RICE UNIVERSITY

A Reduced Basis Method for Molecular Dynamics Simulation

by

Rachel Elisabeth Vincent-Finley

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

Doctor of Philosophy

Approved, Thesis Committee:

Danny C. Sorensen, Chairman
Noah G. Harding Professor of
Computational and Applied Mathematics

Steven J. Cox
Professor of Computational and Applied
Mathematics

Yin Zhang
Professor of Computational and Applied
Mathematics

B. Montgomery Pettitt
Hugh Roy and Lille Cranz Cullen
Distinguished Professor of Chemistry
University of Houston

Houston, Texas

February, 2005

Abstract

A Reduced Basis Method for Molecular Dynamics Simulation

by

Rachel Elisabeth Vincent-Finley

In this dissertation, we develop a method for molecular simulation based on principal component analysis (PCA) of a molecular dynamics trajectory and least squares approximation of a potential energy function. Molecular dynamics (MD) simulation is a computational tool used to study molecular systems as they evolve through time. With respect to protein dynamics, local motions, such as bond stretching, occur within femtoseconds, while rigid body and large-scale motions, occur within a range of nanoseconds to seconds. To capture motion at all levels, time steps on the order of a femtosecond are employed when solving the equations of motion and simulations must continue long enough to capture the desired large-scale motion.

To date, simulations of solvated proteins on the order of nanoseconds have been reported. It is typically the case that simulations of a few nanoseconds do not provide adequate information for the study of large-scale motions. Thus, the development of

techniques that allow longer simulation times can advance the study of protein function and dynamics. In this dissertation we use principal component analysis (PCA) to identify the dominant characteristics of an MD trajectory and to represent the coordinates with respect to these characteristics. We augment PCA with an updating scheme based on a reduced representation of a molecule and consider equations of motion with respect to the reduced representation. We apply our method to butane and BPTI and compare the results to standard MD simulations of these molecules. Our results indicate that the molecular activity with respect to our simulation method is analogous to that observed in the standard MD simulation with simulations on the order of picoseconds.

Acknowledgements

I would like to thank my advisor, Dr. Dan Sorensen, for supporting this research and my advisor, Dr. B. Montgomery Pettitt, for providing wonderful insight into this intriguing application. I thank them both for patiently encouraging me through the research process. I would like to acknowledge and thank my committee members, Dr. Steven Cox and Dr. Yin Zhang, for their interest in this research, and my professor, Dr. Richard Tapia, for offering advice and insight about research and life.

I thank all of my colleagues in the CAAM department who helped to ease my transition into graduate school. Their experience, tips on studying and research, collaboration and friendship assisted me in navigating through my graduate requirements and research.

I would like to acknowledge the late Michael Pearlman, the former systems administrator for CAAM and STAT. Michael went beyond the call of duty to make computing in CAAM and STAT a pleasant experience. I thank the current and former staff of CAAM - Daria Lawrence, Fran Moshiri, Ginger Wright, Ivy Gonzalez, Margaret Poon, and Eric Aune - for helping make the administrative details of graduate school run smoothly.

I would like to thank Dr. Gillian Lynch and Dr. Brian Beck from the Institute for Molecular Design at the University of Houston for sharing their knowledge of scientific computing and answering countless questions about computer code.

I would like to thank my mathematics professors at Bryn Mawr College, for cultivating my interests in mathematics and for encouraging me to pursue graduate degrees in mathematics. I extend special thanks to Dr. Rhonda Hughes and Dr. Danielle Carr for introducing me to scientific computing and maintaining contact with me through my graduate career. I thank the Mellon Foundation for supporting my academic interests through the former Mellon Minority Undergraduate Fellowship program during my tenure at Bryn Mawr College.

I thank my parents, siblings, grandparents, uncles, aunts and extended family for their encouragement and prayers throughout my academic endeavors. I thank my family at Lilly Grove Baptist Church for making me feel at home away from home, especially the Mickey, Ervin and Leadon families.

Finally, I am extremely grateful to my husband, Stephen, for believing in me and supporting me through my research and writing challenges. He wiped away my tears of frustration, and encouraged me to persevere and put in the long hours necessary to complete this work. I am forever grateful to him for being my love, my partner, and for bringing into my life a loving extended family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"The purpose of computing is insight, not numbers." - Richard Hamming

Molecular dynamics (MD) simulation is a powerful tool used to study molecular motion with respect to classical mechanics. With respect to protein dynamics, local motions, such as bond stretching, occur within femtoseconds, while rigid body and large-scale motions, occur within a range of nanoseconds to seconds. To capture motion at all levels, time steps on the order of a femtosecond are employed when solving the equations of motion and simulations must continue long enough to capture the desired large-scale motion. To date, simulations of solvated proteins on the order of nanoseconds have been reported, however, it is typically the case that simulations of this length do not provide adequate sampling for the study of large-scale motion.

In this dissertation we develop a method for performing molecular simulations with respect to a kD coordinate space. Given a standard MD trajectory we use

principal component analysis (PCA) to identify the k dominant characteristics of the trajectory and construct a kD representation of the atomic coordinates with respect to these k characteristics. We then construct a radial basis function (RBF) network based on a kD representation of the trajectory and the values of the potential function evaluated at the conformations in the trajectory. The RBF network provides a model of the molecules potential energy surface. Using this model we construct equations of motion and perform simulations with respect to the constructed kD representation.

We apply our method to butane and BPTI and compare the simulations to standard MD simulations of these molecules. Our results indicate that the molecular activity with respect to our simulation method is analogous to that observed in the standard MD simulation with simulations on the order of picoseconds. The reduced representation of the molecule and potential energy surface allows us to efficiently simulate test molecules by reducing the storage required to represent the molecules and the computation required to simulate the motion of the molecules.

This area of research promises to move classical MD beyond its time step and storage limitations. Previous MD research has focused on reducing the complexity of a molecular system by dividing internal coordinates two classes- coordinates that are deemed important to overall dynamics and coordinates that are negligible (see [2], [3], [8], [18], [56], [66], [67]). The important coordinates then undergo further analysis.

One of the earliest reported discussions of reduced coordinates was presented by Gō and Scheraga [25], who made use of the standard set of internal coordinates -

bond lengths, bond angles and dihedral angles - as well as external variables (overall translation and rotation). They classified bond lengths and bond angles as hard variables, while the dihedral angles and external variable were classified as soft variables when calculating configurational entropy. Karplus and Kushick [40] generalized this notion by defining soft or important internal coordinates to include not only dihedral angles, but also internal coordinates that were strongly correlated. Further evidence of the validity of this generalization was provided in [45].

Building on the idea of soft or important coordinates and hard or nonessential internal coordinates, we consider dynamics with respect to a coordinate system defined using PCA of a trajectory matrix. This dissertation provides further evidence of the usefulness of a reduced representation provided by PCA of an MD trajectory in the study and prediction of molecular motion.

We provide a brief overview of protein structure and introduce notation and definitions used in this dissertation in Section 1.1. We then provide an overview of this dissertation in Section 1.2.

1.1 Protein Structure

A molecule is a chemical unit made up of smaller units called atoms, the basic building blocks of matter. Amino acids are molecules made up of three basic units - an amino group, a carboxyl group, and a side chain - all of which are bonded to a central carbon atom (see Figure 1.1). An amino group is composed of a nitrogen atom and

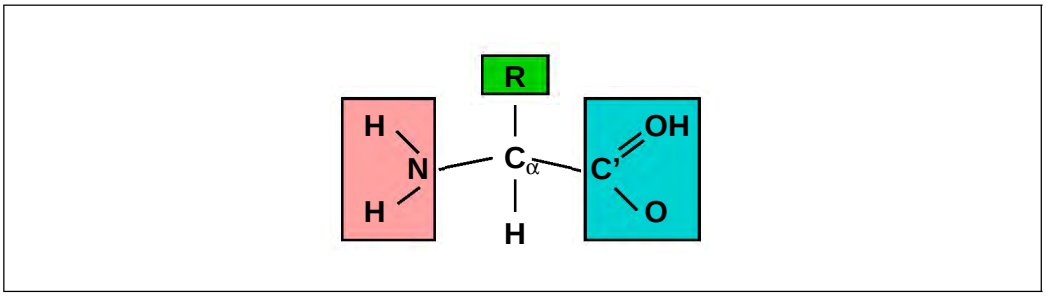


Figure 1.1: Basic structure of an amino acid molecule. The amino group (on the left) is composed of a nitrogen atom and two hydrogen atoms and the molecular formula is NH_2. The carboxyl group (on the right) is composed of a carbon atom, two oxygen atoms and a hydrogen with molecular formula COOH.

two hydrogen atoms and the molecular formula is NH_2. A carboxyl group is composed of a carbon atom, two oxygen atoms and a hydrogen with molecular formula COOH. The central carbon atom, denoted C_α, is bonded to a hydrogen atom, the nitrogen atom of the amino group, and the carbon atom of the carboxyl group. The side chain is the distinguishing feature between amino acids and this structure also shares a bond with the C_α atom.

Of the known amino acids only twenty amino acids are found in proteins. When an amino acid bonds to another amino acid the carbon atom in the carboxyl group of amino acid p, denoted C', bonds to the nitrogen atom in the amino group of amino acid $p+1$ forming a peptide bond (see Figure 1.2). During this process a water molecule (H_2O) is released. The primary structure of a protein is the polypeptide chain of amino acids. An amino acid unit along the polypeptide chain is also called a residue.

The primary structure of a protein folds to form a tightly packed three-dimensional

(3D) structure called the native structure. Understanding protein folding, the process by which the primary structure folds into a 3D structure, is a thriving area of research. Our interests lie in the 3D structure referred to as the tertiary structure. It is in this form that the protein is biologically active and able to perform its function.

The backbone, or main chain, of the protein is made up of the repeating unit $\mathrm{NH}-\mathrm{C}_\alpha\mathrm{H}-\mathrm{C}' =\mathrm{O}$. The degree of rotation about the bond between N and C_α is denoted ϕ; and the degree of rotation about the bond between C_α and C' is denoted ψ. Substructures, or secondary structures, common to proteins are alpha helices and beta sheets. An alpha helix is formed when consecutive residues have the same ϕ and ψ angles. These values are approximately $\phi \approx -60^o$ and $\psi \approx -50^o$. Alpha helices can contain as little as four residues or as many as forty residues. The average rise per residue is 1.5 Ångstrøms. A beta sheet is formed when regions of an amino acid sequence align adjacent to each other. These regions, known as beta strands, are about five to ten residues long [11].

As previously noted, the function of a protein is determined by its 3D structure and this structure is continuously experiencing fluctuations, thus takes various steric forms. A conformation, or configuration, is an arbitrary static structure of a molecule that comes about through free rotation of atoms about single chemical bonds. A molecule containing n atoms can be represented by $3n$ Cartesian coordinates. These coordinates can be combined to form a $3n$D vector. Generally for proteins the atoms are ordered with respect to the residue order in the polypeptide chain. The atomic

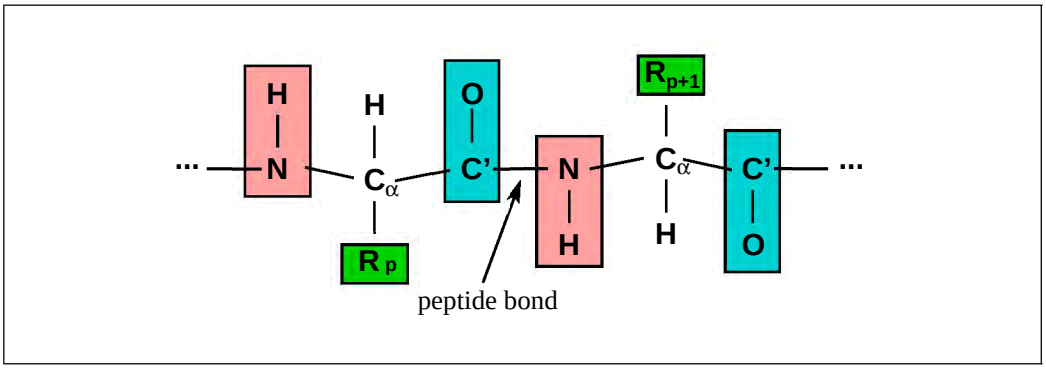


Figure 1.2: The carbon atom in the carboxyl group of amino acid p bonds to the nitrogen atom in the amino group of amino acid $p+1$ forming a peptide bond.

positions of a molecule containing n atoms at time $t_i = ih$ are denoted $\mathbf{r}(t_i) = \mathbf{r}^i \in \mathbb{R}^{3n}$ where h is the length of the discrete time step. Cartesian coordinates in the x, y, and z directions of atom a are stored in components $\mathbf{r}_{3a-2}$, $\mathbf{r}_{3a-1}$, and $\mathbf{r}_{3a}$, respectively. A trajectory is a collection of conformations from an MD simulation over a period of time. The conformation space of a molecule is a $3n$D space containing the collection of all points $\mathbf{r}^i$ that represent a configuration of the molecule. Molecular dynamics simulation provides a means to explore the conformation space, to study the motion of a protein, and to gain insight into the function of a protein.

In the following sections we introduce definitions and notation used throughout this dissertation. Additional notation will be introduced as needed.

1.1.1 Notation

- $\mathbb{R}$ denotes the field of real numbers and $\mathbb{R}^m$ is a vector space of dimension m over the real numbers. We will use the shorthand notation mD when discussing

m-dimensional vectors.

- A vector with a superscript denotes enumeration, eg., $\mathbf{r}^i$ denotes the ith vector in a set of vectors. A vector with a subscript denotes a component of the vector, eg., $\mathbf{r}_i$ denotes the ith component of $\mathbf{r}$. We will use the shorthand $\mathbf{r}^i \equiv \mathbf{r}(ih)$ in accordance to this notation convention where i is an index over time.

- The vector $\mathbf{v}(t) \in \mathbb{R}^{3n}$ represents the atomic velocities at time t. The jth component of $\mathbf{v}(t)$ is the first time derivative of position

$$\mathbf{v}_j = \dot{\mathbf{r}}_j = \frac{\mathbf{d}\mathbf{r}_j}{\mathbf{d}t}.$$

  Atomic accelerations at time t are represented by $\mathbf{a}(t) \in \mathbb{R}^{3n}$. The jth component of $\mathbf{a}_j(t)$,

$$\mathbf{a}_j = \ddot{\mathbf{r}}_j = \frac{\mathbf{d}\mathbf{r}_j^2}{\mathbf{d}^2 t}$$

  is the second time derivative of position. Higher time derivatives of position, $\mathbf{r}^{(\tau)}(t) = \{\mathbf{d}\mathbf{r}_j^\tau / \mathbf{d}^\tau t\}$, will be introduced as needed.

- $\mathcal{V}(\mathbf{r}(t))$ denotes a potential energy function.

- $\mathrm{M} \in \mathbb{R}^{3n}$ is a diagonal matrix containing atomic masses in triplicate, i.e., $\mathrm{M} = diag(m_1 I, \cdots, m_n I)$ where I is the 3 by 3 identity matrix.

- k_B represents the Boltzmann constant, a quantity that relates temperature to energy,

$$k_B = 1.3806503 \times 10^{-23} JK^{-1}$$

where joule (J) is an energy unit and Kelvin (K) is a temperature unit. Mass is represented in atomic mass units. Distance is represented with units of nanometers (nm $= 10^{-9}$ m), and time is represented in units of femtoseconds (fs $= 10^{-15}$ s), picoseconds (ps $= 10^{-12}$ s), and nanoseconds (ns).

1.1.2 Internal Coordinates

Internal coordinates define the location of the atoms in a molecule relative to the other atoms in the molecule. Here we define distance between atoms, angles and dihedral angles formed by bonded atoms.

Let $\mathbf{d}_{ij}$ denote the vector from atom j to atom i in a molecule:

$$\mathbf{d}_{ij} = [\mathbf{r}_{3i-2} - \mathbf{r}_{3j-2}, \mathbf{r}_{3i-1} - \mathbf{r}_{3j-1}, \mathbf{r}_{3i} - \mathbf{r}_{3j}],$$

and let $d_{ij} = \|\mathbf{d}_{ij}\|_2$ denote the two-norm of vector $\mathbf{d}_{ij}$. A bond angle, θ_{ijk}, formed by bonded atoms $i - j - k$ is defined as follows:

$$\cos\theta_{ijk} = \frac{\mathbf{d}_{ij} \bullet \mathbf{d}_{kj}}{d_{ij} d_{kj}}. \tag{1.1}$$

A dihedral angle, ϕ_{ijkl}, formed by bonded atoms $i - j - k - l$ is defined as follows:

$$\cos\phi_{ijkl} = \mathbf{n}_{\mathbf{d}_{ij}\mathbf{d}_{jk}} \bullet \mathbf{n}_{\mathbf{d}_{jk}\mathbf{d}_{kl}}. \tag{1.2}$$

Here, $\mathbf{n}_{\mathbf{d}_{ij}\mathbf{d}_{jk}}$ and $\mathbf{n}_{\mathbf{d}_{jk}\mathbf{d}_{kl}}$ are the unit normal vectors to the planes spanned by vector pairs $\{\mathbf{d}_{ij}, \mathbf{d}_{jk}\}$ and $\{\mathbf{d}_{jk}, \mathbf{d}_{kl}\}$, respectively. In Cartesian coordinate space a nonlinear molecule has $3n - 6$ internal degrees of freedom ($3n - 5$ for linear molecules).

1.2 Organization of the Dissertation

In Chapter 2 we will provide an overview of molecular dynamics (MD) simulation highlighting the general procedures and implementation. This discussion provides an overview of the equations of motion and the common numerical integration techniques used to solve them.

In Chapter 3 we discuss methods used to analyze results obtained from an MD simulation. We provide definitions of correlation functions, power spectrum, principal component analysis (PCA), and the singular value decomposition (SVD) which is used to compute principal components is also defined.

In Chapter 4 we discuss the use of PCA in the context of MD simulation, highlighting literature that is relevant to this work. Given a molecule containing n atoms, we consider working in a $3n$ dimensional ($3n$D) space defined by a concatenation of the 3D Cartesian coordinates of the atoms. We use information provided by PCA of a trajectory matrix to select a kD space where k is significantly smaller than $3n$.

Chapter 5 chronicles the methods we use to build our reduced dynamics code. We begin by defining the projection of an MD trajectory onto a kD space. We then use this projection and potential energy information from a standard MD simulation to generate a model of a kD potential energy surface. We introduce the theory underlying our reduced MD scheme. An outline of the Fortran subroutines used to implement this method is provided in Appendix A.

In Chapter 6 we consider the errors associated with reduced simulation. We

discuss the approximation properties of RBF networks and the local error associated with using the Verlet method to numerically solve the equations of motion.

In Chapter 7 we discuss how we utilize a standard MD simulation package, Extended System Program (ESP), to begin our simulation and discuss the protocol used to continue a simulation in a reduced setting (ie. computing the starting basis, rESP, updating the basis, computing force, returning to original space). We compare computed trajectories with the true trajectory by comparing power spectra computed from standard MD data to power spectra obtained from reduced simulation data.

Finally in Chapter 8 we provide a review of the developments of this dissertation and suggest directions for future consideration.

Chapter 2

Molecular Dynamics

"The purpose [of MD] is never to produce reliable trajectories... in the real physical world individual trajectories have no meaning..." [8]

Molecular dynamics (MD) is a computational technique used to study the interactions of a molecular system as it evolves in time. In this chapter we provide an overview of classical MD. For a more detailed introductory discussion see *Introduction to Protein Structure* by Branden and Tooze [11], *Molecular Modelling, Principles and Applications* by Leach [43], or see [1], [28], [54].

We provide a description of an empirical potential energy function in Section 2.1, followed by an overview of the numerical integration techniques used in MD simulations in Section 2.2. In Section 2.3 we provide an overview of how a standard MD program is set up and discuss the particular program used in this work, Extended System Program.

2.1 Potential Energy Function

The Schrödinger equation describes the motion of a particle in space subject to an external field ($\mathcal{U}$) [43]. The time-independent form is

$$\mathcal{H}\Psi(\mathbf{r}_n, \mathbf{r}_e) = E\Psi(\mathbf{r}_n, \mathbf{r}_e). \tag{2.1}$$

Here E is the energy of the system, Ψ is the wavefunction which is a function of the nuclei and electron coordinates, $\mathbf{r}_n$ and $\mathbf{r}_e$, respectively, and $\mathcal{H}$ is the Hamiltonian operator

$$\mathcal{H} = \left\{ -\frac{\bar{h}^2}{2m}\nabla^2 + \mathcal{U} \right\}.$$

The value $\bar{h}$ is Planck's constant divided by 2π, m is mass of the particle, and the symbol ∇^2 represents $\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$. The wavefunction gives the probability of finding the particle at a certain position. The solution of this partial differential eigenvalue problem requires obtaining energies and wavefunctions such that the effect of the Hamiltonian on the wavefunction equals the (scalar) multiplication of the wavefunction by the E (see Equation 2.1).

Equation 2.1 cannot be solved analytically for molecular systems. The Born-Oppenheimer approximation asserts that the motion of the electron can be decoupled from the motion of the nuclei since the weight of an electron is several orders of magnitude lighter than the nuclei [43]:

$$\mathcal{H}\Psi_e(\mathbf{r}_e; \mathbf{r}_n) = E\Psi_e(\mathbf{r}_e; \mathbf{r}_n) \tag{2.2}$$

$$\mathcal{H}\Psi_n(\mathbf{r}_n) = E\Psi_n(\mathbf{r}_n). \tag{2.3}$$

Equation 2.2 is solved for fixed arrangements of the nuclei and defines an energy E, called the potential energy surface (PES), that is a function of nuclear coordinates, $\mathbf{r}_n$. Minimum points on the PES are of particular interest because they correspond to stable configurations of a molecule. Maximum points on the PES correspond to configurations that are unstable, while saddle points on the PES correspond to transition configurations that connect minimum energy regions of a potential energy surface. After solving Equation 2.2, Equation 2.3 is solved and its solution describes the motion of the nuclei on the potential energy surface.

Molecular dynamics uses an empirical fit to the potential energy surface, called a force field, in lieu of solving Equation 2.2. A force field characterizes the potential energy surface (PES) using information about a subset of molecules to infer parameters for a larger set of molecules. A full description of a force field includes a list of atom types, a list of atomic charges, atom-typing rules, parameters for the function terms, rules for assigning parameters that are not explicitly defined, and functional forms for the components of the energy expression which we denote $\mathcal{V}$. With this information energy and force can be calculated for a molecular system.

Molecular dynamics is concerned with the solution of the Hamiltonian system

$$\dot{\mathbf{p}} = -H_q(\mathbf{p}, \mathbf{q}), \quad \dot{\mathbf{q}} = H_p(\mathbf{p}, \mathbf{q}) \tag{2.4}$$

where the Hamiltonian

$$H(\mathbf{p}, \mathbf{q}) = H(p_1, \cdots, p_d, q_1, \cdots, q_d),$$

represents the total energy of the system and is a function of the generalized coordinates, q_i, and the momentum, p_i for $i = 1, \cdots, d$ where d is the number of degrees of freedom. The functions H_q and H_p represent the partial derivative of H with respect to coordinates and momenta, respectively, and $\mathbf{p}$ and $\mathbf{q}$ are dependent on time t. Along the solution curve of Equation 2.4 the Hamiltonian is constant.

Specifically the total energy is of the form

$$H(\mathbf{p}, \mathbf{q}) = \frac{1}{2}\mathbf{p}^T M^{-1}\mathbf{p} + \mathcal{V}(\mathbf{q})$$

which is a separable Hamiltonian, $H(\mathbf{p}, \mathbf{q}) = T(\mathbf{p}) + \mathcal{V}(\mathbf{q})$, where $T(\mathbf{p})$ is a quadratic form and $\mathcal{V}(\mathbf{q})$ is a given potential function. The Hamiltonian system (2.4) is then

$$\dot{\mathbf{p}} = -\nabla\mathcal{V}(\mathbf{q}), \quad \dot{\mathbf{q}} = M^{-1}\mathbf{p} \tag{2.5}$$

where M is a diagonal matrix containing atomic masses and $\{\nabla\mathcal{V}(\mathbf{q})\}_i = \frac{\partial\mathcal{V}}{\partial\mathbf{q}_i}$, is the force associated with the ith component of the coordinate vector.

The evaluation of the potential function is the core computation of an MD simulation. In the following section we discuss the key interactions that contribute to the potential energy function (see Figure 2.1) and force calculation. The particular empirical potential energy function discussed is the CHARMM potential energy function [12].

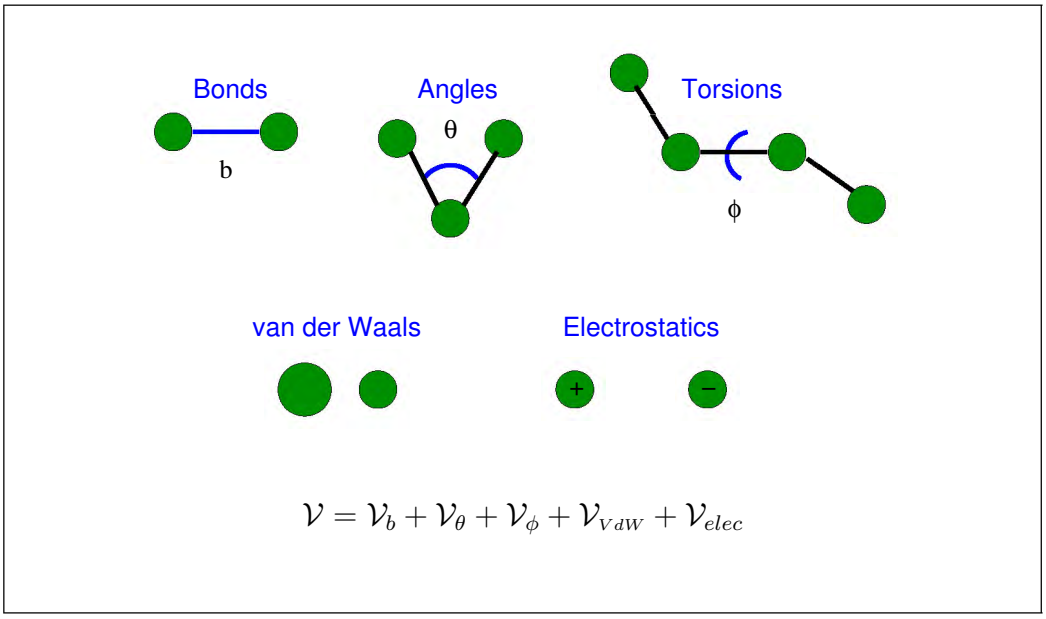


Figure 2.1: Bond lengths, bond angles, dihedral angles and nonbonded interactions contribute to the potential energy function [62].

2.1.1 Bonded Interactions

The bond ($\mathcal{V}_b$) and bond angle ($\mathcal{V}_\theta$) potentials are modeled by harmonic functions:

$$\mathcal{V}_b = \frac{1}{2}\sum_{bonds} k_b(b - b_0)^2 \text{ and } \mathcal{V}_\theta = \frac{1}{2}\sum_{\substack{bond\\angles}} k_\theta(\theta - \theta_0)^2. \tag{2.6}$$

Variables b and θ correspond to the actual bond lengths and bond angles. Parameters b_0, θ_0 and k_* ($* = b$ or θ) represent reference bond lengths, reference bond angles, and force constants, respectively. The higher the force constant the more difficult it is to deform the bond or bond angle. These terms depend on the chemical type of the bonded atoms and sums are over all bonded atoms and bond angles, respectively.

The dihedral angle potential ($\mathcal{V}_\phi$), also called the torsion potential, is modeled by

a cosine series:

$$\mathcal{V}_\phi = \sum_{\substack{dihedral \\ angles}} k_\phi(1 + \cos(n\phi - \psi)). \tag{2.7}$$

Four bonded atoms define ϕ, and Equation (2.7) represents the rotation about the bond between the middle pair of atoms (see Figure 2.1). Here n, the multiplicity, equals the number of minimum energy points that occur as the dihedral angle rotates 360^o, and ψ, the phase shift, determines where the dihedral angle passes through an energy minimum. The value k_ϕ represents the energy cost associated with angle rotation, often referred to as the barrier height.

2.1.2 Nonbonded Interactions

Nonbonded interactions also contribute to the potential energy of a molecule. These contributions are a function of the distance, d_{ij}, between nonbonded atom i and atom j.

The Lennard-Jones 12-6 function models the Van der Waals repulsive and attractive forces:

$$\mathcal{V}_{VdW} = \sum_{\substack{nonbonded \\ pairs}} 4\epsilon_{ij} \left[\left(\frac{\sigma_{ij}}{d_{ij}} \right)^{12} - \left(\frac{\sigma_{ij}}{d_{ij}} \right)^{6} \right]. \tag{2.8}$$

The collision diameter, σ_{ij}, corresponds to the separation distance at which the Lennard-Jones function equals zero. The well depth, ϵ_{ij}, is the magnitude of minimum energy of the Lennard-Jones function. Each of these parameters depends on the interacting atoms [1].

Equation 2.8 has an (absolute) minimum at distance $d_{ij} = \sigma_{ij}\sqrt[6]{2}$. At short distances electron-electron and nucleus-nucleus interactions are strong leading to a repulsive force, modeled by the first term of Equation (2.8). Changes in the charge distributions of the atoms' electron clouds give rise to dipole-dipole interactions which lead to attractive forces, modeled by the second term of Equation (2.8).

We calculate the electrostatic interactions between nonbonded atoms i and j using Coulomb's law:

$$\mathcal{V}_{elec} = \sum_{\substack{nonbonded \\ pairs}} \frac{q_i q_j}{4\pi\epsilon_0 d_{ij}}. \tag{2.9}$$

Here q_i and q_j are the partial charges on atoms i and j. Equation (2.9) models the electric force acting on a point charge, q_i as a result of the presence of a second point charge, q_j. The parameter ϵ_0, the dielectric constant, specifies the relative speed that an electric signal travels in a material.

For the Lennard-Jones and electrostatic interactions summations are over nonbonded pairs considering atom i and atom j. Evaluation of these interactions is typically the most computationally intensive and time consuming portion of an MD simulation ([1], [12], [43], [54]). For example, the computational complexity of electrostatic force evaluation is $\mathcal{O}(N_e^2)$ where N_e is the number of point charges in the molecule. Efficient algorithms exist for large systems (particle-mesh Ewald method, for example) which reduces the computational complexity to $\mathcal{O}(N_e log(N_e))$ [17].

2.1.3 Potential Energy Gradient

To obtain the force exerted on each atom at time t we must evaluate the gradient of the potential energy function. Derivatives of potential energy terms are calculated analytically using the chain rule containing partial derivatives of $\mathcal{V}$ with respect to internal coordinates - bond lengths (b), bond angles (θ), dihedral angle (ϕ) and interatomic distances (d) - and partial derivatives of internal coordinates with respect to Cartesian coordinates. For example, the bond potential between atoms i and j as given in Equation (2.6) is

$$\mathcal{V}_{b_{ij}} = \frac{1}{2} k_b (b_{ij} - b_0)^2$$

and the distance between bonded atoms i and j is

$$b_{ij} = ((x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2)^{1/2}.$$

The partial derivative of $\mathcal{V}_{b_{ij}}$ with respect to component α_β is

$$\frac{\partial \mathcal{V}_{b_{ij}}}{\partial \alpha_\beta} = \frac{\partial \mathcal{V}_{b_{ij}}}{\partial b_{ij}} \frac{\partial b_{ij}}{\partial \alpha_\beta}, \tag{2.10}$$

where $\alpha \in \{x, y, z\}$ and β is an index over the the interacting atoms. Thus we have

$$\frac{\partial \mathcal{V}_{b_{ij}}}{\partial b_{ij}} = k_b (b_{ij} - b_0) \quad \text{and} \quad \frac{\partial b_{ij}}{\partial \alpha_\beta} = \pm \frac{\alpha_i - \alpha_j}{b_{ij}},$$

thus,

$$\frac{\partial \mathcal{V}_{b_{ij}}}{\partial \alpha_\beta} = \pm (\alpha_i - \alpha_j) k_b \frac{b_{ij} - b_0}{b_{ij}},$$

(+, if $\beta = i$, and $-$, if $\beta = j$).

We obtain derivatives of the other potential energy components in a similar manner [48]. All derivatives with respect to the ith component of $\mathbf{r}$ are then combined to obtain the total force acting on that component. Thus the computation of the force is a straightforward byproduct of the calculation of the potential energy.

2.2 Numerical Integration

Recall that molecular dynamics is concerned with the solution of the Hamiltonian system

$$\dot{\mathbf{p}} = -\nabla\mathcal{V}(\mathbf{q}), \quad \dot{\mathbf{q}} = M^{-1}\mathbf{p}$$

where the momenta $\mathbf{p} = M\mathbf{v}$ and the coordinates are simply the Cartesian coordinates of the atoms in the system $\mathbf{q} = \mathbf{r}$. This system of first order differential equations is equivalent to the system of second order differential equations

$$\ddot{\mathbf{r}} = f(\mathbf{r}) \tag{2.11}$$

where $f(\mathbf{r}) = -M^{-1}\nabla\mathcal{V}(\mathbf{r})$ does not depend on $\dot{\mathbf{r}} = \mathbf{v}$. Due to the nature of the potential energy function the equations of motion cannot be solved analytically and must be solved numerically. Calculation of the potential energy is the most computationally intensive portion of an MD simulation, thus a numerical integration method applicable in the context of MD should include at most one energy (and gradient) calculation per iteration.

In the following sections we present numerical integration methods used to solve

the equations of motion. To begin we describe the Verlet method in Section 2.2.1 and Gear predictor-corrector methods in Section 2.2.2. We then provide an overview of multiple time step methods in Section 2.2.3 which account for the varying time scales of molecular activity during the integration process.

2.2.1 The Verlet Method

Let $t = ih$ denote discrete points in time where h is the discretization step, and let $\mathbf{r}^i$, $\mathbf{v}^i$, and $\mathbf{a}^i$ be approximations to $\mathbf{r}(ih)$, $\mathbf{v}(ih)$, and $\mathbf{a}(ih)$, respectively. An underlying assumption when using numerical methods to solve the equations of motion is that positions, velocities and accelerations can be approximated by Taylor series expansions. Consider the forward and backward Taylor series expansion of $\mathbf{r}(t)$ with time step h:

$$\begin{aligned} \mathbf{r}(t+h) &= \mathbf{r}(t) + h\mathbf{v}(t) + \frac{h^2}{2}\mathbf{a}(t) + \frac{h^3}{6}\mathbf{b}(t) + \mathcal{O}(h^4) \\ \mathbf{r}(t-h) &= \mathbf{r}(t) - h\mathbf{v}(t) + \frac{h^2}{2}\mathbf{a}(t) - \frac{h^3}{6}\mathbf{b}(t) + \mathcal{O}(h^4), \end{aligned}$$

where $\mathbf{r}(t)$, $\mathbf{v}(t)$, and $\mathbf{a}(t)$ represent position, velocity and acceleration vectors, respectively, and $\mathbf{b}(t)$ denotes the third time derivative of $\mathbf{r}(t)$, that is $\mathbf{b}(t) = \frac{\mathbf{d}\mathbf{r}_j^3}{\mathbf{d}^3 t}$. Adding these two expressions, we obtain

$$\mathbf{r}(t+h) = 2\mathbf{r}(t) - \mathbf{r}(t-h) + h^2\mathbf{a}(t) + \mathcal{O}(h^4). \tag{2.12}$$

Loup Verlet (1967) proposed a method based on Equation 2.12 for the numerical

integration of the system second order differential equations 2.11

$$\mathbf{r}^{i+1} = 2\mathbf{r}^{i} - \mathbf{r}^{i-1} + h^2\mathbf{a}^{i}. \tag{2.13}$$

Equation 2.13 iteratively updates $\mathbf{r}$ with respect to each component. Velocities do not appear explicitly in this formula, but can be computed from the positions using the formula

$$\mathbf{v}^{i} = \frac{\mathbf{r}^{i+1} - \mathbf{r}^{i-1}}{2h}.$$

This method is known as the Verlet method [68] in molecular dynamics literature, the Störmer method [1] in astronomy literature, and the explicit central difference method in numerical analysis literature.

The Verlet method is a widely used for numerical integration in MD for it is easily implemented, time reversible (i.e. $\mathbf{r}^{i+1}$ and $\mathbf{r}^{i-1}$ are interchangeable in Equation 2.13) and has good stability for moderately large time steps [28]. Furthermore, the Verlet method is symplectic. That is, it provides an exact solution to a discrete Hamiltonian system that closely resembles the continuous system of interest. Important consequences of symplecticity are preservation of geometry and conservation of total energy. Thus under proper simulation conditions the structural integrity of a molecule is maintained during MD simulation.

Variations of the Verlet method have been developed which allow for more precise calculation the velocity. Swope, *et al* (1982) [64] introduced an implementation of

[1] C. Störmer (1907) introduced this method into astronomy calculations.

the Verlet method that updates positions and velocities as follows:

$$\mathbf{r}^{i+1} = \mathbf{r}^i + h\mathbf{v}^i + \frac{h^2}{2}\mathbf{a}^i \tag{2.14}$$

$$\mathbf{v}^{i+1} = \mathbf{v}^i + \frac{h}{2}[\mathbf{a}^i + \mathbf{a}^{i+1}]. \tag{2.15}$$

Equations 2.14 and 2.15 are known as the velocity Verlet method. In practice the velocities are updated in two steps to incorporate accelerations at time t and $t + h$:

$$\begin{aligned} \mathbf{v}^{i+\frac{1}{2}} &= \mathbf{v}^i + \frac{h}{2}\mathbf{a}^i \\ \mathbf{a}^{i+1} &= -\mathrm{M}^{-1}\nabla\mathcal{V}(\mathbf{r}^{i+1}) \\ \mathbf{v}^{i+1} &= \mathbf{v}^{i+\frac{1}{2}} + \frac{h}{2}\mathbf{a}^{i+1}. \end{aligned}$$

During each iteration positions, velocities and accelerations ($9n$ words) are needed to update terms. Mathematically the velocity Verlet method is equivalent to the Verlet method. However, the velocity Verlet method performs better numerically [64].

2.2.2 Gear Predictor-Corrector Methods

Predictor-corrector methods make up another class of methods commonly used in MD simulations. We consider the most prominent of these methods, the Gear predictor-corrector methods [22], which uses scaled time derivatives of positions to predict and correct positions at time $t + h$. The fourth order Gear algorithm in matrix form is as

Order (η)	g_0	g_1	g_2	g_3	g_4	g_5
2	0	1	1			
3	$\frac{1}{6}$	$\frac{5}{6}$	1	$\frac{1}{3}$		
4	$\frac{19}{120}$	$\frac{3}{4}$	1	$\frac{1}{2}$	$\frac{1}{12}$	
5	$\frac{3}{20}$	$\frac{251}{360}$	1	$\frac{11}{18}$	$\frac{1}{6}$	$\frac{1}{60}$

Table 2.1: Gear corrector parameters for a second-order differential equations with predictor of order η [22],[1].

follows:

$$\begin{pmatrix} \mathbf{r}^{i+1} \\ \dot{\mathbf{r}}^{i+1} \\ \ddot{\mathbf{r}}^{i+1} \\ \mathbf{r}^{(3)\ i+1} \\ \mathbf{r}^{(4)\ i+1} \end{pmatrix}_P = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 0 & 1 & 3 & 6 \\ 0 & 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{r}^{i} \\ \dot{\mathbf{r}}^{i} \\ \ddot{\mathbf{r}}^{i} \\ \mathbf{r}^{(3)\ i} \\ \mathbf{r}^{(4)\ i} \end{pmatrix}.$$

The predicted positions (denoted by the subscript P) are then used to evaluate new forces. From the computed forces we obtain the acceleration at time $t + h$ and use the difference $\Delta\ddot{\mathbf{r}} = \ddot{\mathbf{r}}^{i+1} - \ddot{\mathbf{r}}_P^{i+1}$ to correct the predicted values. The corrector step

updates $\mathbf{r}$ and its derivatives using this information:

$$\begin{pmatrix} \mathbf{r}^{i+1} \\ \dot{\mathbf{r}}^{i+1} \\ \ddot{\mathbf{r}}^{i+1} \\ \mathbf{r}^{(3)\ i+1} \\ \mathbf{r}^{(4)\ i+1} \end{pmatrix}_C = \begin{pmatrix} \mathbf{r}^{i+1} \\ \dot{\mathbf{r}}^{i+1} \\ \ddot{\mathbf{r}}^{i+1} \\ \mathbf{r}^{(3)\ i+1} \\ \mathbf{r}^{(4)\ i+1} \end{pmatrix}_P + \begin{pmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \end{pmatrix} \Delta\ddot{\mathbf{r}}.$$

Table 2.1 displays the parameters (c_i) used in $\eta-$th order Gear predictor-corrector algorithms. During each iteration the fourth order Gear method requires positions, velocities, old and new accelerations, as well as higher order derivatives ($18n$ words) to update terms.

When one desires high accuracy the Gear methods perform well, while the Verlet method and its variants, though less accurate, are more stable for large time steps and are time reversible [8]. Advances in multiple time step methods attempt to combine these features (accuracy and stability) to produce robust methods for molecular simulation. We provide an overview of multiple time step methods in the next section.

2.2.3 Multiple Time Step Methods

As noted in Section 2.1.2, evaluation of nonbonded interactions is the most computationally intensive portion of an MD simulation. Atoms that are separated by a substantial distance interact at low frequency, thus these forces do not change significantly with respect to a standard time step.

Multiple time step (MTS) methods take into account the range of times in which molecular activity occurs and use more than one time step during numerical integration. Figure 2.2 outlines a typical MTS method [65]. Here we consider an algorithm that splits force into two parts, $F = F_{short} + F_{long}$. Short range, F_{short}, and long range, F_{long}, forces are typically defined with respect to a splitting function based upon interatomic distances. Short range atomic interactions determine the value of time step h. After m steps, velocities are corrected with respect to F_{long} ([10], [35]).

2.2.4 Integration Time Step

With respect to protein dynamics, local motions, such as bond stretching, occur within femtoseconds (fs = 10^{-15} second), while rigid body and large-scale motions occur within a range of nanoseconds (ns = 10^{-9} second) to seconds [63]. To resolve local motions, an integration time step of approximately 0.5 fs (for example, with respect to O-H bond stretching) must be used [60]. The introduction of the SHAKE [58] and RATTLE [4] algorithms allowed the constraint of bond lengths to their reference lengths, and made it possible to increase the integration time step (1 to 2 fs).

2.3 MD Simulation

In the previous sections we described a potential energy function (Section 2.1) and numerical integration methods used to solve the equations of motion (Section 2.2). In

this section we outline the steps needed to run an MD simulation. We then describe the particular program used for our simulations.

2.3.1 Set up and Simulation

Figure 2.3 [63] provides an overview of the stages of an MD simulation. The first stage of a simulation includes building the system of interest. Initial atomic coordinates as well as a molecule's topology must be supplied to a program. Such information is obtained from theoretical information or experimental data (NMA data, X-Ray crystallography, etc.) which is user defined or retrieved from a database such as the Protein Data Bank [5]. The structure then undergoes energy minimization to relieve adverse atomic interactions present in the initial configuration.

Initial velocities are randomly assigned using the Maxwell-Boltzmann distribution with respect to a particular temperature, T:

$$p(\mathbf{v}_{k\{\alpha\}}) = \left(\frac{m_{k\{\alpha\}}}{2\pi k_B T}\right)^{1/2} \exp\left[-\frac{1}{2}\frac{m_{k\{\alpha\}}\mathbf{v}_{k\{\alpha\}}^2}{k_B T}\right]. \tag{2.16}$$

Here $\mathbf{v}_{k\{\alpha\}}$ represents the velocity of the kth atom in the α direction where $\alpha \in \{x, y, z\}$; $m_{k\{\alpha\}}$ is the associated atomic mass; and k_B is the Boltzmann constant. The Maxwell-Boltzmann distribution is a Gaussian distribution [43].

The system may then undergo various intervals of heating to bring the system to the desired temperature. This may entail adjusting velocities or coupling the system with an external heat bath.

The next stage of dynamics, equilibration dynamics, is used to stabilize the system.

Generally, this entails monitoring energies, temperature, and pressure, depending on the thermodynamic ensemble desired (see Section 2.3.2), to make sure these quantities are fluctuating minimally about a desired mean.

Once the system is properly specified, production dynamics begins. During this stage atomic position, velocities, and the desired simulation parameters are stored for future analysis.

There are many programs available for performing molecular dynamics. Those most widely used in the scientific community include **CHARMM** (**C**hemistry at **HAR**vard **M**acromolecular **M**echanics) [12], **AMBER** (**A**ssisted **M**odel **B**uilding with **E**nergy **R**efinement) [52], and **NAMD** (**N**ot **A**nother **M**olecular **D**ynamics) program [38]. In this work we use **ESP** (**E**xtended **S**ystem **P**rogram) [61] to perform MD simulations courtesy of the Department of Chemistry at the University of Houston. We provide a brief overview of ESP in the following section, Section 2.3.2.

2.3.2 Extended System Program

Figure (2.4) illustrates the organization of ESP. The main program, ESP, facilitates the construction of the molecular system with topology and parameter files provided by the user. Topology files include information about atom types, charges, and connectivities. Atom types are assigned to identify elements and molecular orbital environments, charges are assigned to each atom type. Connectivities between atoms include lists of bonded atoms, bond angles, and torsion angles. Other simulation pa-

rameters such as integration time step, simulation length, system temperature, etc., are incorporated into ESP via a user defined data file.

Parameter files contain force constants necessary to describe the bond, angle, and torsion energies as well as nonbonded interactions. These files may also suggest parameters for setting up energy calculations. For example, the minimum distance at which to consider nonbonded interactions.

Once the system is constructed with user defined initial coordinates and velocities, the program transfers control to one of the second level subroutines - RUNNVE, MTSNVE, or RUNNVT. ESP supports the canonical (RUNNVT) and the microcanonical (RUNNVE) ensembles with the option of using multiple integration time steps in conjunction with the microcanonical ensemble (MTSNVE). The microcanonical ensemble is characterized by a fixed number of atoms (N), fixed volume (V), and fixed energy (E). The canonical ensemble is characterized by a fixed number of atoms, fixed volume and fixed temperature (T).

Numerical integration of the equations of motion occurs in the second level subroutines with calls to the third and fourth level subroutines to obtain force and energy calculations, respectively. CHARMM parameterization [12] is used to define a forcefield, and the velocity Verlet method (see Section 2.2.1) is used to perform numerical integration.

Algorithm 1 Multiple Time Step Integration Algorithm

Input: $\mathbf{r}$, $\mathbf{v}$, $F_{short}(\mathbf{r})$, $F_{long}(\mathbf{r})$ from previous iteration step
Output: $\mathbf{r}$, $\mathbf{v}$

1. **for** $i = 1 : 3 * n_{atoms}$
   $\mathbf{v}_i \leftarrow \mathbf{v}_i + \frac{h_m}{2} F_{long}(\mathbf{r})_i$
   end
2. **for** $ii = 1 : m$
   2.1. **for** $i = 1 : 3 * n_{atoms}$
   $\mathbf{r}_i \leftarrow \mathbf{r}_i + h\mathbf{v}_i + \frac{h^2}{2} F_{short}(\mathbf{r})_i$
   $\mathbf{v}_i \leftarrow \mathbf{v}_i + \frac{h}{2} F_{short}(\mathbf{r})_i$
   end
   2.2. **call** $F_{short}(\mathbf{r})$
   2.3. **for** $i = 1 : 3 * n_{atoms}$
   $\mathbf{v}_i \leftarrow \mathbf{v}_i + \frac{h}{2} F_{short}(\mathbf{r})_i$
   end
   end
3. **call** $F_{long}(\mathbf{r})$
4. **for** $i = 1 : 3 * n_{atoms}$
   $\mathbf{v}_i \leftarrow \mathbf{v}_i + \frac{h_m}{2} F_{long}(\mathbf{r})_i$
   end

Figure 2.2: Code for one iteration of multiple time step algorithm updating positions and velocities using step sizes h and $h_m = mh$ [65].

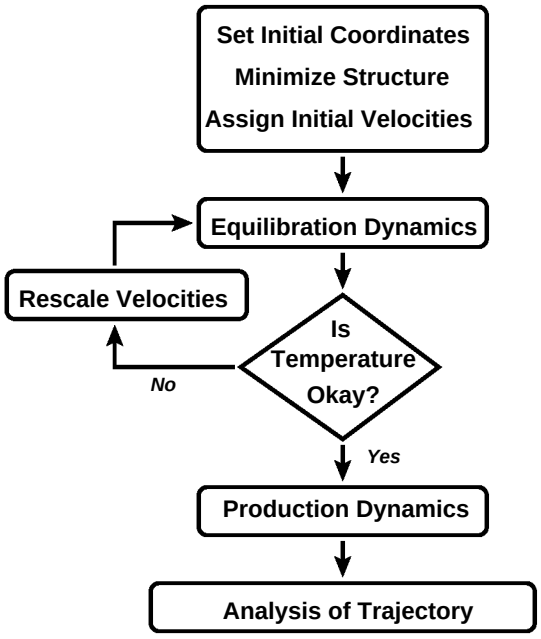


Figure 2.3: Flow chart outlining a standard MD simulation [63].

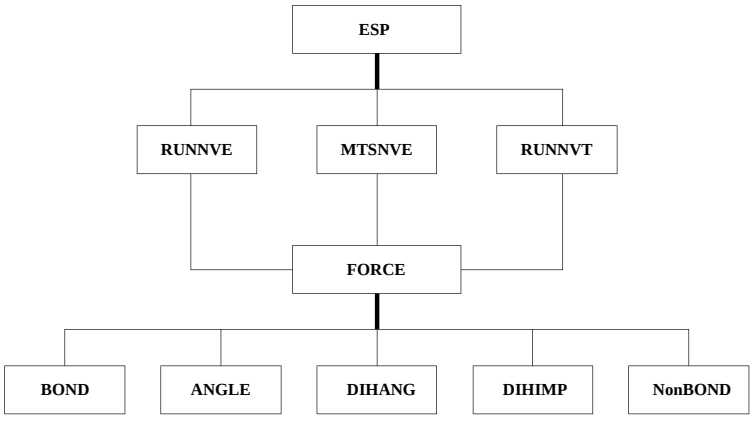


Figure 2.4: Organization of ESP.

Chapter 3

Analysis of MD Trajectory

During the production dynamics stage of an MD simulation the coordinates, velocities, and energies are stored at user defined intervals for analysis. The information obtained from a single time step is of little value in isolation. However, the conformers sampled by an MD simulation belong to the same statistical ensemble. A statistical ensemble is a collection of configurations with different microscopic characteristic (for example, atomic coordinates and velocities) that have similar macroscopic characteristics - for example, the energy (E), volume (V), temperature (T), pressure (P), and number of particles (N). The microcanonical, or NVE, ensemble, has a fixed number of atoms, volume, and energy. Properties of the configurations in an ensemble are then averaged to generate the macroscopic observations of interest.

MD simulation provides time ordered information detailing how a system evolves with respect to classical mechanics in its ensemble context. Ergodic theory [41] states

that ensemble averages can be replaced by time averages. Thus macroscopic observations and correlations of dynamic properties can be computed using MD data.

We provide a brief discussion of time correlations in Section 3.1 and the related power spectrum in Section 3.2. In Section 3.3 we introduce matrix decompositions used in the analysis of MD trajectories. In Chapter 4 we describe the use of Principal Component Analysis in analyzing an MD trajectory.

3.1 Correlation

Time correlations associate a dynamic property with another time shifted dynamic property. A property of interest may be a component of a position vector ($\mathbf{r}_i(t)$) or a component of a velocity vector ($\mathbf{v}_i(t)$). We define a discrete time correlation in Section 3.1.1 and give examples of its use and interpretation in MD. In Section 3.1.2 describe an algorithm for computing discrete correlations.

3.1.1 Definition

Consider two time dependent functions, $a(t)$ and $b(t)$, that are sampled at times $t_p = p\Delta t$. Here Δt is the sampling rate, the time interval between consecutive samples (i.e. $\Delta t = t_{p+1} - t_p$) and $p = \{0, \cdots, N-1\}$. Suppose that the samples are periodic with period N. The discrete correlation of $a(t)$ and $b(t)$ associated with time

shift $l\Delta t$ is defined

$$\mathrm{Corr}(a,b)_l \equiv \sum_{p=0}^{N-1} a_{l+p} b_p,$$

where $a_p \equiv a(t_p)$. The correlation of a function with itself is called the *autocorrelation*.

Let A_n and B_n denote the discrete Fourier transforms (DFT) of a_p and b_p, respectively,

$$A_n \equiv \sum_{p=0}^{N-1} a_p e^{2\pi i p n/N} \quad \text{and} \quad B_n \equiv \sum_{p=0}^{N-1} b_p e^{2\pi i p n/N}$$

where $n = 0, \cdots, \frac{N}{2}$. The *discrete correlation theorem* notes that $\mathrm{Corr}(a,b)_l$ is the Fourier transform pair of the product $A_n B_n^*$, eg., one can use the discrete Fourier transform to move from one representation to the other [53]. Here the asterisk (*) denotes complex conjugation. This relationship allows the use of the *fast Fourier transform* (FFT) to compute correlations.

Correlations provide information about how a dynamic property is related to another time shifted dynamic property. Consider the discrete autocorrelation of the jth component of velocity:

$$\mathrm{Corr}(\mathbf{v}_j, \mathbf{v}_j)_l = \frac{C}{N-l} \sum_{p=0}^{N-1-l} \mathbf{v}_{j(l+p)} \cdot \mathbf{v}_{j(p)}, \quad l = 0, \cdots, N-1. \tag{3.1}$$

Note that the summation is over $N-1-l$ values versus $N-1$ values. This is due to the fact that the data we receive from MD is not periodic as required by the discrete correlation theorem. Thus, the index $l+p$ cannot exceed the number of available values $\mathbf{v}_j$. We will discuss how to resolve the lack to periodicity in Section 3.1.2. Also note that a normalization factor C is included to insure that $\mathrm{Corr}(\mathbf{v}_j, \mathbf{v}_j)_0 = 1$ and

the weighting $1/(N-l)$ is included to reflect changes in the number of items included in the summation as l varies.

To incorporate velocity components in all directions (x, y, and z) for all atoms in the system, we consider Equation (3.2) which is simply summation over all velocity autocorrelations (see Equation (3.1)) weighted by the number of atoms in the system N_a:

$$\mathrm{Corr}(\mathbf{v},\mathbf{v})_l = \frac{C}{N_a(N-l)} \sum_{j=1}^{N_a} \sum_{\alpha=1}^{3} \sum_{p=0}^{N-1-l} \mathbf{v}_{j\alpha(l+p)} \cdot \mathbf{v}_{j\alpha(p)}, \quad l = 0, \cdots, N-1. \tag{3.2}$$

Here α is an index over the x, y, and z components of velocity. Computing the discrete velocity autocorrelation as noted in Equation (3.2) is a process of order N^2. The computational cost is greatly reduced by taking advantage of the discrete correlation theorem. Section 3.1.2 outlines how this is achieved. Before we move to that discussion, however, we provide a brief interpretation of the velocity autocorrelation.

What does the velocity autocorrelation reveal about the associated MD simulation? Generally, this information reveals the nature of force interactions in the system. Newton's first law of motion states that an object in a state of uniform motion maintains that motion unless an external force is applied to the object. Thus velocities in a system with no atomic interactions remain unchanged and the resulting autocorrelations would be constant. In the case of weak force interactions, changes in velocity would be gradual producing an autocorrelation that decays exponentially. When strong force interactions are present atoms in the system tend to fluctuate about stable positions. What results are changes in the velocity in the positive and

Algorithm 2 Computation of Autocorrelation Function

Input: a_p
Output: $Corr(a,a)_l$

a. compute the DFT of a_p using an FFT subroutine:
$A_n \equiv \sum_{p=0}^{2N-1} a_p e^{2\pi ipn/(2N-1)}$;

b. compute the square modulus of the DFT obtained in the previous step:
$G_n = |A_n|^2$;

c. compute the inverse DFT of G_n using an FFT subroutine:
$g_p \equiv \sum_{p=0}^{2N-1} G_n e^{-2\pi ipn/(2N-1)}$;

d. apply the proper normalization:
$\mathrm{Corr}(a,a)_l = \frac{1}{N-l} g_l$.

Figure 3.1: Procedure used to compute the autocorrelation function of a time series $a_p \equiv a(t_p)$.

negative directions of fluctuation, producing an oscillating autocorrelation. The oscillations are damped by force perturbations inherent in the simulated system. The extent of the damped oscillation depends on the nature of the material. We now outline how an autocorrelation is computed using the FFT.

3.1.2 Computation

The discrete correlation theorem assumes that the sampled functions are periodic. This is generally not true for the MD data. We induce periodicity by adding N zeros to the end of our data set. In essence we have for the time dependent quantity a_p that

$a_{2(N-1)+1} = a_1$. The zero padding absorbs errors that may result from non-periodic data ([53], [1]). What follows is an outline of how time autocorrelations are computed using the FFT.

Given a set of discretely sampled values, $a_p \equiv a(t_p)$, for $p = 0, \cdots, N-1$, we append N zeros to the set, so that $a_p = 0$ for $p = N, \cdots, 2N-1$. The computation proceeds as outlined in Figure 3.1. Information pertaining to the original data is stored while the information associated with the zero padding is discarded. Thus we obtain an autocorrelation by application of the discrete correlation theorem very efficiently using the FFT ([21], [41], [1], [53]).

3.2 Power Spectrum

Given a time series sampled at equal intervals, a_p where $p = 0, 1, \cdots, N-1$, the discrete Fourier transform of the N points is defined

$$A_n \equiv \sum_{p=0}^{N-1} a_p \exp(2\pi i p n/N). \tag{3.3}$$

It is often convenient to consider data in the frequency domain. Parseval's theorem states that the total power in a signal can be computed in the time domain or the frequency domain [53]. The discrete form of Parseval's theorem is

$$\sum_{p=0}^{N-1} |a_p|^2 = \frac{1}{N} \sum_{n=0}^{N-1} |A_n|^2 \tag{3.4}$$

for Fourier transform pair $\{a_p\}_{p=0,\cdots,N-1}$ and $\{A_n\}_{n=0,\cdots,N-1}$.

In Section 3.2.1 we define a power spectrum and discuss its relevance and interpretation with respect to MD, and in Section 3.2.2 we discuss the methods used to compute a power spectrum.

3.2.1 Definition

The power spectrum, also called the power spectral density (PSD), is defined as the amount of power per unit of frequency as a function of frequency. It describes how the power of a time series is distributed over frequency, where power is defined as the average energy of the time series. Mathematically, the power spectrum is proportional to the square modulus of the Fourier transform of a time series. Given the discrete Fourier transform of a time series sampled at equally spaced intervals (see Equation 3.3) an estimate of the power spectrum at zero and positive frequencies is defined as follows

$$P(0) = P(f_0) = \frac{1}{N^2}|A_0|^2 \tag{3.5}$$

$$P(f_p) = \frac{1}{N^2}\left[|A_p|^2 + |A_{N-p}|^2\right], \quad p = 1, 2, \cdots, \left(\frac{N}{2} - 1\right) \tag{3.6}$$

$$P(f_c) = P(f_{N/2}) = \frac{1}{N^2}|A_{N/2}|^2. \tag{3.7}$$

We perform this computation for each component of the velocity using steps (a.) and (b.) of Algorithm 2.

3.2.1.1 Interpretation of Power Spectrum

The Fourier transform of the velocity autocorrelation provides information about the density of the normal modes in a harmonic system [1]. With respect to molecular mechanics, the Fourier transform of the velocity autocorrelation function reveals the underlying frequencies of the molecular processes. Thus the power spectrum provides information about the dynamic behavior of atomic interactions and can be compared with the infra-red (IR) spectrum of a molecule. The IR spectrum of a molecule shows which frequencies of infrared radiation are absorbed by the molecule and can be used to identify the functional groups in a molecule.

When plotting a power spectrum versus frequency, we observe pronounced peaks in the neighborhood of the normal mode frequencies. The normal vibrational modes are the characteristic vibrations of a system about a local energy minimum. It is customary to present power spectrum with respect to wavenumber units which is the inverse of the wavelength in centimeters (cm^{-1}). Multiplying the wavenumbers by the speed of light (2.99792×10^{10} cm/sec) converts the value to frequency units, inverse seconds (sec^{-1}).

3.2.1.2 Normal Mode Analysis

As an aside we briefly outline how normal modes are calculated. First the potential energy is minimized. At a local energy minimum

$$\frac{\partial \mathcal{V}(\mathbf{r})}{\partial \mathbf{r}_i} = 0 \text{ and } \frac{\partial^2 \mathcal{V}(\mathbf{r})}{\partial \mathbf{r}_i^2} \geq 0,$$

for $i = 1, \cdots, 3N_a$. The molecular configurations at which minimum energies are achieved correspond to equilibrium arrangements of the system.We are interested in atomic fluctuations about local minimum configurations. Let $\Delta\mathbf{r}_i$ denote the displacement of $\mathbf{r}_i$ from its equilibrium position. We denote mass weighted coordinates as follows $R_j = {m_j}^{1/2}\Delta\mathbf{r}_i$ where m_j is the mass of the ith atom and $i = 3j - 2 : 3j$ for $j = 1 : N_a$. Coordinates are mass weighted because the magnitude of atomic force interactions are affected by the mass of each atom.

The Hessian matrix of the potential energies with respect to mass weighted displacements is defined, $\mathbf{H}_{ij} = \frac{\partial^2 \mathcal{V}(\mathbf{r})}{\partial R_i \partial R_j}$. The Hessian matrix is diagonalized to obtain the normal modes (the eigenvectors of $\mathbf{H}$) and the associated frequencies (the square roots of the eigenvalues). We now turn our attention to the computation of the power spectrum.

3.2.2 Computation

As noted earlier, one can use the first two steps for computing an autocorrelation to compute a power spectrum (see Section 3.1.2). We note here some coding details and summarize the routines that accomplish these tasks in Appendix A. The present code implements the Fast Fourier Transform with a Bartlett window function using overlapping data. The window function is defined as follows:

$$w_j = 1 - \left| \frac{j - \frac{1}{2}N}{\frac{1}{2}N} \right|. \tag{3.8}$$

The purpose of a window function is to provide a more accurate estimate of the power spectrum. Given the window function the power spectrum estimate becomes

$$B_p \equiv \sum_{j=0}^{N-1} a_j w_j \exp(2\pi i j p/N) \quad \text{for } p = 0, \cdots, N-1 \tag{3.9}$$

$$P(0) = P(f_0) = \frac{1}{W_{ss}} |B_0|^2 \tag{3.10}$$

$$P(f_p) = \frac{1}{W_{ss}} \left[|B_p|^2 + |B_{N-p}|^2 \right] \quad p = 1, 2, \cdots, \left(\frac{N}{2} - 1 \right) \tag{3.11}$$

$$P(f_c) = P(f_{N/2}) = \frac{1}{W_{ss}} |B_{N/2}|^2 \tag{3.12}$$

where

$$W_{ss} \equiv N \sum_{j=0}^{N-1} w_j^2.$$

The data is partitioned into K segments each containing $2M$ consecutive points. Data overlapping occurs since consecutive segments share M points. The FFT of each segment is computed and the average over the K segments is the estimated power spectrum. This method of computation is particularly efficient when dealing with large data sets in that the FFT is computed in parts, thus it is not necessary to consider the entire data set at once.

The spectrum, $\hat{\mathbf{r}}_\alpha(\omega)$, of atomic vibrations, derived by a Fourier transform of the trajectory of atom α, determines the contribution of the motion of atom α to the infrared spectrum of a protein [27]. We consider the spectrum of atomic vibrations with respect to two ranges - (0 - 500 cm^{-1}) and (500 - 3500 cm^{-1}) - which will be designated the low-frequency and high-frequency ranges, respectively.

3.2.3 Atomic RMS Fluctuations

Let $\langle \mathbf{r}_\alpha \rangle \equiv \sum_{t_i} \mathbf{r}_\alpha(t_i)/nt$ denote the average position of atom α with respect to nt integration time steps. Let h denote the integration step size. The average displacement of atom α from its mean value is computed

$$\sigma_\alpha = \left[\sum_{t_i} \left((\mathbf{r}_\alpha(t_i) - \langle \mathbf{r}_\alpha \rangle)^2 \right) / nt \right]^{1/2}.$$

This quantity measures atomic mobility, which may also be determined via experimental techniques such as X-ray or neutron scattering. Thus comparing σ_α with values obtained from experimental techniques provides a gauge of the quality of the MD simulation [27].

3.3 Matrix Analysis

In this section we provide an overview of principal component analysis. We begin in Section 3.3.1 by defining descriptive statistics that will be useful in our analysis. In Section 3.3.2 we describe principal component analysis, a method commonly used to identify dominant features in a data set. Finally, in Section 3.3.3 we introduce the singular value decomposition.

3.3.1 Basic Statistics

Consider a set of n data vectors, $\mathbf{x}^j \in \mathbb{R}^m$ where x_{ij} denotes the ith component of the jth data vector. Each vector $\mathbf{x}^j$ represents an observation and its components

x_{ij} are variables that describe the observation. Let $\bar{x}_i = \frac{1}{n}\sum_{j=1}^{n} x_{ij}$ denote the mean over the ith components of all data vectors. The sample variance is a measure of the dispersion of a set of variables around its mean value. If we consider the pth component of each data vector the sample variance is computed as follows

$$s^2 = \frac{1}{n}\sum_{j=1}^{n}(x_{pj} - \bar{x}_p)^2,$$

which is the average of the squared deviations from mean values. The measure of the variance of two variables with respect to each other is the covariance. The covariance of components p and q over all data vectors is

$$cov(p,q) = \frac{1}{n}\sum_{j=1}^{n}(x_{pj} - \bar{x}_p)(x_{qj} - \bar{x}_q)$$

The matrix with components $\mathbf{C}_{pq} = cov(p,q)$ is called the covariance matrix where $\mathbf{C} \in \mathbb{R}^{m\times m}$ is symmetric and positive semidefinite. Given the matrix with data vectors as columns

$$\mathbf{X} = [\mathbf{x}^1 \;\; \mathbf{x}^2 \;\; \cdots \;\; \mathbf{x}^n] \in \mathbb{R}^{m\times n},$$

we can represent the covariance matrix as follows

$$\mathbf{C} = \frac{1}{n}\hat{\mathbf{X}}\hat{\mathbf{X}}^T$$

where the ijth component of $\hat{\mathbf{X}}$ is $x_{ij} - \bar{x}_i$.

3.3.2 Principal Component Analysis

The goal of principal component analysis (PCA) is to represent data described by a large number of interrelated variables, m, using k new uncorrelated variables called

the principal components, where k is significantly smaller than m. PCA is a multivariate method in statistics that identifies the variance inherent in the variables that describe the data and constructs a new set of uncorrelated variables that retains a majority of the total variance.

Given a set of data, we construct a covariance matrix, $\mathbf{C}$. $\mathbf{C}$ is a real, symmetric matrix, thus by the spectral theorem, there exists an orthogonal matrix $\mathbf{U}$ that diagonalizes $\mathbf{C}$, i.e., $\mathbf{C} = \mathbf{U}\Lambda\mathbf{U}^T$, where the columns of $\mathbf{U}$ are eigenvectors of $\mathbf{C}$ and the diagonal matrix Λ contains the eigenvalues of $\mathbf{C}$. We order the columns of $\mathbf{U}$ such that the associated eigenvalues are in nonincreasing order.

The principal components of $\mathbf{X}$ are computed by an orthogonal transformation $\mathbf{Y} = \mathbf{U}^T\hat{\mathbf{X}}$. The rows of $\mathbf{Y}$ are the principal components. That is, the pth row of $\mathbf{Y}$ is defined $\mathbf{y}^p = (\mathbf{u}^p)^T\hat{\mathbf{X}}$ where $\mathbf{u}^p$ is the pth column of $\mathbf{U}$. The first component of $\mathbf{y}^p$ is first principal component associated with $\hat{\mathbf{x}}^p$. Generally, the ith component of $\mathbf{y}^p$ is the ith principal component associated with $\hat{\mathbf{x}}^p$. The ith eigenvalue, Λ_{ii}, is equal to the variance of $\{y_{ij}\}_{j=1,\cdots,n}$, the components of ith row of $\mathbf{Y}$. The total variance of the principal components is given by the sum $\sum_{i=1}^{r}\Lambda_{ii}$ where r represents the number of nonzero eigenvalues [19].

Let $\mathbf{U}_k = [\mathbf{u}^1 \;\; \mathbf{u}^2 \;\; \cdots \;\; \mathbf{u}^k]$ denote the matrix whose columns are the eigenvectors associated with the k largest eigenvalues. A kD representation of the data is obtained by an orthogonal transformation and is defined as follows: $\mathbf{Y}_k = \mathbf{U}_k^T\mathbf{X}$.

3.3.3 Singular Value Decomposition

We use the singular value decomposition (SVD) to compute the principal components. Since the principal components associated with the directions of highest variance are of interest, we consider a truncated SVD (TSVD).

The SVD is a matrix factorization that decomposes a rectangular matrix $\mathbf{W} \in \mathbb{R}^{m \times n}$ into the product of three matrices $\mathbf{W} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ where

$$\mathbf{U} = [\mathbf{u}^1 \cdots \mathbf{u}^n] \in \mathbb{R}^{m \times n}, \quad \mathbf{V} = [\mathbf{v}^1 \cdots \mathbf{v}^n] \in \mathbb{R}^{n \times n}$$

$$\mathbf{U}^T\mathbf{U} = \mathbf{I}_n, \quad \mathbf{V}\mathbf{V}^T = \mathbf{V}^T\mathbf{V} = \mathbf{I}_n, \text{ and}$$

$\mathbf{S}$ is an $n \times n$ diagonal matrix where $n \leq m$. The vectors $\mathbf{u}^i$ and $\mathbf{v}^i$ are the and left singular vectors and right singular vectors of $\mathbf{W}$, respectively. The diagonal elements of matrix $\mathbf{S}$, s_{ii}, are the singular values of $\mathbf{W}$ where $s_{11} \geq s_{22} \geq \cdots s_{nn} \geq 0$ [26].

Consider $\mathbf{W} = \frac{1}{\sqrt{n}}\hat{\mathbf{X}} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ where $\hat{\mathbf{X}}$ is the matrix introduced in Section 3.3.1. Then the covariance matrix can be written

$$\mathbf{C} = \frac{1}{n}\hat{\mathbf{X}}\hat{\mathbf{X}}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T\mathbf{V}\mathbf{S}^T\mathbf{U}^T = \mathbf{U}\mathbf{S}^2\mathbf{U}^T.$$

Thus with respect to PCA the SVD provides an efficient method to compute the eigenvectors and eigenvalues of $\mathbf{C}$. The eigenvectors are the left singular vectors of $\mathbf{W}$, and the singular values of $\mathbf{W}$ are the square roots of the eigenvalues of $\mathbf{C}$. Furthermore, we obtain in the right singular vectors a standardized version of the principal components [37]. Recall that $\mathbf{Y} = \mathbf{U}^T\hat{\mathbf{X}}$. Then by substituting the SVD of $\hat{\mathbf{X}}$ we obtain $\mathbf{Y} = \sqrt{n}\mathbf{S}\mathbf{V}^T$.

The TSVD provides a rank k representation of $\mathbf{W}$:

$$\mathbf{W} \approx \mathbf{W}_k = \mathbf{U}_k \mathbf{S}_k \mathbf{V}_k^T = \sum_{i=1}^{k} \sigma_i \mathbf{u}^i \mathbf{v}^{i\ T}, \tag{3.13}$$

and has the unique property $\|\mathbf{W} - \mathbf{W}_k\|_2 = \sigma_{k+1}$, i.e., it is the best rank k approximation of $\mathbf{W}$ [26]. Furthermore, the TSVD can be updated as new data becomes available. We discuss SVD updating in Section 5.2.4. In the following chapter we outline the procedure used to compute the principal components of an MD trajectory.

Chapter 4

PCA of an MD Trajectory

During an MD simulation the molecular system moves freely in Cartesian space. For the purposes of analyzing the internal motion of a molecule we must consider the MD data with respect to the same frame of reference. Given a trajectory from a reliable MD simulation, we superimpose the conformations onto the 3D coordinates of a reference conformation by performing an RMSD fit. We describe the procedure we used in the following section.

4.1 Defining Coordinates

4.1.1 RMSD Fit

Recall that atomic positions of a molecule at time t_τ are denoted $\mathbf{r}^\tau \in \mathbb{R}^{3n_a}$ where Cartesian coordinates in the x, y, and z directions of atom a are stored in components

Algorithm 3 RMSD Fit to a Reference Structure

Input: Reference structure, $\mathcal{A}_{REF}$, and current structure, $\mathcal{A}_\tau \in \mathbb{R}^{n_a \times 3}$
Output: Γ that minimizes $\|\mathcal{A}_{REF} - \mathcal{A}_\tau \Gamma\|_F$ subject to $\Gamma^T \Gamma = \mathbf{I}_3$.

1. $B = \mathcal{A}_\tau^T \mathcal{A}_{REF}$
2. $B = \mathbf{U}\mathbf{S}\mathbf{V}^T$
3. $\Gamma = \mathbf{V}\mathbf{U}^T$
4. $\mathcal{A}_\tau \leftarrow \mathcal{A}_\tau \Gamma$

Figure 4.1: Procedure used to compute the RMSD fit of an MD trajectory to a reference structure.

$\mathbf{r}_{3a-2}$, $\mathbf{r}_{3a-1}$, and $\mathbf{r}_{3a}$, respectively, and n_a is the number of atoms in the molecule. For the purposes of the present procedure we consider an n_a by 3 matrix for each conformation. We construct column vectors $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ where $\mathbf{x}_a = m_a \mathbf{r}_{3a-2}$, $\mathbf{y}_a = m_a \mathbf{r}_{3a-1}$, and $\mathbf{z}_a = m_a \mathbf{r}_{3a}$, and a is an index over the n_a atoms in the molecule, and m_a is the mass of atom a. The centers of mass in the x, y, and z directions are

$$\mathbf{x}_{CM} = \frac{\sum_{i=1}^{n_a} \mathbf{x}_i}{m_{TOT}}, \quad \mathbf{y}_{CM} = \frac{\sum_{i=1}^{n_a} \mathbf{y}_i}{m_{TOT}}, \quad \mathbf{z}_{CM} = \frac{\sum_{i=1}^{n_a} \mathbf{z}_i}{m_{TOT}},$$

where $m_{TOT} = \sum_{a=1}^{n_a} m_a$ represents the total mass of the n_a atom system.

We translate the center of mass of the system to the origin by subtracting the appropriate center of mass from the components of $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ and form the following

matrix

$$\mathcal{A}_\tau = \begin{bmatrix} \mathbf{x}_1 - \mathbf{x}_{CM} & \mathbf{y}_1 - \mathbf{y}_{CM} & \mathbf{z}_1 - \mathbf{z}_{CM} \\ \mathbf{x}_2 - \mathbf{x}_{CM} & \mathbf{y}_2 - \mathbf{y}_{CM} & \mathbf{z}_2 - \mathbf{z}_{CM} \\ \vdots & \vdots & \vdots \\ \mathbf{x}_n - \mathbf{x}_{CM} & \mathbf{y}_n - \mathbf{y}_{CM} & \mathbf{z}_n - \mathbf{z}_{CM} \end{bmatrix}.$$

This is done for all conformations in the trajectory.

Next we compute a rotation based on the root-mean-square deviation (RMSD) of each conformation with respect to a reference configuration. The RMSD is used to estimate structural similarity between molecules and is defined as follows

$$\text{RMSD}(\mathcal{A}_{REF}, \mathcal{A}_\tau) = \min_\Gamma \|\mathcal{A}_{REF} - \mathcal{A}_\tau \Gamma\|_F \tag{4.1}$$

where $\mathcal{A}_{REF}$ is a reference configuration, Γ is a rotation matrix, and $\|\cdot\|_F$ denotes the Frobenius matrix norm. We use a minimum energy structure as the reference configuration.

For each time step t_τ we construct a matrix $B = \mathcal{A}_\tau^T \mathcal{A}_{REF} \in \mathbb{R}^{3\times 3}$ and compute the SVD of $B = \mathbf{USV}^T$. Using the resulting matrices we construct a rotation matrix $\Gamma = \mathbf{VU}^T$. The rotation matrix Γ produced by this procedure satisfies Equation 4.1 [26]. The procedure for computing rotation matrix Γ is summarized in Figure 4.1.

We consider translation and rotation with respect to the C_α atoms of a molecule. This is done to achieve a superposition similar to that accomplished by the zero angular momentum condition as concluded in various simulation studies ([34], [39], [2]).

4.1.2 Absolute and Mean Adjusted Coordinates

We replace the atomic coordinates by their translation and rotation free counterparts, making the substitution $\mathbf{r}^\tau = \mathcal{A}_\tau(:)$ where $\mathcal{A}_\tau(:)$ denotes the vector defined by concatenating (in order) the columns of $\mathcal{A}_\tau$ using the adjusted coordinates. Consider the sample matrix

$$\mathbf{R} = [\mathbf{r}^1 \quad \mathbf{r}^2 \quad \cdots \quad \mathbf{r}^m] \in \mathbb{R}^{n\times m}$$

where $\mathbf{r}^i \in \mathbb{R}^n$ contains the atomic coordinates of a molecule and $n = 3n_a$. Let $\bar{\mathbf{r}}$ be the vector containing the sample mean where the jth component is $\bar{\mathbf{r}}_j = \frac{1}{m}\sum_{i=1,m} \mathbf{r}^i_j$. Define $\bar{\mathbf{R}}$ as follows

$$\bar{\mathbf{R}} = [\bar{\mathbf{r}} \cdots \bar{\mathbf{r}}] = \frac{1}{m}\mathbf{R}\mathbf{e}\mathbf{e}^T \in \mathbb{R}^{n\times m}$$

where $\mathbf{e} \in \mathbb{R}^m$ has all components equal to 1. The columns of $\bar{\mathbf{R}}$ are identical, that is each component in the ith row is the sample mean of the ith atomic coordinate. The matrix $\mathbf{R}$ contains absolute coordinates, and the matrix $\mathbf{R} - \bar{\mathbf{R}}$ contains mean adjusted coordinates.

We use the SVD to analyze and to predict the behavior of dynamic systems. Since we will be considering simulations where the number of coordinates, n, is (much) less than the number of time points, m, we define

$$\mathcal{R} = \mathbf{R}^T \quad \text{and} \quad \hat{\mathcal{R}} = (\mathbf{R} - \bar{\mathbf{R}})^T.$$

The covariance matrix of the MD trajectory is diagonalized

$$\mathbf{K} = \frac{1}{m}\hat{\mathcal{R}}^T\hat{\mathcal{R}} = \mathbf{Q}\Lambda\mathbf{Q} \tag{4.2}$$

where $\Lambda = \text{diag}(\lambda_1,\ \lambda_2,\ \cdots,\ \lambda_n)$ and the matrix

$$\mathbf{Q} = [\mathbf{q}^1\ \mathbf{q}^2\ \cdots\ \mathbf{q}^n],\quad \mathbf{Q}^T\mathbf{Q} = \mathbf{I}_n$$

contains the eigenvectors of $\mathbf{K}$ arranged such that the associated eigenvalues, λ_j are in nonincreasing order. The eigenvectors can be partitioned such that each atom has a set of eigenvectors that represent the directions preferred by the atom [57]. The eigenvalues are the root mean square fluctuation along these axes.

We consider the eigenvectors associated with the $3n_a - 6$ largest eigenvalues. The six smallest eigenvalues are approximately zero and correspond to the overall translation and rotation of the molecule during simulation, thus the associated eigenvectors do not contribute significantly to the internal motion of the molecule.

4.2 Important Space of Atomic Motion

Consider the following partitioning of $\mathbf{Q}$,

$$\mathbf{Q} = [\underbrace{\mathbf{Q}_k}_{k}\quad \underbrace{\tilde{\mathbf{Q}}_k}_{m-k}].$$

The columns of $\mathbf{Q}$ form an orthonormal basis for $\mathbb{R}^m$. Internal motion can be divided into two subspaces - one in which "important" or correlated molecular motion occurs and another in which all other motion occurs. The eigenvectors associated with the k largest eigenvalues, the columns of $\mathbf{Q}_k$, form a basis for the former subspace, while the columns of $\tilde{\mathbf{Q}}_k$ form a basis for the latter subspace.

Previous studies have reported values of k significantly smaller than $3n_a - 6$, the dimension of the configuration space ([2], [57], [56], [67], [66]). Amadei *et al.* reported that 90% of the overall motion was described by the first $k = 35$ eigenvectors in a study of lysozyme considering all atoms ($3n_a = 3792$) [2]. When considering only the C_α atoms ($3n_a = 387$), 90% of the motion was described by the first $k = 20$ eigenvectors. In a study of myoglobin, Romo reported that 70% of the side-chain activity was described when $k = 20$ ($3n_a = 459$) [56].

These studies suggest that considering protein motion with respect to the important or essential subspace [2] could provide a means to significantly reduce computational overhead and therefore allow longer simulation times. We discuss an analysis of equations of motion with respect to an essential subspace presented by Amadei *et al.* [2] in the following section. In Section 4.3 we discuss coordinate selection and procedures for choosing k.

4.2.1 Orthogonal Transformation of Equations of Motion

In this section we consider an orthogonal transformation of the equations of motion as presented by Amadei *et al.* [2]. This analysis provides compelling evidence in favor of using information obtained from PCA in conjunction with the equations of motion to reduce the computation required to solve the equations of motion. Particularly, if an empirical potential function can be reformulated with respect to transformed coordinates, then the equations of motion can be written such that functionally important

motion can be separated from other motion.

We present the findings of Amadei *et al.* [2] in the form of a proposition and provide a discussion of the results.

Proposition 4.2.1 *Let* $\mathbf{r}_\tau \equiv \mathbf{r}(t_\tau)$ *denote mean adjusted molecular coordinates at time* t_τ *obtained from an MD trajectory. Assume that overall translation and rotation have been removed as outlined in Algorithm 3. Let* $\mathbf{K}$ *be the covariance matrix associated with the trajectory as defined in Equation 4.2, where* $\mathbf{Q}$ *is an orthogonal matrix that diagonalizes* $\mathbf{K}$*, and* Λ *is a diagonal matrix containing the eigenvalues of* $\mathbf{K}$*,* λ_i*, in nonincreasing order. Consider the coordinate transformation* $\mathbf{r} \equiv L(\mathbf{y}) = \mathbf{Q}\mathbf{y}$ *where* $L : \mathbb{R}^{3n_a} \rightarrow \mathbb{R}^{3n_a}$ *is an orthogonal linear mapping and* n_a *is the number of atoms in the system. Construct vectors* ξ *and* $\mathbf{s}$ *such that*

$$\xi = [\mathbf{y}_1 \quad \cdots \quad \mathbf{y}_k] = [\xi_1 \quad \cdots \quad \xi_k] \text{ and} \tag{4.3}$$

$$\mathbf{s} = [\mathbf{y}_{k+1} \quad \cdots \quad \mathbf{y}_{3n_a}] = [s_1 \quad \cdots \quad s_{3n_a-k}]. \tag{4.4}$$

Assume that the components of $\mathbf{s}$ *are normally distributed. Then the motion with respect to the* ξ *coordinates can be described (approximately) independently from the motion with respect to the* $\mathbf{s}$ *coordinates.*

Amadei *et al.* [2] call the k-dimensional vector space containing vectors ξ the *essential subspace*. While the $m - k$ dimensional vector space containing vectors $\mathbf{s}$ is considered nonessential and it is suggested that coordinates in this space behave relatively as full constraints. Thus this vector space is presumed to be negligible in an

MD calculation. We present the argument outlining the validity of such constraints in the following discussion.

4.2.1.1 Analysis of Motion

Consider a system in a canonical ensemble, that is, a system composed of n_a particles, occupying volume V with absolute temperature T. The configurational probability density function (PDF), also called the statistical distribution function, for atomic coordinates $\mathbf{r}$ is

$$\rho(\mathbf{r}^0) = \frac{\exp(-\beta\mathcal{V}(\mathbf{r}^0))}{\int_V \exp(-\beta\mathcal{V}(\mathbf{r}))d\mathbf{r}} \tag{4.5}$$

where $\beta = 1/k_B T$ and k_B is the Boltzmann constant. This value represents the fraction of configuration points per unit volume at point $\mathbf{r}^0 \in \mathbb{R}^{3n_a}$ [70].

Given the coordinate transformation, $\mathbf{r} \equiv L(\mathbf{y}) = \mathbf{Q}\mathbf{y}$, we can express the distribution function with respect to coordinates $\mathbf{y}$. This is accomplished by first performing a change of variables for the integral[1] in the denominator of Equation 4.5. The Jacobian matrix of mapping L is an orthogonal matrix, specifically $\mathbf{Q}$, thus the Jacobian determinant is ± 1. Since an orthogonal transformation is either a rotation or a rotoinversion [69], interatomic distances and angles are preserved by the mapping

[1] **Theorem (The Change of Variables Theorem [20])** *Suppose that the mapping $\Psi : \mathcal{O} \to \mathbb{R}^n$ is a smooth change of variables on the open subset $\mathcal{O}$ of $\mathbb{R}^n$. Let D be an open Jordan domain such that $K = D \cup \partial D$ is contained in $\mathcal{O}$. Then $\Psi(K)$ is a Jordan domain that has the property that for any continuous function $f : \Psi(K) \to \mathbb{R}$, the following integral transformation formula holds:*

$$\int_{\Psi(K)} f(\mathbf{x})d\mathbf{x} = \int_K f(\Psi(\mathbf{u}))|\det \mathbf{D}\Psi(u)|d\mathbf{u}.$$

L. Thus we can write,

$$\mathcal{V}(\mathbf{Qy}) = \mathcal{V}(\mathbf{y})$$

leading to the following PDF

$$\rho(\mathbf{y}^0) = \frac{\exp(-\beta\mathcal{V}(\mathbf{y}^0))}{\int_{V_\mathbf{y}} \exp(-\beta\mathcal{V}(\mathbf{y}))d\mathbf{y}}, \tag{4.6}$$

where $V_\mathbf{y}$ denotes the volume occupied by transformed coordinates $\mathbf{y}$.

Consider the second-order Taylor expansion[2] of the potential function about $\mathbf{q}_{s_0} = (\xi; \mathbf{0}_{3n_a-k})$,

$$\mathcal{V}(\mathbf{q}_{s_0} + \mathbf{h}) = \mathcal{V}(\mathbf{q}_{s_0}) + \sum_i s_i \frac{\partial \mathcal{V}}{\partial s_i}(\mathbf{q}_{s_0}) + \frac{1}{2}\sum_{i,j} s_i s_j \frac{\partial^2 \mathcal{V}}{\partial s_i \partial s_j}(\mathbf{q}_{s_0}) + R_2(\mathbf{h}, \mathbf{q}_{s_0}) \tag{4.7}$$

where $\mathbf{h} = (\mathbf{0}_{3n_a-k}; \mathbf{s})$ and $i, j = k+1, \cdots, 3n_a$. Substituting Equation 4.7 into Equation 4.6 leads to the following PDF

$$\rho(\xi^0; \mathbf{s}^0) = \frac{\exp(-\beta\mathcal{V}(\mathbf{q}_{s_0}))\exp(-\beta\Delta\mathcal{V}(\xi^0; \mathbf{s}^0))}{\int \exp(-\beta\mathcal{V}(\mathbf{q}_{s_0}))d\xi \int \exp(-\beta\Delta\mathcal{V}(\xi; \mathbf{s}))d\mathbf{s}} \tag{4.8}$$

where

$$\Delta\mathcal{V}(\xi; \mathbf{s}) = \sum_i \frac{\partial \mathcal{V}}{\partial s_i}(\mathbf{q}_{s_0})s_i + \frac{1}{2}\sum_{i,j} \frac{\partial^2 \mathcal{V}}{\partial s_i \partial s_j}(\mathbf{q}_{s_0})s_i s_j. \tag{4.9}$$

We obtain a PDF for the $\mathbf{s}$ variables by integrating Equation 4.8 over all possible values ξ

$$\rho(\mathbf{s}^0) = \int \rho(\xi^0; \mathbf{s}^0)d\xi = \int \frac{\exp(-\beta\mathcal{V}(\mathbf{q}_{s_0}))\exp(-\beta\Delta\mathcal{V}(\xi^0; \mathbf{s}^0))}{\int \exp(-\beta\mathcal{V}(\mathbf{q}_{s_0}))d\xi \int \exp(-\beta\Delta\mathcal{V}(\xi; \mathbf{s}))d\mathbf{s}} d\xi. \tag{4.10}$$

[2] **Theorem (Second-Order Taylor Formula [46])** *Let $f : U \subset \mathbb{R}^n \to \mathbb{R}$ have continuous partial derivatives of third order. Then we may write*

$$f(\mathbf{x}_0 + \mathbf{h}) = f(\mathbf{x}_0) + \sum_{i=1}^n h_i \frac{\partial f}{\partial x_i}(\mathbf{x}_0) + \frac{1}{2}\sum_{i,j=1}^n h_i h_j \frac{\partial^2 f}{\partial s_i \partial s_j}(\mathbf{x}_0) + R_2(\mathbf{h}, \mathbf{x}_0)$$

where $R_2(\mathbf{h}, \mathbf{x}_0)/\|\mathbf{h}\|^2 \to 0$ as $\mathbf{h} \to \mathbf{0}$ and the second sum is over all i's and j's between 1 and n (so there are n^2 terms).

Now suppose an MD simulation resulted in a trajectory such that each component of $\mathbf{s}$ has a normal distribution. Then the collective PDF for $\mathbf{s}$ is

$$\rho(\mathbf{s}) \simeq \prod_i \frac{1}{\sigma_i\sqrt{2\pi}} \exp(-s_i^2/2\sigma_i^2) \tag{4.11}$$

where the variance of s_i is $\sigma_i^2 = \lambda_i$, the ith eigenvalue of $\mathbf{K}$. Equation 4.11 is true provided that

$$\exp(-\beta\Delta\mathcal{V}(\xi;\mathbf{s})) \simeq \exp(-\sum_i s_i^2/2\sigma_i^2) \tag{4.12}$$

and thus

$$\frac{\partial\mathcal{V}}{\partial s_i}(\mathbf{q}_{s_0})s_i \simeq 0 \quad \text{and} \quad \frac{\partial^2\mathcal{V}}{\partial s_i s_j}(\mathbf{q}_{s_0}) \simeq (k_BT/\sigma_i^2)\delta_{ij}. \tag{4.13}$$

The potential function can now be expressed as follows

$$\mathcal{V}(\xi;\mathbf{s}) \simeq \mathcal{V}(\mathbf{q}_{s_0}) + \frac{1}{2}\sum_i (kT/\lambda_i)s_i^2, i = n+1, \cdots, 3n_a - 6 \tag{4.14}$$

and the PDF becomes

$$\rho(\xi^0;\mathbf{s}^0) \simeq \rho(\xi^0)\prod_i \frac{1}{\sqrt{\lambda_i 2\pi}} \exp(-s_i^2/2\lambda_i) \tag{4.15}$$

where

$$\rho(\xi^0) = \frac{\exp(-\beta\mathcal{V}(\mathbf{q}_{s_0}))}{\int \exp(-\beta\mathcal{V}(\mathbf{q}_{s_0}))d\xi}. \tag{4.16}$$

Let $\hat{\xi}^i$ denote the unit vectors corresponding to the hyperplane containing coordinates ξ $(i = 1, \cdots, k)$, and let $\hat{\mathbf{s}}^i$ denote the unit vectors corresponding to the hyperplane containing coordinates $\mathbf{s}$ $(i = 1, \cdots, 3n_a - 6 - n)$. Let $\hat{\hat{\mathbf{s}}}^i$ denote unit

vectors that are linearly independent from $\hat{\xi}_i$ and satisfy[3]

$$\hat{\xi}_i^T M \hat{\mathbf{s}}_j = \hat{\mathbf{s}}_j^T M \hat{\xi}_i = 0. \tag{4.17}$$

Then $\{\hat{\xi}^i, \hat{s}^j\}$ is a basis set for $\mathbb{R}^{3n_a}$, and $\mathbf{r}$ and $\mathbf{y}$ can be expressed with respect to this basis

$$\mathbf{r}(t) = \Theta_1 \tilde{\mathbf{y}}(t) \tag{4.18}$$

$$\mathbf{y}(t) = \Theta_2 \tilde{\mathbf{y}}(t) \tag{4.19}$$

where Θ_1 and Θ_2 are transformation matrices. Then $\tilde{\mathbf{y}}$ can be represented as $\tilde{\mathbf{y}} = (\tilde{\xi}; \tilde{\mathbf{s}})$ where $\tilde{\xi}$ represents coordinates corresponding to $\hat{\xi}^i$ and $\tilde{\mathbf{s}}$ represents coordinates corresponding to $\hat{\hat{\mathbf{s}}}_j$. We can then express coordinates of $\mathbf{r}$ and $\mathbf{y} = (\xi; \mathbf{s})$ as follows

$$r_l(t) = \sum_i (\hat{\xi}_i \cdot \hat{\mathbf{r}}_l)\tilde{\xi}_i(t) + \sum_j (\hat{\hat{s}}_j \cdot \hat{\mathbf{r}}_l)\tilde{s}_j(t) \tag{4.20}$$

$$\xi_l(t) = \sum_i (\hat{\xi}_i \cdot \hat{\xi}_l)\tilde{\xi}_i(t) + \sum_j (\hat{\hat{s}}_j \cdot \hat{\xi}_l)\tilde{s}_j(t) \tag{4.21}$$

$$s_l(t) = \sum_i (\hat{\xi}_i \cdot \hat{s}_l)\tilde{\xi}_i(t) + \sum_j (\hat{\hat{s}}_j \cdot \hat{s}_l)\tilde{s}_j(t) \tag{4.22}$$

where $\hat{\mathbf{r}}_l$ is the unit vector corresponding to r_l, $i = 1, \cdots, k$ and $j = 1, \cdots, 3n_a - k$. These representations reduce to

$$\xi_l(t) = \tilde{\xi}_l(t) + \sum_j (\hat{\hat{s}}_j \cdot \hat{\xi}_l)\tilde{s}_j(t) \tag{4.23}$$

$$s_l(t) = \sum_j (\hat{\hat{s}}_j \cdot \hat{s}_l)\tilde{s}_j(t) \tag{4.24}$$

[3] **Theorem ([71] sections 39-41; [26])** *If $V_1 \in \mathbb{R}^{n\times(n-r)}$ has orthonormal columns, then there exists $V_2 \in \mathbb{R}^{n\times(n-r)}$ such that $V = [V_1 V_2]$ is orthogonal. Note that range$(V_1)^\perp$ = range(V_2).*

given the definitions of $\tilde{\xi}_i$ and $\hat{\mathbf{s}}_j$, and we can then represent the potential function as follows (see Equations 4.14, 4.23, and 4.24)

$$\mathcal{V}(\tilde{\xi};\tilde{\mathbf{s}}) \simeq \mathcal{V}(\tilde{\xi};0) + \frac{1}{2}\sum_{ij}\kappa_{ij}\tilde{s}_i\tilde{s}_j \tag{4.25}$$

where κ_{ij} are force constants.

We now rewrite the equations of motion using Equation 4.18 and obtain

$$\Gamma\ddot{\tilde{\mathbf{y}}} = -\nabla_{\tilde{\mathbf{y}}}\mathcal{V}(\tilde{\mathbf{y}}) \tag{4.26}$$

where

$$\Gamma = \Theta_1^T M \Theta_1 \tag{4.27}$$

is a block-diagonal matrix (see Equation 4.17)

$$\begin{bmatrix} \Gamma^{\tilde{\xi}} & 0 \\ 0 & \Gamma^{\tilde{\mathbf{s}}} \end{bmatrix} \begin{bmatrix} \ddot{\tilde{\xi}} \\ \ddot{\tilde{\mathbf{s}}} \end{bmatrix} = \begin{bmatrix} -\nabla_{\tilde{\xi}}\mathcal{V}(\tilde{\xi};\tilde{\mathbf{s}}) \\ -\nabla_{\tilde{\mathbf{s}}}\mathcal{V}(\tilde{\xi};\tilde{\mathbf{s}}) \end{bmatrix} \tag{4.28}$$

and

$$\Gamma_{ij}^{\tilde{\xi}} = \hat{\xi}^{i\ T} M \hat{\xi}^j, i,j = 1,\cdots,k \tag{4.29}$$

$$\Gamma_{ij}^{\tilde{\mathbf{s}}} = \hat{\mathbf{s}}^{i\ T} M \hat{s}^j, i,j = 1,\cdots,3n_a - 6. \tag{4.30}$$

Given Equation 4.14 we approximate

$$-\nabla_{\tilde{\xi}}\mathcal{V}(\tilde{\xi};\tilde{\mathbf{s}}) \simeq -\nabla_{\tilde{\xi}}\mathcal{V}(\tilde{\xi};0), \tag{4.31}$$

and coupled with Equations 4.28 obtain

$$\Gamma_{\tilde{\xi}}\ddot{\tilde{\xi}} \simeq -\nabla_{\tilde{\xi}}\mathcal{V}(\tilde{\xi};0). \tag{4.32}$$

Equation 4.32 provides an approximate representation of the equations of motion with respect to the essential or important coordinates. That is, the $\tilde{\xi}$ coordinates can be considered independently from the $\tilde{\mathbf{s}}$ coordinates with the approximation becoming exact as $k_B T/\lambda_i \to \infty$ for each $i = k+1, \cdots, 3n_a - 6$.

This analysis supports separating the equations of motion with respect to important coordinates. However, an underlying assumption throughout this analysis is that the parameterization of a potential energy surface remains valid or can be reformulated under the suggested separation of coordinates. The current literature suggests that a reformulation of the potential energy surface would be necessary, however it is intractable given the highly nonlinear nature of a potential energy surface [9].

Our purpose here is to define and analyze equations of motion with respect to transformed (least squares) coordinates. In Chapter 5 we construct an approximation to a potential energy surface given an MD trajectory and associated potential energy calculations. We conclude this chapter with a discussion comparing absolute and mean atomic adjusted coordinates.

4.3 Coordinate Selection

In this section we consider the use of absolute coordinates versus mean adjusted coordinates when studying MD trajectories. Recall matrices

$$\mathcal{R} \in \mathbb{R}^{m \times n} \quad \text{and} \quad \hat{\mathcal{R}} \in \mathbb{R}^{m \times n}$$

where $\mathcal{R}$ contains absolute coordinates and $\hat{\mathcal{R}}$ contains mean adjusted coordinates, m is the number of time points and n is the number of coordinates.

The right singular vectors of $\mathcal{R}$ or $\hat{\mathcal{R}}$ (the left singular vectors of $\mathbf{R}$ or $\mathbf{R} - \bar{\mathbf{R}}$, respectively) form orthogonal basis for the coordinate space. We seek approximate solutions to the equations of motion

$$\mathrm{M}\ddot{\mathbf{r}} = f(\mathbf{r}) \text{ where } f(\mathbf{r}) = -\nabla\mathcal{V}(\mathbf{r})$$

which allow us to take advantage of a reduced representation of the configuration space provided by a previous MD simulation. We begin by discussing the relationship between the SVD of the the absolute coordinates, $\mathcal{R}$, and the mean adjusted coordinates, $\hat{\mathcal{R}}$.

4.3.1 Thin SVD

Consider the SVD of $\mathcal{R}$ and $\hat{\mathcal{R}}$,

$$\mathcal{R} = \mathbf{U}\mathbf{S}\mathbf{V}^T \text{ and } \hat{\mathcal{R}} = \hat{\mathbf{U}}\hat{\mathbf{S}}\hat{\mathbf{V}}^T$$

where $\mathbf{U}, \hat{\mathbf{U}} \in \mathbb{R}^{m\times n}$, $\mathbf{V}, \hat{\mathbf{V}} \in \mathbb{R}^{n\times n}$, with $\mathbf{U}^T\mathbf{U} = \hat{\mathbf{U}}^T\hat{\mathbf{U}} = \mathbf{I}_n = \mathbf{V}^T\mathbf{V} = \hat{\mathbf{V}}^T\hat{\mathbf{V}}$, $\mathbf{S} = \mathrm{diag}(s_1, \cdots, s_n)$, and $\hat{\mathbf{S}} = \mathrm{diag}(\hat{s}_1, \cdots, \hat{s}_n)$. The covariance matrix can be represented

$$\mathbf{K} = \frac{1}{m}\hat{\mathcal{R}}^T\hat{\mathcal{R}} = \frac{1}{m}\hat{\mathbf{V}}\hat{\mathbf{S}}^2\hat{\mathbf{V}}^T,$$

and we see that the eigenvectors of $\mathbf{K}$, the columns of $\mathbf{Q}$ in Equation 4.2, are represented here by the columns of $\hat{\mathbf{V}}$, and the eigenvalues are the squared singular values.

We study the SVD of $\hat{\mathcal{R}}$ by considering the related symmetric system, $\hat{\mathcal{R}}^T\hat{\mathcal{R}}$, and observe that this system is a diagonal matrix plus a rank one matrix. By definition

$$\hat{\mathcal{R}} = (\mathbf{R} - \bar{\mathbf{R}})^T = (\mathbf{I} - \frac{1}{m}\mathbf{e}\mathbf{e}^T)\mathcal{R},$$

thus

$$\begin{aligned}
\hat{\mathcal{R}}^T\hat{\mathcal{R}} &= \hat{\mathcal{R}}^T(\mathbf{I} - \frac{1}{m}\mathbf{e}\mathbf{e}^T)^2\hat{\mathcal{R}} && (4.33)\\
&= \mathbf{V}\mathbf{S}\mathbf{U}^T(\mathbf{I} - \frac{1}{m}\mathbf{e}\mathbf{e}^T)\mathbf{U}\mathbf{S}\mathbf{V}^T && (4.34)\\
&= \mathbf{V}(\mathbf{S}^2 - \frac{1}{m}\mathbf{S}\mathbf{U}^T\mathbf{e}\mathbf{e}^T\mathbf{U}\mathbf{S})\mathbf{V}^T && (4.35)\\
&= \mathbf{V}(\mathbf{S}^2 - \frac{1}{m}\mathbf{z}\mathbf{z}^T)\mathbf{V}^T && (4.36)\\
&= \mathbf{V}(\mathbf{S}^2 + \rho\hat{\mathbf{z}}\hat{\mathbf{z}})\mathbf{V}^T && (4.37)
\end{aligned}$$

where $\hat{\mathbf{z}} = \mathbf{z}/\|\mathbf{z}\|_2$ and $\rho = -\frac{1}{m}\|\mathbf{z}\|_2^2$.

The matrix $\hat{\mathcal{R}}$ is obtained from an MD trajectory thus $r = \text{rank}(\hat{\mathcal{R}}) \leq n - 6$. Let $\mathbf{D} = \text{diag}(s_1^2, \cdots, s_r^2)$, then

$$\mathbf{S}^2 = \begin{bmatrix} \mathbf{D} & 0 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad \hat{\mathbf{z}} = \begin{bmatrix} \hat{\mathbf{z}}_1 \\ \hat{\mathbf{z}}_2 \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{z}}_1 \\ 0 \end{bmatrix}$$

and

$$\mathbf{S}^2 + \rho\hat{\mathbf{z}}\hat{\mathbf{z}}^T = \begin{bmatrix} \mathbf{D} & 0 \\ 0 & 0 \end{bmatrix} + \rho \begin{bmatrix} \hat{\mathbf{z}}_1\hat{\mathbf{z}}_1^T & 0 \\ 0 & 0 \end{bmatrix}.$$

It is well established[4] that the eigenvalues, λ_i, of $\mathbf{D} + \rho\hat{\mathbf{z}}_1\hat{\mathbf{z}}_1^T$ have the property

$$s_1^2 > \lambda_1 > s_2^2 > \cdots > s_r^2 > \lambda_r,$$

[4]**Theorem ([26])** *Suppose* $\mathbf{D} = diag(d_1, \cdots, d_k) \in \mathbb{R}^{k\times k}$ *and that the diagonal entries satisfy* $d_1 \geq \cdots \geq d_k$. *Assume that* $\rho \neq 0$ *and that* $\mathbf{z} \in \mathbb{R}^k$ *has no zero components. If* $\mathbf{Q} \in \mathbb{R}^{k\times k}$ *is*

and the eigenvectors, $\mathbf{w}^i$, are multiples of $(\mathbf{D} - \lambda_i \mathbf{I})^{-1}\hat{\mathbf{z}}_1$.

Let $\mathbf{W} = [\mathbf{w}^1, \cdots, \mathbf{w}^r] \in \mathbb{R}^{r \times r}$ be the matrix containing the eigenvectors of $\mathbf{D} + \rho\hat{\mathbf{z}}_1\hat{\mathbf{z}}_1^T$. The matrix containing the eigenvectors of $\mathbf{S}^2 + \rho\hat{\mathbf{z}}\hat{\mathbf{z}}^T$ is $\mathbf{X} = [\tilde{\mathbf{I}}_1\mathbf{W}, \tilde{\mathbf{I}}_2]$ where $\tilde{\mathbf{I}}_1 = [e^1 \cdots e^r]$ and $\tilde{\mathbf{I}}_2 = [e^{r+1} \cdots e^n]$, and e^i denotes the ith canonical vector

$$e^i = (\underbrace{0, \cdots, 0}_{i-1}, 1, \underbrace{0, \cdots, 0}_{n-i})^T \in \mathbb{R}^n.$$

If the orthogonal decomposition of $\mathbf{S}^2 + \rho\hat{\mathbf{z}}\hat{\mathbf{z}}^T$ is

$$\mathbf{S}^2 + \rho\hat{\mathbf{z}}\hat{\mathbf{z}}^T = \mathbf{X}\Lambda\mathbf{X}^T,$$

then $\hat{\mathbf{V}} = \mathbf{V}\mathbf{X}$ and the singular values of $\hat{\mathcal{R}}$ are $\hat{s}_i = \lambda_i^{1/2}$ for $i = 1 : r$ and $\hat{s}_i = 0$ for $i = r + 1 : n$. The first r columns of $\hat{\mathbf{U}}$ can be computed using the relationship $\hat{\mathbf{u}}^i = \frac{1}{\hat{s}_i}\hat{\mathcal{R}}\hat{\mathbf{v}}^i$ for $i = 1 : r$.

4.3.2 Truncated SVD

Let $\mathcal{R}_k$ and $\hat{\mathcal{R}}_k$ denote the best rank k approximations of $\mathcal{R}$ and $\hat{\mathcal{R}}$, respectively,

$$\mathcal{R} \approx \mathcal{R}_k = \mathbf{U}_k\mathbf{S}_k\mathbf{V}_k^T \text{ and } \hat{\mathcal{R}} \approx \hat{\mathcal{R}}_k = \hat{\mathbf{U}}_k\hat{\mathbf{S}}_k\hat{\mathbf{V}}_k^T.$$

In the previous section we saw that the right singular vectors of $\hat{\mathcal{R}}$ (the left singular vectors of $\mathbf{R} - \bar{\mathbf{R}}$) are the columns of $\hat{\mathbf{V}} = \mathbf{V}\mathbf{X}$, thus the matrix containing the first

---

orthogonal such that

$$\mathbf{Q}^T(\mathbf{D} + \rho\mathbf{z}\mathbf{z}^T)\mathbf{Q} = diag(\lambda_1, \cdots, \lambda_k)$$

with $\lambda_1 \geq \cdots \geq \lambda_k$ *and* $\mathbf{Q} = [\mathbf{q}^1, \cdots, \mathbf{q}^k]$, *then*

(a) The λ_i *are the* k *zeros of* $f(\lambda) = 1 + \rho\mathbf{z}^T(\mathbf{D} - \lambda\mathbf{I})^{-1}\mathbf{z}$.

(b) If $\rho > 0$, *then* $\lambda_1 > d_1 > \lambda_2 > \cdots > d_k$. *If* $\rho < 0$, *then* $d_1 > \lambda_1 > d_2 > \cdots > d_k > \lambda_k$.

(c) The eigenvector $\mathbf{q}^i$ *is a multiple of* $(\mathbf{D} - \lambda_i\mathbf{I})^{-1}\mathbf{z}$.

Index	Eigenvalues			
1 - 4	3.1266e+02	1.1626e+02	9.4376e+01	8.6457e+01
5 - 8	8.2372e+01	7.1588e+01	6.7394e+01	6.4604e+01
9 - 12	6.1301e+01	4.9726e+01	2.2584e+01	5.6031e+00
13 - 16	5.2369e+00	4.6835e+00	3.5589e+00	2.9849e+00
17 - 20	2.6260e+00	2.5335e+00	1.3294e+00	1.0537e+00
21 - 24	9.7310e-01	9.3433e-01	7.1021e-01	6.5308e-01
25 - 28	5.8324e-01	5.6634e-01	4.8389e-01	4.4330e-01
29 - 32	3.8222e-01	3.3230e-01	8.4873e-02	8.2981e-02
33 - 36	7.0623e-02	6.1314e-02	5.3039e-02	3.7945e-02

Table 4.1: First $3n_a - 6$ eigenvalues of a butane trajectory.

k columns of $\mathbf{V}$ can be represented by $\hat{\mathbf{V}}_k = \mathbf{V}\mathbf{X}_k$ where $\mathbf{X}_k$ represents the first k columns of $\mathbf{X}$. The columns of $\hat{\mathbf{V}}_k$ represent an orthogonal transformation of $\mathbf{X}_k$ with respect to $\mathbf{V}$. Recall that the columns of $\mathbf{V}$ form an orthogonal basis for the row space of $\mathcal{R}$ (column space of $\mathbf{R}$) which is the space of sampled atomic coordinates.

4.3.3 Choosing k

We close this chapter with a discussion of methods for choosing k, the dimension of the new coordinate space. The methods for choosing k described in PCA literature are generally ad hoc, thus the user must take care to understand the details of their problem when applying the methods. We describe here the scree graph, the log-eigenvalue (LEV) graph, and the cumulative percentage of total variation.

4.3.3.1 Scree Graph

A scree graph is a plot of the eigenvalues λ_i of $\mathbf{K}$ in nonincreasing order versus the eigenvalue index (see 4.2(a)). Cattell (1966) described these graphs as scree

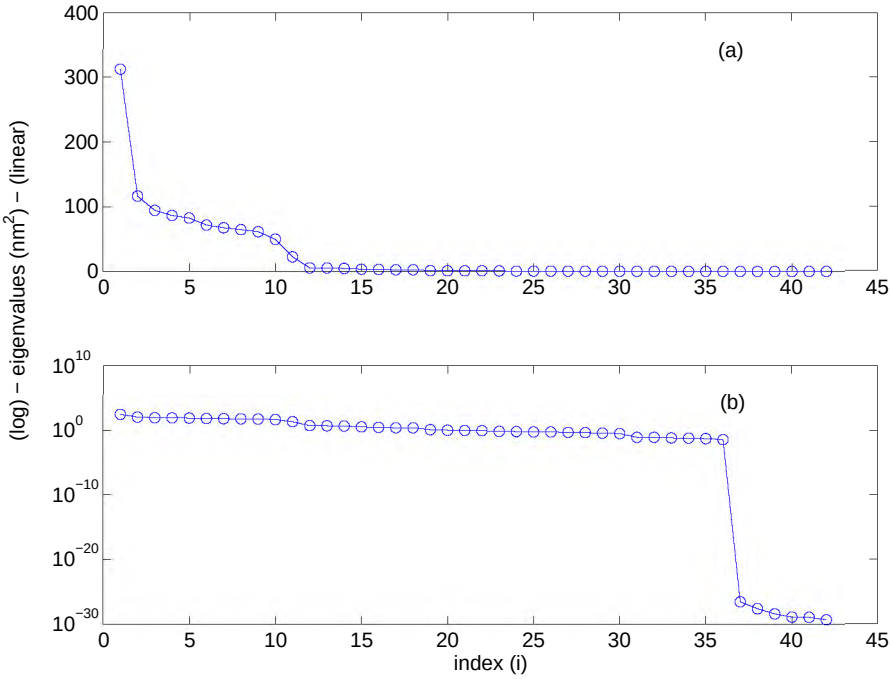


Figure 4.2: Eigenvalues of trajectory matrix (nm^2) in non-increasing order: (a) Scree Graph: Λ_{ii} versus i, (b) LEV Graph: $\log(\Lambda_{ii})$ versus i.

because such graphs often resemble a scree slope, a wedge-shaped accumulation of rock fragments at the base of a steep rock face [15]. A scree graph is used to determine graphically a value k. The general rule of thumb for the use of a scree graph is as follows. Compute the slope between eigenvalues i and $i+1$ for all i. When the slope is relatively constant, or graphically the plot resembles a straight line, the first index associated with this occurrence is chosen as the cut off k. In the case that the slope is relatively constant in multiple regions of the graph, the cut off is chosen with respect to the first such occurrence.

We provide a scree graph for the eigenvalues of a butane MD trajectory in Figure 4.2(a) (See also Table 4.1). Butane (C_4H_{10}) contains 14 atoms ($n_a = 14$). We use

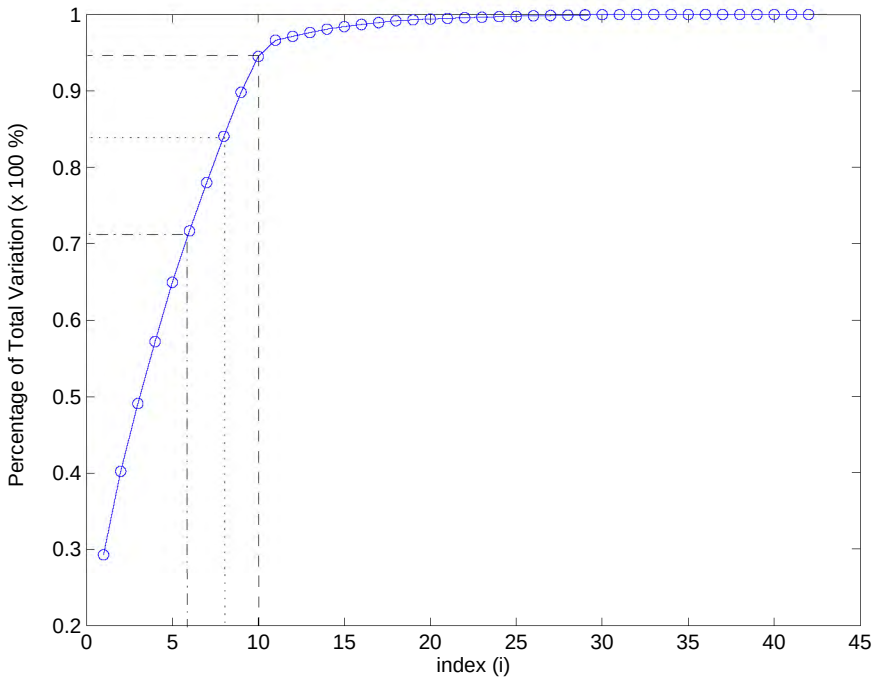


Figure 4.3: Cumulative percentage of the total variation captured by the first i principal components.

10, 000 configurations from a 25 ps simulation in this calculation. According to the rule just outlined, the cut off for our example is $k = 12$.

4.3.3.2 Log-Eigenvalue Graph

The LEV graph is a plot of $\log(\lambda_i)$ in non-increasing order versus the eigenvalue index (see 4.2(b)). This graph was named and discussed by Craddock and Flood (1969) when considering a meteorological application [16]. The rule of thumb for selecting a cut off when using a LEV graph is similar to that noted for use with a scree graph, however, generally the two methods produce different cut offs.

We provide the LEV graph for the eigenvalues of a butane MD trajectory in Figure 4.2(b). According to the rule for the use of a LEV graph the cut off is $k = 1$. Note

that setting the cut off to $k = 36$ is an obvious choice ($3n_a - 6 = 36$). This example illustrates that in the case of the scree graph and the LEV graph, if the slope of the graph gradually becomes less steep, the procedures become less effective in identifying a cut off location.

4.3.3.3 Cumulative Percentage of Total Variation

The cumulative percentage of the total variation captured by the first k principal components, or in the context of atomic motion, the cumulative percentage of atomic fluctuation associated with the first k principal vectors [2], is defined as follows

$$P_k = \frac{\sum_{i=1}^{k} \Lambda_{ii}}{\sum_{i=1}^{p} \Lambda_{ii}} = \frac{\sum_{i=1}^{k} s_{ii}^2}{\sum_{i=1}^{p} s_{ii}^2}$$

where $p = 3n_a - 6$. As a rule of thumb one can choose a cut off percentage P^* between 70% and 90% where the cut off k is the smallest integer k for which $P_k \geq P^*$ [37]. With respect to our butane example $P^* = 70\%$ results in a cut off of $k = 6$, $P^* = 80\%$ results in a cut off of $k = 8$, and $P^* = 90\%$ results in a cut off of $k = 10$ (see Figure 4.3).

We use the scree graph, the LEV graph and the cumulative percentage of the total variation in concert to choose cut off values k.

Chapter 5

Methods

In this chapter we present a method for molecular simulation that takes advantage of a reduced basis defined by SVD analysis of an MD trajectory. We seek to explore the configuration space in the directions "preferred" by a molecule and to suggest reasonable simulation parameters for test molecules. We begin by discussing in Section 5.1 one of the primary difficulties that we encountered in developing our reduced basis method. We then discuss the construction of a kD approximation to a potential energy surface in Section 5.2.

5.1 The Problem

A primary difficulty in developing a method for performing dynamics with respect to a reduced set of coordinates is the determination of a potential energy surface that is a function of these new coordinates. Given an MD trajectory that has been translated and rotated in the manner described in Section 3.3.2, we consider the truncated SVD

$$\hat{\mathcal{R}} \approx \mathbf{U}_k \mathbf{S}_k \mathbf{V}_k^T$$

where $\hat{\mathcal{R}}$ contains coordinates from an MD trajectory organized as described in Section 4.1.2.

The first step is to perform a coordinate transformation from Euclidean coordinates to coordinates in the range of $\mathbf{U}_k$. This requires the solution of the overdetermined systems $\mathbf{U}_k\mathbf{y}^i = \mathbf{r}^i$ where i is an index over a subset of the coordinates produced during MD. The vectors $\hat{\mathbf{y}}^i = \mathbf{U}_k^T\mathbf{r}^i \in \mathbb{R}^k$ are the least squares solutions and the vectors $\hat{\mathbf{r}}^i = \mathbf{U}_k\hat{\mathbf{y}}^i$ are the projections of the atomic coordinates, $\mathbf{r}^i$, onto the range of $\mathbf{U}_k$. We call $\hat{\mathbf{y}}^i$ the kD representation of atomic coordinates $\mathbf{r}^i$ with respect to $\mathbf{U}_k$.

Using these new coordinates, we make the substitution $\hat{\mathbf{r}} \approx \mathbf{r}$ into the equations of motion,

$$\mathrm{M}\ddot{\hat{\mathbf{r}}}(t) = \mathrm{M}\mathbf{U}_k\ddot{\hat{\mathbf{y}}}(t) = -\nabla\mathcal{V}(\hat{\mathbf{r}}(t)).$$

Multiplying on the left by $\mathbf{U}_k^T$ we obtain

$$\hat{\mathrm{M}}\ddot{\hat{\mathbf{y}}}(t) = -\mathbf{U}_k^T\nabla\mathcal{V}(\hat{\mathbf{r}}(t)) \tag{5.1}$$

where $\hat{\mathrm{M}} = \mathbf{U}_k^T\mathrm{M}\mathbf{U}_k$. In Equation 5.1, as in the original system, the potential energy function depends on $3n_a$ coordinates. An ideal scenario would be to efficiently solve Equation 5.1 taking advantage of the reduced representation given by $\hat{\mathbf{y}}^i$.

Our initial inclination was to consider the directional derivatives of the potential energy function in the directions of the first k left singular vectors. For h sufficiently

small

$$\mathbf{u}^{j\ T}\nabla\mathcal{V}(\mathbf{r}) = \lim_{\tau\to 0}\frac{1}{\tau}[\mathcal{V}(\mathbf{r}+\mathbf{u}^{j}\tau)-\mathcal{V}(\mathbf{r})] \tag{5.2}$$

$$\approx \frac{1}{h}[\mathcal{V}(\mathbf{r}+\mathbf{u}^{j}h)-\mathcal{V}(\mathbf{r})] \quad j=1,\cdots,k \tag{5.3}$$

where $\mathbf{u}^j$ represents the jth left singular vector, the jth column of $\mathbf{U}_k$. Equation 5.3 requires $k+1$ evaluations of the potential energy function per iteration to approximate the directional derivative. As noted in Section 2.2, the evaluation of the potential function is the dominant computational task during an iteration of MD simulation, and the partial derivatives of the potential energy function are calculated analytically using the chain rule (see Section 2.1.3). Thus given an evaluation of the potential energy function, it is straightforward to compute the partial derivatives. Since the approximation of the directional derivatives requires an increase in the number of potential function evaluations, we gain nothing computationally from this approximation.

A reformulation of the potential energy function with respect to the kD representation of the atomic coordinates has proven to be an intractable feat [9]. Our goal here is to present a method for approximating the potential energy surface of a particular molecule with respect to a kD representation of the atomic coordinates. In the following sections we outline a procedure that uses coordinate and energy information from a standard MD simulation coupled with SVD analysis.

5.2 Model of kD Energy Function

In Section 2.1 we introduced components of an empirical potential energy function. This function provides an approximation of the energy surface based on CHARMm parameterization [12] and its basic functional form is

$$\begin{aligned}\mathcal{V}(\mathbf{r}(t)) &= \sum_{bonds} k_b(b-b_0)^2 + \sum_{\substack{bond\\angles}} k_\theta(\theta-\theta_0)^2 + \sum_{\substack{dihedral\\angles}} |k_\phi| - k_\phi\cos(n\phi) \\ &+ \sum_{\substack{nonbonded\\pairs}} \left\{ 4\epsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right] + \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}} \right\}.\end{aligned}$$

In this section we outline a method for generating representative potential energies and forces with respect to kD coordinates.

5.2.1 Preliminary Study

We performed a preliminary study to observe the affects of projecting the $3n$D force generated by a standard MD simulation onto a kD space defined by PCA and used this information to update positions and velocities. The procedure outlined in Figure 5.1 shows how kD positions, velocities, and forces were initialized and updated. The final atomic coordinates and velocities from a standard MD trajectory served as the initial coordinates for the reduced system. The coordinates were adjusted to reflect the removal of translation and rotation.

The first three steps of Algorithm 4 initialize the kD system. After we update the kD coordinates in step 4 of Algorithm 4, we project the result onto the range of $\mathbf{U}_k$, $\mathbf{r}^i \equiv \mathbf{U}_k\tilde{\mathbf{r}}^i$. This serves as our approximate to the atomic ($3n$D) coordinates that

would have been produced during a standard MD simulation. We use $\mathbf{r}^i$ to compute the full force $\mathbf{f}^i = -\nabla\mathcal{V}(\mathbf{r}^i)$, which is then used to define force with respect to kD coordinates, $\tilde{\mathbf{f}}^i = \mathbf{U}_k^T\mathbf{f}^i$. We use this information to update the kD velocities in the current iteration and kD coordinates in the next iteration.

In a truly reduced simulation we would not compute the full force during each iteration since it is the computationally intensive portion of an MD simulation that we wish to overcome. The goal of this exercise was to provide some insight into the nature of atomic activity with respect to a kD representation of a molecular system. Recall that in the new coordinate system we are no longer considering individual atomic positions but the system as a whole with respect to independent degrees of freedom.

Our task then is to replace step 4b in Algorithm 4 with a function with respect to $\tilde{\mathbf{r}}$ that provides a reasonable approximate of $\mathbf{U}_k\mathbf{f}$ without computing $\mathbf{f} = -\nabla\mathcal{V}(\mathbf{r})$ at each step of iteration. Such a function is provided by the linear least squares model introduced in the next section.

Recall the kD representation of atomic coordinates, $\mathbf{r}^i$, with respect to $\mathbf{U}_k$ is $\mathbf{y}^i = \mathbf{U}_k^T\mathbf{r}^i$. Here we drop the $\hat{}$ notation. The potential energy at conformation $\mathbf{r}^i$ is represented by $\mathcal{V}(\mathbf{r}^i)$ and is calculated during an MD simulation. We associate this energy with the kD coordinate, $\mathbf{y}^i$, and seek to generate a function, $\mathcal{G}$, that follows the rule

$$\mathcal{G}(\mathbf{y}^i) \equiv \mathcal{V}(\mathbf{r}^i). \tag{5.4}$$

With this goal in mind we construct a linear least squares model with respect to the data $\{\mathbf{y}^i, \mathcal{V}(\mathbf{r}^i)\}$. Step 4b in Algorithm 4 is then replaced by

$$\tilde{\mathbf{f}}^{\tau+1} = -\nabla \mathcal{G}(\tilde{\mathbf{r}}^{\tau+1}).$$

Algorithm 4 Simulation in kD

Input: $\mathbf{r}$, $\mathbf{v}$, $\mathbf{U}_k$, $\hat{\mathrm{M}} \equiv \mathbf{U}_k^T \mathrm{M} \mathbf{U}_k$
Output: $\mathbf{r}$, $\mathbf{v}$

1. $\mathbf{f}^0 =$ **call** Force$(\mathbf{r}^0)$, $\quad \tilde{\mathbf{f}}^0 = \mathbf{U}_k^T \mathbf{f}$
2. $\tilde{\mathbf{r}}^0 = \mathbf{U}_k^T \mathbf{r}$, $\quad \tilde{\mathbf{v}}^0 = \mathbf{U}_k^T \mathbf{v}$,
3. $\tilde{\mathbf{a}}^0 = \hat{\mathrm{M}}^{-1} \tilde{\mathbf{f}}^0$
4. **for** $\tau = 0 : n_{ts} - 1$
   a. **for** $i = 1 : n_{atoms}$
      $\tilde{\mathbf{r}}_i^{\tau+1} = \tilde{\mathbf{r}}_i^{\tau} + h\tilde{\mathbf{v}}_i^{\tau} + \frac{h^2}{2}\tilde{\mathbf{a}}_i^{\tau}$, $\quad \tilde{\mathbf{v}}_i^{\tau+\frac{1}{2}} = \tilde{\mathbf{v}}_i^{\tau} + \frac{h}{2}\tilde{\mathbf{a}}_i^{\tau}$
      end
   b. $\mathbf{r}^{\tau+1} = \mathbf{U}_k \tilde{\mathbf{r}}^{\tau+1}$
      $\mathbf{f}^{\tau+1} =$ **call** Force$(\mathbf{r}^{\tau+1})$, $\quad \tilde{\mathbf{f}}^{\tau+1} = \mathbf{U}_k^T \mathbf{f}^{\tau+1}$
   c. $\tilde{\mathbf{a}}^{\tau+1} = \hat{\mathrm{M}}^{-1} \tilde{\mathbf{f}}^{\tau+1}$
      for $i = 1 : n_{atoms}$
      $\tilde{\mathbf{v}}_i^{\tau+1} = \tilde{\mathbf{v}}_i^{\tau+\frac{1}{2}} + \frac{h}{2}\tilde{\mathbf{a}}_i^{\tau+1}$
      end
   end

Figure 5.1: Procedure used to project $3n$D coordinates onto kD space and update using full force.

5.2.2 General Linear Least Squares Problem

We construct a model that has the general form

$$\mathcal{G}(\mathbf{y}) = \sum_{j=1}^{M} a_j g_j(\mathbf{y}) \tag{5.5}$$

where $g_j : \mathbb{R}^k \rightarrow \mathbb{R}$ are basis functions and the coefficient $a_j \in \mathbb{R}$ is the weight of the jth basis function. This model is linear in the sense that it depends on the parameters a_j linearly, i.e., the model is a linear combination of the basis functions. We seek to model the function that generated the data $\mathcal{V}(\mathbf{r}^i)$ given input $\mathbf{y}^i$. Given an appropriate selection of basis functions, the coefficients a_j must be chosen in such a way that $\mathcal{G}$ obeys the rule given in Equation 5.4 as closely as possible.

The "chi-square" merit function, denoted χ^2, given in Equation 5.6, provides a measure of how well Equation 5.5 agrees with the data, $\{\mathbf{y}^i, \mathcal{V}(\mathbf{r}^i)\}$,

$$\chi^2 = \sum_{i=1}^{N} \left[\frac{E_i - \sum_{j=1}^{M} a_j g_j(\mathbf{y}^i)}{\sigma_i} \right]^2 . \tag{5.6}$$

Here N is the number of variable vectors $\mathbf{y}^i$, $E_i \equiv \mathcal{V}(\mathbf{r}^i)$ is the energy with respect to conformation $\mathbf{r}^i$, and σ_i is the standard deviation of the ith data vector. In principle, the smaller the value χ^2, the better Equation 5.5 models the underlying function that generated the data. Parameters a_j in Equation 5.5 are selected such that χ^2 is minimized over all vectors $[a_1 \cdots a_M]$. It is generally expected that on average the value $\chi^2 \approx \nu$ where $\nu = N - M$ [53].

The system of interest is constructed as follows. A design matrix, $\mathbf{D} \in \mathbb{R}^{N \times M}$, is built where the ijth component is the jth basis function evaluated at the ith kD

coordinate divided by the energy sample standard deviation

$$\mathbf{D}_{ij} = \frac{g_j(\mathbf{y}^i)}{\sigma_i}.$$

Note that N, the number of data points, is larger than M, the number of basis functions. We construct a vector $\mathbf{b} \in \mathbb{R}^N$ where the ith component is the target value of the ith coordinate, E_i, scaled by σ_i,

$$\mathbf{b}_i = \frac{E_i}{\sigma_i},$$

and the vector $\mathbf{a} \in \mathbb{R}^M$ represent the coefficients, a_j, to be determined. A least squares solution to the system

$$\mathbf{Da} = \mathbf{b}, \tag{5.7}$$

is a vector, $\mathbf{a}^* \in \mathbb{R}^M$, that minimizes χ^2.

We present methods for solving a least squares system in Section 5.2.2.1 (the method of normal equations) and Section 5.2.2.2 (the SVD method). We then turn our attention to choosing appropriate basis functions for our model. Our model must be differentiable everywhere, thus the basis functions we choose must be differentiable We also want to choose functions that deal well with multidimensional scattered data. In Section 5.2.3 we introduce radial functions which serve as our basis functions.

5.2.2.1 Solution using Normal Equations

A necessary condition for $\mathbf{a} = \mathbf{a}^*$ to be a local minimizer of χ^2 (see Equation 5.6) is that the gradient of χ^2 evaluated at $\mathbf{a}^*$ equals the zero vector ($\in \mathbb{R}^M$). This condition

leads to the following system of equations in matrix notation

$$\mathbf{D}^T\mathbf{D}\mathbf{a} = \mathbf{D}^T\mathbf{b}. \tag{5.8}$$

Equation 5.8 denotes the normal equations of the least squares problem. If $\mathbf{D}$ is full rank, then $\mathbf{D}^T\mathbf{D}$ is nonsingular, and the linear system can be solved using Cholesky factorization[1]. Specifically, let $\mathbf{C} = \mathbf{D}^T\mathbf{D}$ and $\mathbf{d} = \mathbf{D}^T\mathbf{b}$. Given the Cholesky decomposition of $\mathbf{C}$, the least squares solution can be obtained by solving the following triangular system

$$\mathbf{G}\mathbf{q} = \mathbf{d}$$

and using its solution $\mathbf{q}^*$ in the triangular system

$$\mathbf{G}^T\mathbf{a} = \mathbf{q}^*$$

to obtain the least squares solution $\mathbf{a}_{LS}$. However, information about the original LS system is lost in the formation of $\mathbf{C}$. Furthermore, if $\mathbf{D}$ is ill-conditioned or rank deficient this method is quite inaccurate. The SVD is used in these cases. In the next section we present a theorem which outlines the use of the SVD to solve the least squares problem.

5.2.2.2 Solution using the SVD

Given the SVD of the design matrix, $\mathbf{D}$, the least squares solution to Equation 5.7 is obtained as noted in Theorem 5.2.1.

[1] **Theorem (Cholesky Factorization [26])** *If $\mathbf{C} \in \mathbb{R}^{M\times M}$ is symmetric positive definite, then there exists a unique lower triangular matrix $\mathbf{G} \in \mathbb{R}^{M\times M}$ with positive diagonal entries such that $\mathbf{C} = \mathbf{G}\mathbf{G}^T$.*

Theorem 5.2.1 (SVD Solution to Least Squares Problem [26]) *Suppose* $\mathbf{U}^T\mathbf{D}\mathbf{V} = \mathbf{S}$ *is the SVD of* $\mathbf{D} \in \mathbb{R}^{N\times M}$ *with* $r = rank(\mathbf{D})$. *If* $\mathbf{U} = [\mathbf{u}^1, \cdots, \mathbf{u}^N]$ *and* $\mathbf{V} = [\mathbf{v}^1, \cdots, \mathbf{v}^M]$ *are column partitionings and* $\mathbf{b} \in \mathbb{R}^N$, *then*

$$\mathbf{a}_{LS} = \sum_{i=1}^{r} \frac{(\mathbf{u}^i)^T\mathbf{b}}{s_i}\mathbf{v}^i \tag{5.9}$$

minimizes $\|\mathbf{D}\mathbf{a} - \mathbf{b}\|_2$ *and has the smallest 2-norm of all minimizers. Moreover*

$$\rho^2_{LS} = \|\mathbf{D}\mathbf{a}_{LS} - \mathbf{b}\|_2^2 = \sum_{i=r+1}^{N} ((\mathbf{u}^i)^T\mathbf{b})^2 \tag{5.10}$$

where ρ^2_{LS} *denotes the least squares residual.*

This solution method does not require the design matrix to be full rank. Furthermore, it provides an efficient formula for computing the least squares residual. We now turn our attention to describing the basis functions used in our model.

5.2.3 Radial Functions

Radial functions are a class of functions $g : \mathbb{R}^m \to \mathbb{R}$ with the general form

$$g(\|\mathbf{x} - \mathbf{c}\|; \eta).$$

The vector $\mathbf{c} \in \mathbb{R}^m$ is the center of the radial function, $\eta \in \mathbb{R}$ is the width, or scale parameter, and $\|\cdot\|$ is a vector norm. Examples of radial functions include

1. the Gaussian function: $g(z) = \exp(-\frac{z^2}{\eta^2})$,
2. the multiquadric function: $g(z) = (1 + \frac{z^2}{\eta^2})^{1/2}$,

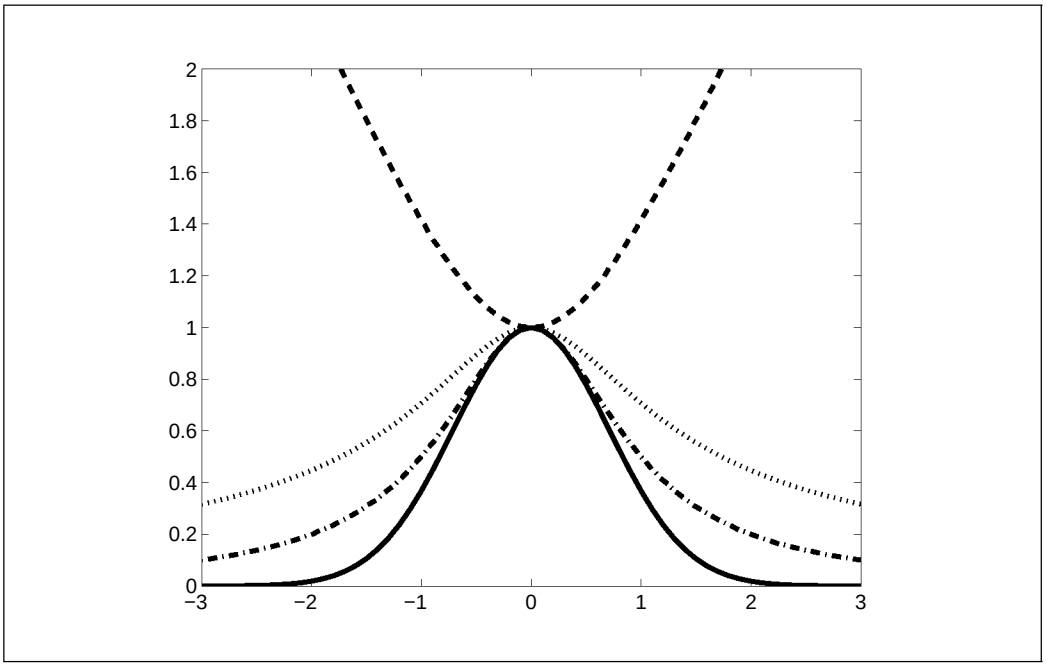


Figure 5.2: One dimensional Gaussian (—), multiquadric (- - -), inverse multiquadric ($\cdots$), and Cauchy (-·-) radial functions with center $c = 0$ and scale $\eta = 1$.

3. the inverse multiquadric function: $g(z) = (1 + \frac{z^2}{\eta^2})^{-1/2}$, and

4. the Cauchy function: $g(z) = (1 + \frac{z^2}{\eta^2})^{-1}$,

where $z = \|\mathbf{x} - \mathbf{c}\|$. These functions are shown in Figure 5.2 for one dimensional input using the Euclidean norm with $c = 0$ and $\eta = 1$. These functions are symmetric about the center and the output values decrease (or increase) as input values move away from the center. Specifically, as $|x - c|$ increases, $|g(|x - c|)|$ decreases (or increases).

Radial functions with local responses, that is, functions that give a significant response only in a neighborhood near its center, such as the Gaussian function (see Example 1), are particularly attractive in biological modelling, because in such cases a response is typically finite [49]. Linear combinations of radial functions model a

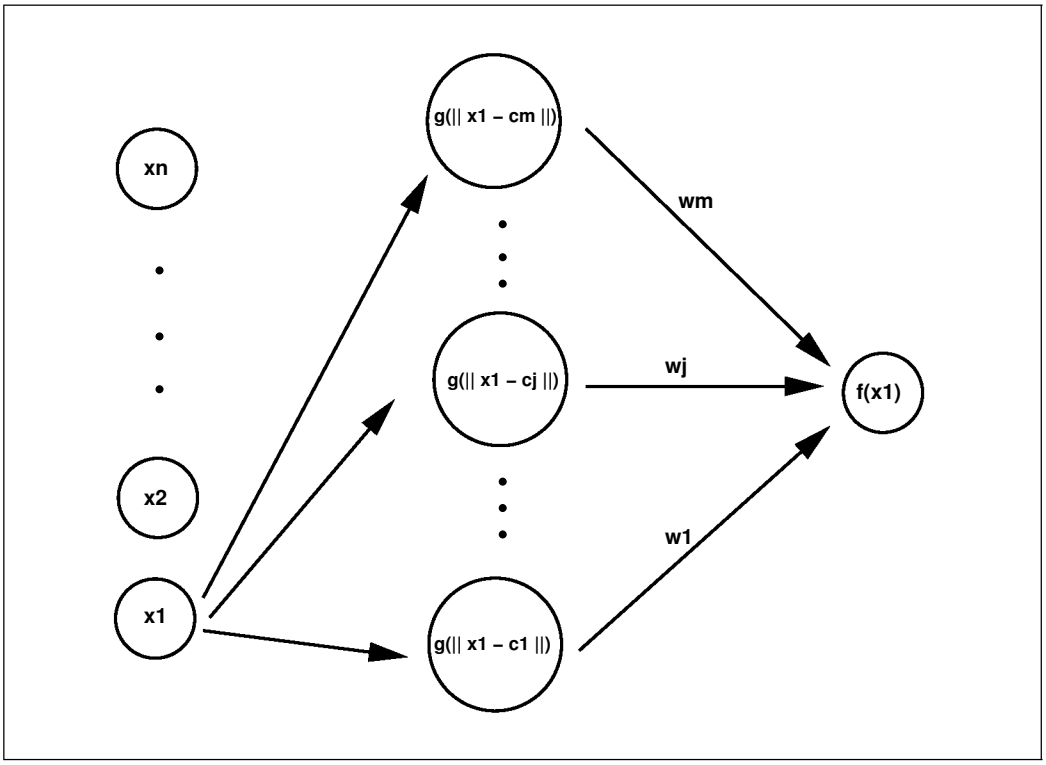


Figure 5.3: A RBFN is a feedforward network with three layers - an input layer ($\mathbf{x}^i$), a kernel layer ($g(\|\mathbf{x}-\mathbf{c}^j\|)$), and an output layer ($f(\mathbf{x}^i)$). Here w_j is the weight applied to basis function $g(\|\mathbf{x}-\mathbf{c}^j\|)$ and $j = 1, \cdots, m$.

large class of functions ([24], [32], [50], [51]) and have the general form

$$f(\mathbf{x}) = \sum_{j=1}^{M} w_i g(\|\mathbf{x}-\mathbf{c}^j\|). \tag{5.11}$$

Equation 5.11 is called a radial basis function (RBF) network in neural network literature. An RBF network is a feedforward network with three layers - an input layer, a kernel layer composed of m basis functions, and an output layer consisting of the resulting function. Figure 5.3 displays a schematic of an RBFN. When the input value is $\mathbf{x}^1$ each of the functions on the kernel layer are evaluated at $\mathbf{x}^1$. The resulting values are multiplied by the appropriate weight and then summed to obtain

the output, $f(\mathbf{x}^1)$.

When designing an RBF network one must select basis functions, g_j, with associated widths, η_j, the number of basis functions, and the location of RBF centers, $\mathbf{c}^j$. The selection of the basis functions is generally dependent on the particular application. Determining the number of basis functions to use in a model is related to the model order selection problem [23]. For initialization purposes we chose the number of basis functions based on the distribution of the data.

After selecting the type and number of RBFs to include in the model, the centers ($\mathbf{c}^j$), the widths (η_j), and the weights (w_j) must be determined. The process of selecting these parameters, often referred to as training an RBF network, is outlined in the following section.

5.2.3.1 Training an RBF Network

We present a two-stage training method to compute centers, widths and weights for an RBF network. As the name of this method suggests the training procedure is performed in two stages. The centers and widths are defined during the first stage of training, and the weights are determined using the centers and widths obtained from the first stage of training. Only the independent (input) variables are used during the first stage of training, thus this process is called unsupervised learning.

The goal of the first stage of training is to determine the placement of basis functions that best represents the input data and its density. Particularly, we want

to select centers for the RBF in locations where there is data and set the widths of the basis functions to represent the local variance of the data taking into account nearby basis functions.

There are various alternatives for selecting basis function centers. The simplest method is to select a random subset of the data points to serve as centers. At first pass this is generally a reasonable initial guess when using a method that iteratively alters the centers. However, this selection procedure is only as good as the centers are representative of the entire data set. A representative choice of centers can be obtained by clustering the input data. More precisely, given N data points $\mathbf{x}^i$ in $\mathbb{R}^m$, our goal is to generate M clusters, or subsets, of the data such that within the clusters some similarity criterion is satisfied.

We use a k-means algorithm to cluster the kD representations into M groups, where M is some predefined factor of the number of data points. k-means is a least squares partitioning method that divides a collection of data points into k groups [30]. Particularly, a k-means algorithm divides the data into clusters such that within each cluster the sum-of-squares criterion

$$\mathcal{S}_M = \sum_{j=1}^{M} \sum_{i \in G_j} \|\mathbf{x}^i - \bar{\mathbf{x}}^j\|^2$$

achieves a local minimum [31]. The symbol $\bar{\mathbf{x}}^j$ denotes the geometric centroid of the cluster G_j. We randomly choose M data points to serve as the initial centroids. The k-means algorithm then randomly assigns data points $\mathbf{x}^i$ to clusters, computes the centroids and iteratively alters the cluster assignments until an algorithm specific

stopping criterion is achieved.

The widths associated with each basis function play an important role in determining the overall nature of a RBFN. In the simplest case all of the widths can be set to the same value. For example, the widths can be set to a multiple of the average distance between all centers

$$\eta = \frac{\kappa}{M} \sum_{i,j;i<j} \|\mathbf{c}^i - \mathbf{c}^j\| \quad \text{for all i}$$

where $\kappa \in \mathbb{R}^+$ and i, j are indices over the centers. It is more customary to vary the widths taking into account nearby basis functions. A straight forward width selection method sets the width of the ith basis function to a multiple of the average distance of its center, $\mathbf{c}^i$, from L of its nearest neighbors

$$\eta_i = \frac{\kappa}{L} \sum_{j}^{L} \|\mathbf{c}^i - \mathbf{c}^j\|.$$

Once the centers for the basis functions and their associated widths are set, Equation 5.11 reduces to the general linear least squares problem in Equation 5.5 where

$$g_j(\mathbf{y}) \equiv g(\|\mathbf{y} - \mathbf{c}^j\|)$$

and $a_j \equiv w_j$ are the weights that will be determined via the SVD solution to the least squares problem.

5.2.4 SVD Updating

We would like to incorporate new information obtained from reduced simulation into the basis describing the reduced representation. This is accomplished by updating

Algorithm 5 SVD Updating Algorithm

Input: $\mathbf{A}_k = \mathbf{U}_k\mathbf{S}_k\mathbf{V}_k^T \in \mathbb{R}^{m\times n}$, $\mathbf{A}_{new} \in \mathbb{R}^{m\times j}$
Output: $\mathbf{S}_k^+$, $\mathbf{U}_k^+$, $\mathbf{V}_k^+$

1. compute $QR = (I - \mathbf{U}_k\mathbf{U}_k^T)\mathbf{A}_{new}$
2. compute $B = \mathbf{U}_k^T\mathbf{A}_{new}$
3. build $D = \begin{bmatrix} \mathbf{S}_k & \mathbf{U}_k^T\mathbf{A}_{new} \\ 0 & R \end{bmatrix}$
4. compute $\hat{\mathbf{U}}_k\hat{\mathbf{S}}_k\hat{\mathbf{V}}_k^T \approx D$
5. update $\mathbf{S}_k^+ \leftarrow \hat{\mathbf{S}}_k$, $\mathbf{U}_k^+ \leftarrow [\mathbf{U}_k, Q]\hat{\mathbf{U}}_k$, $(\mathbf{V}_k^+)^T \leftarrow \hat{\mathbf{V}}_k^T \begin{bmatrix} \mathbf{V}_k^T & 0 \\ 0 & I \end{bmatrix}$.

Figure 5.4: SVD updating algorithm.

the SVD.

Consider the TSVD of a matrix, $\mathbf{A} \approx \mathbf{A}_k = \mathbf{U}_k\mathbf{S}_k\mathbf{V}_k^T \in \mathbb{R}^{m\times n}$. Suppose we append j new columns, $\mathbf{A}_{new} \in \mathbb{R}^{m\times j}$, to $\mathbf{A}_k$. Our goal is to compute the TSVD of

$$[\mathbf{A}_k, \mathbf{A}_{new}] \in \mathbb{R}^{m\times n+j} \tag{5.12}$$

taking advantage of the known SVD of $\mathbf{A}_k$ ([14], [13]).

This updating procedure combines QR-updating with the SVD [47]. Consider the following factorization of the updated matrix

$$[\mathbf{U}_k\mathbf{S}_k\mathbf{V}_k^T, \mathbf{A}_{new}] = [\mathbf{U}_k, Q] \begin{bmatrix} \mathbf{S}_k & \mathbf{U}_k^T\mathbf{A}_{new} \\ 0 & R \end{bmatrix} \begin{bmatrix} \mathbf{V}_k^T & 0 \\ 0 & I \end{bmatrix} \tag{5.13}$$

where $QR = (I - \mathbf{U}_k\mathbf{U}_k^T)\mathbf{A}_{new}$, $Q^TQ = I_j$ and R is $j \times j$ upper triangular. The TSVD

of the second matrix in the factorization is computed:

$$\hat{\mathbf{U}}_k \hat{\mathbf{S}}_k \hat{\mathbf{V}}_k^T \approx \begin{bmatrix} \mathbf{S}_k & \mathbf{U}_k^T \mathbf{A}_{new} \\ 0 & R \end{bmatrix} \in \mathbb{R}^{(k+j)\times(k+j)}. \tag{5.14}$$

Using this new decomposition and previously computed information, we obtain the matrices $\mathbf{U}_k^+$, $\mathbf{S}_k^+$, and $\mathbf{V}_k^+$.

The updating procedure is summarized in Figure 5.4. To obtain an updated TSVD we compute a TSVD in Equation 5.14, a small matrix relative to the original matrix Equation 5.13, and perform matrix multiplications. Once new SVD information is obtained, we must again construct a kD model of the energy function based on the updated left singular vectors.

Chapter 6

Analysis of Reduced Simulation

In the previous chapter we presented a kD model of an empirical potential energy surface. Given an MD trajectory we compute a linear least squares model based on a kD representation of the trajectory and the values of the potential function evaluated at the conformations in the trajectory. The model is as follows

$$\mathcal{G}(\mathbf{y}) = \sum_{j=1}^{M} a_j g_j(\mathbf{y}) \tag{6.1}$$

where $g_j(\mathbf{y}) = \exp(-\|\mathbf{y} - \mathbf{y}^j\|^2/\eta_j^2)$ is the jth of M basis functions.

Using this model we perform simulations with respect to kD coordinates. Specifically, we solve the following system numerically

$$\hat{\mathrm{M}}\ddot{\mathbf{y}} = -\nabla\mathcal{G}(\mathbf{y}). \tag{6.2}$$

The solution of Equation 6.2 is approximated using the velocity Verlet method as is the case with our standard MD simulation. Our present goal is to access the behavior

and performance of this method in comparison to a standard MD simulation. To this end we consider the following questions:

- How well does $\mathcal{G}(\mathbf{y})$ approximate $\mathcal{V}(\mathbf{r})$ where $\mathbf{y} = \mathbf{U}_k^T\mathbf{r}$?
- What is the local behavior of Equation 6.2?

We address the first question in Section 6.1 where we discuss the approximation properties of RBF networks. In Section 6.2 we provide an overview of results about the local error associated with the use of a symplectic integrator. Finally, we discuss the error associated with the force approximation provided by the gradient of the RBF network and other topics for future consideration in Section 6.3.

6.1 RBF and the Best Approximation Property

In this section we discuss the approximation properties of RBF networks. We discuss the concept of best approximation and the associated error bounds. Girosi *et al.* showed that when M, the number of basis functions, equals N, the number of data points, RBF networks can approximate continuous functions arbitrarily well [24].

More precisely, given a function ψ belonging to a set of functions $\mathcal{E}$ and a non-empty subset, $\mathcal{E}_0 \subset \mathcal{E}$, the approximation problem is to find a function in $\mathcal{E}_0$ that best approximates ψ. The distance of ψ from $\mathcal{E}_0$ is defined

$$d(\psi, \mathcal{E}_0) = \inf_{f \in \mathcal{E}_0} d(\psi, f)$$

where $d(\psi, f)$ is a distance function defined on $\mathcal{E}$. If there exists a $f^* \in \mathcal{E}_0$ such that

$$d(\psi, \mathcal{E}_0) = d(\psi, f^*),$$

f^* is called the best approximation of ψ in $\mathcal{E}_0$. If for each $\psi \in \mathcal{E}$ there is at least one $f \in \mathcal{E}_0$ that is a best approximation of ψ, the set $\mathcal{E}_0$ is an existence set. The set is a uniqueness set if at most one $f \in \mathcal{E}_0$ is a best approximation of ψ. A set that is both an existence set and a uniqueness set is called a Tchebycheff set. Girosi *et al.* proved that if $\mathcal{E}_0$ is a compact set in a metric space $\mathcal{E}$, then $\mathcal{E}_0$ is an existence set [24].

When considering Gaussian radial functions the approximated solution belongs to G^M, the set of Gaussian superpositions,

$$G^M \equiv \{\mathcal{G} \in C[\mathbf{X}] | \mathcal{G}(\mathbf{x}) = \sum_{j=1}^{M} a_j \exp\left(\frac{\|\mathbf{y} - \mathbf{c}^j\|}{\eta_j}\right)^2 ; \mathbf{y}, \mathbf{c}^j \in \mathbb{R}^k; a_j, \eta_j \in \mathbb{R}\},$$

where $C[\mathbf{X}]$ denotes the set of continuous functions on $\mathbf{X} \subset \mathbb{R}^k$. The set $G^M \subset C[\mathbf{X}]$ is an existence set, and if $C[\mathbf{X}]$ is strictly convex, G^M is a Tchebycheff set. This is the case when we consider $C[\mathbf{X}]$ with an L_p-norm for $1 < p < \infty$ [55]. Furthermore, Girosi *et al.* showed that G^M is dense in $C[\mathbf{X}]$ when $\mathbf{X}$ is a compact subset of $\mathbb{R}^k$ [24]. Thus RBF networks with Gaussian radial functions can approximate continuous functions arbitrarily well.

In most cases we are considering data with noise or large data sets, thus it is computationally infeasible to consider the linear system that results from Equation 6.1 when $M = N$. When considering noisy data or a large data set we seek a function approximation in the form of Equation 6.1 with $M < N$. In order to maintain the

best approximation property for the set of Gaussian superpositions when the number of basis functions is less than the number of data points, we implement a two step training method. The first step fixes the centers of the radial functions and the second step computes the weights a_j (see Section 5.2.3.1).

Barron [6] derived a lower bound on the error associated with the best approximation provided by an RBF network. The error can be characterized as follows

$$\sup_{\psi \in \mathcal{E}} d(\psi, \text{span}\{g_1, g_2, \cdots, g_M\}) \geq \kappa \frac{c}{k} \left(\frac{1}{M}\right)^{\frac{1}{k}} \tag{6.3}$$

where κ is a universal constant such that $\kappa \geq \frac{1}{8\pi \exp(\pi-1)}$, c is a constant that depends on the basis functions, k is the dimension of the input, and M is the number of basis functions. This is the worst case scenario based on a fixed number of basis functions chosen with no prior knowledge of ψ. That is, the basis functions are not adjusted during the training process. Better error bounds are obtained if the basis functions are allowed to adjust during the training process. However, for our present goal, obtaining a reduced representation of the equations of motion, a fixed basis approximation of a potential energy surface with respect to reduced coordinates provides adequate information.

6.2 Störmer/Verlet Method

In this section we provide an overview of backward error analysis for numerical symplectic integrators presented in [29] and consider the Störmer/Verlet method. Con-

sider an ordinary differential equation

$$\dot{\mathbf{c}} = f(\mathbf{c}), \quad \mathbf{c}(0) = \mathbf{c}_0. \tag{6.4}$$

The goal of backward error analysis is to identify a perturbed or modified differential equation, $\dot{\tilde{\mathbf{c}}} = f_h(\tilde{\mathbf{c}})$, where

$$\dot{\tilde{\mathbf{c}}} = f(\tilde{\mathbf{c}}) + hf_2(\tilde{\mathbf{c}}) + h^2 f_3(\tilde{\mathbf{c}}) + \cdots, \tag{6.5}$$

whose exact solution is the same as the approximate solution of the original differential equation. Then an optimal truncation of Equation 6.5 is determined

$$\dot{\tilde{\mathbf{c}}} = F_N(\tilde{\mathbf{c}}), \qquad F_N(\tilde{\mathbf{c}}) = f(\tilde{\mathbf{c}}) + hf_2(\tilde{\mathbf{c}}) + \cdots + h^{N-1} f_N(\tilde{\mathbf{c}}), \tag{6.6}$$

and an error bound for the difference between the numerical and exact solutions of the truncated differential equation is presented (see Theorem 6.2.4). It is assumed that $f(\mathbf{c})$ is analytic in a complex neighborhood of $\mathbf{c}_0$ and that

$$\|f(\mathbf{c})\| \leq \upsilon \quad \text{for all} \quad \mathbf{c} \in B_{2\rho}(\mathbf{c}_0),$$

where $B_{2\rho}(\mathbf{c}_0) \equiv \{\mathbf{c} \in \mathcal{C}^D : \|\mathbf{c} - \mathbf{c}_0\| \leq 2\rho\}$. Recall that a Hamiltonian system has the form

$$\dot{\mathbf{p}} = -H_q(\mathbf{p}, \mathbf{q}), \quad \dot{\mathbf{q}} = H_p(\mathbf{p}, \mathbf{q}) \tag{6.7}$$

where $\mathbf{c} = (\mathbf{p}, \mathbf{q})^T \in \mathbb{R}^{2d}$ and $f(\mathbf{c}) = (-H_q(\mathbf{p}, \mathbf{q}), H_p(\mathbf{p}, \mathbf{q}))^T$. Let $p(t, p_0, q_0)$, $q(t, p_0, q_0)$ be the solution of the system corresponding to the initial values $p(0) = p_0$, $q(0) = q_0$.

Definition 6.2.1 The *flow* over time t is a mapping, $\varphi_t : \mathcal{U} \to \mathbb{R}^{2d}$, that advances the solution by time t and is denoted by

$$\varphi_t(p_0, q_0) = (p(t, p_0, q_0), q(t, p_0, q_0)),$$

where $\mathcal{U} \subset \mathbb{R}^{2d}$ is an open set.

The solutions to a differential equation produced by a numerical method produces a discrete flow.

Definition 6.2.2 A formula to approximate the solution to a differential equation (6.4), denoted by the mapping $\Phi_h : \mathbf{c}^i \mapsto \mathbf{c}^{i+1}$, is called the *discrete* or *numerical flow* and h is the time step of the method.

Recall the Störmer/Verlet method which when applied to the equations of motion, iteratively updates positions, $\mathbf{r}$, with respect to each component as follows

$$\mathbf{r}^{i+1} = 2\mathbf{r}^i - \mathbf{r}^{i-1} + h^2\mathbf{a}^i.$$

A one-step implementation of this method updates positions and velocities, $\mathbf{v}$, as follows

$$\mathbf{v}^{i+1/2} = \mathbf{v}^i + \frac{h}{2}\mathbf{a}^i \tag{6.8}$$

$$\mathbf{r}^{i+1} = \mathbf{r}^i + h\mathbf{v}^{i+1/2} \tag{6.9}$$

$$\mathbf{v}^{i+1} = \mathbf{v}^{i+1/2} + \frac{h}{2}\mathbf{a}^{i+1}. \tag{6.10}$$

Equations 6.8 through 6.10 represent the numerical flow of the Störmer/Verlet method.

6.2.1 Symplectic Integrator

Definition 6.2.3 A differential map $z : \mathcal{U} \to \mathbb{R}^{2d}$ is called *symplectic* if the Jacobian matrix $z'(p,q)$ is everywhere symplectic, that is, if

$$z'(p,q)^T J z'(p,q) = J \quad \text{where} \quad J = \begin{pmatrix} 0 & \mathrm{I} \\ -\mathrm{I} & 0 \end{pmatrix},$$

I is the identity matrix of dimension d and $\mathcal{U} \subset \mathbb{R}^{2d}$ is an open set.

A numerical one-step method is called *symplectic* if the one-step map, Φ_h, is symplectic whenever the method is applied to a smooth Hamiltonian system

Theorem 6.2.1 *The Störmer/Verlet scheme in Equations 6.8 through 6.10 is a symplectic method of order 2.*

Proof: See [29] (see also [59]).

The following results pertain to the symplecity of the flow of a Hamiltonian system. Particularly, the next two theorems assert that symplecity of the flow is a characteristic property of Hamiltonian systems. Furthermore, the application of a symplectic integrator to a Hamiltonian system is itself Hamiltonian (see [29] for proofs).

Theorem 6.2.2 (Poincaré, 1899) *Let $H(p,q)$ be a twice differentiable function on $U \subset \mathbb{R}^{2d}$. Then, for each fixed t, the flow φ_t is a symplectic transformation wherever it is defined [29].*

Theorem 6.2.3 *If a symplectic method $\Phi_h(\mathbf{c})$ is applied to a Hamiltonian system with a smooth Hamiltonian $H : \mathbb{R}^{2d} \to \mathbb{R}$, then the modified equation (6.5) is*

also Hamiltonian. More precisely, there exist smooth functions $H_j : \mathbb{R}^{2d} \to \mathbb{R}$ *for* $j = 2, 3, \cdots$, *such that* $f_j(\mathbf{c}) = J^{-1}\nabla H_j(\mathbf{c})$.

We now turn our attention to estimating the local error, the difference between the numerical and exact solutions of the truncated differential equation.

6.2.2 Estimate of the Local Error Bound

In this section we outline the steps taken to provide an estimate of a local error bound as presented in [29]. See [29] for a full discussion of these results. The Taylor's series expansion of the numerical flow is

$$\Phi_h(\mathbf{c}) = \mathbf{c} + hf(\mathbf{c}) + h^2 d_2(\mathbf{c}) + h^3 d_3(\mathbf{c}) + \cdots \tag{6.11}$$

where the functions $d_i(\mathbf{c})$ depend on $f(\mathbf{c})$ and its derivatives.

The first step is to provide estimates for functions $d_j(\mathbf{c})$ using Equation 6.11, also known as Cauchy's inequality. The functions $d_j(\mathbf{c})$ are bounded as follows

$$\|d_j(\mathbf{c})\| \leq \mu\upsilon \left(\frac{2\kappa\upsilon}{\rho}\right)^{j-1} \quad \text{for} \quad \|\mathbf{c} - \mathbf{c}_0\| \leq \rho. \tag{6.12}$$

The coefficients, $f_j(\mathbf{c})$ of the modified equation are then estimated resulting in the following estimate of the local error of a numerical method.

Theorem 6.2.4 *Let* $f(\mathbf{c})$ *be analytic in* $B_{2\rho}(\mathbf{c}^0)$, *let the coefficients* $d_j(\mathbf{c})$ *of the method (6.11) be analytic in* $B_\rho(\mathbf{c}^0)$, *and assume (6.5) and (6.12) hold. If* $h \leq h_0/4$ *with* $h_0 = \rho/(e\eta\upsilon)$, *then there exists* $N = N(h)$ *(namely* N *equal to the largest integer satisfying* $hN \leq h_0$*) such that the difference between the numerical solution*

$\mathbf{c}^1 = \Phi_h(\mathbf{c}^0)$ *and the exact solution* $\tilde{\varphi}_{N,t}(\mathbf{c}^0)$ *of the truncated modified equation (6.6) satisfies*

$$\|\Phi_h(\mathbf{c}^0) - \tilde{\varphi}_{N,h}(\mathbf{c}^0)\| \leq h\gamma\upsilon e^{-h_0/h},$$

where $\gamma = e(2 + 1.65\eta + \mu)$ *depends only on the method.*

When Theorem 6.2.4 is applied to the Störmer/Verlet method we obtain information about the energy conservation of symplectic methods. Recall from Theorem 6.2.3 that the modified equation of a Hamiltonian system is itself Hamiltonian. Consider the truncated modified Hamiltonian

$$\tilde{H}(\mathbf{c}) = H(\mathbf{c}) + h^p H_{p+1}(\mathbf{c}) + \cdots + h^{N-1} H_N(\mathbf{c}). \tag{6.13}$$

The following theorem provides results that explain the (approximate) energy conservation of symplectic methods applied to Hamiltonian systems.

Theorem 6.2.5 ([7]) *Consider a Hamiltonian system with analytic* $H : D \to \mathbb{R}$ *(where* $D \subset \mathbb{R}^{2d}$*), and apply a symplectic numerical method* $\Phi_h(\mathbf{c})$ *with step size* h*. If the numerical solution stays in the compact set* $K \subset D$*, then there exist* h_0 *and* $N = N(h)$ *(as in Theorem 6.2.4) such that*

$$\begin{aligned} \tilde{H}(\mathbf{c}^i) &= \tilde{H}(\mathbf{c}^0) + \mathcal{O}(e^{-h_0/2h}) \\ H(\mathbf{c}^i) &= H(\mathbf{c}^0) + \mathcal{O}(h^p) \end{aligned}$$

over exponentially long time intervals $ih \leq e^{h_0/2h}$.

6.3 Open Questions

We conclude this chapter with items for future consideration. We have replaced the right side of an orthogonal transformation of the equations of motion with the gradient of a RBF network:

$$\begin{aligned} \mathbf{U}_k^T \mathrm{M} \mathbf{U}_k \ddot{\mathbf{y}}(t) &= -\mathbf{U}_k^T \nabla \mathcal{V}(\mathbf{r}(t)) \\ &\approx -\nabla \mathcal{G}(\mathbf{y}). \end{aligned}$$

Here $\mathbf{U}_k$ is the left singular vector matrix obtained from the SVD of a standard MD trajectory, $\mathbf{y} = \mathbf{U}_k^T \mathbf{r}$, and $\hat{\mathrm{M}} = \mathbf{U}_k^T \mathrm{M} \mathbf{U}_k$ is the transformed mass matrix. Our "reduced" equations of motion are then

$$\hat{\mathrm{M}} \ddot{\mathbf{y}}(t) \approx -\nabla \mathcal{G}(\mathbf{y}) \tag{6.14}$$

and we solve this system of equations to obtain updated kD coordinates.

Recall that the purpose for this approximation is to replace the empirical potential function with a reasonable approximation that is easily evaluated and is based on PCA of a previously computed trajectory. Assessing the error introduced by this substitution and deriving a bound for this error will assist in clearly defining the strengths and limitations of this method and serve as a guide in refining this method.

Our model function, $\mathcal{G}$, is a RBF network and we discussed the error associated with function approximation using RBF networks in Section 6.1. We use numerical integration to solve Equation 6.14. We discussed in Section 6.2 the local error

associated with using the Störmer/Verlet method to numerically solve differential equations.

A model function, $\mathcal{G}$, depends on the molecule under consideration, the MD trajectory used to define the initial PCA, and the choice of k via the potential energies and the kD representations used to generate the model function. Items for future consideration are to quantify the cumulative error associated with the substitution of the gradient $\nabla\mathcal{G}(\mathbf{y})$ to approximate the projection $\mathbf{U}_k^T\nabla\mathcal{V}(\mathbf{r})$, to assess the contribution to the error at each level of approximation (PCA, RBF network, and numerical solution of reduced equations of motion), and to identify how the error propagates in the solution of the reduced equations of motion (see Equation 6.14).

In the next chapter we discuss how the potential function approximation and simulation with respect to a kD representations affect molecular motion. We provide qualitative assessments of the molecular activity associated with the solutions of Equation 6.14 by comparing the power spectra associated with standard and reduced simulations and reviewing cross-correlation plots.

Chapter 7

Simulations

In this chapter we consider the application of our reduced simulation method to butane and BPTI. Standard MD simulations were performed using the MD program ESP (see Section 2.3.2). Test systems were performed in vacuum versus in the natural solvent environment to limit the size of the systems. We chose not to include solvent in our test systems for several reasons. First, the protein-solvent interactions are evident in the PCA of a system (see for example [25] and [34]). The equations of motion are coupled, thus reductions of the full system would not isolate molecular activity from protein-solvent interactions. Our primary interest is the internal molecular activity. Second, the solvent-solvent interactions introduce "noise" into PCA and complicates the analysis of the molecule. Third, and related to the first two concerns, we want to consider a reduced set of equations of motion. These equations are defined with respect to a basis determined by PCA of an MD trajectory. Thus it is desired that

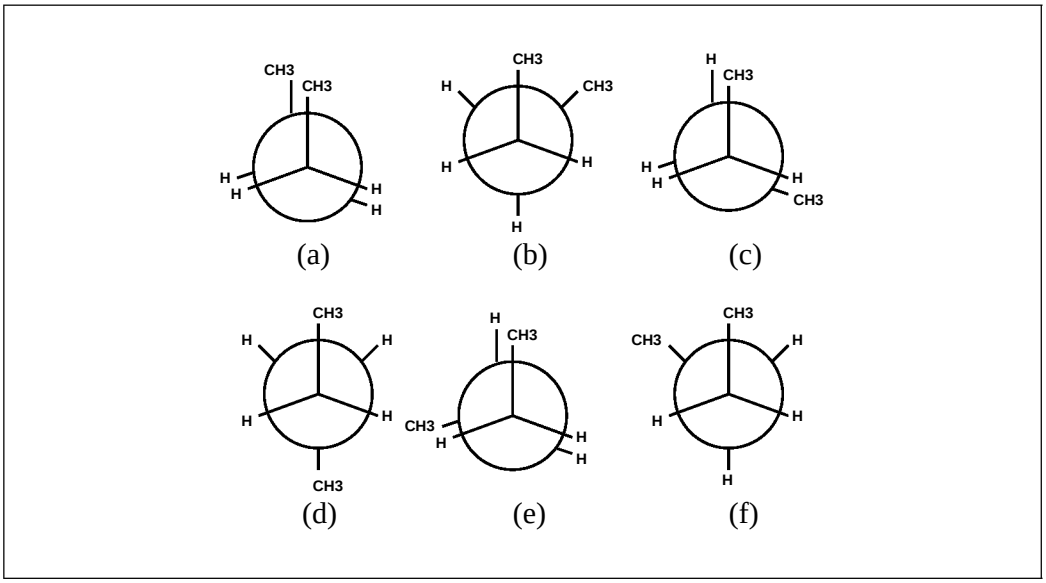


Figure 7.1: Newman projections of butane conformations: (a) Syn, $\psi = 0^o, 360^o$, (b) Gauche, $\psi = 60^o$, (c) Eclipsed, $\psi = 120^o$, (d) Anti, $\psi = 180^o$, (e) Eclipsed, $\psi = 240^o$, (f) Gauche, $\psi = 300^o$.

the only interactions inherent in the trajectory represent interatomic activity.

7.1 Butane

Butane provides a simple case study for initial tests of our reduced basis method. Though it is significantly smaller than the macromolecules we ultimately intend to study, butane contains the relevant degrees of freedom (bond stretching, angle bending, and dihedral torsion) found in larger systems. Furthermore, because butane is a simple molecule, we are able to consider lengthy simulations and explore the conformation space.

Butane is an alkane containing four carbons and ten hydrogens, $CH_3(CH_2)_2CH_3$.

The molecular weight of butane is 58.124 atomic mass units and the critical temperature[1] is 425.15 K. Figure 7.1 displays Newman projections[2] of butane conformations that result from rotation of the molecule about its central carbon-carbon bond.

The syn conformation of butane occurs when the methyl groups (CH_3) are aligned and the dihedral angle is 0^o or 360^o. The eclipsed conformations occur when the methyl groups align with hydrogen atoms and the dihedral angle is 120^o or 240^o. The gauche conformations have dihedral angles of 60^o and 240^o. The anti conformation has dihedral angle of 180^o.

Consider the potential energy of butane as a function of the dihedral angle resulting from the molecule's rotation about the central carbon-carbon bond, holding bond lengths and bond angles constant. Local minima of this function occur when butane is in the anti or gauche conformations. These conformations represent stable conformations of butane. The global minimum occurs when butane is in the anti conformation. Local maxima of the function occur when butane is in the syn or eclipsed conformations. The global maximum occurs when butane is in the syn conformation.

[1]The *critical temperature* of a substance is temperature at and above which a substance cannot exist as a liquid.

[2]A *Newman projection* represents the head-on look down the bond of interest. The circle the Newman projection represents the atom in front of the bond, and the lines radiating from the center are the bonds of that atom. The bonds of the rear atom emerge from the sides of the circle. Newman projections can be characterized by the angles formed between bonds on the front atom and bonds on the rear atom. Such angles are called dihedral angles.

7.1.1 Simulation

7.1.1.1 MD Simulation

Molecular dynamics simulations were performed using ESP distribution 0800 with the NVE ensemble and was run on a Sun Ultra 10 workstation. Bonds were constrained using SHAKE [58], and the integration time step for the velocity Verlet algorithm was 1 fs. The initial simulation temperature was set to 300 K and was controlled over 100 ps using velocity rescaling. The simulation was then run with no rescaling for 50 ps. Simulation continued for 100 ps and position and velocity data were collected every 5 fs. The average simulation temperature was 303.21 K.

7.1.1.2 kD Representation

Given a standard MD trajectory, we remove translation and rotation using the procedure outlined in Figure 4.1. We then compute the truncated SVD of the trajectory matrix, $\mathbf{U}_k\mathbf{S}_k\mathbf{V}_k^T = \frac{1}{\sqrt{n_{ts}}}\mathbf{R}$, where n_{ts} is the number of conformations used to form the trajectory matrix. In this case $n_{ts} = 20,000$ and represents every fifth conformation over the 100 ps simulation. The kD representation of atomic coordinates, $\mathbf{r}^i$, with respect to $\mathbf{U}_k$ is $\mathbf{y}^i = \mathbf{U}_k^T\mathbf{r}^i$ as previously defined.

7.1.2 Analysis

We provide a scree graph for the eigenvalues (squared singular values) of a butane MD trajectory in Figure 7.2(a) and the LEV graph for the eigenvalues of a butane

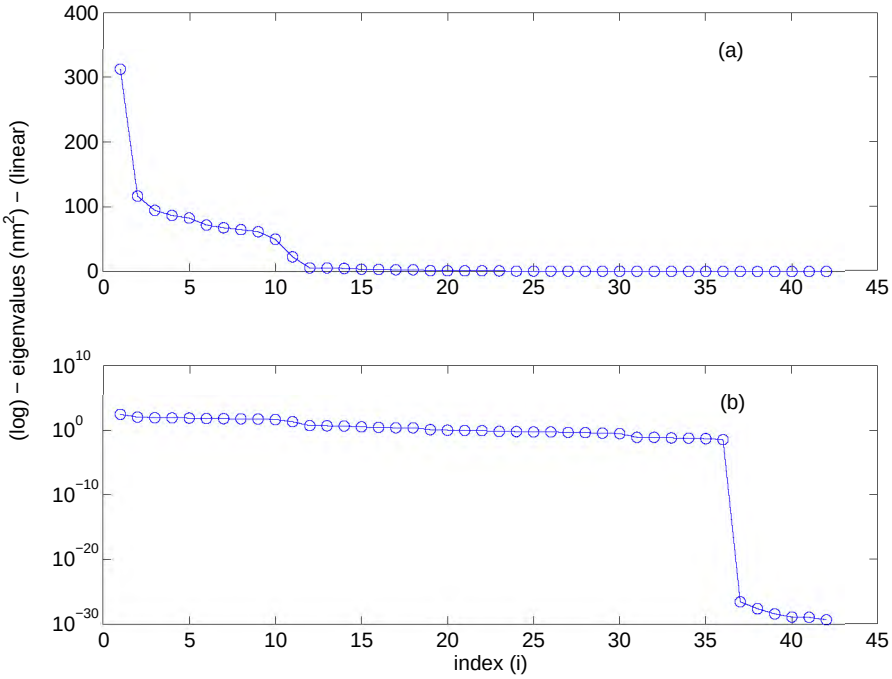


Figure 7.2: Eigenvalues of trajectory matrix (nm^2) in non-increasing order: (a) Scree Graph: Λ_{ii} versus i, (b) LEV Graph: $\log(\Lambda_{ii})$ versus i.

MD trajectory in Figure 7.2(b). It is not clear from these two figures how we should choose k. We rely on the graph of the cumulative percentage of the total variation (see Figure 7.3) to suggest an appropriate choice of k. With respect to our butane example cut off values of $k = 6$, $k = 8$, and $k = 10$ retain 70%, 80%, and 90% of the total variation, respectively. We chose various values for k (90% of the total variation and higher) to access the performance of this procedure.

We present here simulation data for $k = \{10, 26, 36\}$. Each simulation contained 100,000 integration steps with a time step of 1 fs providing 100 ps of simulation time. Position and velocity information was stored every 5 fs. We computed power spectra for reduced trajectories and compared them to a power spectrum from a standard

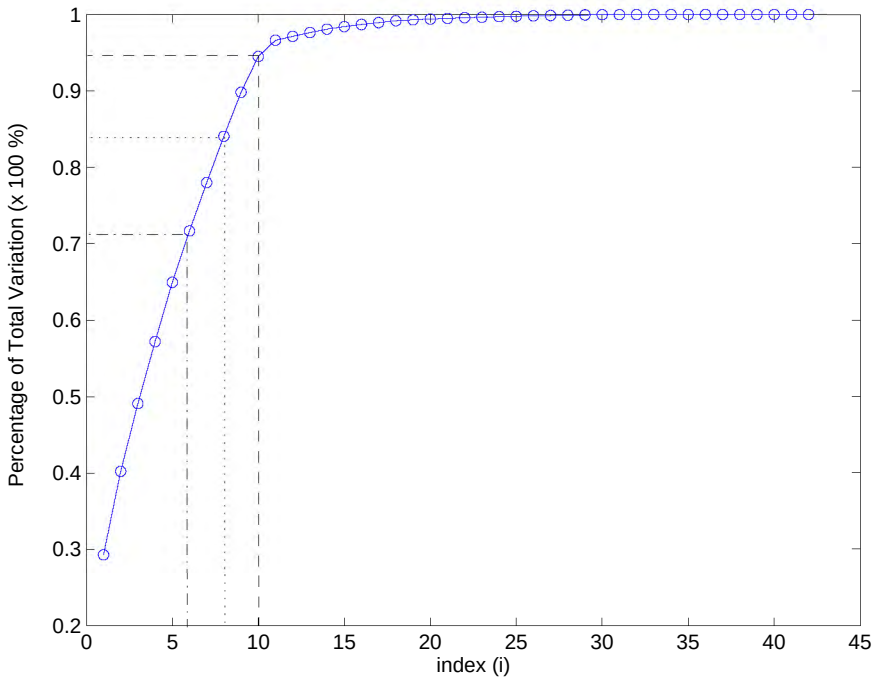
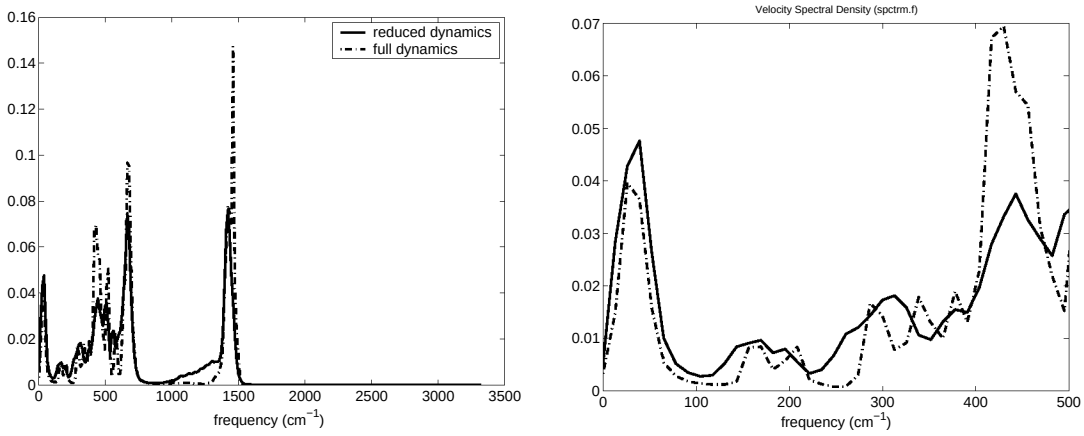


Figure 7.3: Cumulative percentage of the total variation captured by the first i principal components.

butane simulation.

We observe in Figures 7.4 that for $k = 36$ qualitatively the power spectra of the reduced and standard simulations behave similarly noting that they contain peaks at approximately the same frequencies, indicating similar interatomic activity during the simulations. As k decreases, the kD power spectrum degenerates with respect to high frequencies, while low frequencies (less than 500) are approximated fairly well. This observation is quite significant since we seek to maintain the large scale molecular motion which is associated with the low frequency activity of the power spectrum. We quantify this observation by computing the percentage of the total power that is captured in the low frequency range, 0 to 500 cm^{-1}. For the standard MD power



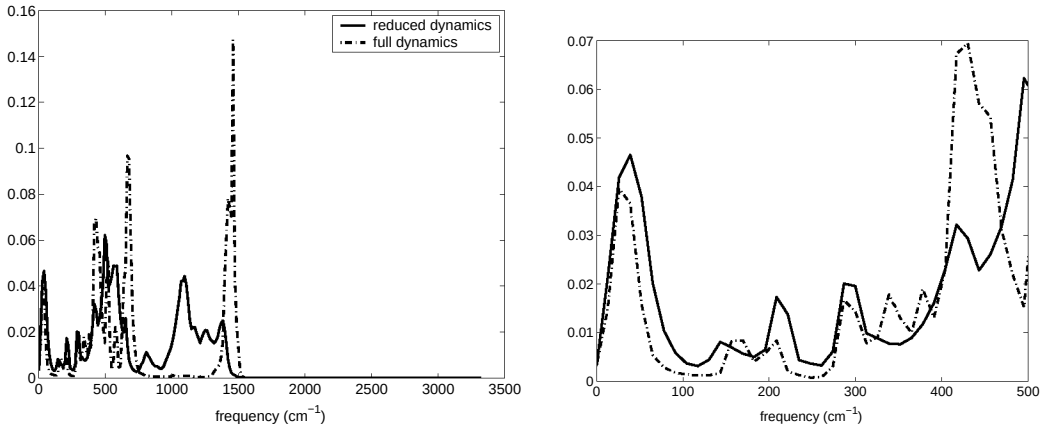
(a) Full Power Spectrum (b) Magnification of 0 - 500 cm^{-1}

Figure 7.4: Power spectral density, nev = 36 versus reference.

spectrum approximately 34% of the total power is present within the frequency range of 0 to 500 cm^{-1}. In the reduced power spectra approximately 35%, 32%, and 26% of the total power is present within the frequency range of 0 to 500 cm^{-1} for $k = 36$, 26, 10, respectively.

Figure 7.5 displays the power spectrum for reduced simulation with $k = 26$. We observe that qualitatively the power spectrum in Figure 7.1.2 behaves similarly to that in Figure 7.1.2.

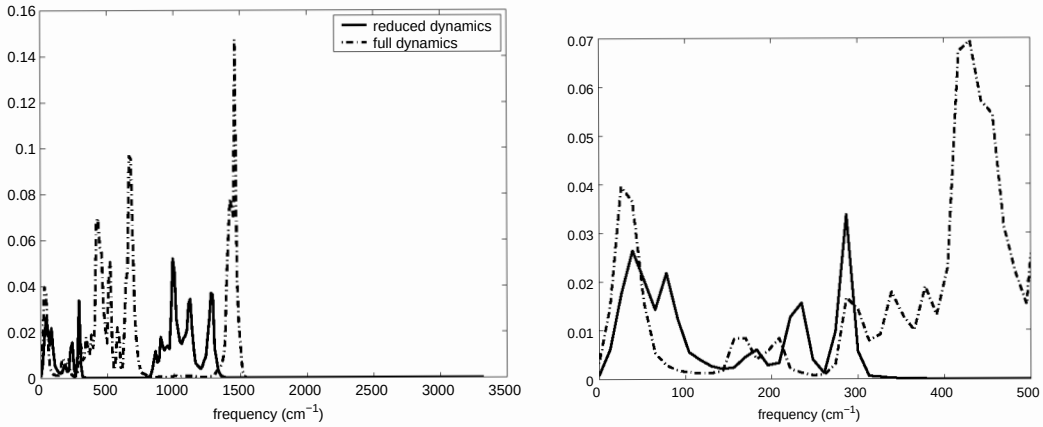
Figure 7.6 displays the power spectrum for reduced simulation with $k = 10$. At this point the degeneration of the power spectrum is more pronounced in the low frequency range. Given that coordinates responsible for high frequency motion are excluded from the kD representation, we anticipated that this behavior would be misrepresented in the power spectra from the reduced simulation. These results indicate



(a) Full Spectrum (b) Magnification of 0 - 500 cm^{-1}

Figure 7.5: Power spectral density, nev = 26 versus reference.

that low frequency motion, most likely to be functionally significant, is reasonably maintained by the reduced simulation method.



(a) Full Spectrum

(b) Magnification of 0 - 500 cm^{-1}

Figure 7.6: Power spectral density, nev = 10 versus reference.

7.2 BPTI

Bovine pancreatic trypsin inhibitor (BPTI) is a protein containing 58 residues (892 atoms). BPTI contains two alpha helices - one C-terminal α-helix (residues 47 to 56) and one 3_{10} helix (residues 3 to 7) - and two strands of antiparallel beta sheet (residues 16 to 24 and residues 29 to 36). Three disulfide bonds, generally important in the folding, structure and function of a protein, serve to stabilize the tertiary structure of the protein.

BPTI belongs to a class of proteins called serine protease inhibitors. It binds to and inactivates trypsin, a digestive enzyme, found in the pancreas. This interaction is irreversible and serves to protect the pancreas from self-digestion. When trypsin and BPTI bind, the Lys 15 side chain of BPTI occupies the specificity pocket of trypsin.

BPTI is a small protein containing both alpha helices and beta sheets, furthermore it is stable due to the three disulfide bonds. Thus BPTI provides a favorable model for general studies of protein structure. The first protein simulation on BPTI was performed in 1977 [63]. Since then BPTI has been widely studied.

Numerous comparative studies consider normal mode (NM) analysis and principal component analysis to gain insight into structural and dynamic properties of proteins ([34], [33], [36], [42]). Ichiye and Karplus [34] performed a comparative study of atomic fluctuations in MD and NM simulations. The results in their work indicate that a solvent simulation provides the most realistic simulation environment. Particularly, the solvent environment is significant to the long-range motion. However, for

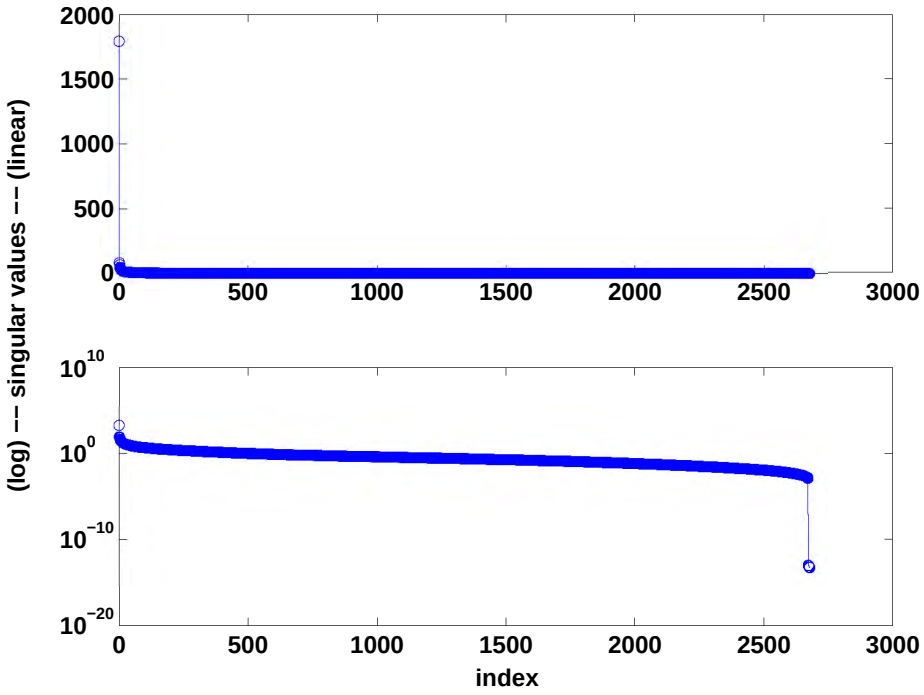


Figure 7.7: Singular values of trajectory matrix (nm).

simplification we perform simulations in vacuum.

7.2.1 Simulation

7.2.1.1 MD Simulation

We use the program ESP to perform MD simulations in vacuum using the NVT ensemble. The particular molecule we consider includes four internal water molecules. Temperature is set at 300 Kelvin. Bond lengths were constrained using the SHAKE algorithm [58], and an integration time step of 1 fs was used in the velocity Verlet method. The total simulation time was 100 ps. Positions and velocities were stored every 10 fs, thus 10,000 conformations were available for analysis.

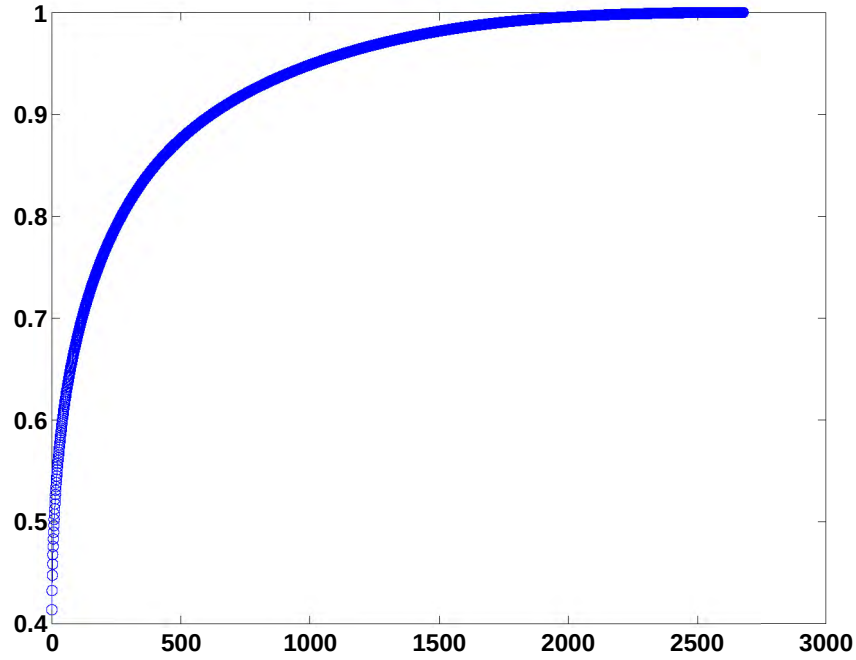


Figure 7.8: Relative contribution of first i LSV to positional fluctuation.

7.2.1.2 PCA of trajectory

After removing overall translation and rotation from the atomic coordinates using the procedure outlined in Figure 4.1, we construct the matrix of mean adjusted coordinates and perform a standard PCA. Figure 7.7 displays the eigenvalues obtained from PCA. We observe that one mode is prominent.

Figure 7.8 displays the cumulative percentage of the total variation as a function of the eigenvalue index. We observe that over 40% of the total variation can be attributed to the first principal component, and approximately 50% of the total variation can be attributed to the first 9 principal components.

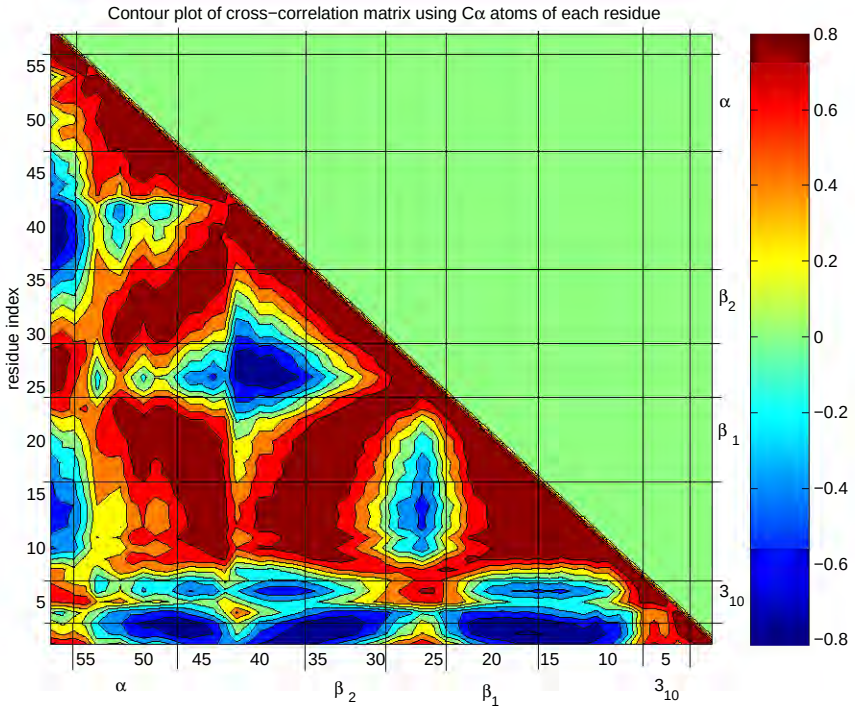


Figure 7.9: Contour plot of cross-correlation matrix, C, using the C_α atom of each residue. C is a symmetric matrix thus we only display the lower triangular portion of the matrix.

7.2.2 Analysis

In this section we consider the cross-correlations of residues in BPTI. Our purpose is to show that the correlations between residues are maintained in reduced simulation. We present the results for $k = 100$.

For a set of discrete points, $\mathbf{r}^\tau$, from an MD trajectory the cross-correlation between atoms i and j, or normalized covariance, is calculated as follows

$$C(i,j) = \frac{c(i,j)}{(c(i,i)c(j,j))^{1/2}}$$

where

$$c(i,j) = \frac{1}{n_t}\sum_{n=1}^{n_t}(\mathbf{r}_i^\tau - \bar{\mathbf{r}}_i)(\mathbf{r}_j^\tau - \bar{\mathbf{r}}_j).$$

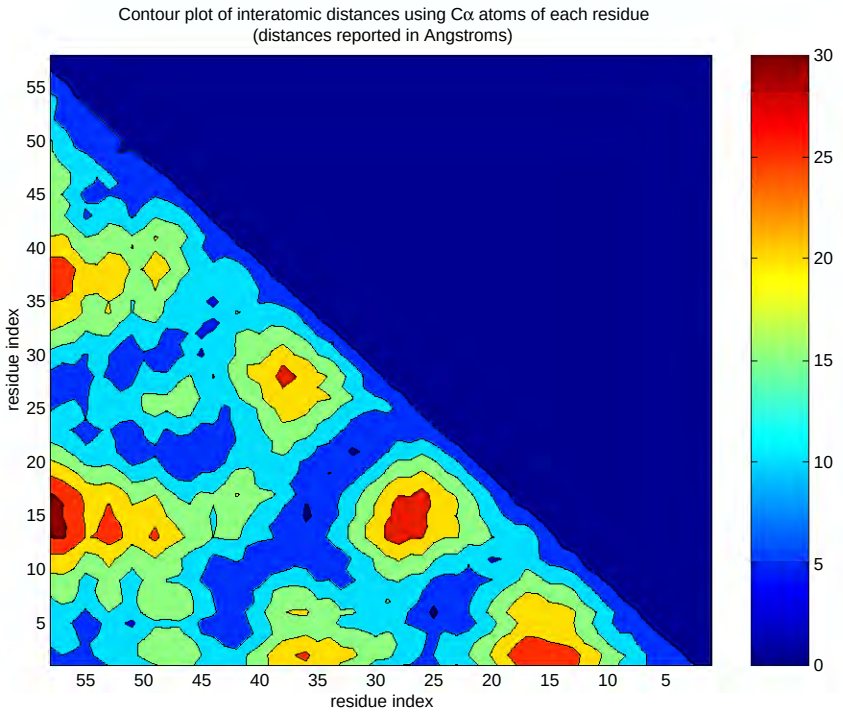


Figure 7.10: Contour plot of interatomic distances using the C_α atom of each residue. The distance matrix is symmetric thus we display the lower triangular portion of the matrix.

The cross-correlation quantifies the qualitative correspondence between parts of the molecule. The motion of atoms i and j are completely correlated when $C(i,j) = 1$. By this we mean that the motions have the same phase and period. The motion of atoms i and j are completely anti-correlated when $C(i,j) = -1$. This measure is most useful in identifying the level of correlation between atoms with the angle between the directions of motion is close to 0^o or 180^o [34].

We examine the correlations between residues i and j. Figure 7.9 displays a contour plot of a cross-correlation matrix using the C_α atom of each residue. The x and y axes contain the residue indices i and j, respectively. The contour plot displays

the level curves of the matrix C. Since this matrix is symmetric, we only display the lower triangular portion. The cross-correlation is represented by the height above (or below) the $x - y$ plane. The height along the diagonal ($i = j$) is the self correlation and by definition is equal to 1.

The C-terminal α-helix (residues 47 to 56) exhibits high correlations with the two strands of antiparallel beta sheet (residues 16 to 24 and residues 29 to 36). While there is a negative correlation between the C-terminal α-helix and the 3_{10}-helix. The two strands of antiparallel beta sheet are highly correlated. We observe a high anticorrelation between residues 7 to 24 and residues 1 to 3 and a high anticorrelation between residues 30 to 51 and residues 1 to 3.

Figure 7.10 displays a contour plot of interatomic distances using C_α atoms of each residue. The x and y axes represent the residue index i and j, respectively. The distance matrix is symmetric thus we display the lower triangular portion. When comparing Figures 7.9 and 7.10 we generally observe that residues that are closer together in distance are more positively correlated.

Chapter 8

Conclusion

In this dissertation we have presented a method for performing molecular simulations with respect to a kD coordinate space. Given a standard molecular dynamics we consider the truncated SVD of the resulting trajectory matrix. We presented a method for approximating the potential energy surface of a particular molecule with respect to a kD representation of the atomic coordinates and potential energies from a standard MD simulation.

We applied our method to butane and BPTI and compared the results to standard MD simulations of these molecules. Our results indicate that the molecular activity with respect to our simulation method is analogous to that observed in the standard MD simulation with simulations on the order of picoseconds. This work provides further evidence of the usefulness of a reduced representation provided by PCA of an MD trajectory in the study and prediction of molecular motion.

Our current implementation considers each molecule in vacuum in order to simplify our analysis. However, as noted in various studies, the solvent environment is most like a protein's natural environment (see [34] for example). We are currently investigating ways in which solvent affects can be incorporated into a kD simulation scheme. Future work will focus on further error analysis of this method and the incorporation of solvent affects into our reduced simulation model. This will include the simulation of larger proteins and protein-protein interactions, and automation of the simulation process.

Appendix A

Overview of Fortran Subroutines

In this appendix we provide an overview of the Fortran subroutines developed for use in this project. We constructed subroutines to accomplish the following tasks:

i. Compute the SVD of a standard MD trajectory and define a kD representation of the trajectory.

ii. Model a potential energy surface with respect to kD representations of a molecule and the known potential energies.

iii. Use the potential surface model to update kD coordinates.

iv. Construct a $3n$D trajectory from the new kD coordinates.

v. Analyze data.

In the following sections we list the input and output for various components of the project and provide brief descriptions of this information.

INPUT	**OUTPUT**
FPOS: trajectory file **nstep**: number of conformers in file **ncoord**: number of coordinates in the system **FPDB**: reference structure **nev**: number of singular vectors computed (*arp)	svd(FPOS)

Table A.1: Input and output for functions *svdarp.f* and *svdlap.f*

A.1 SVD

ARPACK (ARnoldi PACKage) and LAPACK (Linear Algebra PACKage) are used to compute the SVD of trajectory matrices. ARPACK is a collection of fortran subroutines that uses the implicitly restarted Arnoldi method to efficiently compute the k largest or smallest (algebraically or in magnitude) eigenvalues and associated eigenvectors of an n by n matrix. One can also compute the singular values and vectors of an m by n matrix. The singular value computation provides a rank k approximation of an m by n matrix (assuming $m \geq n$) and only requires $O(nk)$ storage [44].

When full SVD computations are necessary we use the LAPACK subroutine *dgesvd.f*. Table A.1 lists the input required by the SVD drivers, *svdarp.f* and *svdlap.f*.

INPUT	**OUTPUT**
FPOS: kD trajectory file	a_j: model coefficients
FENE: energy file	η_j: function widths
nstep: number of conformers in file	
nev: number of singular vectors computed (*arp)	
ns: read increment for training	
nsc: read increment for centers	

Table A.2: Input and output for function *lsfit.f*

A.2 LS Fit

Once we have computed the SVD of an MD trajectory and extracted the appropriate potential energy information from an MD simulation, we have the necessary data to begin constructing a model of the potential energy surface. Using k user defined left singular vectors, we first compute a kD representation of the MD trajectory with respect to $\mathbf{U}_k$

$$\mathbf{y}^i = \mathbf{U}_k^T \mathbf{r}^i \in \mathbb{R}^k.$$

The potential energy evaluated at $\mathbf{r}^i$, $\mathcal{V}(\mathbf{r}^i)$, is associated with $\mathbf{y}^i$.

We seek a function that best fits the data $\{\mathbf{y}^i, \mathcal{V}(\mathbf{r}^i)\}$ in the least squares sense, where the function is of the form

$$\mathcal{G}(\mathbf{y}) = \sum_{j=1}^{M} a_j \varphi_j(\mathbf{y}), \tag{A.1}$$

and $\varphi_j(\mathbf{y}) = \exp(-\frac{\|\mathbf{y}-\mathbf{c}^j\|^2}{\eta_j^2})$ are Gaussian radial functions. The general least linear squares problem is solved using the SVD method described in Section 5.2.2.2.

The task here is to compute the coefficients for the least squares model in Equation A.1 and the gradient of the model. The ith component of the gradient vector is

$$\nabla \mathcal{G}(\mathbf{y})_{(i)} = \sum_{j=1}^{M} a_j \frac{\partial \varphi_j(\mathbf{y})}{\partial \mathbf{y}_i} = -\sum_{j=1}^{M} \frac{2a_j}{\eta_j^2}(\mathbf{y}_i - \mathbf{c}_i^j) exp\left(\frac{\|\mathbf{y}-\mathbf{c}^j\|}{\eta_j^2}\right). \tag{A.2}$$

Equation A.2 represents the force acting on the kD coordinates based on a least squares approximation of the potential energy surface.

The computations are facilitated by the program *drive_lsfit.f* with the calling sequence:

```
drive_lsfit.x kD*.pos *.ener nstep nev nskip nskipc.
```

The user is prompted to supply files containing the independent (kD*.pos) and dependent variables of interest (*.ener). In this case the independent variables are kD representations of atomic coordinates produced by an MD simulation, and the dependent variables are the associated potential energies. The user is also prompted to supply the number of data entries in the files (nstep), the dimension, k, of the independent variable (nev), the increment at which to read in data points (nskip), and the increment at which to read in data points that will serve as the initial centers for the RBFs (nskipc). Table A.2 summarizes the information required as input into *lsfit.f*.

INPUT	**OUTPUT**
FPOS: kD trajectory file	$\mathbf{y}^+$: updated kD coordinates
nstep: number of time steps	$\hat{\mathbf{r}}^+$: corresponding $3n$D coordinates
nev: number of left singular vectors	time averages
MOL: file containing molecule parameters	

Table A.3: Input and output for subroutine *runred.f*.

A.3 kD Simulation

Once we have defined our model parameters, we consider equations of motion in k dimensions

$$\hat{\mathrm{M}}\ddot{\mathbf{y}}(t) \approx -\nabla\mathcal{G}(\mathbf{y}).$$

The model function is used to compute kD energies and the forces acting on the kD coordinates. Recall that we have defined the force acting on the ith component of y as $f_j = \frac{\partial\mathcal{G}(\mathbf{y})}{\partial\mathbf{y}_j}$. This process is facilitated by *runred.f*, a subroutine incorporated in Extended System Program (see Section 2.3.2). Table A.3 outlines the basic input requirements of this subroutine.

A.4 Power Spectrum

In Section 3.2 we defined the power spectrum. Here we present the Fortran program used to compute estimates of the power spectrum of a molecule. The subroutine

INPUT	**OUTPUT**
FVEL: velocity file from MD simulation **natom**: number of atoms in the system **nstep**: number of time steps **ovropt**: logical data - overlap data **nseg**: number of data segments **lenseg**: length of each segment	**psd**: power spectrum

Table A.4: Input and output for function *power.f*.

spctrm.f provided by *Numerical Recipes in Fortran 77* [53] is an integral part of these computations. The calling sequence is as follows:

```
usage: power.x *.vel natom nstep ovropt nseg lenseg.
```

The function *power.f* takes the FFT of each component of the velocity produced during an MD simulation. The user must supply a file containing the velocity data (FVEL), as well as the number of atoms in the system (natom), and the number of velocity vectors in the file (nstep). The data is partitioned into nseg segments each containing lenseg data points. The function takes the FFT of 2∗lenseg data points. The user has the option to overlap the segments by letting ovropt = 0. In this case segments i and $i + 1$ have lenseg data points in common, and the power spectrum approximation is based on (2∗nseg+1)∗lenseg data points. If the data is not overlapped the values nseg and lenseg should be chosen such that the number of

data points used to approximate the power spectrum, 4∗nseg∗lenseg, does not exceed the available data. Table A.4 summarizes the input required by the function *power.f*.

Bibliography

[1] M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids*. Oxford Science Publications, 1987.

[2] A. Amadei, A. B. M. Linssen, and H. J. C. Berendsen. Essential dynamics of proteins. *PROTEINS: Structure, Function, and Genetics*, 17:412–425, 1993.

[3] A. Amadei, A. B. M. Linssen, B. L. de Groot, D. M. F. van Aalten, and H. J. C. Berendsen. An efficient method for sampling the essential subspace of proteins. *Journal of Biomolecular Structure and Dynamics*, 13(4):615–625, 1996.

[4] H. C. Andersen. Rattle: A "velocity" version of the shake algorithm for molecular dynamics calculations. *Journal of Computational Physics*, 52:24–34, 1983.

[5] Protein Data Bank. *http://www.rcsb.org/pdb/index.html*.

[6] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions of Information Theory*, 39(3):930–945, 1993.

[7] G. Benettin and A. Giorgilli. On the hamiltonian interpolation of near to identity symplectic integration algorithms. *Journal of Statistical Physics*, 74:1117–1143, 1994.

[8] H. J. Berendsen and W. F. Van Gunsteren. Practical algorithms for dynamics simulations. *Molecular-dynamics simulation of statistical-mechanical systems; Varenna on Lake Como, Villa Monastero, 23 July-2 August 1985*, pages 43–65, January 1986. International School of Physics 'Enrico Fermi' (1985); Varenna, Italy).

[9] H. J. C. Berendsen. Molecular dynamics simulations: The limits and beyond. In P. Deuflhard, J. Hermans, B. Leimkuhler, A. E. Mark, S. Reich, and R. D. Skeel, editors, *Computational Molecular Dynamics: Challenges, Methods, Ideas*, Lecture Notes in Computational Science and Engineering, pages 3–36. Springer-Verlag, 1999. Proceedings of the 2nd International Symposium on Algorithms for Macromolecular Modelling, Berlin, May 21-24, 1997.

[10] T. C. Bishop, R. D. Skeel, and K. Schulten. Difficulties with multiple time stepping and fast multipole algorithm in molecular dynamics. *Journal of Computational Chemistry*, 18(14):1785–1791, 1997.

[11] C. Branden and J. Tooze. *Introduction to Protein Structure*. Garland Publishing, Inc., New York, New York, second edition, 1999.

[12] B. Brooks, R. Bruccoleri, B. Olafson, D. States, S. Swaminathan, and M. Karplus. Charmm: A program for macromolecular energy, minimization, and dynamics calculations. *Journal of Computational Chemistry*, 4:187–217, 1983.

[13] J. Bunch and C. Nielson. Updating the singular value decomposition. *Numerische Mathematik*, 31:111–129, 1978.

[14] J. Bunch, C. Nielson, and D. Sorensen. Rank one modification of the symmetric eigenvalue problem. *Numerische Mathematik*, 31:31–48, 1978.

[15] R. B. Cattell. The scree test for the number of factors. *J. Multiv. Behav. Res.*, 1:245–276, 1966.

[16] J. M. Craddock and C. R. Flood. Eigenvectors for representing the 500 mb hemispheric fields. *Q. J. R. Met. Soc.*, 95:576–593, 1969.

[17] T. Darden, D York, and L. Pedersen. Particle mesh ewald: An $n \cdot log(n)$ method for ewald sums in large systems. *Journal of Chemical Physics*, 98(12):10089–10092, June 1993.

[18] B. L. de Groot, A. Amadei, D. M. F. van Aalten, and H. J. C. Berendsen. Towards an exhaustive sampling of the configurational spaces of the two forms of the peptide hormone guanylin. *Journal of Biomolecular Structural Dynamics*, 13:741–751, 1996.

[19] K. I. Diamantaras and S. Y. Kung. *Principal Component Neural Networks, Theory and Applications*. John Wiley & Sons, Inc., 1996.

[20] P. M. Fitzpatrick. *Advanced Calculus: A Course in Mathematical Analysis*. PWS Publishing Company, 1996.

[21] R. P. Futrelle and D. J. McGinty. Calculation of spectra and correlation functions from molecular dynamics data using the fast fourier transform. *Chemical Physics Letters*, 12(2):285–287, December 1971.

[22] C. W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall Series in Automatic Computation. Prentice-Hall, Inc., 1971.

[23] J. Ghosh and A. Nag. An overview of radial basis function networks.

[24] F. Girosi and T. Poggio. Networks and the best approximation property. Technical Report 1164/45, Massachusetts Institute of Technology Artificial Intelligence Laboratory and Center for Biological Information Processing Whitaker College, October 1989. A. I. Memo No. 1164 and C.B.I.P. Paper No. 45.

[25] N. Gō and H. A. Scheraga. On the use of classical statistical mechanics in the treatment of polymer chain conformation. *Macromolecules*, 9:535–542, 1976.

[26] G. H. Golub and C. F. Van Loan. *Matrix Computations*. John Hopkins University Press, Baltimore, Maryland, third edition, 1996.

[27] H. Grubmüller and P. Tavan. Multiple time step algorithms for molecular dynamics simulations of proteins: How good are they? *Journal of Computational Chemistry*, 19(13):1534–1552, 1998.

[28] J. M. Haile. *Molecular Dynamics Simulation, Elementary Methods*. Wiley Professional Paperback Series. John Wiley & Sons, Inc., 1992.

[29] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration; Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer, 2002.

[30] J. A. Hartigan. *Clustering Algorithms*. John Wiley & Sons, Inc., 1975.

[31] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

[32] E. J. Hartman, J. D. Keeler, and J. M. Kowalski. Layered neural networks with gaussian hidden units as universal approximations. *Neural Computation*, 2(2):210–215, 1990.

[33] S. Hayward, A. Kitao, and N. Gō. Harmonic and anharmonic aspects in the dynamics of bpti: A normal mode analysis and principal component analysis. *Protein Science*, 3:936–943, 1994.

[34] T. Ichiye and M. Karplus. Collective motions in proteins: A covariance analysis of atomic fluctuations in molecular dynamics and normal mode simulations. *Proteins*, 11:205–217, 1991.

[35] J. Izaguirre, S. Reich, and R. Skeel. Longer time steps for molecular dynamics. *Journal of Chemical Physics*, 110(20):9853–9864, May 1999.

[36] D. Janežič, R. M. Venable, and B. R. Brooks. Harmonic analysis of large systems. iii. comparison with molecular dynamics. *Journal of Computational Chemistry*, 16(12):1554–1566, 1995.

[37] I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer-Verlag, 1986.

[38] L. Kalé, R. Skeel, M. Bhandarkar, R. Brunner, A. Gursoy, N. Krawetz, J. Phillips, A. Shinozaki, K. Varadarajan, and K. Schulten. Namd2: Greater scalability for parallel molecular dynamics. *Journal of Computational Physics*, 151:283–312, 1999.

[39] M. Karplus and T. Ichiye. Comment on a "fluctuation and cross correlation analysis of protein motions observed in nanosecond molecular dynamics simulations". *Journal of Molecular Biology*, 263:120–122, 1996.

[40] M. Karplus and J. N. Kushick. Method for estimating the configurational entropy of macromolecules. *American Chemical Society*, 14:325–332, 1981.

[41] E. Kestemont and J. Van Craen. On the computation of correlation functions in molecular dynamics experiments. *Journal of Computational Physics*, 22:451–458, 1976.

[42] A. Kitao and N. Gō. Investigating protein dynamics in collective coordinate space. *Current Opinion in Structural Biology*, 9:164–169, 1999.

[43] A. R. Leach. *Molecular Modelling, Principles and Applications*. Prentice Hall, second edition, 2001.

[44] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK User's Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, Philadelphia, 1998.

[45] R. M. Levy, M. Karplus, J. Kushick, and D. Perahia. Evaluation of the configurational entropy for proteins: Application to molecular dynamics simulations of an $\alpha-$helix. *American Chemical Society*, 17:1370–1374, 1984.

[46] J. E. Marsden and A. J. Tromba. *Vector Calculus*. W. H. Freeman and Company, fourth edition, 1996.

[47] M. Moonen, P. Van Dooren, and J. Vandewalle. An svd updating algorithm for subspace tracking. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1015–1038, 1992.

[48] S. R. Niketić and K. Rasmussen. *The Consistent Force Field: A Documentation*, volume 3 of *Lecture Notes in Chemistry*. Springer-Verlag, 1977.

[49] M. J. L. Orr. Introduction to radial basis function networks. Technical report, Centre for Cognitive Science, University of Edinburgh, 2, Buccleuch Place, Edinburgh EH8 9LW, Scotland, April 1996.

[50] J. Park and I. W. Sandberg. Universal approximation using radial basis function networks. *Neural computation*, 3(2):246–257, 1991.

[51] J. Park and I. W. Sandberg. Approximation and radial basis function networks. *Neural computation*, 5(2):305–316, 1993.

[52] D.A. Pearlman, D.A. Case, J.W. Caldwell, W.R. Ross, III T.E. Cheatham, S. DeBolt, D. Ferguson, G. Seibel, and P. Kollman. Amber, a computer program for applying molecular mechanics, normal mode analysis, molecular dynamics and free energy calculations to elucidate the structures and energies of molecules. *Computer Physics Communications*, 91:1–41, 1995.

[53] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in Fortran 77*, volume 1. Cambridge University Press, second edition, 1992.

[54] D. C. Rapaport. *The Art of Molecular Dynamics Simulation*. Cambridge University Press, New York, New York, 1995.

[55] J. R. Rice. *The approximation of functions*, volume 1. Addison-Wesley, Reading, MA, 1964.

[56] T. Romo. *Identification and Modeling of Protein Conformational Substates*. PhD thesis, Department of Biochemistry and Cell Biology, Rice University, Houston, Texas, 1998.

[57] T. Romo, J. B. Clarage, D. C. Sorensen, and Jr. G. N. Phillips. Automatic identification of discrete substates in proteins: Singular value decomposition analysis of time-averaged crystallographic refinements. *Proteins*, 22:311–321, 1995.

[58] J. Ryckaert, G. Ciccotti, and H. J. C. Berendsen. Numerical integration of the cartesian equations of motion of a system with constraints: Molecular dynamics of n-alkanes. *Journal of Computational Physics*, 23:327–341, 1977.

[59] J.M. Sanz-Serna and M. P. Calvo. *Numerical Hamiltonian Problems*, volume 7 of *Applied Mathematics and Mathematical Computation*. Chapman & Hall, 1994.

[60] T. Schlick, E. Barth, and M. Mandziuk. Biomolecular dynamics at long timesteps: Bridging the timescale gap between simulation and experimentation. *Annual Review of Biophysics & Biomolecular Structure*, 26:181–222, 1997.

[61] P. E. Smith and B. Montgomery Pettitt. Extended system program for molecular dynamics. Department of Chemistry, University of Houston Board of Regents, 1991.

[62] P. Steinbach. Introduction to macromolecular simulation. *http://cmm.info.nih.gov/intro_simulation/course_for_.html*.

[63] R. Stote, A. Dejaegere, D. Kuznetsov, and Falquet L. Theory of molecular dynamics simulations. *http://www.ch.embnet.org/MD_tutorial/*, October 1999.

[64] W.C. Swope, H.C. Andersen, P.H. Berens, and K.R. Wilson. A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters. *Journal of Chemical Physics*, 76(1):637–649, 1982.

[65] M. Tuckerman, B. J. Berne, and G. J. Martyna. Reversible multiple time scale molecular dynamics. *Journal of Chemical Physics*, 97(3):1990–2001, August 1992.

[66] D. M. F. van Aalten, A. Amadei, R. Bywater, J. B. C. Findlay, H. J. C. Berendsen, and C. Sander. A comparison of structural and dynamic properties of different simulation methods applied to sh3. *Biophysical Journal*, 70:684–692, February 1996.

[67] D. M. F. van Aalten, A. Amadei, A. B. M. Linssen, V. G. H. Eijsink, G. Vriend, and H. J. C. Berendsen. The essential dynamics of thermolysin: Confirmation of hinge-bending motion and comparison of simulations in vacuum and water. *PROTEINS: Structure, Function, and Genetics*, 22:45–54, 1995.

[68] L. Verlet. Computer "experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Physical Review*, 159(1):98–103, July 1967.

[69] E. W. Weisstein, et al. Orthogonal transformation. MathWorld–A Wolfram Web Resource. *http://mathworld.wolfram.com/OrthogonalTransformation.html*.

[70] R. E. Wilde and S. Singh. *Statistical Mechanics: Fundamentals and Modern Applications*. John Wiley & Sons, Inc, 1998.

[71] J. H. Wilkinson. *The Algebraic Eigenvalue Problem*. Monographs on Numerical Analysis. Clarendon Press, Oxford, 1965.