

$$\big|\mathbf{cd}\big] = \exp\big[-\gamma(\mathbf{r}_1 - \mathbf{C})^2 - \delta(\mathbf{r}_1 - \mathbf{D})^2\big] \prod_{i=x,y,z} (i_1 - C_i)^{c_i} (i_1 - D_i)^{d_i}$$

to make the following connection. We now present a notation for a general one-electron function which can describe the pairs just given and their derivatives with respect to the nuclear coordinates. The notation makes mnemonic reference to the usual quantum bra and ket notation by framing the integrals of interest as inner

products with the kernel r_{12}^{-1}. That is

$$\big[\mathbf{ab}|\mathbf{cd}\big] = \int\int d\mathbf{r}_1\, d\mathbf{r}_2 \big[\mathbf{ab}\big|r_{12}^{-1}\big|\mathbf{cd}\big]$$

With this in mind we denote a *primitive bra* as

$$\left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{array}\right| = \frac{(2\alpha)^{a'}(2\beta)^{b'}}{(2\zeta)^{p'}} \frac{\partial^{\bar{a}+\bar{b}}}{\partial A_x^{\bar{a}_x}\partial A_y^{\bar{a}_y}\partial A_z^{\bar{a}_z}\partial B_x^{\bar{b}_x}\partial B_y^{\bar{b}_y}\partial B_z^{\bar{b}_z}} \left|\begin{array}{ccc} \mathbf{0} & \mathbf{0} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ 0 & 0 & 0 \end{array}\right]$$

where

$$\left[\begin{array}{ccc|} \mathbf{0} & \mathbf{0} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ 0 & 0 & 0 \end{array}\right| = \exp[-\alpha(\mathbf{r}-\mathbf{A})^2 - \beta(\mathbf{r}-B)^2] \prod_{i=x,y,z} (i-A_i)^{a_i}(i-B_i)^{b_i}\zeta^{p_i/2} H_{p_i}\big[\zeta^{1/2}(i-P_i)\big]$$

Here H_n indicates a Hermite polynomial, $\zeta = \alpha + \beta$, and $\mathbf{P}$ is the product center $(\alpha\mathbf{A} + \beta\mathbf{B})/\zeta$. Similarly a *primitive ket* is written as

$$\left|\begin{array}{ccc} \bar{\mathbf{c}} & \bar{\mathbf{d}} & \\ \mathbf{c} & \mathbf{d} & \mathbf{q} \\ c' & d' & q' \end{array}\right] = \frac{(2\gamma)^{c'}(2\delta)^{d'}}{(2\eta)^{q'}} \frac{\partial^{\bar{c}+\bar{d}}}{\partial C_x^{\bar{c}_x}\partial C_y^{\bar{c}_y}\partial C_z^{\bar{c}_z}\partial D_x^{\bar{d}_x}\partial D_y^{\bar{d}_y}\partial D_z^{\bar{d}_z}} \left|\begin{array}{ccc} \mathbf{0} & \mathbf{0} & \\ \mathbf{c} & \mathbf{d} & \mathbf{q} \\ 0 & 0 & 0 \end{array}\right]$$

where

$$\left|\begin{array}{ccc} \mathbf{0} & \mathbf{0} & \\ \mathbf{c} & \mathbf{d} & \mathbf{q} \\ 0 & 0 & 0 \end{array}\right] = \exp[-\gamma(\mathbf{r}-\mathbf{C})^2 - \delta(\mathbf{r}-D)^2] \prod_{i=x,y,z} (i-C_i)^{c_i}(i-D_i)^{d_i}\eta^{q_i/2} H_{q_i}\big[\eta^{1/2}(i-Q_i)\big]$$

with $\eta = \gamma + \delta$ and $\mathbf{Q} = (\gamma\mathbf{C} + \delta\mathbf{D})/\eta$. From this, recalling the definition of an inner product, we can make the immediate association of a *primitive braket* as

$$\left[\begin{array}{ccc|ccc} \bar{\mathbf{a}} & \bar{\mathbf{b}} & & \bar{\mathbf{c}} & \bar{\mathbf{d}} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} & \mathbf{c} & \mathbf{d} & \mathbf{q} \\ a' & b' & p' & c' & d' & q' \end{array}\right] = \int\int d\mathbf{r}_1\, d\mathbf{r}_2 \left[\begin{array}{ccc|c|ccc} \bar{\mathbf{a}} & \bar{\mathbf{b}} & & & \bar{\mathbf{c}} & \bar{\mathbf{d}} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} & r_{12}^{-1} & \mathbf{c} & \mathbf{d} & \mathbf{q} \\ a' & b' & p' & & c' & d' & q' \end{array}\right]$$

This notation may appear complicated, and indeed a great deal of information is summarized using it. However, if one keeps in mind that the first row refers to derivatives, the second row to the combination of Gaussian and Hermite functions and the third row describes a scaling factor, it becomes relatively easy to understand the basics of equations with this notation at a glance. The generalization of the one-electron functions to include Hermite polynomials makes possible the recurrence relations that follow.

It is straightforward to identify a *contracted bra* as

$$\left(\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} & \\ a' & b' & p' & \end{array}\right| = \sum_{k=1}^{K_a}\sum_{\ell=1}^{K_b} D_k^{(a)} D_\ell^{(b)} \left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} & \\ a' & b' & p' & \end{array}\right|$$

and, correspondingly, a *contracted ket* as

$$\left|\begin{array}{ccc} \bar{\mathbf{c}} & \bar{\mathbf{d}} & \\ \mathbf{c} & \mathbf{d} & \mathbf{q} \\ c' & d' & q' \end{array}\right) = \sum_{m=1}^{K_c}\sum_{n=1}^{K_d} D_m^{(c)} D_n^{(d)} \left|\begin{array}{ccc} \bar{\mathbf{c}} & \bar{\mathbf{d}} & \\ \mathbf{c} & \mathbf{d} & \mathbf{q} \\ c' & d' & q' \end{array}\right]$$

and, finally, a *contracted braket* as

$$\left(\begin{array}{ccc|ccc} \bar{\mathbf{a}} & \bar{\mathbf{b}} & & \bar{\mathbf{c}} & \bar{\mathbf{d}} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} & \mathbf{c} & \mathbf{d} & \mathbf{q} \\ a' & b' & p' & c' & d' & q' \end{array}\right) = \sum_{k=1}^{K_a}\sum_{\ell=1}^{K_b}\sum_{m=1}^{K_c}\sum_{n=1}^{K_d} D_k^{(a)} D_\ell^{(b)} D_m^{(c)} D_n^{(d)} \left[\begin{array}{ccc|ccc} \bar{\mathbf{a}} & \bar{\mathbf{b}} & & \bar{\mathbf{c}} & \bar{\mathbf{d}} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} & \mathbf{c} & \mathbf{d} & \mathbf{q} \\ a' & b' & p' & c' & d' & q' \end{array}\right]$$

In this notation, any contracted ERI or derivative equals a contracted braket with $a' = b' = p' = c' = d' = q' = 0$ and $\mathbf{p} = \mathbf{q} = 0$.

2.11.3 Recurrence Relationships

The notation of the previous section allows us to write complicated relations between bras in a compact fashion. In this section, through the use of a few elementary identities, we will derive a set of recurrence relationships. We will use the following, starting with Leibnitz' rule for the nth derivative of a product.

$$\frac{\partial^{n_A+n_B}}{\partial A^{n_A}\partial B^{n_B}}\big[(A-B)f(A,B)\big] = (A-B)\frac{\partial^{n_A+n_B} f}{\partial A^{n_A}\partial B^{n_B}} + n_A\frac{\partial^{n_A+n_B-1} f}{\partial A^{n_A-1}\partial B^{n_B}} - n_B\frac{\partial^{n_A+n_B-1} f}{\partial A^{n_A}\partial B^{n_B-1}} \quad (2.66a)$$

$$(x - B) = (x - A) + (A - B) \tag{2.66b}$$

$$(\mathbf{P} - \mathbf{A}) = (2\beta/2\zeta)(\mathbf{B} - \mathbf{A}) \tag{2.66c}$$

$$xH_n(x) = nH_{n-1}(x) + H_{n+1}(x)/2 \tag{2.66d}$$

$$\frac{dH_n(x)}{dx} = 2nH_{n-1}(x) \tag{2.66e}$$

Applying Eq. (2.66b) to Eq. (2.11.2) gives

$$\begin{aligned}\begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{bmatrix} &= (i - A_i) \begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a} & \mathbf{b}-\mathbf{1}_i & \mathbf{p} \\ a' & b' & p' \end{bmatrix} + (A_i - B_i) \begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a} & \mathbf{b}-\mathbf{1}_i & \mathbf{p} \\ a' & b' & p' \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a}+\mathbf{1}_i & \mathbf{b}-\mathbf{1}_i & \mathbf{p} \\ a' & b' & p' \end{bmatrix} + (A_i - B_i) \begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a} & \mathbf{b}-\mathbf{1}_i & \mathbf{p} \\ a' & b' & p' \end{bmatrix}\end{aligned} \tag{2.67}$$

Applying Eq. (2.66b), then Eqs. (2.66d) and (2.66c), respectively, yields

$$\begin{aligned}&\begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{bmatrix} = (i - P_i) \begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a}-\mathbf{1}_i & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{bmatrix} + (P_i - A_i) \begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a}-\mathbf{1}_i & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{bmatrix} \\ &= p_i \begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a}-\mathbf{1}_i & \mathbf{b} & \mathbf{p}-\mathbf{1}_i \\ a' & b' & p' \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a}-\mathbf{1}_i & \mathbf{b} & \mathbf{p}+\mathbf{1}_i \\ a' & b' & p'+1 \end{bmatrix} + (B_i - A_i) \begin{bmatrix} \mathbf{0} & \mathbf{0} & \\ \mathbf{a}-\mathbf{1}_i & \mathbf{b} & \mathbf{p} \\ a' & b'+1 & p'+1 \end{bmatrix}\end{aligned} \tag{2.68}$$

The $(\bar{\mathbf{a}} \rightarrow \mathbf{a})$ *Recurrence Relation*

Differentiating a bra once with respect to A_i and using Eq. (2.66e), we obtain

$$\left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{array}\right| = \left[\begin{array}{ccc|} \bar{\mathbf{a}}{-}\mathbf{1}_i & \bar{\mathbf{b}} & \\ \mathbf{a}{+}\mathbf{1}_i & \mathbf{b} & \mathbf{p} \\ a'{+}1 & b' & p' \end{array}\right| - a_i \left[\begin{array}{ccc|} \bar{\mathbf{a}}{-}\mathbf{1}_i & \bar{\mathbf{b}} & \\ \mathbf{a}{-}\mathbf{1}_i & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{array}\right| - p_i \left[\begin{array}{ccc|} \bar{\mathbf{a}}{-}\mathbf{1}_i & \bar{\mathbf{b}} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p}{-}\mathbf{1}_i \\ a'{+}1 & b' & p' \end{array}\right| \tag{2.69}$$

This equation relates the derivative of an integral to integrals of higher and lower angular momenta. It has origins in part in the identity shown in Eq. (2.31).

The $(\mathbf{b} \rightarrow \mathbf{a})$ *Recurrence Relation*

Applying Leibnitz' rule Eq. (2.66a) to differentiate Eq. (2.67) we have

$$\left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{array}\right| = \left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & \\ \mathbf{a}{+}\mathbf{1}_i & \mathbf{b}{-}\mathbf{1}_i & \mathbf{p} \\ a' & b' & p' \end{array}\right| + (A_i - B_i) \left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & \\ \mathbf{a} & \mathbf{b}{-}\mathbf{1}_i & \mathbf{p} \\ a' & b' & p' \end{array}\right| + \bar{a_i} \left[\begin{array}{ccc|} \bar{\mathbf{a}}{-}\mathbf{1}_i & \bar{\mathbf{b}} & \\ \mathbf{a} & \mathbf{b}{-}\mathbf{1}_i & \mathbf{p} \\ a' & b' & p' \end{array}\right| - \bar{b_i} \left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}}{-}\mathbf{1}_i & \\ \mathbf{a} & \mathbf{b}{-}\mathbf{1}_i & \mathbf{p} \\ a' & b' & p' \end{array}\right| \tag{2.70}$$

Note the common feature on the right hand side is the reduction of the angular momentum $\mathbf{b}$ by one in every term.

The $(\mathbf{a} \rightarrow \mathbf{p})$ *Recurrence Relation*

Applying Eq. (2.66a) to differentiate Eq. (2.68) gives

$$\begin{aligned}
\left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{array}\right| &= p_i \left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & \\ \mathbf{a}{-}\mathbf{1}_i & \mathbf{b} & \mathbf{p}{-}\mathbf{1}_i \\ a' & b' & p' \end{array}\right| \\
&+ \left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & \\ \mathbf{a}{-}\mathbf{1}_i & \mathbf{b} & \mathbf{p}{+}\mathbf{1}_i \\ a' & b' & p'{+}1 \end{array}\right| + (B_i - A_i) \left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}} & \\ \mathbf{a} & \mathbf{b}{-}\mathbf{1}_i & \mathbf{p} \\ a' & b'{+}1 & p'{+}1 \end{array}\right| \\
&- \bar{a}_i \left[\begin{array}{ccc|} \bar{\mathbf{a}}{-}\mathbf{1}_i & \bar{\mathbf{b}} & \\ \mathbf{a}{-}\mathbf{1}_i & \mathbf{b} & \mathbf{p} \\ a' & b'{+}1 & p'{+}1 \end{array}\right| + \bar{b}_i \left[\begin{array}{ccc|} \bar{\mathbf{a}} & \bar{\mathbf{b}}{-}\mathbf{1}_i & \\ \mathbf{a}{-}\mathbf{1}_i & \mathbf{b} & \mathbf{p} \\ a' & b'{+}1 & p'{+}1 \end{array}\right| \qquad (2.71)
\end{aligned}$$

Here the angular momentum $\mathbf{a}$ is reduced in every term on the right hand side. The Gaussian coefficients are reduced in order in trade for increasing the order of the Hermite polynomials.

The recurrence relations above were derived for uncontracted primitive bras. The resulting coefficients are, in all cases, independent of any primitive quartet specific variables (e.g. the exponents α and so on). Therefore, these relations hold for contracted bras without modification. Gill *et al.* derived two more recurrence relationships. Our application only requires ERIs and their first derivatives, for which the equations given suffice. Following the sequence in which they were

given, we see that Eq. (2.69) can be iteratively applied to write any bra in terms of bras with $\bar{\mathbf{a}} = \bar{\mathbf{b}} = 0$, i.e. all derivative terms can be expressed as linear combinations of integral only terms. Then Eq. (2.70) can be iteratively applied to write such terms as combinations where $\mathbf{b} = 0$. Finally Eq. (2.71) can reduce these terms to combinations with $\mathbf{a} = 0$.

2.11.4 Construction of pq-brakets

As was shown in the last section, using equations 2.70 through 2.69 and the corresponding relationships for contracted kets, any ERI or first derivative on an ERI can be systematically reduced to a combination of *contracted pq-brakets*, i.e. brakets of the form

$$\left(\begin{array}{ccc|ccc} \mathbf{0} & \mathbf{0} & & \mathbf{0} & \mathbf{0} & \\ \mathbf{0} & \mathbf{0} & \mathbf{p} & \mathbf{0} & \mathbf{0} & \mathbf{q} \\ a' & b' & p' & c' & d' & q' \end{array}\right)$$

The original integrals over quartets of Gaussians have been transformed to integrals over scaled Hermite polynomials and simple s-type Gaussians. Writing a pq-braket as

$$\left(\begin{array}{ccc|ccc} \mathbf{0} & \mathbf{0} & & \mathbf{0} & \mathbf{0} & \\ \mathbf{0} & \mathbf{0} & \mathbf{p} & \mathbf{0} & \mathbf{0} & \mathbf{q} \\ a' & b' & p' & c' & d' & q' \end{array}\right) = (-1)^{q}{}_{a'b'p'}(\mathbf{p}+\mathbf{q})_{c'd'q'}$$

Gill *et al.* and McMurchie and Davidson[37] identify the right hand side of the previous equations with auxiliary integrals $[\mathbf{r}]^{(0)}$, as

$$_{a'b'p'}(\mathbf{p}+\mathbf{q})_{c'd'q'} = \sum_{k=1}^{K_a}\sum_{\ell=1}^{K_b}\sum_{m=1}^{K_c}\sum_{n=1}^{K_d}\frac{(2\alpha)^{a'}(2\beta)^{b'}}{(2\zeta)^{p'-a-b}}\frac{(2\gamma)^{c'}(2\delta)^{d'}}{(2\eta)^{q'-c-d}}[\mathbf{r}]^{(0)} \qquad (2.72)$$

where $\mathbf{r} = (r_x, r_y, r_z)$ is a triplet of integers and a, b, c, and d are the angular momenta of the quartet of the end-product ERI.

We will now show how to construct the auxiliary integrals $[\mathbf{r}]^{(0)}$ from the elementary quantities of the primitive Gaussians. We define the total angular momentum L of the target braket class as

$$L = a + b + c + d + \bar{a} + \bar{b} + \bar{c} + \bar{d}$$

Starting with the following two-center terms

$$\sigma_P = \frac{1}{2(\alpha+\beta)}$$

$$\mathbf{P} = \frac{\alpha\mathbf{A}+\beta\mathbf{B}}{\alpha+\beta}$$

$$U_P = (8\pi^3)^{1/2}\sigma_P^{a+b+3/2}D^{(a)}D^{(b)}\exp\left[-2\alpha\beta\sigma_P(\mathbf{A}-\mathbf{B})^2\right]$$

$$\sigma_Q = \frac{1}{2(\gamma+\delta)}$$

$$\mathbf{Q} = \frac{\gamma\mathbf{C}+\delta\mathbf{D}}{\gamma+\delta}$$

$$U_Q = (8\pi^3)^{1/2}\sigma_Q^{c+d+3/2}D^{(c)}D^{(d)}\exp\left[-2\gamma\delta\sigma_Q(\mathbf{C}-\mathbf{D})^2\right]$$

we can directly calculate these seven elementary four-center quantities:

$$\mathbf{R} = \mathbf{Q} - \mathbf{P}$$

$$\begin{aligned} R^2 &= R_x^2 + R_y^2 + R_z^2 \\ 2\vartheta^2 &= \frac{1}{\sigma_P + \sigma_Q} \\ 2T &= 2\vartheta^2 R^2 \\ U &= U_P U_Q \end{aligned} \tag{2.73}$$

These quantities are used to form the $[\mathbf{0}]^{(m)}$ integrals, $m = 0, \ldots, L$ defined as

$$[\mathbf{0}]^{(m)} = U(2\vartheta^2)^{m+1/2} G_m(T) \tag{2.74}$$

where

$$G_m(T) = (2/\pi)^{1/2} \int_0^1 dt\, t^{2m} \exp(-Tt^2) \tag{2.75}$$

Here it is important to note that for T less than a value T_c $G_m(T)$ can be calculated via standard series expansion methods. For values greater than T_c, we may profitably employ the asymptotic series

$$G_m(T) \asymp (2/\pi)^{1/2} \frac{\Gamma(m+1/2)}{2T^{m+1/2}} - (2/\pi)^{1/2} \frac{\exp(-T)}{2T} \left(1 + \frac{m-1/2}{T}(1 + \ldots)\right)$$

The series is formally divergent, but the deviation from the true value of the integral is less than the magnitude of the last term included. The series is particularly useful when all but the first term can be neglected. Requiring the ratio of the error and integral estimates be less than a tolerance ϵ leads to the transcendental

equation

$$T^{m-1/2}\exp(-T) = \epsilon\Gamma(m+1/2)$$

which may be solved to determine T_c for a given m. Going back to the definition of T, it becomes apparent that this limit corresponds to the distance at which the charge clouds of electrons 1 and 2 no longer overlap appreciably. The integrals $[\mathbf{0}]^{(m)}$ can then be calculated by the approximate formula

$$[\mathbf{0}]^{(m)} = \frac{U}{R}\prod_{i=1}^{m}\left(\frac{2i-1}{R^2}\right) \tag{2.76}$$

which Gill *et al.* call the classical regime.

Returning to the auxiliary integrals, we make use of the McMurchie-Davidson (MD) recurrence relation[37] whereby

$$[\mathbf{r}]^{(m)} = R_i[\mathbf{r}-\mathbf{1}_i]^{(m+1)} - (r_i-1)[\mathbf{r}-\mathbf{2}_i]^{(m+1)} \tag{2.77}$$

With this identification, we have a complete sequence for formulating any contracted ERI and derivatives of the same. We give a simple example next, after which we shall consider the modifications that can be made for the particular special cases of integrals of interest.

Example – Calculation of a $(p_x s|ss)$ Integral

To illustrate the Gill *et al.* notation and use of the recurrence relationships, we examine the application of the equations above to the (relatively simple) case of a $(p_x s|ss)$ ERI. This would be one of three integrals making up a complete $(ps|ss)$

class. Moving to the braket notation, we have

$$(p_x s|ss) = \left(\begin{array}{lll|ll} \mathbf{0} & & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ (1,0,0) & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & & 0 & 0 & 0 & 0 \end{array}\right)$$

where we have explicitly written out the $\mathbf{a}$ element in the bra rather than using the vector shorthand. Applying the $(\mathbf{a} \rightarrow \mathbf{p})$ relation gives

$$\left(\begin{array}{lll|ll} \mathbf{0} & & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ (1,0,0) & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & & 0 & 0 & 0 & 0 \end{array}\right)$$

$$= \left(\begin{array}{lll|ll} \mathbf{0} & \mathbf{0} & & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & (1,0,0) & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right) + (B_x - A_x) \left(\begin{array}{lll|ll} \mathbf{0} & \mathbf{0} & & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & 1 & 1 & 0 & 0 & 0 \end{array}\right)$$

$$= {}_{001}(1,0,0)_{000} + (B_x - A_x){}_{011}(0,0,0)_{000}$$

where several terms from Eq. (2.71) have dropped out because their coefficients are 0. Recalling Eq. (2.72) we see that

$${}_{001}(1,0,0)_{000} = \sum_{k=1}^{K_a} \sum_{\ell=1}^{K_b} \sum_{m=1}^{K_c} \sum_{n=1}^{K_d} [1,0,0]^{(0)}$$

and

$${}_{011}(0,0,0)_{000} = \sum_{k=1}^{K_a} \sum_{\ell=1}^{K_b} \sum_{m=1}^{K_c} \sum_{n=1}^{K_d} 2\beta_\ell [0,0,0]^{(0)}$$

where the second equation already has an integral reduced to the $[\mathbf{0}]^{(m)}$ form. Application of the MD relation to the $[\mathbf{1}_x]^{(0)}$ term produces

$$[1,0,0]^{(0)} = R_x[\mathbf{0}]^{(1)}$$

This is, of course, the reverse of the path actually used to calculate the target ERI.

2.11.5 Simplifications and Special Cases

The example calculation demonstrates that the actual application of the braket recurrence relationships often simplifies because some terms are eliminated since they have a coefficient equal to zero. In our application, we are only interested in ERIs and gradient terms, so we never have a derivative of order greater than one. We use Eq. (2.69) to write derivative terms as a combination of undifferentiated ERIs for which $\bar{\mathbf{a}} = \bar{\mathbf{b}} = \mathbf{0}$ always. Examining the other two recurrence equations, we see that in both this means two terms are eliminated on the right hand sides from the more general form.

With the exception of integrals of the form $(ab|a'b')$, all the correction terms used have at least one paired set of basis functions that share a common center. For any such bra or ket, coefficients that depend on the difference between coordinates of the functions centers reduce to zero, eliminating more terms. Explicitly, then, for the special case of an undifferentiated coincident center, the contracted versions of Eqs. (2.70) and (2.71) simplify to

$$\left(\begin{array}{ccc|} \mathbf{0} & \mathbf{0} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{array}\right| = \left(\begin{array}{ccc|} \mathbf{0} & \mathbf{0} & \\ \mathbf{a}+\mathbf{1}_i & \mathbf{b}-\mathbf{1}_i & \mathbf{p} \\ a' & b' & p' \end{array}\right|$$

and

$$\left(\begin{array}{ccc|} \mathbf{0} & \mathbf{0} & \\ \mathbf{a} & \mathbf{b} & \mathbf{p} \\ a' & b' & p' \end{array}\right| = p_i \left(\begin{array}{ccc|} \mathbf{0} & \mathbf{0} & \\ \mathbf{a}-\mathbf{1}_i & \mathbf{b} & \mathbf{p}-\mathbf{1}_i \\ a' & b' & p' \end{array}\right| + \left(\begin{array}{ccc|} \mathbf{0} & \mathbf{0} & \\ \mathbf{a}-\mathbf{1}_i & \mathbf{b} & \mathbf{p}+\mathbf{1}_i \\ a' & b' & p'+1 \end{array}\right|$$

respectively.

Finally, a more subtle but powerful optimization can be applied to integrals of the form $(aa'|bc)$ in the limit of the classical regime. Recall the four-center quantity $R = |\mathbf{P} - \mathbf{Q}|$ shown earlier. When a paired set of basis functions have a common center, the product center becomes independent of the specific primitive pair quantities and subsequently R only depends on the bc pair quantities and the center A. The MD recurrence also only multiplies terms by factors of the coordinate differences R_i. Examining Eqs. (2.76) and (2.77) shows that all the auxiliary $[\mathbf{r}]^{(m)}$ functions have the form

$$[\mathbf{r}]^{(m)} = U_{aa'} U_{bc} f(\mathbf{R}_{Abc})$$

where we use the label Abc as a reminder of the dependencies of R, and by extension of the function f. Factoring Eq. (2.72) as

$${}_{a'b'p'}(\mathbf{r})_{c'd'q'} = \left[\sum_{k=1}^{K_a}\sum_{\ell=1}^{K_b} \frac{(2\alpha)^{a'}(2\beta)^{b'}}{(2\zeta)^{p'-a-b}} U_{aa'}\right]\left[\sum_{m=1}^{K_c}\sum_{n=1}^{K_d} \frac{(2\gamma)^{c'}(2\delta)^{d'}}{(2\eta)^{q'-c-d}} U_{bc} f(\mathbf{R}_{Abc})\right]$$

it becomes apparent that the sums can be performed independently. Therefore, in the limit $T > T_c$ of the classical regime, we can form the second sum once for any bc pair and center A, then form the first sum for every function pair aa' centered

at A and multiply to obtain the requisite contracted pq-brakets. We define the coefficient

$$\bar{U}_{a'b'p'} = \sum_{k=1}^{K_a} \sum_{\ell=1}^{K_b} \frac{(2\alpha)^{a'}(2\beta)^{b'}}{(2\zeta)^{p'-a-b}} U_{aa'}$$

for future reference.

2.11.6 Two Electron Corrections – Computer Implementation

In this section we describe the computer code we implemented based on the GHGP relationships. Because our hybrid method requires only certain special subsets of all integrals, we chose to implement four different sets of subroutines, each taking advantage of the special properties of a particular case to optimize the computations. Although the actual computation of an integral from the proper elementary quantities is numerically expensive, great care must be taken in optimizing the drivers that select and prepare integrals for calculation. A more general approach would be easier to implement. However, our approach uses memory efficiently, and creates large batches of integrals of the same shell type. Specialized hardware can often obtain significant speed increases by overlapping operations in long inner loops, so the batch size can be important for that reason (and the code must be written in such a way that a compiler can identify potential optimizations). It also avoids spending too much time moving up and down through layers of subroutines.

The uppermost driver routine is outlined in Figure 2.2. This driver mostly calls other routines based on the correction type flags set for the particular iteration. We have omitted much of the detail here and in the other code outlines in order to more clearly show the important structure.

```
For each angular momentum
  call CALCTCUT to form T cutoffs
Next angular momentum

call ATPR to form atom pair data
call MKDIST to form atom distance table
call SETAPCUT to form pair cutoff
call LOADUP to initialize density and storage arrays

call AAAA to form (aa'|a''a''') integrals

For each Fock matrix strip
  call AABB to form (aa'|bb') integrals
  call AABC to form (aa'|bc) integrals
  call ABAB to form (ab|a'b') integrals
Next strip
```

Figure 2.2: Schematic representation of subroutine **GHGPINT**

Several stages in these integral routines make use of values derived from function pairs in which both functions are centered on a single atom. Since the data only depends on atom type and requires little storage space, we calculate it prior to calling the main integral routines. Subroutine **ATPR** calculates the values of $U_{aa'}$ and $\sigma_{aa'}$ for every primitive contraction on each atom type, along with the set of all $\bar{U}_{a'b'p'}$ which might be needed for classical integrals. The $U_{aa'}$ and $\sigma_{aa'}$ are stored in arrays indexed by three values; contraction number within a pair, pair number on an atom type, and atom type. For a particular pair number and atom type, the values are stored in order sorted by increasing $\sigma_{aa'}$ (and therefore by decreasing values of T when matched with a particular bc pair). $\bar{U}_{a'b'p'}$ values are just sums of the $U_{aa'}$ values needed in cases where the classical approximation holds. With the $U_{aa'}$ terms sorted, a $\bar{U}_{a'b'p'}$ value is generated for the case where

all the contracted primitives for a aa' pair can be done classically, the case where all but the contraction with the largest $\sigma_{aa'}$ can be done classically, and so on down to the case where only the contraction with the smallest $\sigma_{aa'}$ for the pair can be done classically.

SETAPCUT creates an atomic pair distance cutoff array used both in the $(aa'|bc)$ case and the $(ab|a'b')$ case. This cutoff forms a crude estimate of the magnitude of an integral by assuming all terms in Eq. (2.74) are of order unity except the pair coefficients U_P and U_Q and by factoring in the scaling from Eq. (2.72). Multiplying by the largest scaling factors for an integral, requiring that, for example, U_P be greater than a minimum value can be rearranged in the form of a cutoff for the pair distance $|\mathbf{A} - \mathbf{B}|$ where the cutoff value depends on other pair quantities. This cutoff must therefore be calculated for all types of function pairs, then used to throw out inconsequential primitive pairs based on distance. We found empirically that this cutoff could be used to noticeably reduce the computational effort without loss of accuracy.

Special Cases – $(aa'|a''a''')$, $(aa'|bb')$

The algorithms for $(aa'|a''a''')$ and $(aa'|bb')$ are nearly identical. Other than a deeply nested set of loops, they are also quite simple. Most everything necessary has already been arranged in **ATPR**. The structure of the loops allows batches of integrals to be automatically formed by indexing over braket angular momenta, contraction degrees and atom types. No cutoffs are used. The classical approximation for the auxiliary integrals could be used, but the number of integrals is relatively small and there would be a trade off of extra complexity and decreased batch sizes.

Special Case – $(aa'|bc)$

The special case in which one function pair shares a center, the $(aa'|bc)$ case, is probably the most complex. This case accounts for the vast majority of the analytic integrals needed for larger molecules, scaling formally in number as N^3. We devised a scheme that rapidly partitions brakets according to the number of bra and ket primitives that can be treated classically. The scheme uses two cutoffs rather than one, producing a more restrictive criteria for the partitioning than a strict implementation of the GHGP approach, but one that avoids the prohibitive cost of forming an individual cutoff parameter for every primitive braket.

Recall that an integral may be computed using the classical approximation given in Eq. (2.76) whenever the argument T of the function G_m in Eq. (2.75) exceeds some control parameter T_c. Using the definition of T from Eq. (2.73), we may write this condition as

$$\vartheta^2 R^2 > T_c$$

Substituting from Eq. (2.11.4) (and using $\sigma_{aa'}$ and σ_{bc} in place of σ_P and σ_Q respectively), this becomes

$$R^2 > 2(\sigma_{aa'} + \sigma_{bc})T_c$$

R only depends on the bc primitive pair and the center A, but this condition would require a cutoff for every quartet of primitive combinations in order to apply it strictly. Instead, assume, for the sake of illustration, that $\sigma_{aa'} > \sigma_{bc}$. Then it becomes immediately apparent that

$$4\sigma_{aa'}T_c > 2(\sigma_{aa'} + \sigma_{bc})T_c > 4\sigma_{bc}T_c \tag{2.78}$$

Therefore, rather than forming individual cutoffs for all the aa' and bc primitive combinations, an effort that scales as the product of the number of primitives pairs, we instead use a double check based on the leftmost and rightmost terms in the previous inequality, at least one of which is guaranteed to be more restrictive than the precise cutoff of Eq. (2.11.6). As we will show in the following discussion of the computer code, this method scales only as the sum of the number of primitives pairs.

The algorithm is structured using a "cascading bucket" model. At each stage, groups of similar quantities are accumulated until the memory "bucket" allotted fills. That bucket then gets passed to the next stage for processing.

```
For each ket angular momentum
  call RPOINT
  For each atom B
    For each function b on atom B
      For each atom C in strip
        For each function c on atom C
          check pseudooverlap, neighbor matrix
              and correction type
            For each primitive of function b
            For each primitive of function c
              check distance cutoff
            Next function c primitive
            Next function b primitive
            store function pair information indexed
                by contraction degree
            if max storage reached call PROCBC
        Next function c
      Next atom C
    Next function b
  Next atom B
Next ket angular momentum
```

Figure 2.3: Schematic representation of subroutine **AABC**

Figure 2.3 shows the uppermost driver for the $(aa'|bc)$ integrals. **AABC** cycles through all the possible ket angular momenta. With the momenta of the two ket functions fixed, and hence the total momentum of the ket, we call subroutine **RPOINT**, described in detail later, to create a sorted table of the $\sigma_{aa'}$ distance cutoffs for use in determining the classical/non-classical partitionings. The cutoffs depend on the total angular momentum of the braket, so they need to be recomputed for each ket type. Following the construction of the cutoff table, **AABC** selects pairs of basis functions on different centers with the correct momenta. We use the distance cutoff from **SETAPCUT** to eliminate small primitive contractions, resulting in an actual contraction degree for the function pair that is often less than the formal degree. Having determined the actual number of contracted primitives to use, we store the atom and function indexes in a bucket labeled by contraction degree. Once a maximum number of bc pairs with the same contraction degree have been accumulated, we call **PROCBC** to process them.

```
For each atom type
  For each pair of functions on atom type
    For each primitive contraction of the function pair
      store pair/contraction info, σ_aa' distance cutoff
    Next contraction
  Next function pair

  sort information in order of increasing distance cutoff
Next atom type
```

Figure 2.4: Schematic representation of subroutine **RPOINT**

RPOINT is a short routine that builds two arrays, as outlined in Figure 2.4. These arrays contain information about aa' function pairs that will later be matched with bc pairs. R2CUT is an array of distance cutoffs based on $\sigma_{aa'}$, sorted in in-

creasing order. ITOPS stores the function pair index and the contraction number within a function pair that correspond to each R2CUT entry. The arrays are indexed by total primitive number (out of all the contracted primitives on an atom type) and atom type. **RPOINT** loops over all atom types, and each function pair on every atom type, generating the data for all aa' pairs at once. The value of T_c used for the classical distance cutoff depends on the total angular momentum of the integral, and so these tables must be recomputed for each ket type, as discussed previously in the description of **AABC**.

```
For each bc function pair for batch
  For each primitives on atom b
  For each primitives on atom c
    store contraction info
    store classical cutoff information based on σ_bc
  Next atom c
  Next atom b
Next bc function pair

For each atom a
  For each bc function pair for batch
    classify correction type
    For each primitive contractions
      calculate distance from a atom to bc product center
      check σ_bc based classical distance cutoff
        count pair as (possibly) classical
        save minimum a-bc classical distance
    Next primitive contractions

    store a-bc info based on type and
        number of pairs marked as classical
    call PROCABC for batch
  Next bc function pair
Next atom a
```

Figure 2.5: Schematic representation of subroutine **PROCBC**

PROCBC, shown in Figure 2.5, gets called with batches of bc pairs of the same type. The routine starts out by first rechecking the primitive contraction cutoff to determine which contractions get kept for each function pair, but this time keeping track of σ_{bc}, U_{bc}, the product centers R_{bc} and a distance cutoff based on σ_{bc}. We chose not to store this information previously in routine **AABC** because to do so would have required setting aside extra storage space for each type of ket while we accumulated batches. Instead, once **AABC** has a full batch of one type, we can generate the information for that one batch at the cost of redoing the primitive contraction checks.

Next a set of loops over atoms a and pairs bc performs the first check based on the inequalities Eq. (2.78), based on σ_{bc}. This set of loops counts the number of primitive pairs that can be treated classically based on this estimate. The minimum a-bc center distance is saved for use in the next subroutine. Indices are kept organized by the number of contractions to be treated classically. Once the bucket size for a particular classical partitioning has filled, **PROCABC** is called for that batch.

The code enters **PROCABC** with a set of cutoffs ordered by $\sigma_{aa'}$ and a certain classical/non-classical partitioning of the a-bc triples. The triples are sorted according to the minimum a-bc distance found in **PROCBC**. Starting at the beginning of the sorted array of $\sigma_{aa'}$ based cutoffs, we step through the distance array checking the distance cutoff until the classical approximation is no longer valid for that aa' primitive. An index is stored to mark that segment of the batch triples, then we move to the next larger $\sigma_{aa'}$ and repeat the process. Thus for a set of aa' and bc primitives, we have a partitioning indicating which aa' pairs can be treated classically with the premarked classical bc pairs. Some situations where

```
sort a-bc triple info based on minimum a-bc distance

For each a-bc triples in batch
  check σ_aa' based classical distance cutoff
    set pointer for last a-bc triple to be done with
        a particular aa' classical degree
Next a-bc triples

call GENCLDAT

For each aa' function pairs
  For each primitive contractions of function pair
    call MKAABC
  Next primitives
Next aa'
```

Figure 2.6: Schematic representation of driver subroutine **PROCABC**

Eq. (2.76) could apply will be missed, but this approach allows efficient batch processing with less overhead.

It is easy to come up with pathological cases in which this algorithm will do a poor job of identifying terms for which the classical approximation holds. Conceivably this could cause the method to be considerably slower than one that doesn't attempt to use that approximation at all. Experience has shown that in actual use the performance is sufficient to not only provide a proof of concept of the whole TEC approach, but to use in "production" code.

Special Case – $(ab|a'b')$

In this section we discuss another sort of two-center integral, the $(ab|a'b')$ special case. They require a significantly different approach than that used for the other integral types. In the other cases, bras and kets can be matched with little depen-

dence on eachother. Here, of course, the basis function pairs must center on the same atom pair.

```
For each atom A
  For each atom B in strip
    check simple A-B distance cutoff
    store atom pair data by pair class
  Next atom B
Next atom A

For each atom pair class
  For each pair
    sort pairs by distance
  Next pair
Next atom pair class

Estimate memory usage
Determine work array layout
Determine maximum pair block size

For each atom pair class
  For each bra block type
    call GETAB
    call SORTAB
    call CALCAB
    For each ket block type
      call MKABAB
    Next ket block type
  Next bra block type
Next atom pair class
```

Figure 2.7: Schematic representation of **ABAB**

In order to make optimal use of memory, the **ABAB** driver uses only a few fixed size arrays. Information needed to calculate pair data values is placed in a single, large, general-purpose array. Since the layouts in this memory region are

not pre-set, data can be stored in the least amount of space. Although the size of the pair data can not be determined exactly before some pair information processing occurs, certain run-time parameters can be used to estimate it. In this way, the program can automatically and adjustably determine the appropriate number of atom pairs to operate on in each cycle. Later, when the actual size required is known, as a by-product of the way it is calculated, the pair data ends up confined to a single, contiguous, minimal portion of the scratch array. The rest can then be used for batch processing.

The pair data is calculated in three steps, information gathering, sorting, then computation. This implementation was chosen for two reasons other than the memory usage considerations given above. The so-called information gathering step is a necessity. This is where function pairs are selected and their actual contraction degree determined. The pair data could be computed here, ending the process, but the loop lengths are short, making it inefficient to do so. Instead, integer indices are stored which allow the requisite function information to be retrieved later. These indices are then sorted. The sorting method allows the use of a simpler, cheaper (in the sense of memory use) indexing scheme in building quartet batches. It also places the pair information in a single, contiguous block of memory. The flop intensive computation stage can then be performed in one long loop over all the contraction pairs at once.

A schematic outline of the driver is given in Figure 2.7. Atoms are assigned classes according to atomic number, and the number in each class is recorded. Atoms in the same class will have similar basis sets (e.g. oxygen and carbon versus hydrogen). Selecting blocks of atom pairs that come from the same classes improves the likelihood that each pair will contribute members to the same batches.

After pairing up atoms and forming them in to a list arranged by the pair classes, each sub-list is sorted according to interatomic distance. This ensures the lists will fall into blocks with the same contraction degree later when distance cut-offs are applied. The program now performs memory estimates to determine the maximum number of atom pairs to process at once.

The code is now ready to process blocks of pairs. It enters loops over atom pair class, blocks and block types (essentially a loop over bra angular momenta). Routine **GETAB** performs the check to eliminate small primitive contractions (discussed elsewhere in regard to routines **GHGPINT** and **AABC**) and gathers pair index data according to the sort ordering. Again, because of the contraction pair cutoff, the actual pair contraction degree may be less than the product of the contraction degrees of the function pair under consideration. Once routine **GETAB** has finished, the storage needed for the pair data is known. Routine **SORTAB** does a simple sorting of the index arrays into a region past where the pair data will go. Routine **CALCAB** then retrieves the actual pair data and stores it in the beginning region of the scratch array, leaving the rest open for processing batches.

Last comes a loop over ket block types (ket angular momenta). Since the bra and ket pair data is from the same atom pair, the bra pair data can be reused. Batches of function quadruplets are accumulated and passed off for integral assembly through routine **MKABAB**.

2.11.7 Final Transformation – pq-brakets to ERIs

Once the requisite pq-brakets have been constructed, they must be transformed in to the final desired ERIs. This phase of the algorithm is one of the most com-

putationally expensive, especially for integrals over higher angular momentum functions. It is therefore critical that it be implemented efficiently. One should note that the transformation sequence, or path through the recursion relations, for each side (bra or ket) is independent of the other.

```
do 10 i=1,n
straw(i,66)=straw(i,66)+ba(i,1)*straw(i,72)+straw(i,75)
straw(i,67)=straw(i,67)+ba(i,1)*straw(i,73)
straw(i,68)=straw(i,68)+ba(i,1)*straw(i,74)
10 continue

do 20 i=1,n
straw(i,69)=straw(i,69)+ba(i,2)*straw(i,73)+straw(i,75)
straw(i,70)=straw(i,70)+ba(i,2)*straw(i,74)
straw(i,71)=straw(i,71)+ba(i,3)*straw(i,74)+straw(i,75)
20 continue
```

Figure 2.8: Schematic representation of a sample transformation code snippet

Figure 2.8 shows a short example of the code from a typical transformation step. Although one could code a general purpose version of each of the recursion formulas 2.69-2.71, we chose to write special case subroutines for all possible transformations through fd functions. The entire transformation from the base pq-brakets through to the final integrals is handled in a single subroutine. In addition, since significant computational savings can be achieved when the function pair shares a center, we broke the cases into one and two center versions. This approach requires perhaps a great deal more coding than a more generalized implementation such as the one described in GHGP. The routines become quite complex for higher angular momenta. Subroutine **fd2c**, the two-center sequence for fd integrals, is roughly 1400 lines long, for example. In contrast, the repeated use

of a few generalized routines simplifies debugging. Routines used for a variety of transformation stages need only be debugged once, and are themselves relatively simple. However, writing individualized routines typically produces code that compilers can optimize much more effectively. In some cases, this can produce enormous speed increases. Our code is particularly well suited for vector or deep-pipelining architectures, although virtually any modern hardware will benefit. The transforms are all performed using only the memory required to hold the input primitives and the final results. A general routine would require substantial extra memory to hold intermediate results, which would significantly reduce the number of integrals that could be performed at once in a batch. Analyzing how to do this was a painstaking process, one for which we found no automated procedure. Even if one could be formulated, it would almost certainly require a control structure that would be difficult to optimize, either by hand or for a compiler.

2.11.8 ERI Gradients

For first derivative terms, it is quite straightforward to use all the control and calculational structures just described. In fact much of the code proceeds without any deviation. First derivative terms, as noted from equation 2.69, are nearly trivial sums of undifferentiated ERIs. The TEC algorithm simply runs calculations for the component ERIs, and then hands the results off to a routine that combines them.

2.12 Results and Discussion

2.12.1 Overview

We present results here for single point Hartree-Fock calculations using the 6-31G** basis set and preliminary cc-pVTZ basis set.[36,38] In general, the advantages of the PS method increase as the basis set becomes more dense (e.g. for a triple zeta basis) and when the shells in the basis set do not share exponents, as is the case in the present basis. Despite the fact that the present basis is the most advantageous DZP basis for standard two electron methods, we show here that significant timing improvements as compared to GAUSSIAN 92 can be obtained with minimal sacrifice of accuracy.

In previous work, we emphasized the agreement of absolute energies from PS-GVB and GAUSSIAN. However, insistence on such agreement for large molecules leads to loss of efficiency in PS-GVB. The only chemically relevant quantities are energy differences; indeed, it should be remembered that energies obtained by GAUSSIAN are themselves off by hundreds of kcals/mole from the exact differential equation solutions, due to basis set incompleteness. While absolute energy agreement is still in general very good (within a few tenths of a kcal/mole for the vast majority of cases), we have succeeded in developing grids, dealiasing sets, and iteration sequences in which the internal cancellation of error in PS-GVB is very reliable despite absolute energy differences with GAUSSIAN of as much as 1 kcal/mole for large molecules.

We have, in addition, developed parameter sets which do reliably reproduce the absolute energies from GAUSSIAN within 0.2 kcal/mole, even for quite large molecules where summation of the long range Coulomb fields to an accuracy of

one part in 10^7 is very demanding. These parameter sets are $\sim$30% slower than the default parameters; however, they allow the user to compare total energies directly with GAUSSIAN if this is desired.

2.12.2 Total Energies

Table A.1 presents total energies for a wide range of molecules in their equilibrium geometries using GAUSSIAN 92 and the default and tight parameter sets for the 6-31G** basis. For the cases shown here, the parameter set described above gives total energies that are very close to the GAUSSIAN results. These results are obtained with grids that are considerably smaller than those used in other numerical methods in electronic structure theory ($\sim$100 points/atom on most iterations as compared with $\sim$1000 points/atom in typical density functional codes to obtain an accuracy that is significantly worse than what we report here). The improved performance is obtained by the use of pseudospectral methods and two-electron integral corrections as described above.

2.12.3 Relative Energies

Table A.2 compares energy differences for a selected set of molecular torsional barriers and conformations for GAUSSIAN 92 and PS-GVB for our parameter set, again using the 6-31G** basis. In all cases, the relative energies agree to better than 0.1 kcal/mole, independent of the size of the molecule. This demonstrates that PS-GVB now has its own cancellation of error comparable to that in GAUSSIAN 92. Again, we emphasize that the GAUSSIAN 92 results are nowhere near the Hartree-Fock limit due to basis set incompleteness, so that energy differences are in fact the only basis for a fair comparison of the two methods.

We have done preliminary studies of torsional energies as a function of electron correlation. Correlation effects can be on the order of several kcal/mole even for systems as simple as butane or urea. Consequently, the "errors" in PS-GVB energy differences are trivial compared to uncertainties due to basis set and correlation effects. This argument applies even more strongly to bond energies where correlation effects are still larger. In summary, then, the performance of PS-GVB with regard to accuracy is quite adequate at the level of our parameter set optimized for computational efficiency and timing comparisons with GAUSSIAN at this level are meaningful.

*2.12.4 Timing Results: 6-31G** Basis*

Table A.3 presents CPU times for a selected set of molecules as compared to GAUSSIAN 92. For small molecules, a factor of 2 is obtained routinely for 6-31G** while for larger molecules a factor between 3–4 is obtained for both a Cray vector supercomputer and for am IBM Model 580 RISC workstation. These results do not represent a major breakthrough but they do reflect a significant quantitative advantage for the PS method. For other basis sets, the advantages are greater; the larger and more complex the basis set, the more the advantage of PS grows, as stated above. To illustrate this, we present results for the the Dunning correlation-consistent TZP basis below.

2.12.5 Timing Results: cc-pVTZ Basis

In table A.4 we show our preliminary cc-pVTZ[36,38] results for a subset of molecules presented above. The larger molecules were chosen to better illustrate the scaling advantage of the PS method. Up to a factor of 6.5 improvement over

GAUSSIAN 92 run with its default cutoffs is achieved in this size regime, showing the PS method's better scaling with basis set size. It should be noted that this basis set uses general contractions, which forces GAUSSIAN 92 to recalculate elementary integrals. However, PS-GVB must also recalculate these quantities, and therefore this timing comparison is more directly illustrative than those for basis sets with shared exponents, such as 6-31G**, where as yet PS-GVB does not take advantage of this construct in its analytical two-electron package described above. With the expectation that future research will focus on larger molecules, as well as bigger basis sets to better model chemical properties, the utility of such a better scaling algorithm as the PS method in *ab initio* chemistry is apparent.

2.13 Conclusion

We have demonstrated that PS methods are capable of reliable computing energy differences and total energies for the Hartree-Fock equations while displaying substantial acceleration of CPU time as compared to GAUSSIAN 92, which is generally accepted as the standard in the field; for single point Hartree-Fock direct SCF calculations on large molecules without symmetry, we believe that GAUSSIAN 92 is the most efficient conventional electronic structure code available. A graphical user interface for UNIX systems has been constructed which makes PS-GVB easy to use as well.

3

PROTEIN STRUCTURE DETERMINATION VIA X-RAY CRYSTALLOGRAPHY

3.1 Overview

Proteins, a vital class of large molecules key to the functioning of cells, don't have a regular structure like the famous double helix of DNA. Structure here refers specifically to the three dimensional arrangement of atoms in space, also known as the protein fold.[39] Instead, proteins come in a nearly infinite variety of shapes and sizes. It has long been recognized that knowing a protein's structure can supply critical information about its function. One generally speaks of the protein's active site, shorthand for the region in the structure where the chemical activity related to the protein's function takes place. Determining the layout of the amide backbone chain and the amino acid residues around the active site is particularly important.

While great strides have been made in the *de novo* and comparative modeling of protein structures, particularly in the past ten years or so, non-empirical methods have not yet proven reliable enough to supplant experiment. Several experimental methods exist which can establish the conformation of a protein, of which X-ray crystallography remains the technique responsible for the most new

structure determinations. The most common alternatives used are nuclear magnetic resonance (NMR) and electron microscopy (EM), although a few structures have been determined by other methods, including neutron diffraction and powder diffraction.[40] To give a sense of the relative importance of the methods, in 2005 there were 34345 structures total in the Protein Data Bank (PDB).[40–42] Of these, 29004 structures had been determined using X-ray crystallography, 4514 of them new in 2005. 5161 were determined by NMR with 884 new, and 108 had been determined by EM, of which 23 were new.

Histograms of the deposits in the PDB show that almost 40% of structures were solved from data with a resolution in the range of 1.5-2.0 Å. Over 60% fell in the range 1.5-2.5 Å. This, no doubt, reflects the fact that structure determination is significantly easier when experiments achieve resolutions below 2.5 Å. Our work primarily intends to enhance the ability of researchers to solve structures of poorer resolution, 2.5 Å and above.

X-ray techniques suffer from two primary bottlenecks. Not surprisingly, both bottlenecks arise because there are no sets of defined steps that guarantee success, so each requires the attention and effort of a highly trained expert to carry out. Due to the nature of proteins, their irregular structure and size, they are difficult to crystallize. This, of course, affects all following stages of the experimental work. However, that stage is not the focus here. The second, perhaps less obvious issue, is the act of deriving a quality model from the experimental data. This is the problem our work intends to address.

In the following pages we will describe our advances in one critical stage of structure determination from X-ray crystallography data, the initial identification of features and concomitant placement of fragmentary pieces of a potential model.

When high resolution data exists, typical methods might attempt to trace the protein backbone directly, but at poorer resolutions this often fails. In this case one often begins by identifying probable locations of larger scale motifs, particularly the ubiquitous alpha helix and beta sheet conformations.

3.2 X-ray Crystallography

We refer the reader to such texts as those by Drenth or Rhodes for a thorough introduction to X-ray crystallography as it pertains to the study of macromolecular systems such as proteins.[43,44] There is, of course, an extensive body of literature in journals, but there are also a wide range of resources on the World Wide Web, including many tutorials. We will give a comparatively short overview here for context.

All present day X-ray crystallography has some major common steps. Sufficient quantities of the material under study must be produced. Crystals are grown from this material. While this step might seem simple (after all, for many common materials like salt, one can grow crystals at home under quite crude conditions) this is in fact one of the more difficult, often rate limiting steps in the overall process. There is no guarantee that a crystal can be formed for any particular material, and even when done successfully, they are often of poor quality. This limits the resolution obtainable in collecting data.

With a crystal (or, more often, several crystals, some made from modified versions of the material) in hand, one can collect X-ray diffraction data. At this point, one cannot escape mention of the ubiquitous “phase problem” of X-ray crystallography.[45] X-rays, though simply a name for part of the spectrum of light, cannot be focused like visible light through a lense. Rather, the three dimensional

spacial image of a material must be reconstructed from the results of the scattering of X-rays from the electrons surrounding atoms (known as X-ray diffraction). X-rays diffracted from a regular system like a crystal will form a pattern of dark regions and high intensity spots. The spots are usually referred to as reflections. Technically they are not reflections, but Bragg developed a conceptually useful framework in which the spots can be thought of as deriving from waves reflecting off of imaginary planes connected to the crystal lattice points.[46] We will discuss reflections and show the origin of the phase problem explicitly shortly.

3.2.1 Electron Density Maps

Classical electromagnetic theory provides a model for connecting the intensities of reflections measured in an X-ray experiment to a three dimensional image of the electron clouds around the atoms in the specimen.[43,44,47] This picture of the electron distribution is commonly referred to as an electron density map or EDM. The formula for the electron density is

$$\rho(x,y,z) = \frac{1}{V}\sum_{h}\sum_{k}\sum_{\ell}\mathbf{F}(h,k,\ell)\exp[-2\pi i(hx+ky+\ell z)] \tag{3.1}$$

where ρ denotes the electron density at location (x,y,z). V is the volume of the unit cell. The sum takes place over the integers (both positive and negative) h,k and l that designate each reflection, known as the Miller indices. $\mathbf{F}(h,k,l)$ are the so-called structure factors. Note $\mathbf{F}$ is a complex quantity. This is made explicit in

an alternative form of Eq. (3.1):

$$\rho(x,y,z) = \frac{1}{V}\sum_{h}\sum_{k}\sum_{\ell}|F(h,k,\ell)|\exp[-2\pi i(hx+ky+\ell z)+i\alpha(h,k,\ell)] \tag{3.2}$$

The actual measured intensities are proportional to the magnitude of the structure factors. As discussed previously, this means that, in a single experiment, a critical component of the information needed to calculate the EDM is not discernable directly, namely the phases $\alpha(h,k,l)$ in Eq. (3.2). This is the source of the phase problem. Phases must be determined indirectly. Common techniques include using phases from a molecule expected to be closely related structurally to the one under study (referred to as molecular replacement) or using heavy atoms (atoms with relatively high atomic numbers such as selenium or mercury) in isomorphous or anomalous dispersion methods.

3.2.2 Difficulties in Structure Assignment

In principle, the wavelengths of typical X-ray sources (less than 1 Å) are sufficiently short to easily resolve structure at the atomic level, and to derive precise phase information. In practice, protein crystals contain a number of kinds of disorder. These range everywhere from deviation of individual atoms from exact lattice position because of ordinary thermal atomic motion to cracking of the crystal during cooling (a now standard practice). Other factors include impurities, the existence of flexible regions, crystal growth faults and conformational changes. The list is quite long.[48]

These imperfections in crystals restrict the effective resolution obtainable in an experiment. At some maximum scattering angle θ one can no longer adequately

distinguish different reflections. This maximum angle corresponds to a minimum spatial distance. This distance is what is commonly referred to as the resolution of the experiment. We showed previously the majority of models deposited in the PDB have a resolution of 1.5-2.5 Å.

In some cases, the reflections for which phase information can be determined are even more restricted. During model building, it is often possible to use information from a partial model to allow inclusion of additional reflections. This is known as phase extension, phase refinement or simply phase improvement.[49] Many of the models derived from the experimental data used in this work have a higher final than initial effective resolution.

With such seemingly poor resolving power (relative to typical molecular scales) how, then, can we expect to arrive at an accurate model? Typical solutions make use of various forms of outside information (meaning any information not directly derived from the experiment in question). This includes restraining models to match known bond lengths and angles, making use of non-crystallographic symmetry when it exists, and density modification methods such as solvent flattening and histogram matching. Various estimation techniques suggest that the RMSD of typical models is on the order of 0.2-0.3 Å, although the issue is not entirely resolved.[43,50]

The most common single measure of model quality in use today is the R_{free} factor, originally introduced by Brünger,[51,52] although this should be interpreted with caution.[53] The reciprocal space R factor is defined as

$$R = \frac{\sum_{hkl} ||F_{obs}| - k|F_{calc}||}{\sum_{hkl} |F_{obs}|}$$

Here h, k and l refer to the Miller indices, so the sums are over all reflections.

$|F_{obs}|$ and $|F_{calc}|$ are the magnitudes of the observed structure factors and those calculated from the model, respectively. The multiplier k is there to adjust for differences in scaling between the structure factors. It is chosen to minimize the R factor.

It is important to note that the phase information actually has more to do with the quality of a density map than the structure factor magnitudes do. Injudicious phase refinement can lead to a completely wrong interpretation of the map (i.e. a model that, by naive measures, fits the map well, but does not correspond to the actual protein structure).[54,55] The R factor defined above has been demonstrated to suffer from model bias. Model bias occurs because the incorporation of phase information from a partial model will tend to reinforce the EDM's matching of the model. R_{free} avoids this problem by selecting a test set of reflections which are not included in the refinement process.[56,57] Typically around 10% of the reflections are chosen to make up the test set. Since these structure factors don't incorporate model phase information, R_{free} avoids bias introduced during model refinement. The equation for R_{free} is given as

$$R_{free} = \frac{\sum_{hkl \subset T} ||F_{obs}| - k|F_{calc}||}{\sum_{hkl \subset T} |F_{obs}|}$$

This is identical to the expression for R except the sum runs only over $hkl \subset T$, the set of reflections belonging to the test set.

One might naively assume that the R_{free} factor would typically be quite small for a well-determined structure, but this is rarely the case. A histogram of R_{free} values from the PDB shows a balanced distribution with almost 40% of deposited structures falling the the 0.228-0.266 range. Only a fraction of a percent of structures are listed with an R_{free} value of less than 0.1.

3.3 Current Model Building Methods

Automated protein structure determination is a highly active area of research. In a review published in 2000, Lamzin and Perrakis[58] listed only one tool under the heading of "Automated model building", the ARP/wARP suite. Several new additions have become available in a few short years since then. The journal Acta Crystallographica Section D has taken to devoting an entire issue once each year to the "Proceedings of the CCP4 study weekend"[59]*. The 2004 special issue is devoted entirely to model building and refinement.[60] The problem is far from "solved", even in regimes where researchers can successfully build models fairly reliably by hand. In this section we give an overview of the history and current state of the efforts to automate the initial model building process.

Greer published perhaps the first automated method designed to assist in model building.[61] Through an iterative procedure, points are removed from a EDM grid, starting at low density and moving to progressively higher values, leaving points that connect to form a "skeletonized" trace of the protein backbone. C^{α} sites are identified by the branch points where side-chain density forks off from the backbone trace. This technique is still in use today, incorporated into many of the most widely used graphics systems such as O.[62] Levitt's MAID and Oldfield's QUANTA packages both make use of forms of skeletonization as part of their automated build procedures.[63–65]

The CAPRA system also begins by skeletonizing an EDM.[66,67] The remaining trace points are considered candidates for C^{α} placements. The approach then uses pattern-recognition techniques to identify true C^{α} positions. This consists of a two step sequence. First a large number of rotationally invariant quantities are

*CCP4 refers to a widely-used package of software tools for analyzing crystallography data

calculated based of spheres of density around the location in question. Features include quantities such as the moments of inertia, mean density, and other statistics. These values are fed into a neural network. The network output predicts how far the current location lies from a true C^{α} position. These predictions are ranked by increasing distance and a chain built by moving through the list in rank order.

Kleywegt and Jones developed a method specifically intended to help detect secondary structure features.[68] Their program, ESSENS, calculates a crude score at each point in a real-space map reflecting how well the atoms in a template match the neighboring density. Scores are calculated for several rotational orientations of the template. A map consisting of the best score found at each point for all the orientations sampled is produced, along with a PDB file giving the coordinates of the best overall match. In this way, ESSENS created a visual aid for identifying smaller features (e.g. helical fragments as opposed to an entire domain) rather than a true automatic detection tool.

Cowtan extended the approach of Kleywegt *et al.* to create a true automatic feature detection tool.[69,70] The program FFFEAR calculates a masked square residual between the fragment and map densities. For a particular rotational orientation of the fragment, the translational search is carried out using fast Fourier transforms[†]. Combined with other techniques (for example, to deal with density scaling), FFFEAR is capable of performing the equivalent of an exhaustive six-dimensional search for any template specified as a set of PDB style coordinates using a more sophisticated scoring function, in relatively little computer time. Coordinates corresponding to the top matches (100 by default) are output in PDB format. Cowtan also introduces a statistical search function based on Bayesian prob-

[†]In Cowtan's terms "Fast Fourier Feature Recognition", hence the name FFFEAR

ability theory, with some improvement in discrimination. The statistical method has a drawback in that the search models require significant preparation. FFFEAR is included in the popular CCP4 suite of crystallographic tools. Recently, Cowtan published another hybrid approach that uses a similar maximum likelihood search function to identify C^{α} positions, which are then formed into chains to produce a backbone trace.[71]

The output of FFFEAR can be combined with other tools in the CCP4 suite to attempt to assemble a model, but this procedure is not automated. The program RESOLVE adopted Cowtan's FFT search and the use of a maximum likelihood target as the first stage in a fully automated procedure capable of building a complete model. Two templates are used, a six segment helix and a four segment beta sheet form. Best match locations for each template are stored and used as the seed sites for the rest of the process.

Perhaps the earliest approach capable of producing a nearly complete model automatically is the ARP/wARP suite. This method works using an entirely different approach. Models are built iteratively by placing dummy atoms at density peaks, then matching patterns of dummy atoms to known protein motifs, refining the positions, then repeating. One particularly interesting aspect of this method is the fact that dummy atoms can be removed or added at each iteration, introducing the possibility that the model can make discontinuous shifts, unlike most other approaches. However, since ARP/wARP essentially depends on picking density peaks, it can generally only build a significant portion of a model at resolutions below ~ 2.5Å.

We have tried to capture our estimation of the most significant and mature methods for aiding in or automating the initial model building stage of the protein

structure determination process. This list is hardly complete. For a more detailed look at the field we suggest reading Lamzin and Perrakis, Rupp, and Morris.[58, 72, 73] Of particular interest, too, is the paper by Badger comparing the model building capabilities of ARP/wARP, MAID and RESOLVE.[74] The literature unfortunately has a dearth of this sort of direct comparison at present.

3.4 Improving Feature Detection

3.4.1 Overview

In the following sections we will present our efforts at improving the first steps of model building by enhancing feature detection. We follow a path similar to Kleywegt *et al.* in using a molecular fragment as a template in a six-dimensional search. While the program we developed is capable of performing an exhaustive search, rather than rely on this alone we follow the search with a further rigid-body minimization. As we will show, we found that reasonable results, often superior to that of FFFEAR, can be achieved using a much coarser initial search when followed by refinement of the template placement. We will discuss the algorithm in detail in a later section, but the main elements comprise sampling a number of potential template match locations, obtaining a score reflecting the initial fit, then refining the promising initial placements. Key elements for success then are the adequacy of the sampling, the discriminatory power of the scoring function, and level of enhancement obtained in performing refinements. These all intertwine inextricably to shape the final method.

3.4.2 Mathematical Basis

Let us begin, then, by discussing scoring. We primarily use the following scoring function

$$S(x_1, \ldots, x_n) = 1 - \frac{\sum_i f_i m_i}{(\sum_i f_i^2)^{\frac{1}{2}} (\sum_i m_i^2)^{\frac{1}{2}}} \tag{3.3}$$

Here we define $f_i \equiv F_i - \langle F \rangle$ and $m_i \equiv M_i - \langle M \rangle$ where $\langle F \rangle = \frac{\sum_i F_i}{N}$ and $\langle M \rangle = \frac{\sum_i M_i}{N}$. F_i and M_i are the density values of the fragment and the molecule at grid point i, respectively. N is the total number of grid points used in scoring. $x_1, \ldots, x_n$ are the coordinates that specify the fragment pose. Since we move the fragment as a rigid body, there will be six degrees of freedom, although not necessarily six independent coordinates.

The last term on the right hand side of the equation is commonly known as the correlation coefficient. If one thinks of the numerator as the dot product of two vectors (albeit ones made up of density values), it is easy to realize that this term varies between 1 and -1. The particular form, then, of the scoring function varies between 0 and 2, with lower values indicating a better fit. This last detail is done simply so that the refinement can take the form of a minimization problem. There are hidden details in this scoring method. In particular, the choice and number of grid points to use are critical factors both to the discriminatory power obtained and the efficiency of the algorithm. We will discuss these issues when describing the implementation.

Two other values have proven useful in evaluating a fit. The function

$$U(x_1, \ldots, x_n) = 1 - \frac{\sum_i F_i M_i}{(\sum_i F_i^2)^{\frac{1}{2}} (\sum_i M_i^2)^{\frac{1}{2}}} \tag{3.4}$$

uses the values of the densities rather than their deviation from the average. Both this and the primary scoring function can be related to a simple least squares score by including an overall multiplicative factor and a multiplicative factor and an offset, respectively, to the data. Recall the data is scaled arbitrarily and has an unknown offset (corresponding to the unmeasured $\mathbf{F}(0,0,0)$ structure factor). Once again appealing to vector arithmetic, one can see that terms in equation 3.3 correspond to the angle between two unit vectors, whereas equation 3.4 has a correspondence with the scalar product of two arbitrary vectors.

Since both equations 3.3 and 3.4 effectively allow fits within regions of very low density, it turns out to be useful to include a cutoff based on the sum of the squares of the molecular density at the grid points considered. These three values, then, are all considered in evaluating a fit. The use of cutoffs will be discussed further in the implementation section.

The refinement process takes the form of adjusting the fragment position and orientation from the initial pose to find a better local fit. We use a quasi-Newton minimizer[75,76] to do this, which essentially requires that the score have an analytic gradient. To evaluate the components of the gradient $\frac{\partial S}{\partial x_j}$ we first note that m_i is independent of the coordinates x. Breaking down the substituent pieces of the formula, we have

$$\frac{\partial \sum_i f_i m_i}{\partial x_j} = \sum_i m_i \frac{\partial f_i}{\partial x_j}$$

$$\frac{\partial}{\partial x_j}(\sum_i f_i^2)^{-\frac{1}{2}} = -(\sum_k f_k^2)^{-\frac{3}{2}} \sum_i f_i \frac{\partial f_i}{\partial x_j}$$

Factoring out $\frac{\partial f_i}{\partial x_j}$ we have then the general form[‡]

$$\frac{\partial S}{\partial x_j} = -(\sum_k f_k^2)^{-\frac{1}{2}}(\sum_k m_k^2)^{-\frac{1}{2}} \sum_i [m_i - f_i(\sum_k f_k m_k)(\sum_k f_k^2)^{-1}]\frac{\partial f_i}{\partial x_j} \quad (3.5)$$

Since $\langle F \rangle$ is independent of gridpoint, for any set of coefficients $\{c_i : \sum_i c_i = 0\}$, it follows that

$$\sum_i c_i \frac{\partial f_i}{\partial x_j} = \sum_i c_i \frac{\partial F_i}{\partial x_j}$$

The derivative term $\frac{\partial f_i}{\partial x_j}$ always has multipliers that meet this requirement, so it remains then only to find expressions for $\frac{\partial F_i}{\partial x_j}$ in order to evaluate Eq. (3.5). Following the method of Agarwal and others, we approximate the electron density surrounding an atom with a sum of Gaussians.[59,77,78] The fragment density at a point **r** in space is given as

$$F(\mathbf{r}) = \sum_a \sum_n C_{an} \exp[-\lambda_{an}(\mathbf{r} - \mathbf{r}_a)^2] \quad (3.6)$$

where the sum a is over all atoms and the sum n is over the number of terms in the Gaussian approximation for the density contributed by an atom. $\mathbf{r}_a$ denotes the position of atom a in space. Denoting grid location i by $\mathbf{r}_i$, we have $F_i = F(\mathbf{r}_i)$. Though not explicitly shown, the exponential factor λ incorporates a temperature factor.

The positions of the atoms at the sample pose are obtained by rotating and translating the reference fragment. Hence

$$\mathbf{r}_a = \mathbf{R} \cdot \mathbf{r}'_a + \mathbf{T}$$

[‡]This assumes all the coordinates are independent.

where $\mathbf{r}'_a$ is the original position of atom a, $\mathbf{R}$ is the rotation matrix and $\mathbf{T}$ is the translation vector needed to transform from the reference to the sample pose. $\mathbf{R}$ and $\mathbf{T}$ are given by

$$\mathbf{R} = \begin{pmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2-q_1^2+q_2^2-q_3^2 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2-q_1^2-q_2^2+q_3^2 \end{pmatrix}$$

$$\mathbf{T} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

$\mathbf{T}$ is a straight-forward Cartesian translation vector. In calculating $\mathbf{R}$ we chose to use a quaternion representation. We have identify as coordinates $(x_1, \ldots, x_7) \equiv (q_0, \ldots, q_3, a, b, c)$. With this choice, the coordinates used to locate a pose are not all independent. Rather, the four values q_0-q_3 must satisfy the constraint $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$. Strictly speaking, then, equation 3.5 does not give the total derivative of S. We will examine different approaches to address this when discussing the implementation details.

The quaternion representation has a number of advantages over other methods.[79] In particular, we initially used a rotation matrix calculated using Euler angles:[80]

$$\mathbf{R} = \begin{pmatrix} cos(\alpha)cos(\beta)cos(\gamma) & -sin(\alpha)cos(\beta)cos(\gamma) & sin(\beta)cos(\gamma) \\ \quad -sin(\alpha)sin(\gamma) & \quad -cos(\alpha)sin(\gamma) & \\ cos(\alpha)cos(\beta)sin(\gamma) & -sin(\alpha)cos(\beta)sin(\gamma) & sin(\beta)sin(\gamma) \\ \quad +sin(\alpha)cos(\gamma) & \quad +cos(\alpha)cos(\gamma) & \\ -cos(\alpha)sin(\beta) & sin(\alpha)sin(\beta) & cos(\beta) \end{pmatrix}$$

The Eular angle representation suffers some well-known problems, including the difficulty in obtaining a uniform sampling of rotations and degeneracies where widely different sets of angles produce the same rotation.[81] Those reasons alone make the quaternion representation preferable. Experience suggests that convergence of the quasi-Newton minimization technique is superior using quaternions, too.

Continuing with the derivation of the gradient of the scoring function, we have

$$\frac{\partial F_i}{\partial x_j} = 2\sum_a (\mathbf{r}_i - \mathbf{r}_a) \cdot \frac{\partial \mathbf{r}_a}{\partial x_j} \sum_n \lambda_{an} C_{an} \exp[-\lambda_{an}(\mathbf{r}_i - \mathbf{r}_a)^2]$$

$\mathbf{r}'_a$ is independent of the coordinates, so

$$\frac{\partial \mathbf{r}_a}{\partial x_j} = \frac{\partial \mathbf{R}}{\partial x_j} \cdot \mathbf{r}'_a + \frac{\partial \mathbf{T}}{\partial x_j}$$

$\frac{\partial \mathbf{T}}{\partial x_j}$ is particularly simple. We have $\frac{\partial \mathbf{T}}{\partial x_j} = 0$ for $j \in (1, \ldots, 4)$ and, e.g., $\frac{\partial \mathbf{T}}{\partial a} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$. Similarly, $\frac{\partial \mathbf{R}}{\partial x_j} = 0$ for $j \in (5, 6, 7)$. The equations for $\frac{\partial \mathbf{R}}{\partial x_j}$, $j \in (1, \ldots, 4)$ are more complicated. Explicitly,

$$\frac{\partial \mathbf{R}}{\partial q_0} = 2 \begin{pmatrix} q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{pmatrix}$$

$$\frac{\partial \mathbf{R}}{\partial q_1} = 2 \begin{pmatrix} q_1 & q_2 & q_3 \\ q_2 & -q_1 & -q_0 \\ q_3 & q_0 & -q_1 \end{pmatrix}$$

$$\frac{\partial \mathbf{R}}{\partial q_2} = 2 \begin{pmatrix} -q_2 & q_1 & q_0 \\ q_1 & q_2 & q_3 \\ -q_0 & q_3 & -q_2 \end{pmatrix}$$

$$\frac{\partial \mathbf{R}}{\partial q_3} = 2 \begin{pmatrix} -q_3 & -q_0 & q_1 \\ q_0 & -q_3 & q_2 \\ q_1 & q_2 & q_3 \end{pmatrix}$$

With these terms in place we have the necessary analytic expression for the gradient of the scoring function S.

3.5 Algorithm

The core algorithm is shown in Figure 3.1 on the next page. There is supporting code for performing necessary initializations, reading a structured control file and reading electron density maps (not illustrated). The code was written in the C programming language, which is reflected in the style of the schematic representation. Terms ending in empty parentheses indicate subroutines. Indentation is used to reflect the code structure. Subroutine calls and code wrapped by control structure (loops, conditional statements) are indented one level further than the surrounding code.

Routine `dr_fit_fragments()` begins by looping over a list of fragments to be fitted. The list of fragments is specified in the the control file in a format of our own design. Each fragment entry comprises a fragment id, number of atoms, number of scoring rounds, the number of points and cutoffs for each round, a fragment "pad" distance and the coordinates and types of all the atoms. The coordinates are given in a regular orthogonal system, in Ångstroms. They are automatically recentered to a system with the origin at the geometric center of the fragment.

The code then enters a set of nested loops. The outermost loop runs over a sampling of unit quaternions, up to the requested number of rotational poses. The quaternions are taken in order from a preset table designed to provide uniform, deterministic sampling. We will discuss the sampling method in the discussion section. Next there is a loop over divisions, which allows fine-grain sampling of translational offsets. Normally the reference fragment is adjusted to have its geometric center at the origin. If an integral division is specified, sampling is also done starting with the reference fragment centered at integral divisions of the

```
loop over fragments, quaternions, divisions
 gd_gen_search_grid()
  qn_set_rotation()
  as_transform()
  fragment_cell()
  fragment_grid()
   gd_set_density()
   us_sort_grid_index()
   gd_gen_indexes()
 loop over vertex grid positions
  fs_check_fit()
   loop over scoring rounds
    fs_score()
    evaluate scores
     pass: continue loop
     fail: return for next position
   end loop over scoring rounds

   re_lbfgs()
   evaluate scores, convergence
    pass: continue
    fail: return for next position

   as_transform()

   loop over existing clusters
    find lowest RMSD match
   end loop over existing clusters

   if RMSD cutoff met
    update_cluster()
   else
    add_cluster()
   end if
 end loop over vertex grid positions
end loop over divisions, quaternions, fragments
```

Figure 3.1: Schematic representation of `dr_fit_fragments()`

grid spacing in all three dimensions. Rotations are always performed about this origin (i.e. the geometric center and, potentially, a series of small offsets near the geometric center).

With the fragment, rotational pose and center offset determined, we are now ready to generate a set of fragment values which will be used for all the translational searches. Calculating the electron densities over a grid, although optimized, is computationally expensive. The translational search is performed by calculating the fragment densities on a grid with the same axes and grid point spacings as the molecular grid. Since the relative offsets of the grids are matched, translational sampling amounts to selecting different base indices in the molecular grid.

The subroutine `gd_gen_search_grid()` encapsulates all the functions involved in creating the fragment grid. It calls subroutine `qn_set_rotation()` to calculate the rotation matrix $\mathbf{R}$ given the quaternion value for the pose. Subroutine `as_transform()` then effects the rotation of the reference fragment about the origin, that is, it calculates the new atomic positions for each atom. It will also add a translational offset, although not in this initial case. Given a set of atomic coordinates, `fragment_cell()` calculates the vertices of a cell around the atoms. The cell must have the same shape as the unit cell of the crystal (angles and relative axes lengths). The cell is chosen so that no atom comes closer to a cell face than the distance of the pad parameter (that is, the cell size ensures that every atom has a certain radius of room around it in the grid).

Continuing in `gd_gen_search_grid()`, subroutine `fragment_grid()` generates all the Cartesian coordinates of the grid points for the cell. Subroutine `gd_set_density()` calculates the fragment density at each grid point from the sum of the density contributions of all the atoms. The density contributions are

calculated from a standard approximation in the form of a sum of Gaussians.[77] This step is optimized using a table lookup with a linear interpolation. This allows an excellent approximation of density distribution given by equation 3.6, including the temperature factor, without the computational expense of evaluating exponentials. It should be noted that while any temperature factor can be specified at run time, only one factor is applied during a run, to all atoms. An index array of the grid points is sorted in `us_sort_grid_index()`, so the points may be directly accessed ordered by density value, highest to lowest. The search preparations are completed by `gd_gen_indexes()` which calculates the index offsets necessary to access a molecular grid point given a fragment grid point and a base offset. See the description of the translational search that follows.

After generating the fragment grid for this particular rotational orientation, `dr_fit_fragments()` enters a set of loops over molecular grid points. The fragment grid, which has the same cell axes and grid spacing, aligns with the molecular grid. Identifying a vertex point from the fragment grid with a molecular grid point is equivalent to calculating the fragment density with the fragment shifted to a new origin. This amounts to moving the fragment around the molecule in discrete steps. Note this means any single search can only happen at discrete displacements of the fragment center. The code described earlier for shifting the fragment center slightly makes up for this, allowing denser searches.

Once we have a fragment grid, and have selected the molecular grid point at which to overlay this grid, we drop into subroutine `fs_check_fit()`. Subroutine `fs_check_fit()` begins by scoring the now completely determined fragment pose. This is done in a loop over cutoff rounds. Each round includes more points in the score than the previous round, with the idea that extremely poor fits

can be discarded without the cost of computing the full score. There are three cutoffs evaluated at present, the two scores described in section 3.4.2 on page 97 and the sum of the square of the molecular density over the same set of grid points. All values necessary for these three sums are easily accumulated and fed from one round to the next. Because the fragment grid is ordered by density, scoring starts with the highest density grid points and includes terms with progressively lower density. If all the points were included at which the fragment density was non-negligible, the last term of Eq. (3.3) would be the discretized correlation coefficient between the fragment and the molecule. This is both computationally expensive and unnecessary. We will show in the results that adequate discrimination is achieved with a relatively small number of points. The actual score calculation is performed in `fs_score()`, which takes as input the fragment densities, the molecular densities, and an index array to map between fragment and molecular grid points.

If a position passes the initial scoring rounds, we enter the refinement stage. The refinement step searches for a local minimum of the primary scoring function using the limited-memory BFGS quasi-Newton library by Nocedal and collaborators (version 2.1) (hereafter referred to as LBFGS).[75,76,82,83] The refinement algorithm is shown in Figure 3.2 on the next page.

Subroutine `setulb()` is the entry point for the LBFGS library. Calling routines interact through a number of variables and the character array TASK. The first call to `setulb()` initializes the code. Subsequently the surrounding code is driven by the return values in TASK. If TASK returns NEW_X, a step vector array contains the coordinates of a new minimum. If TASK returns FG, the calling routine is expected to supply the value of the function and its gradient at the position

```
set start conditions

loop indefinitely
 setulb()

 if TASK is
  NEW_X:
    save new coordinates
    optionally adjust constraints

  FG:
    rescale quaternion coordinates
    gd_gen_refinement_grid()
      qn_set_rotation()
      as_transform()
      build grid tag table
      gd_set_density()
      us_sort_grid_index()
    fs_score()
    gs_qn_grad()
 end if

 optionally adjust constraints and continue loop

 if converged break loop
end loop

rescale quaternion coordinates
gd_gen_refinement_grid()
fs_score()
```

Figure 3.2: Schematic representation of `re_lbfgs()`

given by the step vector. Finally, TASK can return with a value CONVERGED, meaning one or more of the convergence criteria has been met and the refinement cycle has ended.

Because of the relatively simple form of our scoring function, we can calculate the gradient analytically. To calculate both the score and gradient at a new pose we need to re-evaluate the fragment density. This can be computationally expensive. With the exception of the way in which we select the points to include in the score/gradient calculation, the steps involved in obtaining the fragment densities for a new pose are essentially the same as those for obtaining the original grid. Consequently, subroutine `gd_gen_refinement_grid()` calls many of the same routines as `gd_gen_search_grid()`. We will discuss the approach used to reduce the number of points in more detail further along. See section 3.6.2 on page 111.

Subroutine `gs_qn_grad()` builds the quaternion-based gradient. As pointed out earlier, quaternions do not form a set of independent coordinates. This is dealt with in two ways. Quaternion coordinates are always normalized whenever new ones are generated. And, after the gradient is calculated, we remove the component normal to the three-sphere. This means that to first order a step will rotate a quaternion without changing its length. This ensures that the gradient is not dominated by a false term (the change in the score due to deviation from unit quaternions). Other methods could be used as well. In particular, we tested Lagrange multiplier and Augmented Lagrange multiplier techniques, but these proved unreliable, with no advantage over the simple and direct method described here.[84]

Once the refinement rounds are done, the code performs a straight forward

clustering of the results. A match is compared against the current cluster list. If the RMSD of the atomic distances is less than the control parameter value, the match score is compared to that of the current cluster representative. If the score is better, the new match is kept and the old one thrown out. If no other match has a sufficiently small RMSD, a new cluster is added to the list.

3.6 Algorithm Discussion

3.6.1 Minimization Method

The scoring function used is infinitely differentiable. It is tempting to consider trying to use a true Newton-Raphson method, or otherwise make use of the Hessian of the scoring function. In practice, this does not work very well.[85] The LBFGS code is quite mature and sophisticated, in fact more so than is necessary for a problem with relatively few dimensions. There is no reason to think that other methods would not work equally well, although the ability to apply constraints to the variables is critical.

The quasi-Newton minimization technique works most efficiently and reliably when the function in question can be accurately approximated by quadratic terms over wide regions near any minimum. While $\mathbf{R}$ is quadratic in the variables q, F_i is not. It is intuitively tempting to believe the non-linear $\sin$ and $\cos$ functions that appear as coefficients in the derivatives of the scoring function when using the Euler representation will limit the domains over which a quadratic approximation works well, but an analysis of Eq. (3.6) does not suggest this is actually the case. Coefficients beyond the second derivative in a Taylor expansion of a simplified function of the exponential term in Eq. (3.6) do not drop off more rapidly in mag-

nitude with quaternions. Despite this, and the fact that enforcing the normalization of the quaternions adds a small amount of complexity, other benefits of using this representation (some of which were mentioned previously) persuaded us to this form. While not demonstrated rigorously, experience suggests the minimization is more efficient and stable using quaternions.

3.6.2 Efficiency

Evaluating Eq. (3.6) and the related derivative terms precisely would be prohibitively expensive because of the need to compute numerous exponentials. Since the function $F(\mathbf{r})$ depends only on the magnitude $|\mathbf{r}|$, it is simple to precompute the values once and use a table lookup scheme instead. Tables large enough to hold all the terms with a granularity sufficient to provide the precision needed would require large amounts of memory. Instead, we perform a lookup combined with a linear interpolation. This requires only a little extra computational effort, but roughly doubles the accuracy of a straight lookup, allowing the use of much smaller lookup tables.

During the minimization phase, the fragment is moved to new positions without regard to the grid spacing. The fragment density can only be (usefully) calculated at positions corresponding to molecular grid points. This necessarily means that the scoring function can't "track" the fragment, meaning the positions used to calculate the score move relative to the fragment as the fragment moves. The original scoring points are chosen by calculating the fragment density over a grid contained within a cell framing the fragment and then using a subset of the points with the highest density. The number of scoring points used is typically significantly smaller than the number of points over which the density is evaluated (on

the order of a factor of 10 or more). Clearly it is preferable to recalculate the densities during the minimization only for the grid points that will be used in scoring. Initially we kept the points fixed and recomputed the density only at the positions used for initial scoring. It soon became apparent that the fragment could move far enough that important elements were missed. We therefore developed an algorithm which tags the allowed position closest to where a previous scoring position lies after being shifted, and all of that point's neighbors in the grid. We then obtain the densities for those points and sort them, finally again taking the subset of points ordered by decreasing density. Selecting the gridpoints this way can be done quickly using integer operations, greatly reducing the cost of each minimization step.

With the algorithm in place, the choices of cutoffs and sampling parameters obviously have the largest effect on the final CPU times. While we obtained a great deal of insight during the development and testing of this method, it is computationally intractable to fully explore the range of possibilities. The parameters used are adequate to carry out meaningful comparisons, but are almost certainly far from optimal. It is particularly notable that values of the scoring function are only meaningful to about two significant figures, but the tolerances used in the LBFGS method cause the minimization to converge to roughly four figures. We found that loosening the convergence criteria often led the algorithm to miss many correct placements. Apparently it is fairly common for the scoring function to have relatively flat regions near local minima.

3.6.3 Sampling Rotations

We chose the unit quaternion representation of rotations in three dimensions (i.e. the group $SO(3)$). Coutsias and Romero[86] show that a quaternion

$$q = (\cos(\theta/2), \sin(\theta/2)\mathbf{p})$$

with $\mathbf{p}$ a unit vector, represents a rotation by an angle θ around the axis $\mathbf{p}$. They also show that q and -q represent the same rotation (as can easily be seen from the expression for the rotation matrix given previously). A uniform distribution of unit quaternions with antipodal points identified therefore provides a uniform sampling of rotational poses. We chose the quaternion representation in part because obtaining a uniform sampling of rotations in other representations (notably Euler angles) is quite difficult. With quaternions, the problem becomes one of sampling over a unit three-hemisphere. Note here we are not interested in a random sampling, but rather a deterministic one that increases coverage of $SO(3)$ uniformly as more points are added.

Sampling values on a unit sphere in four dimensions is still a difficult problem.[81,87,88] Exact solutions have only been found for relatively small numbers of points, but many methods exist that provide good approximate solutions. In order to allow a user to determine the number of poses sampled at run-time, a list of quaternions was generated using a program that produces a low-discrepancy, low-dispersion, deterministic sampling over the three-hemisphere.[89,90] Lindemann *et al.*[91] define one uniformity measure, the dispersion of a point set P, as

$$\delta(P, \rho) = \sup_{q \in X} \min_{p \in P} \rho(q, p)$$

where X represents the space of all possible samples and ρ denotes any metric. From this definition one can see that a low-dispersion point set must be uniformly distributed. An intuitive metric is the angle between two points, $\theta = \arccos(\frac{q \cdot p}{|q||p|})$. Examining $\cos(\theta)$ gives an indication of the uniformity of a point set. We computed the maximum and minimum values of $\cos(\theta)$ for each point and its nearest neighbor, and the average of those values over the whole set for sample sets of increasing size for the sequence of quaternions we used. That is, for each point in the set, we calculate $\cos(\theta)$ with every other point and track the minimum values for each point. The maximum values provide an approximation to the definition of dispersion above. The average and the minimum values give a measure of the discrepancy of the distribution. These calculations showed that the selected quaternions do indeed give good uniformity over a large range of sample sizes (currently up to a maximum of 1000).

3.7 Results and Discussion

3.7.1 Comparison

In order to assess our results, we compared fragment searches using the FFFEAR program from the CCP4 suite (version 6.0) against the current work using the experimental results for ten different proteins (supplied by Professor L. Tong, whom we gratefully thank for the assistance). We will discuss FFFEAR in some detail here, but the we direct the reader to the work of Cowtan[69,70] for complete details. We also recommend reading the seminal work of Kleywegt and Jones describing their program ESSENS,[68] an earlier effort to aid in identifying features.

Summary information about the proteins, including the PDB identifier, the

PDB (NESG) ID.	Residues (Chain A)	Initial Resolution	Solvent Fraction	Free R Factor	Final Resolution
1XQ4 (ber40)	139	3.00 Å	0.48	0.286	2.70 Å
1SQ1 (br19)	370	2.80 Å	0.55	0.279	2.80 Å
1TZ9 (efr41)	367	3.00 Å	0.42	0.292	2.90 Å
1TM0 (lr31)	350	2.84 Å	0.45	0.312	2.80 Å
1SQ4 (par14)	278	2.70 Å	0.50	0.262	2.70 Å
1RU8 (pfr23)	232	2.70 Å	0.47	0.277	2.70 Å
1YXB (rr8)	98	2.60 Å	0.51	0.295	2.60 Å
1ZEE (sor52)	403	2.50 Å	0.47	0.275	2.31 Å
1YDO (sr181)	307	2.90 Å	0.50	0.303	2.71 Å
1YVK (sr237)	163	3.20 Å	0.67	0.284	3.01 Å

Table 3.1: A brief summary of the data sets used.

Northeast Structural Genomics Consortium (NESG) identifier,[92] the number of residues in the A chain[§], the resolution of the data initially used for structure determination, the solvent fraction initially used, the Free R measure of the structure quality and the final (phase improved) resolution of the deposited (PDB) model are given in table 3.1. Note the initial resolutions range from 2.50 Å to 3.20 Å. A representative graphic image for chain A of each model can be found in appendix B on page 131. More information can be found in the PDB entries for each protein or on the NESG web site.

We took the original reflection data obtained from SOLVE and performed standard pre-model building modification using RESOLVE version 2.10. That is, we ran RESOLVE supplying only the same estimated solvent content used in the original structure determination and using the "no_build" keyword. With

[§]Most of the models have more than one molecule of the same protein in the asymmetric unit. None contain more than one protein.

this input, RESOLVE performs no model building or phase improvement, but performs other modifications as discussed in Terwilliger et al. and produces modified reflection data.

The CCP4 suite comes with a library of 11 fragments used as the standard set for feature recognition with the program FFFEAR.[93] Of these, one (a 70 segment helix model) has no use here. Two other models, the empirically determined nine segment helix and the nine segment "maximum-likelihood" model only deviate from each other by 0.15 RMSD. The results for these two fragments are quite similar, so in most cases only the data for the first of the two will be shown. (N.B. The CCP4 graphical user interface excludes both the 70 segment helix model and the nine segment maximum-likelihood helix model from the list of fragment choices.)

We ran FFFEAR using standard parameters suggested by the program documentation.[94] The exact keyword input is

```
SOLC <solvent ratio>
SEARCH STEP 10
RESO 1000.0 <resolution>
LABI FP=FP SIGFP=SIGFP PHIO=PHIM FOMO=FOMM
END
```

where the terms in angle brackets (<>) represent the appropriate protein-specific parameters, namely the solvent ratio and upper resolution limit to use.

FFFEAR uses an exhaustive search technique that relies on fast Fourier transforms for efficiency. The translational search takes place at discrete points spaced on the order of 0.5 Å apart in each dimension. With an angular search step of 10 degrees (recommended), FFFEAR sampled 1781, 3154, 6055 or 12139 orientations, depending on the symmetry group of the target.

The base scoring function used by FFFEAR (also known as a translation search

function) is simply a weighted (or masked) squared difference between fragment and map densities. In the notation of Cowtan (2001), for a particular fragment orientation the function is

$$t(x) = \sum_{y} \mu_f(y) \big[\rho_f(y) - \rho(y - x)\big]^2 \tag{3.7}$$

where $\rho_f(x)$ is the fragment density, $\mu_f(x)$ is the fragment mask and $\rho(x)$ is the map density. The program simply reports the lowest values for this function (rather than performing any sort of minimization of the square residual).

The current work, as discussed previously, performs a far less exhaustive sampling. The translational search takes place at points on the order of 4 Å apart in each dimension (i.e. at approximately $\frac{1}{8^3}$ as many positions as FFFEAR). The exact spacing depends on the sample spacing of the density map, which, in turn, varies slightly for convenience in calculating the map from the reflection data via FFT techniques. The maps used were generated from the reflection data using the CCP4 FFT program.[95] The rotational search uniformly sampled 150 orientations. The program does not currently take into account any symmetry. Cutoffs were enforced so that a sample pose had to have either a primary score (Eq. (3.3)) or secondary score (Eq. (3.4)) less than or equal to 1.0 and a mean squared molecular density of not less than 0.008 for each scoring round before refinement would be performed. Scoring is split into two rounds; the first using two points per atom and the second using ten points per atom. The cutoff values must be met each round before proceeding to the next step.

We used the LBFGS package to refine qualified poses. Three important parameters control the behavior of the LBFGS algorithm, the number of BFGS correc-

tions, the termination tolerance and the projected gradient tolerance. We specified 20 corrections (the recommended maximum). The first termination condition is

$$(f_k - f_{k+1})/\max(|f_k|, |f_{k+1}|, 1) \leq \text{ftol*epsmach} \tag{3.8}$$

where f_k denotes the value of the function being minimized at the kth iteration, epsmach is the "machine precision", meaning the smallest value that can be added to 1.0 with a result different from 1.0 in the double precision floating point representation for the computer used, and ftol is the user defined tolerance. In our control input we combined ftol and epsmach so the input parameter more intuitively approximates the number of significant digits in the score that remain the same for termination. Explicitly, this is simply term/epsmach. We used a value of $1.0x10^{-4}$ for term. The second termination condition specifies a tolerance for the projected gradient g.

$$\|\text{proj g}\| \leq \text{pgtol} \tag{3.9}$$

This allows the refinement to end when the magnitude of the gradient vector drops below a certain value, indicating a stationary point has been reached within tolerance. We used a value of $1.0x10^{-2}$. Only converged positions with scores of 0.5 (primary) and 0.5 (secondary) are kept. We used a B factor of 50 unless otherwise noted.

The LBFGS package can perform constrained minimization, allowing the user to fix upper bounds, lower bounds or both on any selection of variables. We constrain the amount the translational coordinates can change during refinement. We set the upper and lower bounds on each translational coordinate to allow a range of $\pm 2.4x$ where x here denotes the incremental distance between grid points

along an axis. This was found to have two benefits. In tests it was found that the fragment could slide a long distance when refined against lower resolution maps. Applying constraints allows us to find a best fit within a limited region. Also, the LBFGS code will, on occasion, attempt a very large step, much larger than one wants for a local search on a molecular scale. At times, the step would even move the fragment completely off the grid. Usually this happens early on in the refinement process. Applying the constraints produces results much more in keeping with the desired search, and can allow a minimization to continue in cases where a move would otherwise cause an outright failure.

Appendix B contains graphs comparing results from the two methods. The FFFEAR results are shown on top, with the corresponding results from the current work below. Each program outputs a set of atomic coordinates for the fragment and score for each placement. A separate analysis program finds the lowest RMSD match between each putative fragment location and the model backbone positions (taking into account both the crystal symmetries and possible offsets by integral factors of the crystal cell dimensions). This match is performed using only backbone atoms (i.e. C^{β} atoms are excluded even if present in the search fragment). Each match is further categorized according to whether it matches atom types in sequence (denoted by green circles), matches in a forward threaded direction, but out of sequence (the black squares), or matches a reverse threading (red triangles). A pose is considered a duplicate if it has an RMSD from a previous pose of less than 1.65 Å.

FFFEAR outputs the top 100 best scoring positions found, while our work lists as many as meet the cutoff criteria. In order to keep the results manageable, we only match the top fifty results. Duplicates are removed from that top fifty,

CCP4 fragment	FFFEAR	current work
emp-helix-9	942.13	10160.67
emp-helixend-9	945.93	9168.89
emp-strand-9	937.97	4431.27
emp-turn_a-9	958.45	8581.89
emp-turn_b-9	943.35	8904.51
ml-helix-9	951.34	10709.23
theor-helix-10	946.76	11041.73
theor-helix-5	952.59	6199.48
theor-strand-10	935.20	4498.60
theor-strand-5	958.09	5013.60

Table 3.2: User CPU times in seconds for searches against 1RU8

so the final number of poses plotted is often less than fifty. In the case of the current work, it is fairly common for few, if any, poses to score below the 0.5 level cutoff. In those cases, only the qualifying poses are shown (again, excluding duplicates). Each plot includes a line drawn at the 1.65 Å RMSD level in red. Roughly speaking, this can be considered the cutoff for "good" matches. We will discuss this in section 3.7.2 on the next page.

Put concisely, then, the graphs in appendix B show plots of accuracy of a pose as measured by RMSD versus the pose score for each method, fragment type and data set.

Table 3.2 shows the user times taken by each program, using searches in the data for 1RU8 as an example. FFFEAR runs in nearly the same amount of time, independent of fragment details. Our approach takes considerably longer, and varies with the longest linear dimension of the fragment.

3.7.2 Discussion

The data sets used in the comparisons were chosen only based on their resolution, without regard to other factors such as the most prominent secondary-structure motifs, model temperature factors, free R factors or any other detail. As can be quickly seen from the model images (Figures B.1 through B.10 on page 134), they include a variety of folds. The resolution range is around where current automated methods are not reliable, but within the spectrum of structures that one might expect to be solvable with good results. These tests, therefore, provide an appropriate and stringent measure of the efficacy of the feature recognition techniques examined.

To make an assessment of good versus poor fits, we, of course, need to define a measure. In describing their work on the application of simulated annealing to crystallographic refinement, Brünger and Rice state that the torsion angle method can correct backbone RMSD "of at least 1.65 Å".[96] Certainly there is no hard and fast rule, but we take this as an appropriate approximate indicator of good fits or "hits". A dashed line indicating the 1.65 Å RMSD mark is included in the comparison graphs in Appendix B.

The current work, like others, is the result of both a number of algorithmic choices and a selection of parameters. The "space" of possibilities to consider are beyond the capabilities of current computing systems to explore thoroughly. However, we believe we attained enough experience through testing to make good heuristic choices for the purposes of this investigation. It is notable that the determination of the run-time parameters was made before any comparisons to FFFEAR, using different test sets. It is particularly interesting to consider the structure of the cutoffs for refinement and their implications. It quickly became clear that,

with the current method, primary scores greater than about 0.5 meant that a pose was incorrect. There is some grey area with scores between about 0.5 and 0.35. A score of below 0.35 starts to reliably indicate a good match. Since the scoring function achieves reasonable discrimination, the question then becomes one of how to balance initial sampling with refinement in order to locate a high number of good final poses.

Using the parameters described in section 3.7.1 on page 114, the run of the CCP4 theor-strand-5 fragment against 1SQ4 (see Figure B.12 on page 136) sampled a total of 292420 trial poses, of which 121871 were refined (over 1/3), requiring 7112279 gradient calculations, meaning there were over 20 gradients for every trial pose or an average of roughly 58 gradients required for each refinement. Gradient calculations are much more expensive than an initial scoring calculation. Recall that, while the entire fragment grid density does not have to be recalculated for each gradient, a portion of it is, while the grid for the initial search need only be calculated once for each sampled orientation (150, in this case). In the test cases examined, FFFEAR sampled factors of between approximately 5000 to 40000 times more poses. Given run-times on the order of hundreds of minutes, it is apparent that to perform the same proportion of refinements while sampling as many points as FFFEAR, the current method could literally take years. In testing, we considered both increasing the sampling with a concomitant lowering of the cutoffs for refinement and decreased sampling with looser cutoffs for refinement. Our experience indicated that, in order to achieve a good number of hits using the more sampling/fewer refinements approach, the sampling had to be so fine as to be prohibitive with our method. The obvious point here is to describe why we chose the parameters described, but, as the comparisons will show, the more significant

point is that even the vastly denser sampling rate of FFFEAR often does not alone perform well.

With ten data sets and limiting ourselves to the nine standard CCP4 fragments, we examined 90 comparisons. We will not discuss them all, but rather a representative selection. Of the 90 comparisons, quite a few show no good hits for either method, but generally in cases where one would not expect any. For example, 1XQ4 is made up almost entirely of beta sheets and loops, with only very short helical segments. Figure B.11 on page 135 shows a fairly typical result¶. Examining the graph, which compares the results for a nine-segment helix template, we see that neither method finds a match with an RMSD of less than about 5 Å, in keeping with the lack of any long helices in the model. Both sets of scores are in ranges that, when compared to other results, can be said with confidence to indicate poor fits. These cases tell us nothing about the relative performance of the methods, except to say that neither produces blatantly incorrect results.

Of the remaining cases, the current method can be seen to perform equally well or better in all but one. Classification of the success of a search is somewhat heuristic, since we assume there must be a certain tolerance for false positives (putative matches with an RMSD greater than 1.65 Å). Examining the graphs, though, it is typically clear when a search is overwhelmed by false positives. The current method achieves more hits in many cases, for example with the five segment strand fragment against 1SQ4 (Figure B.12 on page 136). Typically the results using the empirical helix and strand fragments are similar to those of the theoretical ones, so the illustrations focus mostly on the latter. Other improved outcomes

¶This graph is shown in large format to more easily see what comparison is being made. The legend illustrated in the upper left area of this graph applies to all the others. Smaller graphs are included to further illustrate similar results.

are seen in searches using the plain helical fragments against 1SQ1 (Figures B.14 and B.15) and both the plain helices and the five segment strand against 1TM0 (Figures B.20, B.21 and B.22). Some contain only sparse (but sometimes surprising) hits, as in the case of the empirical "b" turn against 1YVK (Figure B.25 on page 141). In that case the current method gets only one good hit, but with a well separated score, whereas FFFEAR has two hits mixed in with two misplacements (too high a false positive rate). One such hit hardly constitutes greater success, but the case is noteworthy.

Tables B.1 and B.2 show the details of the top fifteen non-redundant matches from the results illustrated in Figures B.12 and B.18, respectively. The first two columns list the score and RMSD for the match. Column three tells if the fragment matched with the proper threading direction, while column four indicates whether the matched fragment and model atom types were the same (i.e. whether the fragment backbone and the model backbone aligned). This is the information represented in the graphs. The last columns of the tables list, from the PDB entries in the model, the atom number, atom type, three-letter residue code, chain ID and residue number of the first atom matched. Examining these entries demonstrates that the matches find hits in several non-overlapping parts of the proteins. That is to say, the hits shown in the graphs are not simply overlapping or near-duplicates in one location, although that does happen. Note this result holds generally for all cases, even when identifying positions in separate chains as equivalent. There is some duplication and overlap, but this occurs with FFFEAR too, although the degree to which this happens varies from case-to-case.

The one case in which FFFEAR finds noticeably better placements is also rather sparse. This is the case of the search of the five segment helix against 1SQ1

(Figure B.29 on page 142). In fact, neither method does particularly well, but FFFEAR does have two good, well distinguished hits.

Careful examination of the graphs suggests that FFFEAR does get the threading direction correct a higher proportion of the time. It is not surprising that the threading direction is somewhat troublesome, given the result by Kleywegt and Jones that demonstrated the ease with which an entire model might be reverse-threaded.[53,62] Obviously one could mitigate this problem by adopting FFFEAR's scoring as a final check. We presume there are other approaches that would guard against this problem, too, but that is outside the scope of this work.

3.8 Conclusion

The solution of protein structures remains a challenging effort, one that, in particular, is tedious yet takes the time and attention of a highly skilled scientist to carry out in many cases. The beginning steps in model building can be critical, especially if phase information from the model is integrated back into the original data in the iterative steps used by the standard protocols. We have presented a highly flexible method for performing initial feature searches. This method has certain novel aspects, particularly the use of a grid-based constrained minimization to obtain superior feature location. We showed that this method produces improved results over similar efforts, albeit at greater cost in computer time. In future work we plan to combine this method with the fast Fourier techniques such as those used in FFFEAR to perform the initial search, with the expectation that a denser initial search followed by minimization will benefit from the best aspects of each technique. Given that the minimization method can clearly converge hits despite relatively sparse sampling, we expect that we could use an initial search that is

simultaneously much denser than practical with the current method, but significantly sparser than that of FFFEAR. Since FFT methods scale as $\mathcal{O}(n \log n)$, this suggests the possibility of performing complete searches (including minimizations) in a matter of minutes per fragment (assuming a translational search at $\frac{1}{2}$ the linear density of FFFEAR), opening the way to performing feature recognition with a much larger library of fragments.

Appendix A

PSEUDOSPECTRAL METHOD COMPARISONS TO GAUSSIAN 92

Molecule	E(G92) (hartrees)	E(PS) (hartrees)	ΔE(PS−G92) (kcal/mole)
C_2H_2	−76.821835	−76.821825	0.0063
C_6H_6	−230.701680	−230.701660	0.0125
C_2H_4S	−475.525899	−475.525982	−0.0521
$C_3S_2H_4$	−910.814150	−910.814115	0.0220
C_4H_4	−153.634912	−153.634860	0.0326
CH_2PH	−380.209898	−380.209865	0.0207
CH_3Cl	−499.088628	−499.088617	0.0069
CH_3SH	−437.664129	−437.664034	0.0596
CH_3SiH_3	−330.279077	−330.279204	−0.0797
CH_2NH	−94.035705	−94.035658	0.0295
CH_3F	−139.038781	−139.038719	0.0389
CH_3CH_2OH	−154.089013	−154.088985	0.0176
H_2CO	−113.869736	−113.869687	0.0307
Glycylglycine	−489.550210	−489.549941	0.1688
Glutamine	−528.646741	−528.646595	0.0916
Glycine 0°	−282.844462	−282.844579	−0.0734
H_2O_2	−150.770782	−150.770599	0.1148
H_3^+	−1.293591	−1.293587	0.0025
HCN	−92.865967	−92.865995	−0.0176
H_2CS	−436.469855	−436.469858	−0.0019
H_2S_2	−796.177451	−796.177420	0.0195
H_3SiCl	−750.181166	−750.181154	0.0075
HCP	−379.106572	−379.106702	−0.0816
HOCl	−534.847156	−534.847147	0.0056
HOCN	−167.729020	−167.729334	−0.1970
CH_3OH	−115.045719	−115.045630	0.0558
CH_4	−40.201399	−40.201470	−0.0445
NH_2CHO	−168.937654	−168.937572	0.0515
NH_2F	−154.959172	−154.959114	0.0364
Porphine	−983.163305	−983.163186	0.0747
P_2H_4	−683.756972	−683.757103	−0.0822
S_3	−1192.441335	−1192.441380	−0.0282
SC_4H_4	−550.917166	−550.917146	0.0125
Si_2H_6	−581.311568	−581.311723	−0.0973
Si_3	−866.607046	−866.607057	−0.0069
Si_5	−1444.431461	−1444.431561	−0.0627
Si_6	−1733.362994	−1733.362841	0.0960
SiF_2	−487.862531	−487.862486	0.0282
SiH_2	−290.002560	−290.002566	−0.0038
SiH_3F	−390.145882	−390.145858	0.0151
SiH_4	−291.230804	−291.230822	−0.0113
SO_3	−621.980612	−621.980732	−0.0753
Tyrosine	−626.232318	−626.232230	0.0552
Uracil	−412.479477	−412.479089	0.2435
H_2O	−76.023615	−76.023596	0.0119

Table A.1: Absolute energy comparisons: 6-31G** basis. Hartree-Fock energies ($1\,\mathrm{kcal/mole} = 0.0016\,\mathrm{au}$). Default cutoffs used with GAUSSIAN 92.

Molecule	E(G92) (hartree)	$\Delta E^{(1)}$ (kcal/mole)	E(PS) (hartree)	$\Delta\Delta E^{(2)}$ (kcal/mole)
glycine 0°	-282.844462	—	-282.844579	—
glycine 150°	-282.841392	1.926	-282.841434	0.047
glycine 180°	-282.841665	1.755	-282.841680	0.064
biphenyl 0°	-460.266416	—	-460.266008	—
biphenyl 22.5°	-460.268991	-1.616	-460.268656	0.045
biphenyl 45.0°	-460.270672	-2.670	-460.270335	-0.045
biphenyl 67.5°	-460.269115	-1.693	-460.268682	-0.015
biphenyl 90.0°	-460.267927	-0.948	-460.267480	-0.024
diphenylether 30°-30°	-535.105210	—	-535.104526	—
diphenylether 40°-40°	-535.111927	-4.216	-535.111174	-0.045
diphenylether 50°-50°	-535.113118	-4.962	-535.112326	-0.068
diphenylether 60°-60°	-535.112484	-4.564	-535.111768	-0.020
diphenylether 70°-70°	-535.111329	-3.839	-535.110672	0.017
diphenylether 80°-80°	-535.110248	-3.161	-535.109641	0.048
diphenylether 90°-90°	-535.109773	-2.863	-535.109104	0.009

Table A.2: Relative energy comparisons: 6-31G** basis. Hartree-Fock energies ($1\,\mathrm{kcal/mole} = 0.0016\,\mathrm{au}$). (1) GAUSSIAN 92 energy differences calculated relative to the top listed energy for each method: the 0° conformers of glycine and biphenyl; the 30°-30° conformer of diphenyl ether. (2) Deviation of PS relative energies from the corresponding GAUSSIAN difference.

Molecule	Number of Basis Fcns	Workstation G92 Time	Workstation PS Time	Supercomputer G92 Time	Supercomputer PS Time
water	25	6.3	13.7	4.35	5.00
glycine 0°	100	187.1	172.9	40.88	24.91
uracil	140	542.3	340.9	76.42	36.93
glutamine	200	1400.5	770.6	186.55	68.84
tyrosine	250	2674.5	1207.8	311.02	104.90
porphine	430	9941.0	3683.9	948.25	275.82

Table A.3: User CPU time comparisons: 6-31G** basis. All times in user CPU seconds. Workstation is an IBM RS/6000 Model 580. Supercomputer is a Cray Y-MP C90. All calculations utilize direct SCF methods with symmetry explicitly turned off. Default cutoffs used with GAUSSIAN 92.

Molecule	Number of Basis Fcns	Supercomputer G92 Time	Supercomputer PS Time	ΔE (kcal/mole)
glycine 0°	170	270.3196	160.3925	-0.0471
uracil	236	728.8067	272.9761	0.0477
glutamine	340	2201.4424	533.4242	-0.1004
diphenylether 30°-30°	415	3615.1448	734.0660	0.0201
tyrosine	424	4322.0474	809.7973	-0.0998
porphine	726	15131.2271	2428.6439	0.1161

Table A.4: User cpu time comparisons: cc-pVTZ basis. All times are user CPU seconds. Supercomputer is a Cray Y-MP C90. All calculations utilize direct SCF methods with symmetry explictly turned off. Default cutoffs used in all cases.

Appendix B

FEATURE DETECTION: COMPARISONS OF CURRENT WORK AND FFFEAR

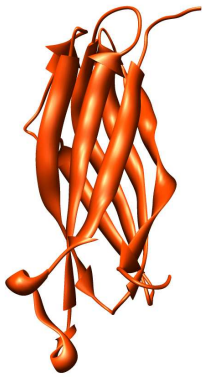


Figure B.1: PDB ID 1XQ4, NESG ID ber40 (chain A only)

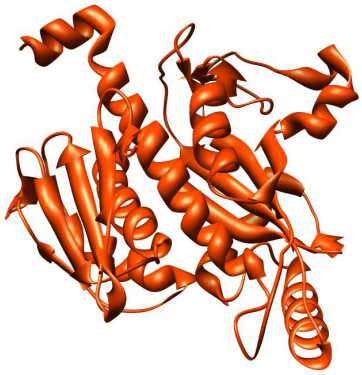


Figure B.2: PDB ID 1SQ1, NESG ID br19

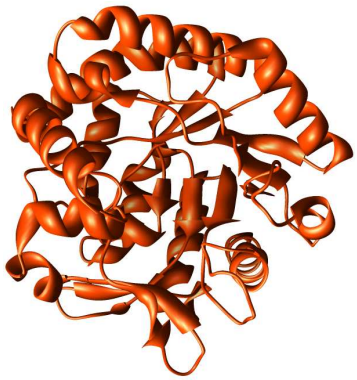


Figure B.3: PDB ID 1TZ9, NESG ID efr41 (chain A only)

Figure B.4: PDB ID 1TM0, NESG ID lr31 (chain A only)

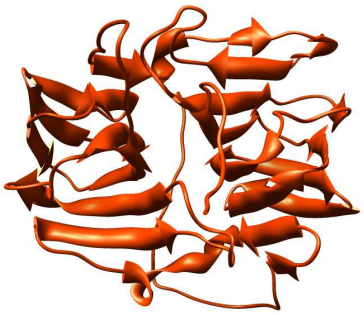


Figure B.5: PDB ID 1SQ4, NESG ID par14 (chain A only)

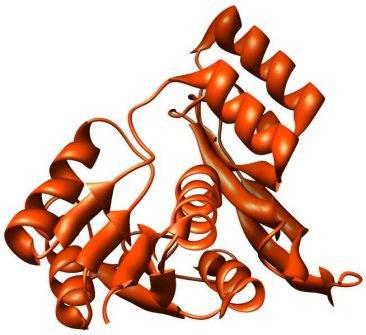


Figure B.6: PDB ID 1RU8, NESG ID pfr23 (chain A only)

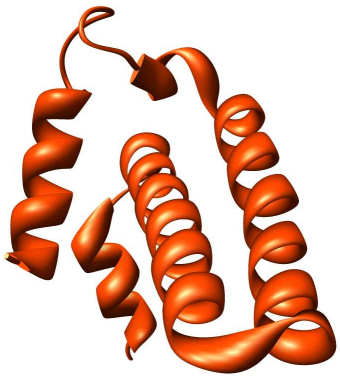


Figure B.7: PDB ID 1YXB, NESG ID rr8 (chain A only)

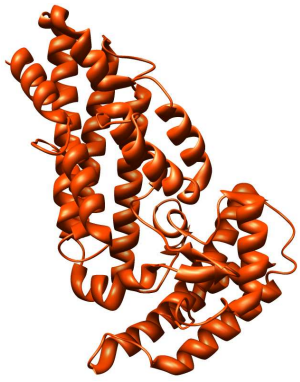


Figure B.8: PDB ID 1ZEE, NESG ID sor52 (chain A only)

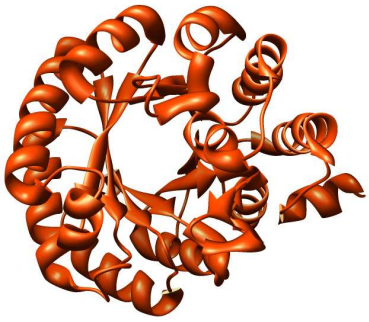


Figure B.9: PDB ID 1YDO, NESG ID sr181 (chain A only)

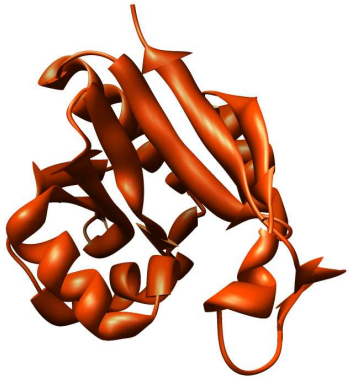


Figure B.10: PDB ID 1YVK, NESG ID sr237 (chain A only)

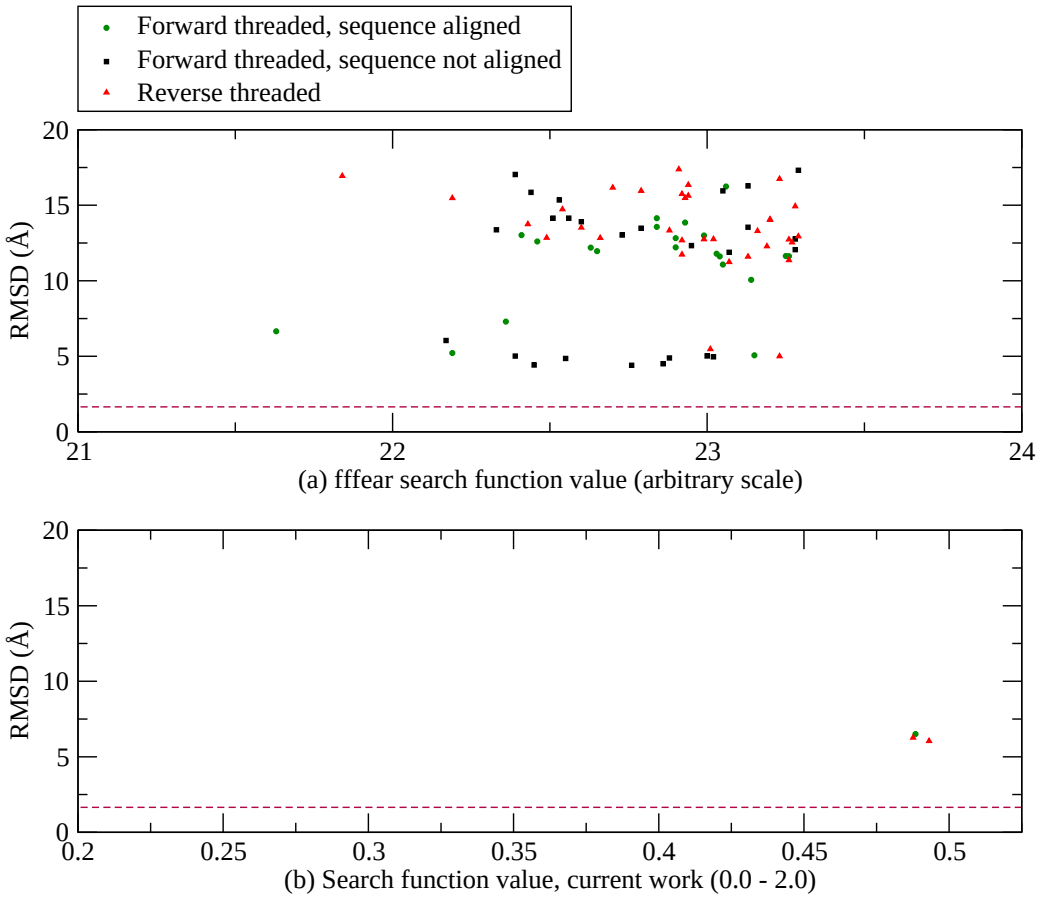


Figure B.11: Search results for the CCP4 emp-helix-9 fragment in a 3.00 Å resolution map of 1XQ4 . The dashed lines show the 1.65Å RMSD mark. Top 50 results each (redundancies removed). (a) Standard FFFEAR search. (b) Current approach, 150 angles sampled, $B = 50$.

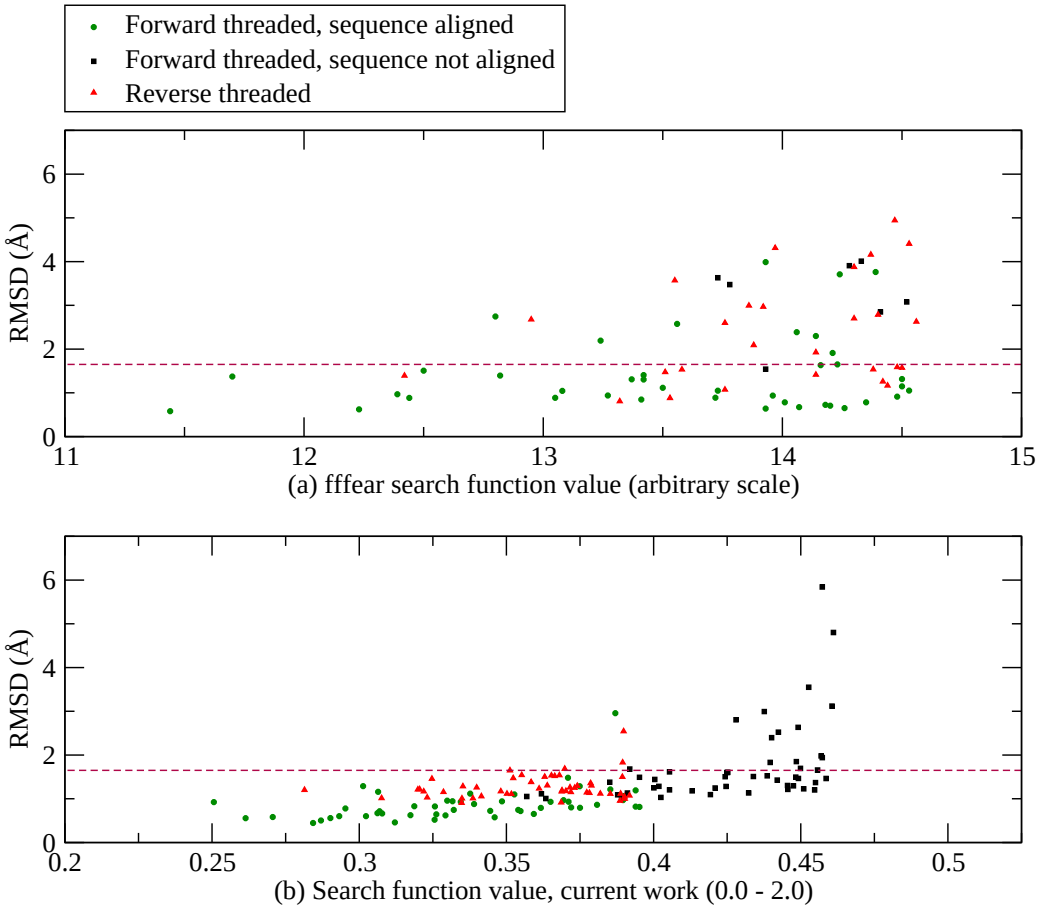


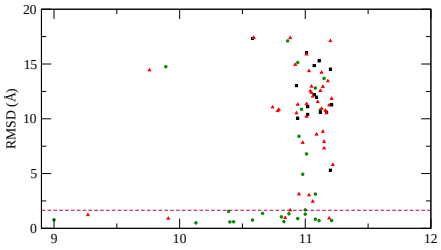
Figure B.12: Search results for the CCP4 theor-strand-5 fragment in a 2.70Å resolution map of 1SQ4 . The dashed lines show the 1.65Å RMSD mark. Top 50 results each (redundancies removed). (a) Standard FFFEAR search. (b) Current approach, 150 angles sampled, $B = 50$.

Score	RMSD	Forward Threaded	Sequence Aligned	PDB Entry				
0.251	0.92	Yes	Yes	4182	N	LEU	B	261
0.261	0.56	Yes	Yes	1652	N	HIS	A	214
0.271	0.58	Yes	Yes	2639	N	PHE	B	68
0.281	1.20	No	No	3059	O	ALA	B	124
0.284	0.45	Yes	Yes	1026	N	PHE	A	138
0.287	0.50	Yes	Yes	3171	N	PHE	B	138
0.290	0.56	Yes	Yes	677	N	ALA	A	92
0.293	0.60	Yes	Yes	1037	N	HIS	A	139
0.295	0.78	Yes	Yes	3610	N	ASP	B	190
0.301	1.29	Yes	Yes	2897	N	SER	B	102
0.302	0.60	Yes	Yes	3643	N	ASN	B	194
0.306	0.67	Yes	Yes	3061	N	ASP	B	125
0.306	1.16	Yes	Yes	286	N	THR	A	43
0.307	0.71	Yes	Yes	2502	N	ASN	B	51
0.308	1.01	No	No	854	CA	GLY	A	116

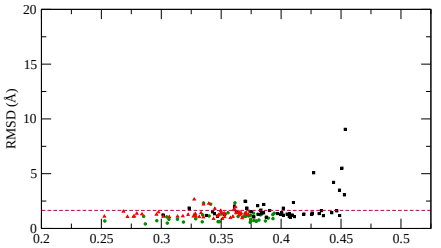
Table B.1: Details of the top fifteen search results for the CCP4 theor-strand-5 fragment using the data for 1SQ4. The entries correspond to the graph of the current work shown in Figure B.12.

Score	RMSD	Forward Threaded	Sequence Aligned	PDB Entry				
0.203	0.64	Yes	Yes	2592	N	ARG	A	336
0.203	0.35	Yes	Yes	5048	N	LEU	B	301
0.210	0.95	No	No	2592	N	ARG	A	336
0.214	1.04	No	No	5304	N	TYR	B	334
0.219	0.31	Yes	Yes	5379	N	ILE	B	343
0.220	0.33	Yes	Yes	5324	N	ARG	B	336
0.222	0.21	Yes	Yes	5034	N	ALA	B	299
0.223	0.74	Yes	Yes	2286	N	ASP	A	297
0.224	0.37	Yes	Yes	2302	N	ALA	A	299
0.226	0.45	Yes	Yes	1420	N	TYR	A	188
0.228	1.04	No	No	525	N	HIS	A	70
0.228	0.90	No	No	5367	N	TYR	B	342
0.228	1.05	No	No	2635	N	TYR	A	342
0.230	0.49	Yes	Yes	2635	N	TYR	A	342
0.231	0.41	Yes	Yes	5367	N	TYR	B	342

Table B.2: Details of the top fifteen search results for the CCP4 theor-helix-5 fragment using the data for 1TZ9. The entries correspond to the graph of the current work shown in Figure B.18.

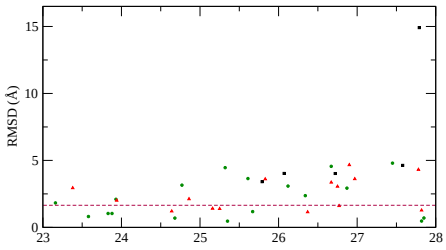


(a) fffear search function value (arbitrary scale)

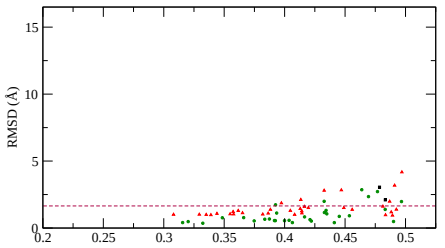


(b) Search function value, current work (0.0 - 2.0)

Figure B.13: 1XQ4 3.00Å CCP4 theor-strand-5

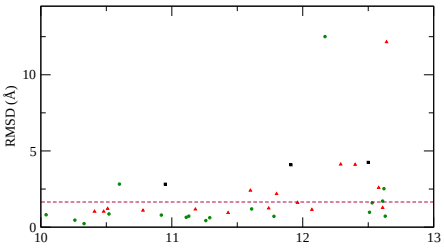


(a) fffear search function value (arbitrary scale)

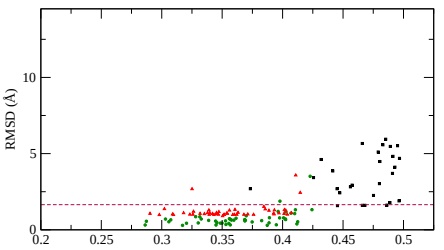


(b) Search function value, current work (0.0 - 2.0)

Figure B.14: 1SQ1 2.80Å CCP4 theor-helix-10

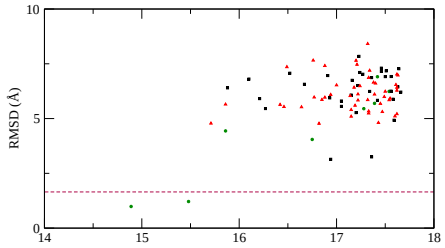


(a) fffear search function value (arbitrary scale)

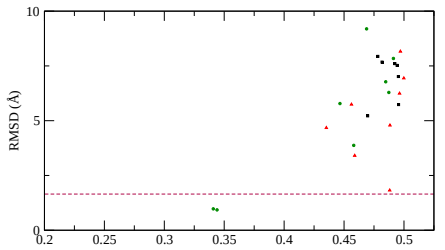


(b) Search function value, current work (0.0 - 2.0)

Figure B.15: 1SQ1 2.80Å CCP4 theor-helix-5

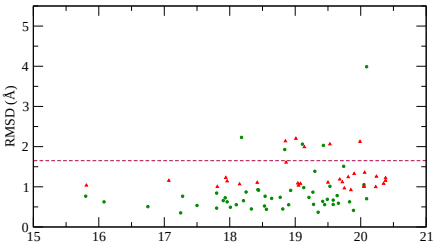


(a) fffear search function value (arbitrary scale)

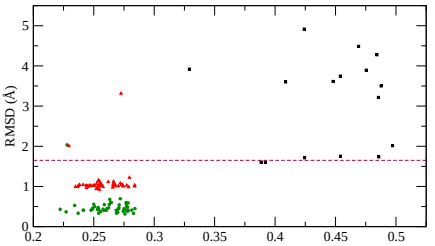


(b) Search function value, current work (0.0 - 2.0)

Figure B.16: 1TZ9 3.00Å CCP4 emp-turn_a-9

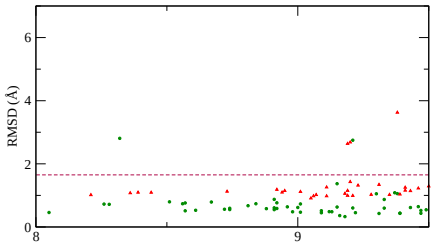


(a) fffear search function value (arbitrary scale)

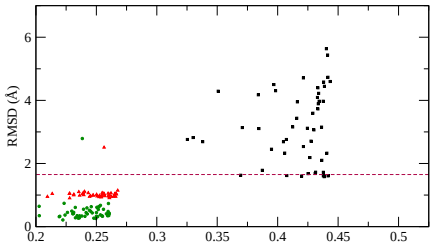


(b) Search function value, current work (0.0 - 2.0)

Figure B.17: 1TZ9 3.00Å CCP4 theor-helix-10

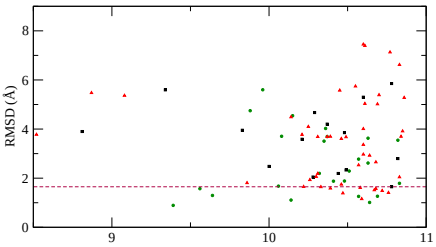


(a) fffear search function value (arbitrary scale)

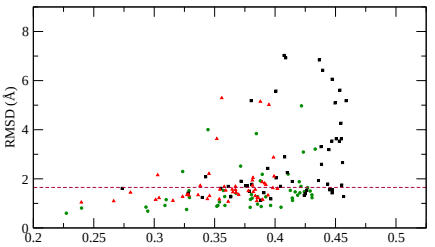


(b) Search function value, current work (0.0 - 2.0)

Figure B.18: 1TZ9 3.00Å CCP4 theor-helix-5

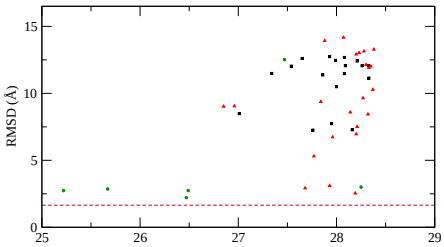


(a) fffear search function value (arbitrary scale)

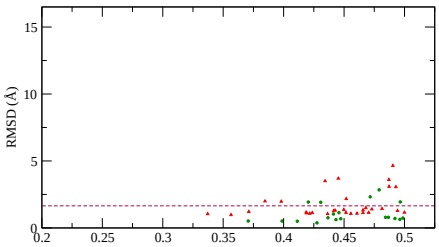


(b) Search function value, current work (0.0 - 2.0)

Figure B.19: 1TZ9 3.00Å CCP4 theor-strand-5

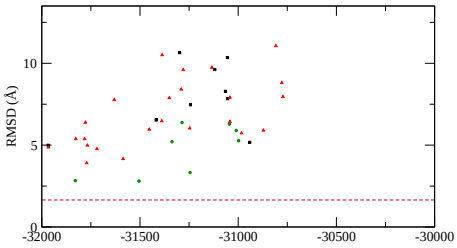


(a) fffear search function value (arbitrary scale)

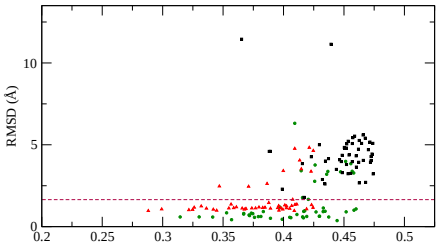


(b) Search function value, current work (0.0 - 2.0)

Figure B.20: 1TM0 2.84Å CCP4 theor-helix-10

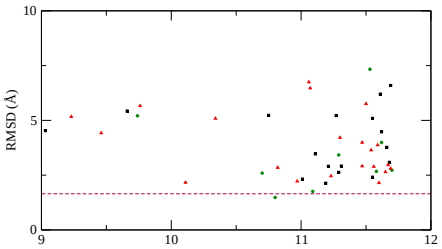


(a) fffear search function value (arbitrary scale)

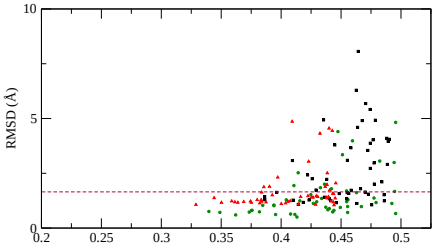


(b) Search function value, current work (0.0 - 2.0)

Figure B.21: 1TM0 2.84Å CCP4 theor-helix-5

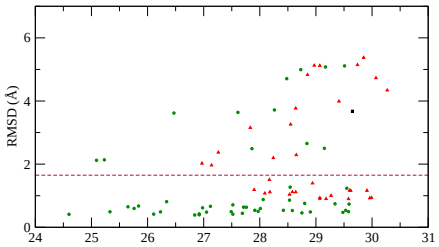


(a) fffear search function value (arbitrary scale)

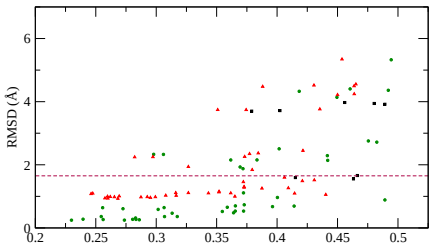


(b) Search function value, current work (0.0 - 2.0)

Figure B.22: 1TM0 2.84Å CCP4 theor-strand-5

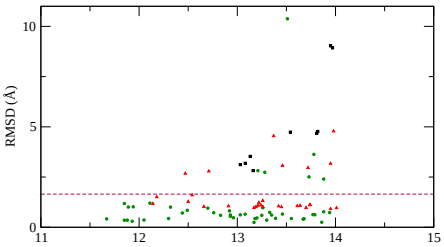


(a) fffear search function value (arbitrary scale)

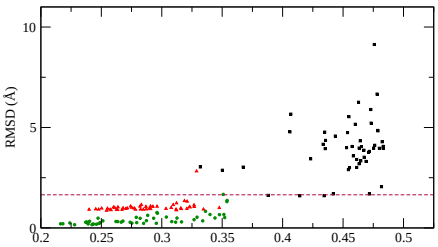


(b) Search function value, current work (0.0 - 2.0)

Figure B.23: 1RU8 2.70Å CCP4 theor-helix-10

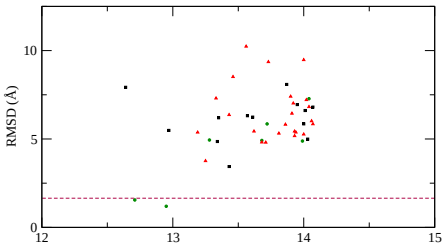


(a) fffear search function value (arbitrary scale)

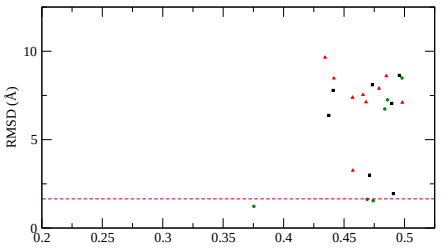


(b) Search function value, current work (0.0 - 2.0)

Figure B.24: 1RU8 2.70Å CCP4 theor-helix-5

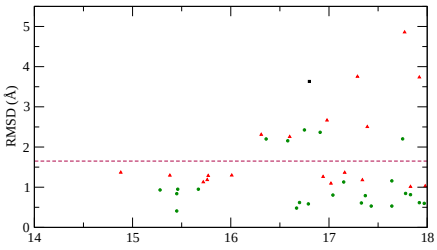


(a) fffear search function value (arbitrary scale)

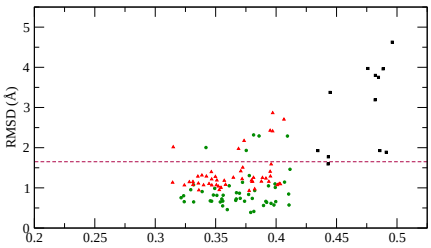


(b) Search function value, current work (0.0 - 2.0)

Figure B.25: 1YVK 3.20Å CCP4 emp-turn_b-9

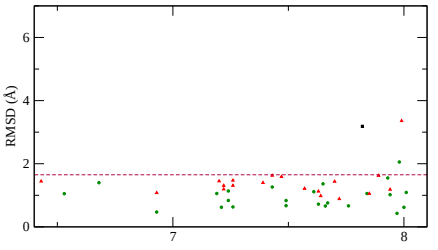


(a) fffear search function value (arbitrary scale)

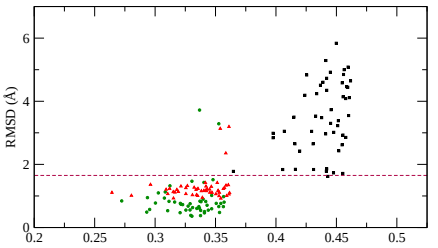


(b) Search function value, current work (0.0 - 2.0)

Figure B.26: 1YVK 3.20Å CCP4 theor-helix-10

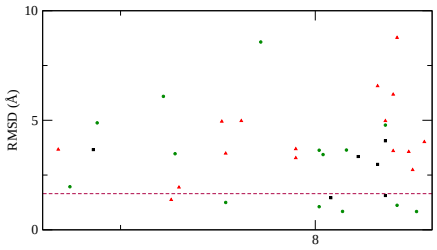


(a) fffear search function value (arbitrary scale)

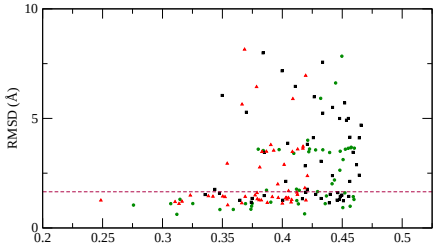


(b) Search function value, current work (0.0 - 2.0)

Figure B.27: 1YVK 3.20Å CCP4 theor-helix-5



(a) fffear search function value (arbitrary scale)



(b) Search function value, current work (0.0 - 2.0)

Figure B.28: 1YVK 3.20Å CCP4 theor-strand-5

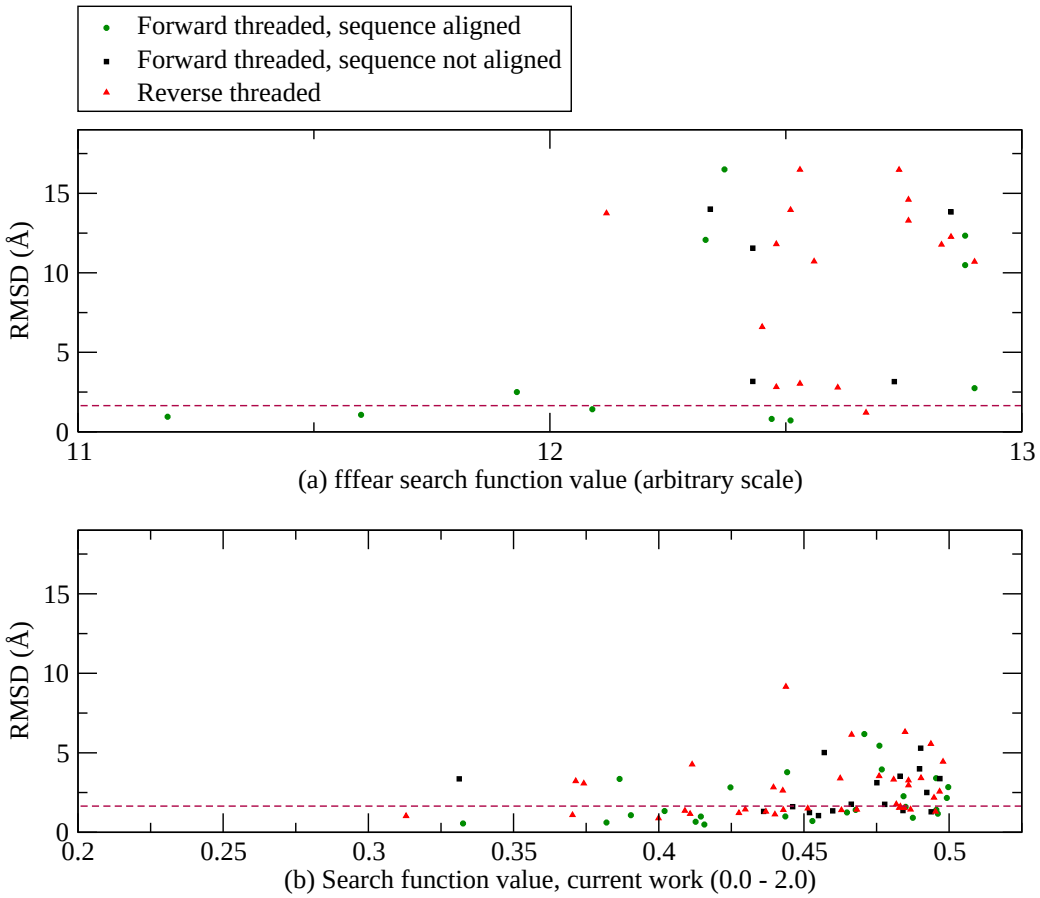


Figure B.29: Search results for the CCP4 theor-strand-5 fragment in a 2.80Å resolution map of 1SQ1 . The dashed lines show the 1.65Å RMSD mark. Top 50 results each (redundancies removed). (a) Standard FFFEAR search. (b) Current approach, 150 angles sampled, $B = 50$.

REFERENCES

1. E. F. Pettersen, T. D. Goddard, C. C. Huang, G. S. Couch, D. M. Greenblatt, E. C. Meng, and T. E. Ferrin, "UCSF Chimera - A Visualization System for Exploratory Research and Analysis," *J. Comput. Chem.* **25**(13), 1605 (2004).

2. H. A. Stern, F. Rittner, B. J. Berne, and R. A. Friesner, "Combined fluctuating charge and polarizable dipole models: Application to a five-site water potential function," *J. Chem. Phys.* **115**(5), 2237 (2001).

3. G. A. Kaminski, R. A. Friesner, J. Tirado-Rives, and W. L. Jorgensen, "Evaluation and reparameterization of the OPLS-AA force field for proteins via comparison with accurate quantum chemical calculations on peptides," *J. Phys. Chem. B* **105**, 6474 (2001).

4. G. A. Kaminski, H. A. Stern, B. J. Berne, R. A. Friesner, Y. X. Cao, R. B. Murphy, R. Zhou, and T. A. Halgren, "Development of a Polarizable Force Field for Proteins via *Ab Initio* Quantum Chemistry: First Generation Model and Gas Phase Tests," *J. Comput. Chem.* **23**, 1515 (2002).

5. M. D. Beachy, D. Chasman, R. B. Murphy, T. A. Halgren, and R. A. Friesner, "Accurate *ab initio* quantum chemical determination of the relative energetics of peptide conformations and assessment of empirical force fields," *J. Am. Chem. Soc.* **119**, 5908 (1997).

6. D. J. Tannor, B. Marten, R. Murphy, R. A. Friesner, D. Sitkoff, A. Nicholls, M. Ringalda, W. A. Goddard, and B. Honig, “Accurate First Principles Calculation of Molecular Charge Distributions and Solvation energies from ab initio quantum mechanics and continuum dielectric theory,” *J. Am. Chem. Soc.* **116**, 11875 (1994).

7. B. Marten, K. Kim, C. Cortis, R. A. Friesner, R. B. Murphy, M. N. Ringnalda, D. Sitkoff, and B. Honig, “New Model for Calculation of Solvation Free Energies: Correction of Self-Consistent Reaction Field Continuum Dielectric Theory for Short-Range Hydrogen-Bonding Effects,” *J. Chem. Phys.* **100**, 11775 (1996).

8. D. M. Philipp and R. A. Friesner, “Mixed *Ab Initio* QM/MM Modeling Using Frozen Orbitals and Tests with Alanine Dipeptide and Tetrapeptide,” *J. Comput. Chem.* **20**(14), 1468 (1999).

9. R. B. Murphy, D. M. Philipp, and R. A. Friesner, “A Mixed Quantum Mechanics/-Molecular Mechanics (QM/MM) Method for Large-Scale Modeling of Chemistry in Protein Environments,” *J. Comput. Chem.* **21**(16), 1442 (2000).

10. R. A. Friesner, “Solution of Self-Consistent Field Electronic Structure Equations by a Pseudospectral Method,” *Chem. Phys. Lett.* **116**, 39 (1985).

11. R. A. Friesner, “Solution of the Hartree-Fock Equations by a Pseudospectral Method: Application to Diatomic Molecules,” *J. Chem. Phys.* **85**(3), 1462 (1986).

12. R. A. Friesner, “Solution of the Hartree-Fock Equations for Polyatomic Molecules by a Pseudospectral Method,” *J. Chem. Phys.* **86**(6), 3522 (1987).

13. R. A. Friesner, “An Automatic Grid Generation Scheme for Pseudospectral Self-Consistent Field Calculations on Polyatomic Molecules,” *J. Phys. Chem.* **92**, 3091 (1988).

14. M. N. Ringnalda, M. Belhadj, and R. A. Friesner, “Pseudospectral Hartree-Fock theory: Applications and algorithmic improvements,” *J. Chem. Phys.* **93**(5), 3397 (1990).

15. M. N. Ringnalda, Y. Won, and R. A. Friesner, “Pseudospectral Hartree-Fock calculations on glycine,” *J. Chem. Phys.* **92**(2), 1163 (1990).

16. Y. Won, J.-G. Lee, M. N. Ringnalda, and R. A. Friesner, “Pseudospectral Hartree-Fock gradient calculations,” *J. Chem. Phys.* **94**(12), 8152 (1991).

17. S. Obara and A. Saika, “Efficient recursive computation of molecular integrals over Cartesian Gaussian functions,” *J. Chem. Phys.* **84**(7), 3963 (1986).

18. P. M. W. Gill, M. Head-Gordon, and J. A. Pople, “Efficient Computation of Two-Electron-Repulsion Integrals and Their nth-Order Derivatives Using Contracted Gaussian Basis Sets,” *J. Phys. Chem.* **94**, 5564 (1990).

19. D. R. Hartree, “The wave mechanics of an atom with a non-Coulomb central field. Part I. Theory and methods,” *Proc. Cambridge Phil. Soc.* **24**, 89 (1928).

20. V. Fock, “Näherungsmethode zur Lösung des quantenmechanischen Mehrkörperproblems,” *Zeitschrift für Physik* **61**, 126 (1930).

21. J. C. Slater, “Note on Hartree’s Method,” *Phys. Rev.* **35**(2), 210 (1930).

22. C. C. J. Roothaan, “New Developments in Molecular Orbital Theory,” *Rev. Mod. Phys.* **23**(2), 69 (1951).

23. G. G. Hall, “The Molecular Orbital Theory of Chemical Valency. VIII. A Method of Calculating Ionization Potentials,” *Proc. R. Soc. London A* **205**(1083), 541 (1951).

24. R. B. Murphy, R. A. Friesner, M. N. Ringnalda, and W. A. Goddard, III, "Pseudospectral contracted configuration interaction from a generalized valence bond reference," *J. Chem. Phys.* **101**(4), 2986 (1994).

25. R. B. Murphy, M. D. Beachy, R. A. Friesner, and M. N. Ringnalda, "Pseudospectral localized Møller-Plesset methods: Theory and calculation of conformational energies," *J. Chem. Phys.* **103**(4), 1481 (1995).

26. R. B. Murphy, W. T. Pollard, and R. A. Friesner, "Pseudospectral localized generalized Møller-Plesset methods with a generalized valence bond reference wave function: Theory and calculation of conformational energies," *J. Chem. Phys.* **106**(12), 5073 (1997).

27. R. B. Murphy, Y. Cao, M. D. Beachy, M. N. Ringnalda, and R. A. Friesner, "Efficient pseudospectral methods for density functional calculations," *J. Chem. Phys.* **112**, 10131 (2000).

28. A. Szabo and N. S. Ostlund, *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*, (Dover Publications, Inc., Mineola) (1996).

29. S. A. Orszag, "Numerical simulation of incompressible flows within simple boundaries. I (Galerkin method for numerical simulation of incompressible boundary flows in box geometries with periodic and free slip conditions, noting Taylor-Green vortex decay)," *Stud. Appl. Math.* **50**, 293 (1971).

30. S. A. Orszag, "Comparison of pseudospectral and spectral approximations." *Stud. Appl. Math.* **51**, 253 (1972).

31. S. A. Orszag and A. T. Patera, "Subcritical Transition to Turbulence in Plane Channel Flows," *Phys. Rev. Lett.* **45**, 989 (1980).

32. H. A. Bethe and R. W. Jackiw, *Intermediate Quantum Mechanics*, (Benjamin/Cummings, Reading, Massachusetts), 2nd edition (1968).

33. S. F. Boys, "Electronic Wave Functions. I. A General Method of Calculation for the Stationary States of Any Molecular System," *Proc. R. Soc. London A* **Vol. 200**(1063), 542 (1950).

34. T. H. Dunning, Jr., "Gaussian Basis Functions for Use in Molecular Calculations. I. Contraction of (9s5p) Atomic Basis Sets for the First-Row Atoms," *J. Chem. Phys.* **53**, 2823 (1970).

35. T. H. Dunning, Jr., "Gaussian basis sets for use in correlated molecular calculations. III. Contraction of (10s6p) Atomic Basis Sets for the First-Row Atoms," *J. Chem. Phys.* **55**, 716 (1971).

36. T. H. Dunning, Jr., "Gaussian basis sets for use in correlated molecular calculations. I. The atoms boron through neon and hydrogen," *J. Chem. Phys.* **90**(2), 1007 (1989).

37. L. E. McMurchie and E. R. Davidson, "One- and two-electron integrals over Cartesian Gaussian functions," *J. Comput. Phys.* **26**, 218 (1978).

38. R. A. Kendall, T. H. Dunning, Jr., and R. J. Harrison, "Electron affinities of the first-row atoms revisited. Systematic basis sets and wave functions," *J. Chem. Phys.* **96**, 6796 (1992).

39. C. Branden and J. Tooze, *Introduction to Protein Structure*, (Garland Publishing, Inc., New York, NY), 2nd edition (1999).

40. "Research Collaboratory for Structural Bioinformatics PDB," [Online] (2000), Available at `http://www.pdb.org/`.

41. H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, "The Protein Data Bank," *Nucleic Acids Res.* **28**, 235 (2000).

42. H. M. Berman, K. Henrick, and H. Nakamura, "Announcing the worldwide Protein Data Bank," *Nature Struct. Biol.* **10**(12), 980 (2003).

43. J. Drenth, *Principles of Protein X-Ray Crystallography*, (Springer, New York), 2nd edition (1999).

44. G. Rhodes, *Crystallography Made Crystal Clear*, (Academic Press, San Diego), 2nd edition (2000).

45. G. Taylor, "The phase problem," *Acta Cryst.* **D59**, 1881 (2003).

46. W. L. Bragg, "The Diffraction of Short Electromagnetic Waves by a Crystal," *Proc. Cambridge Phil. Soc.* **17**, 43 (1912).

47. L. D. Landau, E. M. Lifshitz, and L. P. Pitaevskiĭ, *Electrodynamics of Continuous Media*, (Pergamon Press, New York), 2nd edition (1984).

48. I. Dobrianov, C. Caylor, S. G. Lemay, K. D. Finkelstein, and R. E. Thorne, "X-ray diffraction studies of protein crystal disorder," *J. Cryst. Growth* **196**, 511 (1999).

49. K. Y. J. Zhang, K. D. Cowtan, and P. Main, "Phase improvement by iterative density modification," *International Tables of Crystallography* **F**, 311 (2001).

50. M. A. DePristo, P. I. W. de Bakker, and T. L. Blundell, "Heterogeneity and Inaccuracy in Protein Structures Solved by X-Ray Crystallography," *Structure* **12**, 831 (2004).

51. A. T. Brünger, "Free R value: a novel statistical quantity for assessing the accuracy of crystal structures," *Nature* **355**, 472 (1992).

52. A. T. Brünger, “Assessment of Phase Accuracy by Cross Validation: the Free R Value. Methods and Applications,” *Acta Cryst.* **D49**, 24 (1993).

53. G. J. Kleywegt and T. A. Jones, “Where freedom is given, liberties are taken,” *Structure* **3**, 535 (1995).

54. C. I. Bränden and T. A. Jones, “Between objectivity and subjectivity,” *Nature* **343**, 687 (1990).

55. A. Hodel, S.-H. Kim, and A. T. Brünger, “Model Bias in Macromolecular Crystal Structures,” *Acta Cryst.* **A48**, 851 (1992).

56. A. L. U. Roberts and A. T. Brünger, “Phase Improvement by Cross-Validated Density Modification,” *Acta Cryst.* **D51**, 990 (1995).

57. G. J. Kleywegt and A. T. Brünger, “Checking your imagination: applications of the free R value,” *Structure* **4**, 897 (1996).

58. V. S. Lamzin and A. Perrakis, “Current state of automated crystallographic data analysis,” *Nature Struct. Biol.* **Supp.**, 978 (2000).

59. Collaborative Computational Project, Number 4, “The CCP4 Suite: Programs for Protein Crystallography,” *Acta Cryst.* **D50**, 760 (1994).

60. *Proceedings of the CCP4 study weekend: Model building and refinement*, volume 60 of *Acta Crystallographica Section D* (2004).

61. J. Greer, “Three-dimensional Pattern Recognition: An Approach to Automated Interpretation of Electron Density Maps of Proteins,” *J. Mol. Biol.* **82**, 279 (1974).

62. T. A. Jones, J.-Y. Zou, S. W. Cowan, and M. Kjeldgaard, “Improved Methods for Building Protein Models in Electron Density Maps and the Location of Errors in these Models,” *Acta Cryst.* **A47**, 110 (1991).

63. D. G. Levitt, “A new software routine that automates the fitting of protein X-ray crystallographic electron-density maps,” *Acta Cryst.* **D57**, 1013 (2001).

64. T. J. Oldfield, “Pattern-recognition methods to identify secondary structure within X-ray crystallographic electron-density maps,” *Acta Cryst.* **D58**, 487 (2002).

65. T. J. Oldfield, “Automated tracing of electron-density maps of proteins,” *Acta Cryst.* **D59**, 483 (2003).

66. T. Holton, T. R. Ioerger, J. A. Christopher, and J. C. Sacchettini, “Determining protein structure from eletron-density maps using pattern matching,” *Acta Cryst.* **D56**, 722 (2000).

67. T. R. Ioerger and J. C. Sacchettini, “Automatic modeling of protein backbones in electron-density maps via prediction of C^{α} coordinates,” *Acta Cryst.* **D58**, 2043 (2002).

68. G. J. Kleywegt and T. A. Jones, “Template Convolution to Enhance or Detect Structural Features in Macromolecular Electron-Density Maps,” *Acta Cryst.* **D53**, 179 (1997).

69. K. Cowtan, “Modified Phased Translation Functions and their Application to Molecular-Fragment Location,” *Acta Cryst.* **D54**, 750 (1998).

70. K. Cowtan, “Fast Fourier feature recognition,” *Acta Cryst.* **D57**, 1435 (2001).

71. K. Cowtan, “The Buccaneer software for automated model building,” *Acta Cryst.* **D62**, 1002 (2006).

72. B. Rupp, *Structural Genomics and High Throughput Structural Biology*, (CRC, Boca Raton, Florida) (2006).

73. R. J. Morris, “Statistical pattern recognition for macromolecular crystallographers,” *Acta Cryst.* **D60**, 2133 (2004).

74. J. Badger, “An evaluation of automated model-building procedures for protein crystallography,” *Acta Cryst.* **D59**, 823 (2003).

75. R. H. Byrd, P. Lu, and J. Nocedal, “A Limited Memory Algorithm for Bound Constrained Optimization,” *SIAM J. Scien. Stat. Comput.* **16**(6), 1190 (1995).

76. C. Y. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, “Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization,” *ACM Trans. Math. Softw.* **23**(4), 550 (1997).

77. R. C. Agarwal, “A New Least-Squares Refinement Technique Based on the Fast Fourier Transform Algorithm,” *Acta Cryst.* **A34**, 791 (1978).

78. D. T. Cromer and J. T. Waber, *International Tables for X-ray Crystallography, Vol. IV*, (The Kynoch Press, Birmingham, England) (1974).

79. B. K. P. Horn, “Some Notes on Unit Quaternions and Rotation,” [Online] (2001), Available at `http://people.csail.mit.edu/bkph/articles/Quaternions.pdf`.

80. J. Matthews and R. L. Walker, *Mathematical Methods of Physics*, (Benjamin/Cummings, Menlo Park), 2nd edition (1970).

81. J. J. Kuffner, “Effective Sampling and Distance Metrics for 3D Rigid Body Path Planning,” in *Proc. IEEE International Conference on Robotics and Automation (ICRA 2004)*, pp. 3393–3398 (2004).

82. J. Nocedal, “Updating Quasi-Newton Matrices with Limited Storage,” *Math. Comp.* **35**(151), 773 (1980).

83. D. C. Liu and J. Nocedal, “On the Limited Memory Method for Large Scale Optimization,” *Math. Prog. B* **45**(3), 503 (1989).

84. A. R. Conn, N. Gould, A. Sartenaer, and P. L. Toint, “Convergence of an Augmented Lagrangian Algorithm for Optimization with a Combination of General Equality and Linear Constraints,” *SIAM J. Optim.* **6**(2), 674 (1996).

85. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, (Cambridge University Press, Cambridge, England), 2nd edition (1992).

86. E. A. Coutsias and L. Romero, “The Quaternions with an application to Rigid Body Dynamics,” Technical Report SAND2004-0153, Sandia National Laboratories (2004).

87. E. B. Saff and A. Kuijlaars, “Distributing many points on the sphere,” *Math. Intelligencer* **19**, 5 (1997).

88. D. Rusin, “Frequently-Asked Questions about Spheres,” [Online] (1998), Available at `http://www.math.niu.edu/~rusin/known-math/index/spheres.html`.

89. A. Yershova and S. M. LaValle, “Deterministic Sampling Methods for Spheres and SO(3),” in *Proc. IEEE International Conference on Robotics and Automation* (2004).

90. A. Yershova, “Library for Generating Deterministic Sequences of Samples Over SO(3),” [Online] (), Available at `http://msl.cs.uiuc.edu/~yershova/so3sampling/sampling.tar.gz`.

91. S. R. Lindemann and S. M. LaValle, “Incremental Low-Discrepancy Lattice Methods for Motion Planning,” in *Proceedings IEEE International Conference on Robotics and Automation*, pp. 2920–2927 (2003).

92. “Northeast Structural Genomics Consortium,” [Online] (Accessed July 29, 2006), Available at `http://www.nesg.org/`.

93. “FFFEAR fragment library 1.0 documentation,” [Online] (Accessed July 29, 2006), Available at `http://www.ccp4.ac.uk/dist/html/fffear_fraglib.html`.

94. “FFFEAR (CCP4: Supported Program) documentation,” [Online] (Accessed July 29, 2006), Available at `http://www.ccp4.ac.uk/dist/html/fffear.html`.

95. “FFT (CCP4: Supported Program) documentation,” [Online] (Accessed July 29, 2006), Available at `http://www.ccp4.ac.uk/dist/html/fft.html`.

96. A. T. Brünger and L. M. Rice, “Crystallographic refinement by simulated annealing: methods and applications,” *Methods Enzymol.* **277**, 243 (1997).

ABOUT THE AUTHOR

Burnham H. (Hod) Greeley was born in Honolulu, Hawaii on October 4, 1962. He attended the California Institute of Technology and graduated with a degree in physics in June, 1986.