

---

Masters Theses

Student Theses and Dissertations

---

Fall 2013

## Protein aggregation at solid/liquid interfaces: a Monte Carlo study with explicit and implicit solvent effects

Rehman Fazeem

Follow this and additional works at: [https://scholarsmine.mst.edu/masters\\_theses](https://scholarsmine.mst.edu/masters_theses)

 Part of the [Chemical Engineering Commons](#)

Department:

---

### Recommended Citation

Fazeem, Rehman, "Protein aggregation at solid/liquid interfaces: a Monte Carlo study with explicit and implicit solvent effects" (2013). *Masters Theses*. 7197.  
[https://scholarsmine.mst.edu/masters\\_theses/7197](https://scholarsmine.mst.edu/masters_theses/7197)

This thesis is brought to you by Scholars' Mine, a service of the Missouri S&T Library and Learning Resources. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact [scholarsmine@mst.edu](mailto:scholarsmine@mst.edu).



PROTEIN AGGREGATION AT SOLID/LIQUID INTERFACES: A MONTE CARLO  
STUDY WITH EXPLICIT AND IMPLICIT SOLVENT EFFECTS

by

REHMAN FAZEEM

A THESIS

Presented to the Faculty of the Graduate School of the  
MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN CHEMICAL ENGINEERING

2013

Approved by

Daniel Forciniti, Advisor  
Parthasakha Neogi  
Jee-Ching Wang



## ABSTRACT

Metropolis Monte Carlo (MC) simulations were performed with positively charged peptides in aqueous solution to study changes in peptide conformations at solid/liquid interface, and its effects on protein aggregation. Intermediate-resolution diblock model peptide, comprising of 10 units of ALA (non-polar) and LYS (polar) amino acid residues, was used for the simulations. In the first approach to modeling, solvent effects were considered explicitly. The explicit model was then used to study two peptide molecules, in helical structure, at solid/liquid interface. In order to increase the number of peptide molecules in the simulation box, with reduced computational cost, an implicit solvent model was developed with nonadditive hydrogen bonding and hydrophobic interaction potentials. The implicit model was used to simulate two peptides of helical structure at charged surfaces (to compare with the explicit model), and ten peptides of random coil structure with and without charged surfaces. The peptides were observed to always move towards the negatively charged surface and orient with residues of complimentary charge settling close to the surface, maximizing the electrostatic interactions. On reaching the surface, the peptides partially lose their secondary structure and clusters around the hydrophobic ends; this restructuring and dehydration of the peptides provides the entropic drive for adsorption and subsequent misfolding events. The 2- peptide-water-surface system in explicit model was also simulated with periodic switching of surface charge polarity, to induce a “shaking effect” in order to observe possible peptide configurational changes.

## ACKNOWLEDGMENTS

I thank my advisor, Dr. Daniel Forciniti, for providing me with valuable guidance and insight into my thesis topic. He has been there always to assist me through building the potential model and analyzing the results.

This material is based upon work supported by the National Science Foundation under Grant No. 0933468.

I also thank my committee members, Dr. Parthasakha Neogi and Dr. Jee-Ching Wang, for reviewing my work and for providing their valuable input.

**TABLE OF CONTENTS**

	Page
ABSTRACT .....	iii
ACKNOWLEDGMENTS .....	iv
LIST OF ILLUSTRATIONS .....	vi
SECTION	
1. INTRODUCTION.....	1
2. MODEL DEVELOPMENT.....	7
2.1. EXTENSION OF EXISTING MODEL TO FINITE CONCENTRATIONS.....	7
2.2. MIXING .....	8
2.4. DEVELOPMENT OF IMPLICIT WATER MODEL .....	8
3. RESULTS .....	14
APPENDICES	
A. EXPLICIT MODEL SIMULATION PROGRAM CODE.....	26
B. INSTRUCTIONS TO RUN EXPLICIT MODEL CODE.....	116
C. IMPLICIT MODEL SIMULATION PROGRAM CODE.....	120
D. INSTRUCTIONS TO RUN IMPLICIT MODEL CODE.....	173
E. EXPLICIT MODEL RESULTS.....	175
BIBLIOGRAPHY.....	183
VITA.....	185

## LIST OF ILLUSTRATIONS

Figure	Page
2.1. A schematic figure of the side-chain dependent HB interactions, between two selected Lysine amino acid groups, as proposed in this model. ....	12
3.1 Total energy vs. NMCS for 2-peptide(helix)-surface system simulated using the implicit model. ....	15
3.2 Peptide ETED profile for 2-peptide(helix)-surface system simulated using the implicit model. ....	16
3.3 COM z-location profile for 2-peptide(helix)-surface system simulated using the implicit model. ....	16
3.4 Snapshots of initial (a) and final (b) configurations of 2-peptide(helix)-surface system simulated using the implicit model. ....	17
3.5 “Ribbon” graphic snapshots of final configurations of 2-peptide(helix)-surface system simulated using the explicit (a) and the implicit models (b). ....	18
3.6 Initial configuration obtained by computational “shaking” for 10-peptide(random coil) simulations. ....	19
3.7 Energy profile for 10-peptide(random coil) system simulated using the implicit model. ....	20
3.8 Peptide ETED profile for 10-peptide(random coil) system simulated using the implicit model. ....	20
3.9 COM z-location profile for 10-peptide(random coil) system simulated using the implicit model. ....	21
3.10 “Ball & Stick” snapshot of the final configuration for 10-peptide(random coil) system simulated using the implicit model. ....	21
3.11 “Ribbon” snapshot of the final configuration for 10-peptide(random coil) system simulated using the implicit model. ....	22
3.12 Energy profile for 10-peptide(random coil)-surface system simulated using the implicit model. ....	23
3.13 Peptide ETED profile for 10-peptide(random coil)-surface system simulated using the implicit model. ....	23



3.14 COM z-location profile for 10-peptide(random coil)-surface system simulated using the implicit model. ....	24
3.15 Snapshot for 10-peptide(random coil)-surface system simulated using the implicit model. ....	24
3.16 A close-up view near the negatively charged surface of the final configuration of 10-peptide(random coil)-surface system using the implicit model. ....	25

## 1. INTRODUCTION

Proteins are polymers of amino acids which are covalently linked through peptide bonds. Amino acids are small organic molecules that consist of an  $\alpha$ -carbon atom linked to an amino and carboxylic acid functional groups, a hydrogen atom, and a variable group called a side chain. The linear sequence of amino acids within a protein forms its primary structure. The chemistry of each amino acid plays an important role in the 3D-structure of proteins, as they can participate in non-covalent bonds, such as hydrogen bonds, which define the protein secondary structure. The tertiary structure of proteins is formed by the 3D arrangement of multiple secondary structures in single polypeptide chain. Quaternary protein structure refers to multiple polypeptide chains or subunits in a single protein molecule. Proteins are also prone to aggregation, where unfolded molecules associate and clump together. These aggregate structures may be fibrillar or amorphous.

Proteins support all vital functions of every cell and are often referred to as “biological workhorses”. To carry out their functions, proteins assemble into complex three-dimensional structures through a mechanism called folding. When proteins do not fold correctly (misfolding) they lose their specific activities --adversely affecting various cellular functions. The self-templating ability of misfolded proteins leads to misfolding inheritance by the cell, and their aggregation and/or accumulation (called amyloid plaque) is a signature of several human diseases, such as Alzheimer's, Huntington's chorea, Parkinson's, type two diabetes, systemic amyloidosis, transmissible spongiform encephalopathy etc. These diseases are collectively known as protein misfolding or amyloid diseases. Amyloid diseases are apparently unconnected; however, intermolecular secondary structure (mostly  $\beta$ -sheets) is present in all amyloid aggregates. In spite of the large amount of published material, the effect of surfaces on the formation of the deposits remains obscure. The concentration of amyloid peptides in healthy or sick tissue is of the order of a few nano-moles. Therefore, the presence of a solid/liquid interface may serve as a crowding agent that “catalyses” the formation of the aggregates.

It is believed that virtually any protein may form amyloid fibrils, given appropriate conditions; this suggests that the amyloid fibril is the global free energy minimum assembly formed by polypeptide chains. The experimental pool suggests that

the fibrils minimize their energy by minimizing electrostatic repulsions and maximizing hydrophobic contacts. The current consensus is that all amyloid deposits consists of continuous  $\beta$ -sheets with  $\beta$ -strands oriented perpendicular to the fibril long axis. The remarkable stability and insolubility of the fibril structure poses greater threat for etiology and treatment of amyloid diseases. The aggregation process and the type of aggregate structure are largely dependent on the protein primary structure [1] and, sometimes, a minor change in amino acid sequence changes the aggregation phenomenon.

Protein aggregation may be desirable for situations such as in evolution [2] because it allows cells to acquire complex traits step wise, but is strongly undesirable in processes like protein crystallization, in vitro folding and formulation and delivery. Many vital protein functions, which depend on maintaining the protein native state, are lost as the protein aggregates. Recently it was found that these aggregate structures were not as non-specific as earlier believed [3]. Proteins form amorphous or ordered (fibril) aggregates structures depending on the polypeptide sequence, temperature, solution conditions and protein concentration.

Protein aggregation occurs readily in almost all biopharmaceutical processes and it is one of the top tumbling blocks in the rapid commercialization of protein drug candidates. Even though the protein preparation may have been aggregate-free, aggregates may form during storage. For over a 2 year shelf-life, build-up of aggregate levels as low as 1% can render a product clinically unsafe [4]. It is therefore important to understand the conditions at which protein stability is compromised. The main factors that affect protein stability in protein drug processing and delivery are shear/shaking (involves interaction with interfaces), temperature, pH and protein concentration. Irreversible aggregation, caused by disulfide shuffling or stable hydrophobic association, can have a direct impact on drug potency immunogenicity and unfolded protein response. The administration of a protein drug that aggregates reversibly could be also tricky if the disassociation is slow in the physiological time scale.

Some proteins in folded or unfolded state undergo non-specific self-association into disordered aggregates. When viewed in an electron microscope they look granular without any fibrils and they are called amorphous aggregates. Amorphous aggregation is usually an initial step in amyloid assembly before amyloid fibril formation proceeds.

Among the variety of aggregates formed, the aggregates with amyloid fibril structure are common. They have three characteristic features: (1) birefringent staining using congo red, (2) fibrous morphology observed by electron microscopy and (3) X-ray fiber diffraction pattern consistent with high cross- $\beta$  structure [4]. However, debate still remains on the toxicity of these aggregates as few studies indicate that precursors to these aggregates are much more toxic than the mature fibrillar aggregates [5].

One of the least investigated areas in protein aggregation phenomena is the interface-induced aggregation at solid-liquid and air-liquid boundaries. Although there are some publications on this topic [6-9] there is not consensus among the scientific community about the actual mechanism of interface-induced aggregation. It has been suggested that since proteins are surface active, their structure gets perturbed by surface tension forces at interfaces. These perturbations restructure the protein folding energy landscape to energetically favor partially folded and structurally expanded transition states (which are aggregate prone) over the more compact native states [6]. However, it is found often that the structural conformational change alone cannot explain protein aggregation and hence there are no apparent reasons to believe that aggregation is preceded by a structural conformational change.

Some biopharmaceutical processing operations, like stirring, create air/liquid interfaces. At the interface, the hydrophobicity of air relative to water induces protein alignment, maximizing the exposure of the hydrophobic residues to the air and initiating aggregation. Many proteins aggregate at the air/liquid interface [10]. To suppress this in industrial processes surfactants are commonly used. Surfactants inhibit protein aggregation by competitively accumulating with proteins at hydrophobic surfaces/interfaces and/or by binding directly to proteins. Certain surfactants have the added advantage of increasing solution viscosity, which may limit the motion of the protein backbone and thus reduce aggregation.

Proteins adsorb at solid liquid interfaces. This process, which is driven by electrostatic and hydrophobic interactions, may alter the characteristics of the sorbent surface, and in many cases, of the protein molecules. Adsorption likely reduces the energy barrier for protein aggregation [10].

Adsorption of biopharmaceuticals at solid/liquid interfaces creates serious problems for their production, storage and delivery. For example, insulin gets destabilized by adsorption at hydrophobic surfaces like those formed by air-water or water-polymer. Nucleation precedes the process, followed by intermediate aggregation that serves as precursor for large insulin precipitates [11]. It is suggested that less stable insulin monomers undergo partial unfolding upon adsorption, and combine with neighboring unfolded species initiating nucleation. After reaching a critical size, the intermediate aggregates grow into larger aggregates [11]. Aggregation of insulin reduces its biological potency and obstructs continuous/controlled release pumps and polymeric devices, creating serious problems for drug delivery systems.

Experiments show that presence of negatively charged surfaces in beta amyloid peptide solutions shifts the random coil structure of beta amyloid peptide to a beta sheet conformation [12], under conditions that do not lead to aggregation in solution. This conformational change is triggered by adsorption of beta amyloid peptides, on both hydrophobic and hydrophilic surfaces, over a period of time [12]. After such conformational change, the process may follow two pathways: (1) The beta sheet form of the peptide may act as a template for other peptides to continue adsorbing at the surface, yielding surface-confined aggregates and (2) the peptide in beta sheet form diffuses back into solution to interact with other beta amyloid peptides to form aggregates. It is also observed that the amount of amyloid deposits in beta amyloid solution increases with exposure time to the surface. Following these observations it is suggested that surfaces have a seeding effect on beta amyloid peptides in solution, inducing their aggregation [12].

Kinetic experiments on protein folding indicate that, at least, for most proteins the folding event is spontaneous and reversible, usually occurring rapidly within few seconds. It is evident that considering the huge number of possible conformations, a protein molecule has to follow a specific path to fold into its native state in order to meet the biologically feasible time scale. It has been also suggested that metastable, partially folded, states of a protein exist and they could provide vital information on the folding mechanism [13]. For studying complex folding and aggregation mechanisms, occurring on very fast time scales, access to atomistic details of the process is needed. This has

been realized by combining experimental approaches with powerful theoretical models. Although modern experimental techniques can monitor folding events occurring at nanosecond to microsecond time scales, theoretical approaches facilitate harnessing information at atomic resolution, at least for small peptides [13]. Theoretical models should be robust and should be able to cover different scales (in space and time) for them help identifying the determinants of protein aggregation process and to direct experimental investigations.

In recent years, theoretical and experimental studies on protein folding have focused on characterizing the folding mechanism in detail. Experiments have shown that not all proteins follow specific pathways in the folding process. Search for the native conformation is driven by the bias towards the native state over much of the protein energy landscape. Thus proteins must have tilt, funnel-shaped energy landscape (meaning that many conformations have higher free energy and few have lower energy) to fold quickly and efficiently. Theoretically, an ideal protein landscape will be smooth funnel shaped, however, for real proteins the folding funnel should carry landscape features like kinetics, smoothness or barrier heights. This concept has replaced the random search concept of Levinthal paradox in protein folding problem, and thus the “golf course” model of the protein potential energy surface is no longer valid [14]. A number of experimental examples and theoretical models have shown that essentially any protein or polypeptide can form non-native aggregates in solution, if incubated for sufficient time without intervention of other processes such as chemical or proteolytic degradation. The unfolded/non-native monomeric state of proteins is often identified as the reactive precursors that subsequently assemble to form ordered or disordered oligomers or higher molecular weight aggregates [15]. Studies have shown that a number of non-disease related proteins can also form amyloid fibrils. These studies also revealed that the process of destabilization of the native-structure is crucial in guiding normally soluble proteins into amyloid deposits [5]. This has led to the conclusion that fibril formation tendency is a generic property of the peptide backbone, and thus a generic mechanism of aggregation should exist. Recent experiments show [16] that replacing leucine by isoleucine in a diblock peptide drastically change the amyloid forming properties of the peptides. Those experiments added to the undisputed importance of the solvent on protein conformation

suggest that solvent effects and the specifics of the amino acid side chain **cannot** be ignored.

Thus, understanding the physical-chemical factors shaping a protein folding energy landscape will provide an insight into its misfolding/aggregation process. Most of the theoretical and experimental studies on protein folding mechanism aim at dissecting the folding free energy landscape of interested proteins, particularly by studying the effect of perturbations on their populated folding routes [17].

In this thesis, Metropolis Monte Carlo (MC) simulations were performed with positively charged peptides in aqueous solution to study changes in peptide conformations at solid/liquid interface, and their effects on protein aggregation. An intermediate-resolution diblock model peptide, comprising of 10 units of ALA (non-polar) and LYS (polar) amino acid residues, was used for the simulations. In the first approach to modeling, solvent effects were considered explicitly. The explicit model was then used to study two peptide molecules, in helical structure, at solid/liquid interfaces. In order to increase the number of peptide molecules in the simulation box, with reduced computational cost, an implicit solvent model was developed with non-additive hydrogen bonding and hydrophobic interaction potentials. The implicit model was used to simulate two peptides of helical structure at charged surfaces (to compare with the explicit model), and ten peptides of random coil structure in bulk and at solid/liquid interfaces. The two-peptide-water-surface system using the explicit model was also simulated with periodic switching of surface charge polarity. This was done to simulate the effect of “shaking” on the conformation of short peptides.

## 2. MODEL DEVELOPMENT

Metropolis Markov Chain Monte Carlo (MCMC) Simulations, in the canonical ensemble, was chosen over Molecular Dynamics (MD), because of our own expertise [18-21] and because the focus of study was the final equilibrium state rather than the kinetics. Experiments show that helix formation takes place in a few hundred nanoseconds, while a small beta hairpin takes a microsecond to form. The fastest known time scale for complete folding of a small protein is of order of tens of microseconds. With current computing capabilities, the attainable time scale in MD is too short for studying protein folding with a reasonable model at atomistic resolution. Thus, MCMC simulations are a better choice for the present study.

The existing Monte Carlo code available in Dr. Forciniti's group [20,21] can simulate a peptide of arbitrary length dissolved at infinite dilution in explicit water. The Monte Carlo suite is able to handle peptides at infinite dilution in bulk as well as peptides at infinite dilution at solid/liquid interfaces (where the interface consists of a collection of Lennard Jones sites having a partial charge). This Monte Carlo suite was extended to handle multiple peptides. This extension is very important because a new model described later in this section needed to be benchmarked against the existing atomistic model.

### 2.1. EXTENSION OF EXISTING MODEL TO FINITE CONCENTRATIONS

The existing MCMC code simulates a single peptide molecule in explicit water. The peptide is generated by the molecular modeling software ALCHEMY and a simple point charge (SPC) model is used for water (the extension to other water models is straightforward). The GROMACS force field is used. The surfaces of same charge density were placed in the xy-plane at the two ends of the simulation box. The capability of the current MCMC simulator, limited to run models to study infinite dilution behavior only, was augmented to investigate protein aggregation process, involving multiple peptide molecules. A listing of the modified code is included in Appendix A. Appendix B includes instructions to run the program.



As a first step into handle multiple peptide molecules, it was decided to update the existing explicit model code to simulate a two-peptide-water-surface system. The peptides were identical and all program variables required to simulate a peptide model were duplicated. As part of system initialization, the ALCHEMY generated peptide coordinate file was modified to include the different z-coordinates of the two peptides, keeping the x and y coordinates the same. Thus, two identical peptides were placed in the box along the z-coordinate between the two charged surfaces. Water molecules were then positioned around the peptides in numbers to keep water density inside the simulation box equal to  $1 \text{ g/cm}^3$ . To the existing peptide-surface, peptide-water, water-surface and water-water inter-molecular interactions, peptide-peptide interactions were added. The intramolecular energy estimations for torsional and van der Waal's interactions were also extended for the second peptide molecule. Periodic boundary conditions were used in the x and y directions only, and the charged surfaces at the two ends of the box (in z-direction) ensured all molecules to stay inside the simulation box throughout the run. All the interactions were calculated by the minimum image (MI) convention, which is considered to be reliable for studying the structural features of protein aggregation at interfaces [18].

## **2.2. MIXING**

A “mixing effect” was induced for the explicit 2-peptide(random coil)-water-surface model by continuously changing the polarity of the surfaces throughout the simulation run. It is believed that this will help in understanding, qualitatively, the effects of mixing on protein aggregation. A model diblock peptide, 5ALA-5LYS, was used for this experiment. In order for the peptides to have enough space for adsorption, the surfaces were extended along the y-axis and the peptides were initially arranged along y-axis equidistant from the surfaces. For every 20,000 MC steps the charges on the surfaces were switched and the simulation was continued.

## **2.3. DEVELOPMENT OF IMPLICIT WATER MODEL**

It is evident that atomic-level simulations of the protein aggregation process are computationally uneconomical. To solve this problem, it was decided to use an implicit

model to account for solvent effects and thereby reducing the degrees of freedom introduced by the explicit treatment of water molecules. Takada et al. [22] performed MD simulations of a helical protein and a random hetero polymer to study their folding dynamics. They used a simplified description of the protein, which has a realistic backbone but single sphere side chains, to reduce the number of degrees of freedom. Solvent effects were captured by incorporating hydrogen bonds and hydrophobic interactions implicitly. Anisotropic hydrogen bonds (HB) were modeled by a particular combination of attractive and repulsive forces between donor and acceptor atoms; with the bond strength reflecting the local dielectric constant that is dependent on protein configuration. Hydrophobic (HP) interaction were modeled in terms of local density of the peptide atoms; this is similar (but computationally cheaper) to the nonadditive HP interaction model written in terms of accessible surface area (ASA) [22].

The HB/HP interaction model used by Takada's group [22] was extended to model the adsorption/aggregation of peptides at solvent-surface interfaces. By treating the solvent implicitly, finite peptide concentrations became manageable avoiding heavy computational load. In this new model, all peptide atoms (including the side chain groups) are treated explicitly except for the methyl groups, CH<sub>x</sub> ( $x=1-3$ ), which are treated as united atoms. This is an advance to the model introduced by Takada's team [22] who oversimplified the side chains by modeling them as a united atom. The more detailed description of the peptide molecule and the extension to finite concentrations in this model call for some changes in HP/HB interaction relations as proposed by Takada et al. [22]. These modifications are described below.

The system potential term,  $V$ , is defined as,

$$V = V_{lc} + V_{nonlc} \quad (1)$$

where,  $V_{lc}$  and  $V_{nonlc}$  are the local potential and non-local potential terms, respectively. These terms are defined as,

$$V_{lc} = V_{tor} + V_{vdv-lc} \text{ and,} \quad (2)$$

$$V_{nonlc} = V_{vdw} + V_{HP} + V_{HB} \quad (3)$$

where,  $V_{tor}$  is the torsional potential, which was calculated the same as in our existing explicit model (Refer to Appendix A). The local van der Waals (vdW) potential (only for 1-4 pairs),  $V_{vdw-lc}$ , was estimated along with the non-local vdW potential,  $V_{vdw}$ , by LJ plus a Coulombic term following the approach used in our explicit model (Refer to Appendix A).

Following Takada et al.'s work [22], the hydrophobic interaction potential ( $V_{HP}$ ) was incorporated in the model. It should be notice that the HP potential depends on the local density of the peptide atoms of **all** peptide molecules in the simulation box. For the HP interaction model (described in later section), only interactions among  $C\alpha$  and  $C\beta$  atoms were considered, where the interactions between  $C\beta$  atoms depend on the type of the amino acid.

The existing code was modified to include the hydrogen bond potential ( $V_{HB}$ ). This potential was modeled anisotropic by considering attractive forces between CO and NH while a repulsive force is introduced between NH and NH, NH and  $C\alpha H$ , between  $C\alpha H$  and CO, and between CO and CO. Since all these groups are treated explicitly in the new model, interactions between particular atoms of the groups (as described in Figure 2.1) were selected. The interaction scheme in the HB model is amino acid type specific, as it consists of feasible pair-wise interactions among the backbone atoms and also among side chain atoms. In addition to the intra-molecular amino acid interactions for estimating  $V_{HB}$  as proposed by Takada et al. [19], the new model considered inter-molecular amino acid interactions also. The proposed HB interaction model can be represented as (for  $K^{th}$  peptide molecule),

$$V_{HB, \text{intra}, K} = \epsilon_{HB} \sum_{I, J \in K} S_{HB, IJ} u_{HB}^{a,r}(r_{ij}) + \epsilon_{HB} \sum_{c} S_{HB, c, I} \quad (4)$$

$$V_{\text{HB,inter,K}} = \epsilon_{\text{HB}} \sum_{\text{J} \in \text{K}} \sum_{\substack{\text{peptide Q} \\ \text{Q} \neq \text{K}}} \sum_{\text{I} \in \text{Q}} S_{\text{HB,IJ}} u_{\text{HB}^{\text{a,r}}}(r_{ij}) \quad (5)$$

$$V_{\text{HB,K}} = V_{\text{HB,intra,K}} + \left( \frac{V_{\text{HB,inter,K}}}{\text{NMOLP}} \right) \quad (6)$$

where, I and J represent the amino acids to which the  $i$ th and  $j$ th atoms belong, respectively. The distance between those atoms (in Å units) is stored in the term  $r_{ij}$ . The variables K and Q represent any two peptide molecules in the system. The constant HB strength is represented by the variable  $\epsilon_{\text{HB}}$ .

results in anisotropic HB interactions. These interactions may occur between intramolecular amino acids or between intermolecular amino acids.

The term  $S_{\text{HB,IJ}}$  accounts for the local density dependence of actual hydrogen bonds, and is defined as,

$$S_{\text{HB,IJ}} = \frac{S_{\text{HB,I}} + S_{\text{HB,J}}}{2} \quad (7)$$

$S_{\text{HB,I}}$  is a function of the term  $\sum_{\text{K}} u_{\text{HP}}(r_{\text{IK}})$ ,  $\eta_{\text{HB,min}}$ ,  $\eta_{\text{HB,max}}$ , where  $\sum_{\text{K}} u_{\text{HP}}(r_{\text{IK}})$  monitors the local density on the  $\text{I}^{\text{th}}$  amino acid and  $r_{\text{IK}}$  is the distance between  $\text{C}^{\alpha}$  of  $\text{I}^{\text{th}}$  and  $\text{K}^{\text{th}}$  residues. The functional form of  $u_{\text{HP}}(r)$ , which represents the distance dependent interaction (with pre-set parameters,  $\eta$ ) as a smooth switching function, is

$$u_{\text{HP}} = \begin{cases} 1 & r < \sigma_{\text{HP1}} \\ \frac{1}{2} \left( 1 + \cos \pi \frac{r - \sigma_{\text{HP1}}}{\sigma_{\text{HP2}} - \sigma_{\text{HP1}}} \right) & \sigma_{\text{HP1}} \leq r \leq \sigma_{\text{HP2}} \\ 0 & r > \sigma_{\text{HP2}} \end{cases}$$

The function  $S_{\text{HB}}$  is a smooth switching function and has the explicit form of,

$$S_{\text{HB}}(x, x_{\text{min}}, x_{\text{max}}) = \begin{cases} 1 & x > x_{\text{max}} \\ \frac{1}{2} \left( 1 + \cos \pi \frac{x_{\text{max}} - x}{x_{\text{max}} - x_{\text{min}}} \right) & x_{\text{min}} \leq x \leq x_{\text{max}} \\ 0 & x < x_{\text{min}} \end{cases} \quad (9)$$

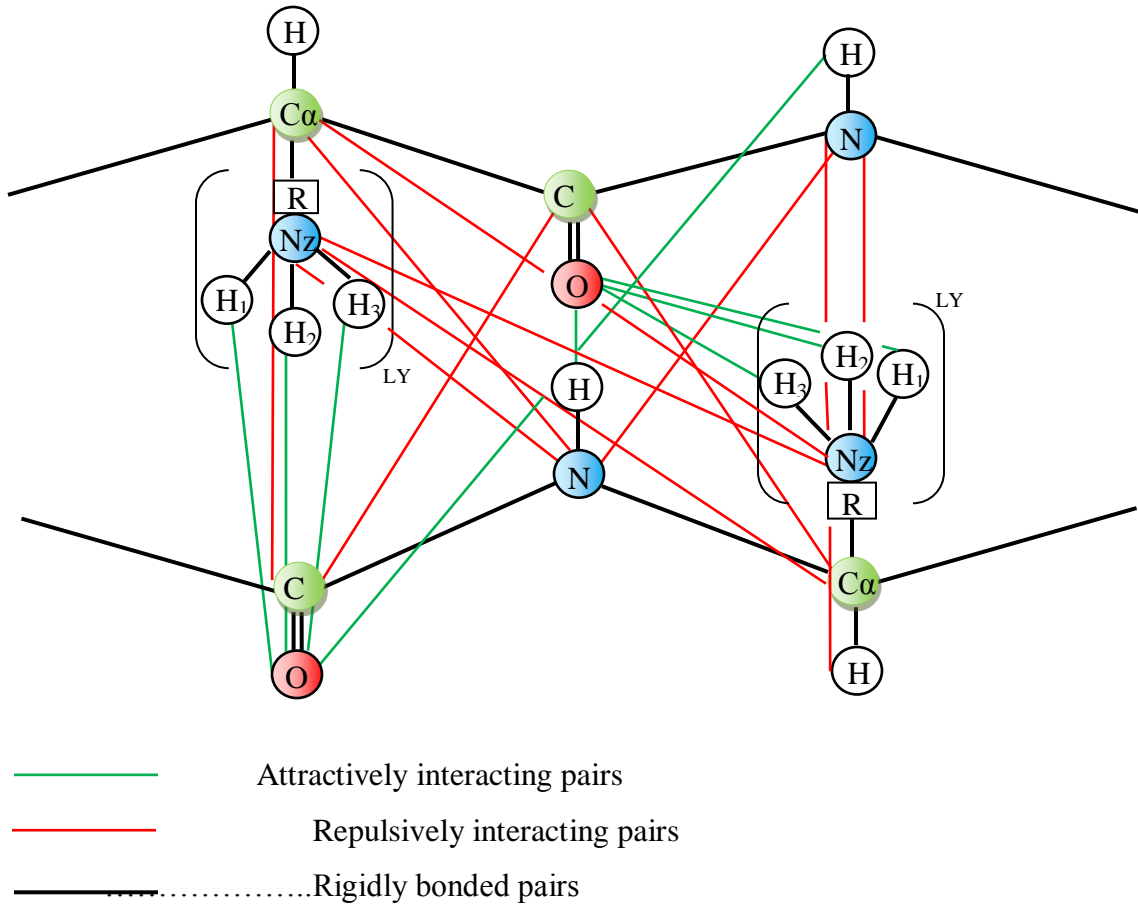


Figure 2.1 A schematic figure of the side-chain dependent HB interactions, between two selected lysine amino acid groups, as proposed in this model. The repulsive forces are half as strong as the attractive ones and the combination of attractive and repulsive forces (8)

The terms  $u_{\text{HB}}^{\text{a,r}}(r_{ij})$  account for the distance dependent part of the potential, which based on the atom type of  $i, j$  may be either attractive or repulsive. (Refer to Figure 2.1 for details of attractive/repulsive interactions between atoms of two neighboring amino acid groups). The interaction has the form,

$$u_{\text{HB}}^{\text{a}}(r_{ij}) = 60 \left[ \frac{1}{12} \left\{ \frac{\sigma_{\text{HB}}}{r_{ij}} \right\}^{12} - \frac{1}{10} \left\{ \frac{\sigma_{\text{HB}}}{r_{ij}} \right\}^{10} \right] \quad (10)$$

$$u_{\text{HB}}^{\text{r}}(r_{ij}) = 30 \left[ \frac{1}{10} \left\{ \frac{\sigma_{\text{HB}}}{r_{ij}} \right\}^{10} \right] \quad (11)$$

where,  $\sigma_{\text{HB}}$  represents the length scale for hydrogen bonding. The model for HP interaction potential for the  $K^{\text{th}}$  peptide is given by,

$$V_{HP,K} = \left[ -\varepsilon_{HP}^{\alpha} \sum_{I,J \in K} S_{HP,I} \left( \sum_{J \neq I} u_{HP}(r_{ij}), \eta_{HP,\alpha} \right) - \right. \\ \left. \left[ \sum_{L,M \in K} \varepsilon_{HP,L}^{\beta} S_{HP,L} \left( \sum_{M \neq L} u_{HP}(r_{LM}), \eta_{HP,\beta} \right) \right] \right] \quad (12)$$

Where I and J represents  $\alpha$  carbons, and L and M represents  $\beta$  carbons in peptide K.  $\varepsilon_{HP}^{\alpha}$  and  $\varepsilon_{HP,L}^{\beta}$  are, respectively, the HP energy parameters for  $\alpha$  carbons and side chains. The function  $S_{HP}$  represents buriedness of the atom and is defined as,

$$S_{HP}(x, xmax) = \begin{cases} 1 & x > xmax \\ \cos\left(\frac{\pi}{2} \frac{xmax - x}{xmax}\right) & x \leq xmax \end{cases} \quad (13)$$

The listing of the Fortran code that implements this implicit model is included in Appendix C. Appendix D includes detail instructions to run the code.

### 3. RESULTS

The explicit model was used to study the adsorption of multiple peptides and to study the effect of mixing on their conformation. The explicit model extended to handle multiple peptides was used to simulate two peptides in a box. The purpose of this exercise was to benchmark the results obtained with the implicit model. Multiple peptides were “computationally shaken” by switching the polarity of the simulation box. For the sake of clarity the results of all the simulations done with the explicit model are presented in Appendix E.

To code the proposed implicit model in FORTRAN, the existing MC code for simulating a single-peptide-surface system in vacuum was modified. In the new code up to fifty identical peptide could be simulated in the user defined simulation box with/without charged surfaces. After extending the existing explicit model interaction subroutines for peptide-surface and intra-peptide interactions (of torsional and vdW forces) to handle multiple peptide system, a subroutine to estimate inter-peptide vdW interactions was coded. Then to calculate the implicit solvent interactions (of HB/HP forces) four different subroutines were coded for each of the following:

1. UHP\_est(): This is used to calculate the distance dependent interaction term  $u_{HP}$  that monitors the local density of each amino acid sites of all the peptide molecules available in the simulation box. The two types of  $u_{HP}$  terms, one defined in terms of  $C\alpha$ - $C\alpha$  and the other in terms of  $C\beta$ - $C\beta$  distances between all pairs of amino acid residues, are determined by this subroutine. For this the  $C\alpha$  and  $C\beta$  atomic positions of the selected peptide molecular model should be provided in the input peptide configuration file (A5-L5.cod for our model).
2. HPenrg(): It is used to estimate the HP interaction energy of the selected peptide molecule.
3. HBenrg\_intra(): It calculates the intramolecular HB interaction energy for the selected peptide molecule.
4. HBenrg\_inter(): This subroutine determines the intermolecular HB interaction energy for a given peptide.

At first, a two-peptide(helix)-surface system was simulated using the newly coded implicit model program. This allows benchmarking the proposed implicit model using the results obtained from the existing explicit model simulation of the same system. The energy profile, end-to-end distance profile and COM z-location profile for this system were plotted vs. the number of Monte Carlo steps (NMCS) (refer to Figures 3.1, 3.2 and 3.3 respectively).

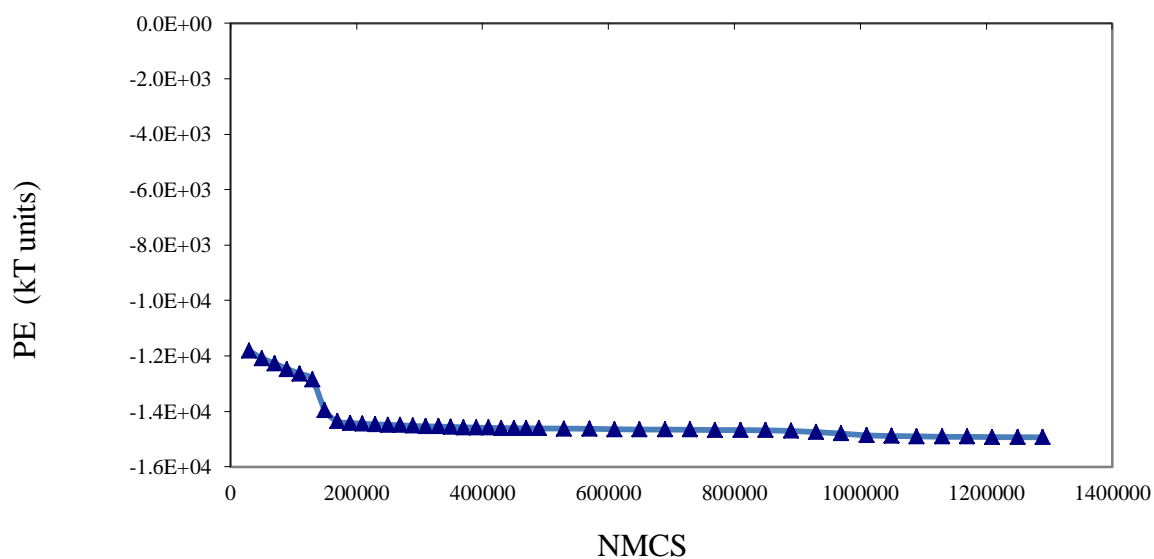


Figure 3.1 Total energy vs. NMCS for 2-peptide(helix)-surface system simulated using the implicit model.



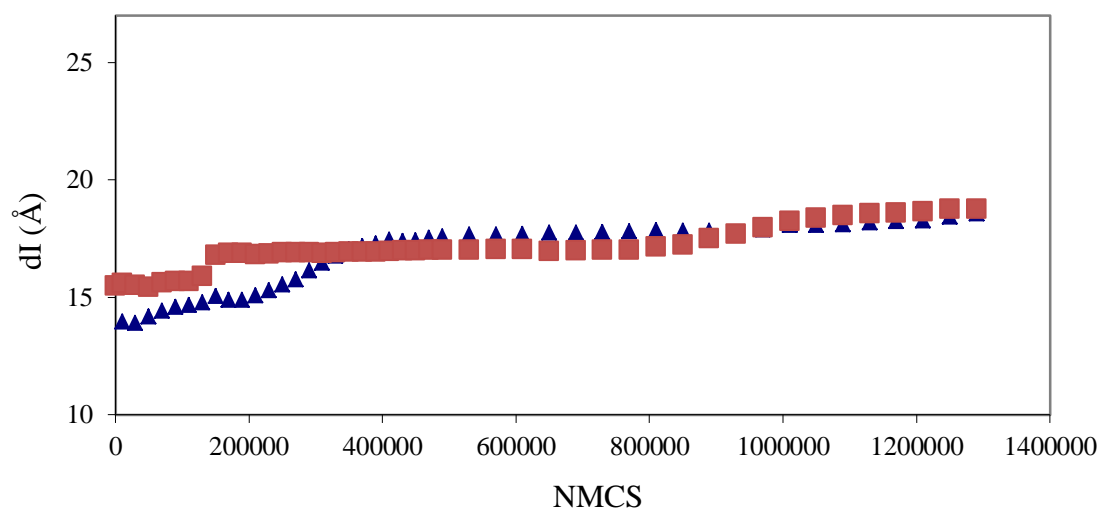


Figure 3.2 Peptide ETED profile for 2-peptide(helix)-surface system simulated using the implicit model.

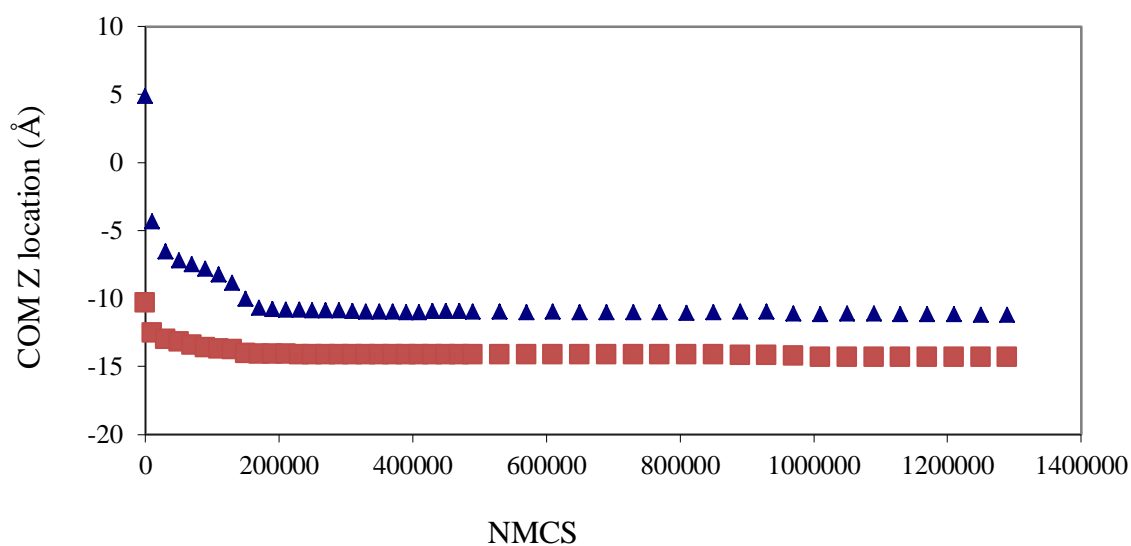


Figure 3.3 COM z-location profile for 2-peptide(helix)-surface system simulated using the implicit model.

Figure 3.4 shows snapshots of the initial and final system configurations. The two peptides move towards the surface and were adsorbed flat on the surface increasing the electrostatic contacts between the oppositely charged residues (LYS) and the surface. The

peptides orient parallel to each other, similar to what was observed in the explicit model run (refer to Appendix E).

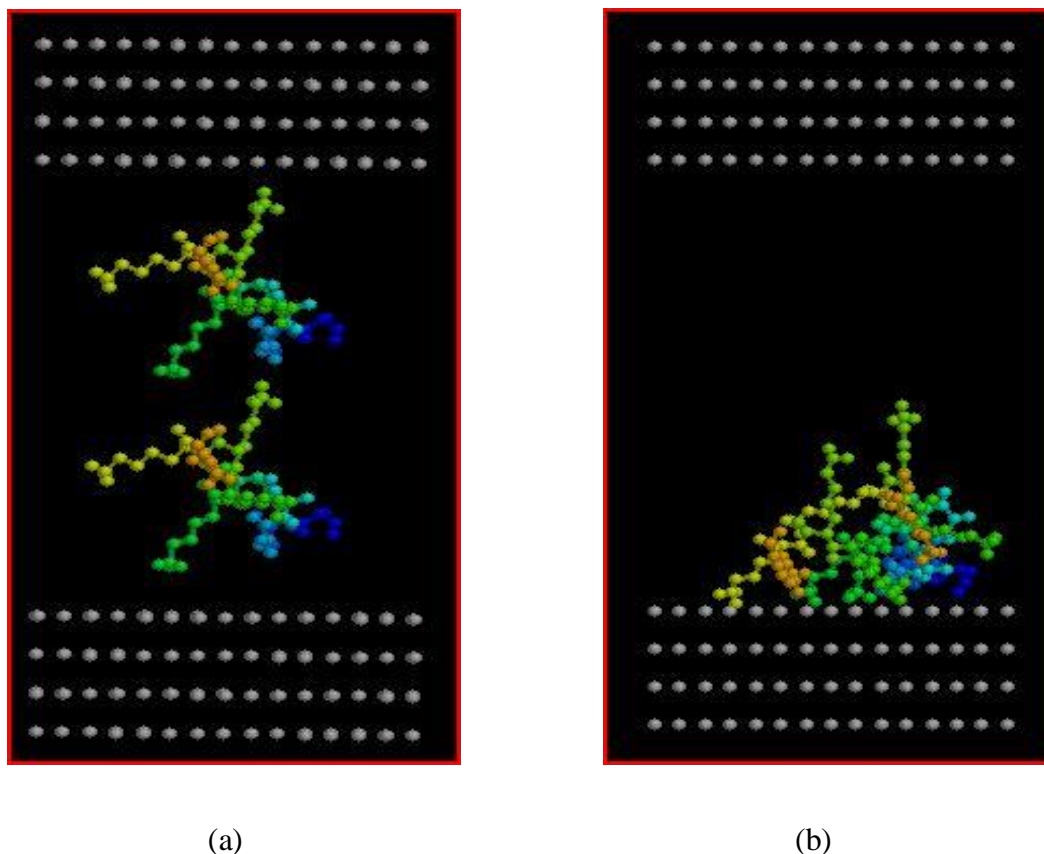


Figure 3.4 Snapshots of initial (a) and final (b) configurations of 2-peptide(helix)-surface system simulated using the implicit model.

The end-to-end distance profiles suggest some re-structuring of the peptide molecules, with both peptides partially losing their secondary helical structure. This change in the secondary structure could be also viewed in the snapshots shown in Figure 3.5. Unlike the explicit model, the peptides in the implicit model follow matching end-to-end distances after reaching equilibrium. The COM z-location profiles and the snapshots indicate that the two peptides lie closer and parallel on the surface than in the explicit model. This could be due to two reasons. First, the water molecules trapped between the surface and the peptides, in the explicit model, affect the interaction between the surface

and the peptide .[19] The second possible reason is the larger peptide structural rearrangement observed in the implicit model; this is evident from longer end-to-end distances for both peptides (18.56 Å and 18.75 Å) compared to those for the explicit model (15 Å and 17.04 Å).

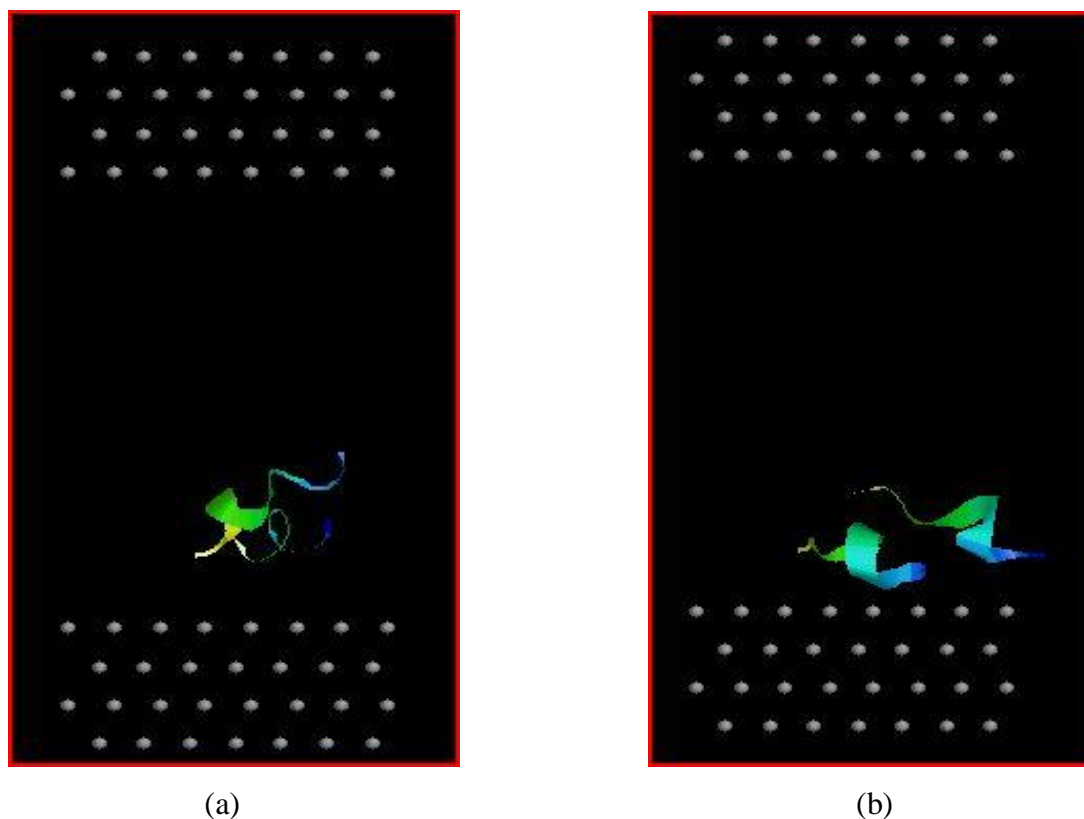


Figure 3.5 “Ribbon” graphic snapshots of final configurations of 2-peptide(helix)-surface system simulated using the explicit (a) and the implicit models (b).

In the second simulation a 10-peptide (5ALA-5LYS) system in bulk and in between two charged surfaces was used. The selected simulation box dimensions (27.748 X 48.061 X 85 Å) were unchanged for both model systems, thus they differ only in the presence of charged surfaces. After placing the peptides in the box, their initial position in the box was randomized by “computational shaking”. Every 20,000 MC steps the polarity of the surfaces was switched generating a random configuration after 0.1 million MC steps. Refer to Figure 3.6 for a snapshot of the final distribution of the peptides after

initial shaking. This configuration was then used as the initial configuration for simulating the 10-peptide system with and without surfaces.

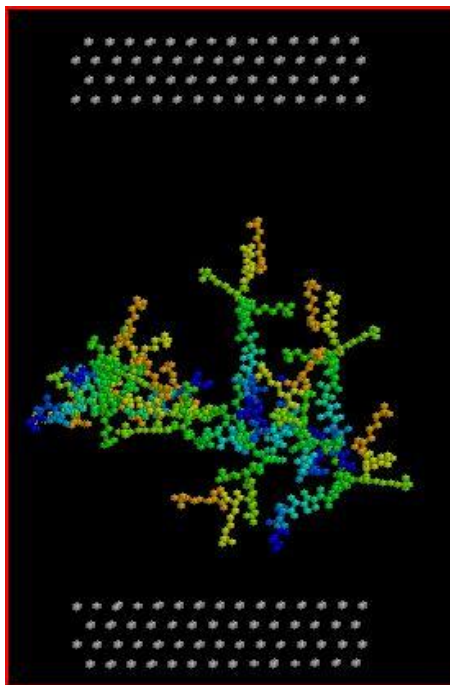


Figure 3.6 Initial configuration obtained by computational “shaking” for 10-peptide(random coil) simulations.

For the 10-peptide system without surfaces, the energy of the box is minimized after completing 0.4 million MC steps. The simulation was continued to complete a total of 0.722 million steps, while the system energy remained at the minimum value corresponding to the equilibrium configuration. Figure 3.7 shows the energy profile curve, and Figures 3.8 and 3.9 show the end-to-end distance profiles and COM z-location profiles for all the 10 peptide molecules inside the simulation box. Figures 3.10 and 3.11 are snapshots of the simulation box after equilibration. After equilibrium, the peptides orient in clusters with their hydrophobic ends forming the inner core. The COM z-location profile also indicates this clustering effect with respect to the z-axis.

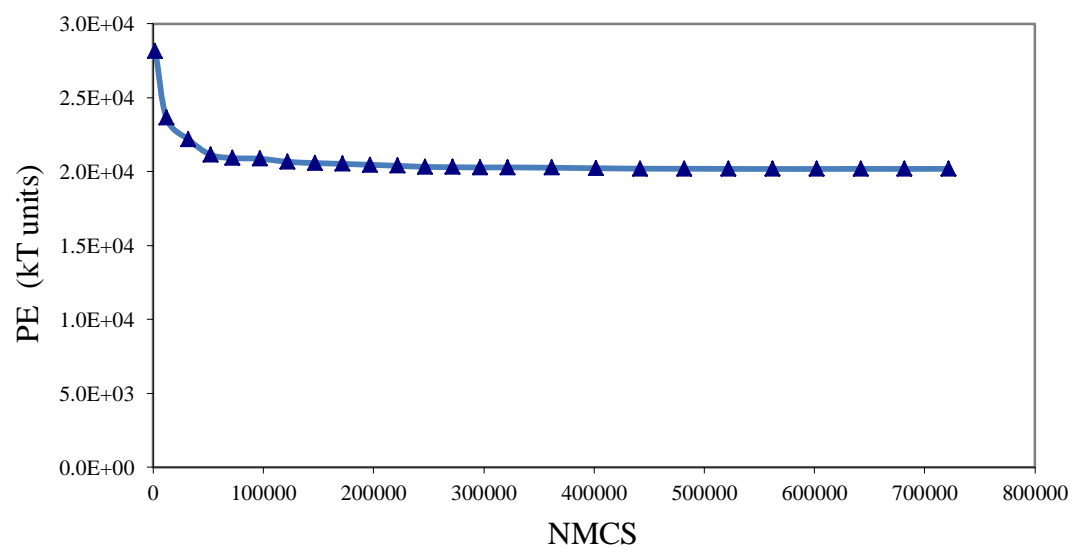


Figure 3.7 Energy profile for 10-peptide(random coil) system simulated using the implicit model.

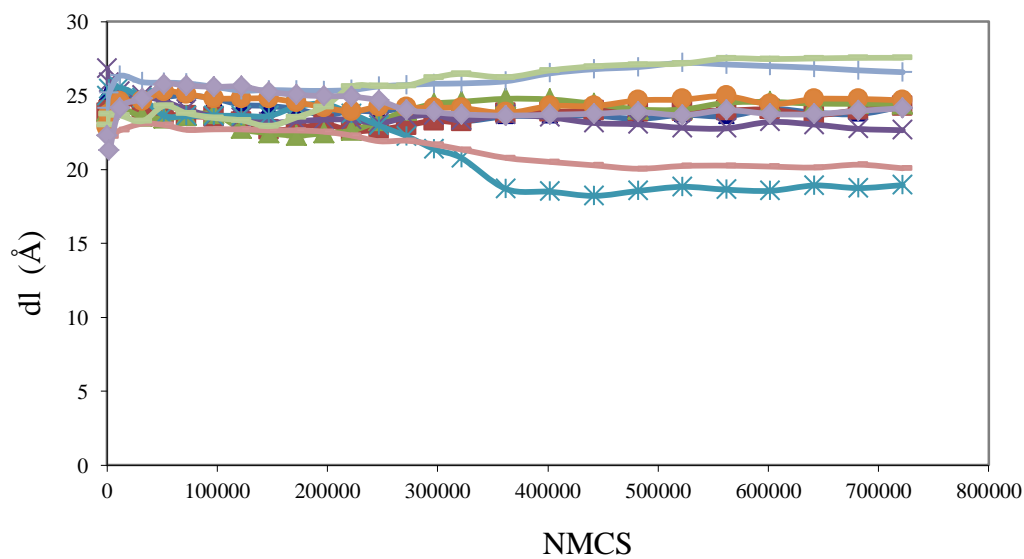


Figure 3.8 Peptide ETED profile for 10-peptide(random coil) system simulated using the implicit model.

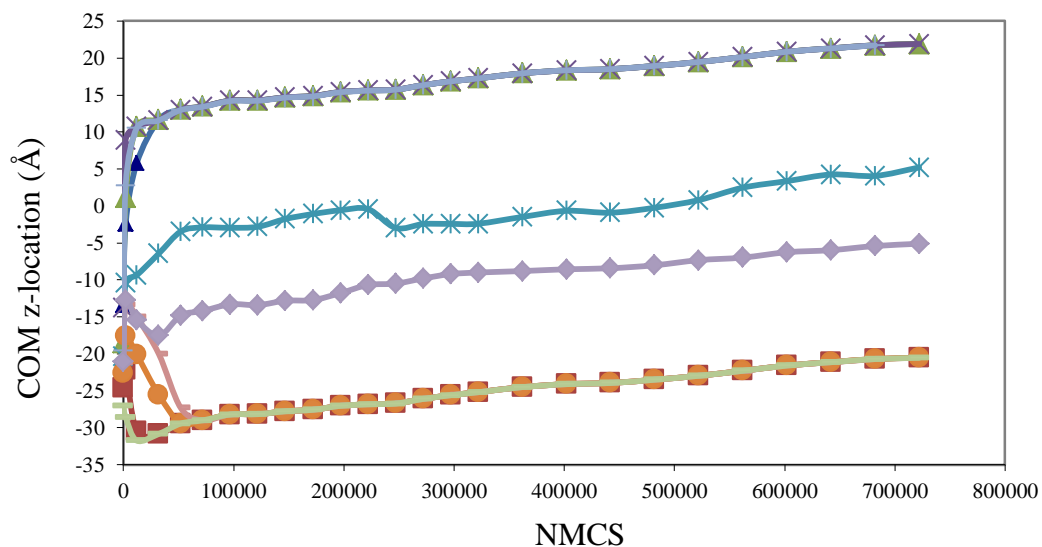


Figure 3.9 COM z-location profile for 10-peptide(random coil) system simulated using the implicit model.

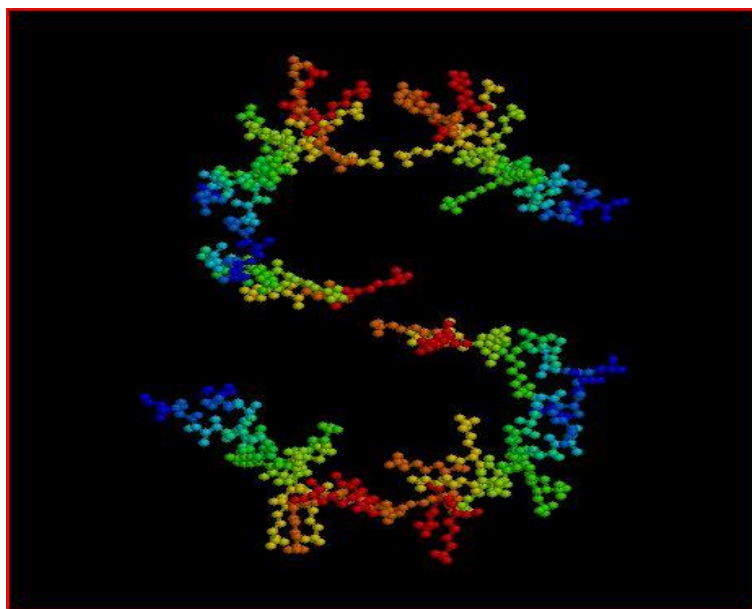


Figure 3.10 “Ball & Stick” snapshot of the final configuration for 10-peptide(random coil) system simulated using the implicit model.

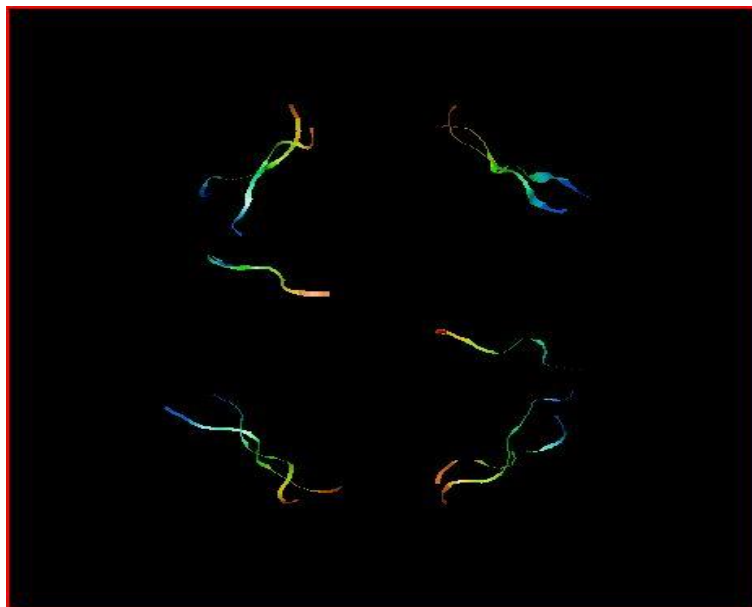


Figure 3.11 “Ribbon” snapshot of the final configuration for 10-peptide(random coil) system simulated using the implicit model.

The system energy, end-to-end distance of all peptides, and the COM z-location of all peptides vs. the number of Monte Carlo steps for the 10-peptide-surface system, simulated using the implicit model and in the presence of charge surfaces are shown in Figures 3.12, 3.13 and 3.14, respectively. The energy reaches its minimum value after 0.7 million MC steps. All peptides moved towards the negatively charged surface, and each of them oriented perpendicular to the surface. The positively charged LYS ends of the peptides are in contact with the surface thereby maximizing electrostatic interactions. The hydrophobic ends, extending out from the surface, clustered together and sink-in closer to the surface. This effect is further evident from the negative slope of the end-to-end distance profile (Figure 3.13) for the peptides. Figure 3.15 is a snapshot of the simulation box during the production run. A close-up view of the snapshot near the negatively charged surface is displayed in Figure 3.16. For the implicit model simulations of 2-peptide and 10-peptide systems with charged surfaces, the change in peptide end-to-end distances depends on their orientation with respect to the surface. The ETED decreased for peptides oriented normal to the surface and it increased for peptides parallel to the surface.

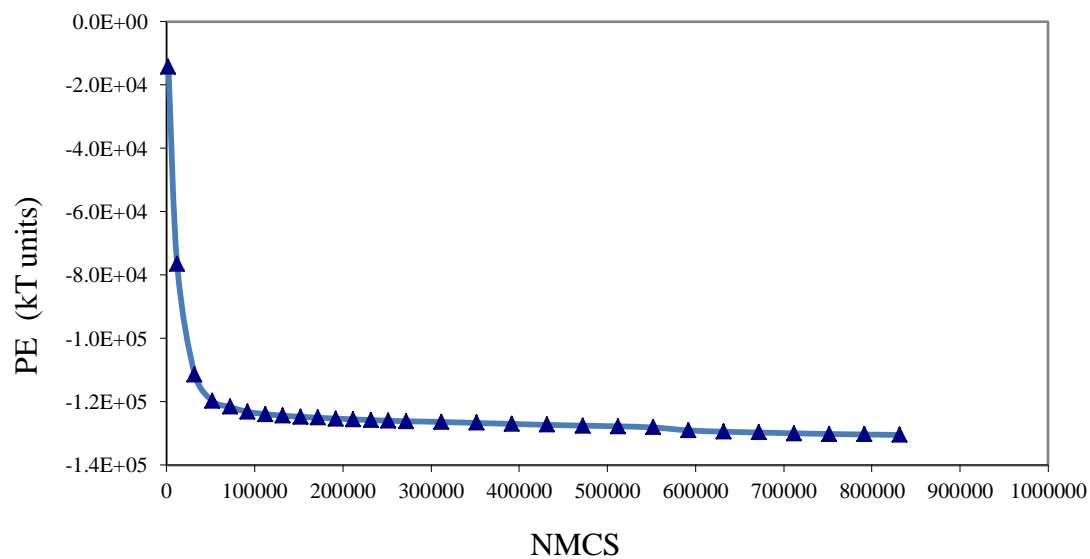


Figure 3.12 Energy profile for 10-peptide(random coil)-surface system simulated using the implicit model.

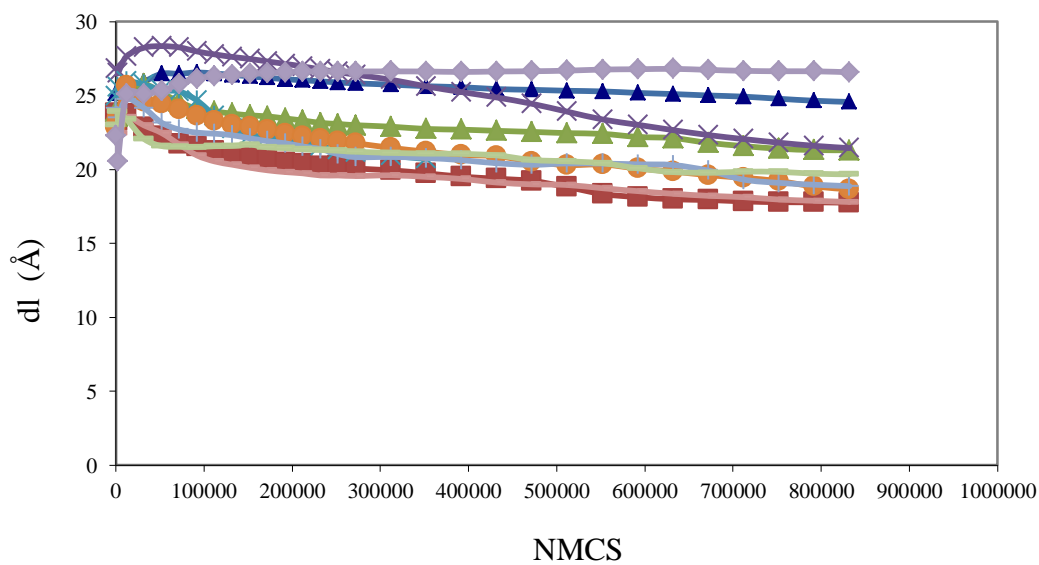


Figure 3.13 Peptide ETED profile for 10-peptide(random coil)-surface system simulated using the implicit model.



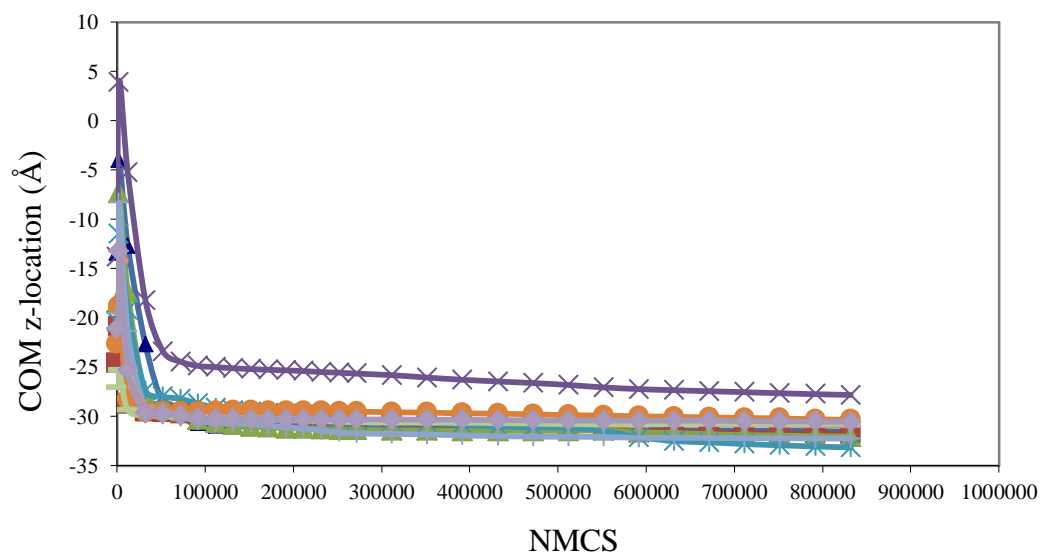


Figure 3.14 COM z-location profile for 10-peptide(random coil)-surface system simulated using the implicit model.

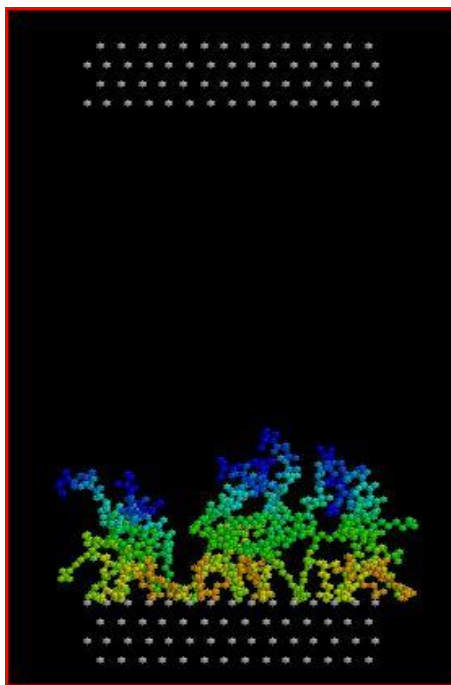


Figure 3.15 Snapshot for 10-peptide(random coil)-surface system simulated using the implicit model.

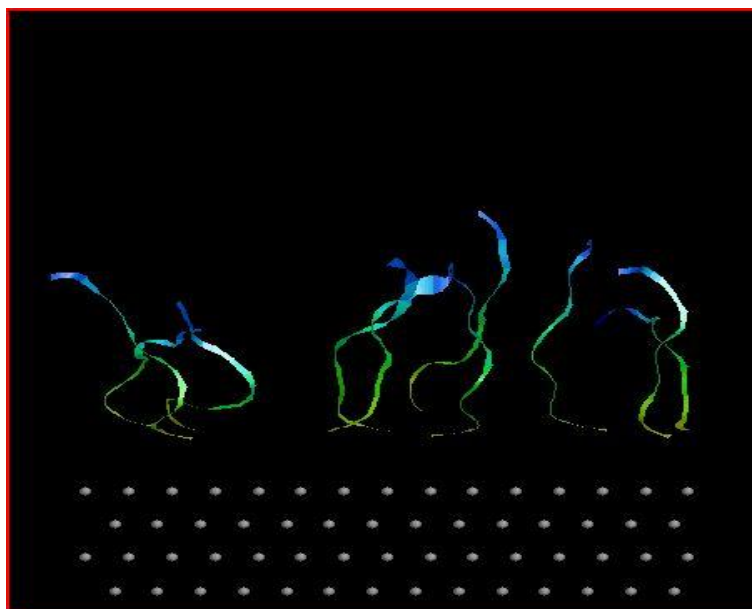


Figure 3.16 A close-up view near the negatively charged surface of the final configuration of 10-peptide(random coil)-surface system using the implicit model.

The implicit model was successfully used to simulate a multi-peptide system in a reasonable computational time. The results for 10-peptide (random coil) systems shows peptide restructuring at the surface and clustering of hydrophobic ends, which may be seen as a hydrophobic collapse effect. These effects are consistent with current knowledge of protein behavior near solid/liquid interfaces [8]. However, the local density dependent HB model did not seem to induce any secondary structure formation in random coil peptide structures. This may be due to the shortness of the model peptides (only 10 residues) used in our simulation. Short peptides are not expected to easily fold into secondary structures [22].

The 2-peptide (helix) simulation in implicit model showed structural changes similar to those observed with the explicit system. However there is a considerable mismatch in the ETED profiles. This could be due to the lack of tuning of HB model parameters and the short peptide size selected for the system model. Although the implicit model may not be able to capture all the solvent effects (like dehydration effect around the peptide and at the surface interface), it could qualitatively capture (with proper tuning) conformational changes which may induce the formation of aggregates.

APPENDIX A

EXPLICIT MODEL SIMULATION  
PROGRAM CODE

C\*\*EXPLICIT SOLVENT MODEL FOR 2-PEP-WATER-SURFACE SYSTEM. EXISTING CODE BY AMOL IS JUST REPLICATED TO ACCOUNT FOR THE ADDITIONAL PEPTIDE MOLECULE. CORRECTION MADE FOR THE INITIAL SYSTEM ENERGY (V) CALCULATION. LIMITS ARE: NATOM=200, RESNUM=20, NMOLW=2000, NSURF=600, NMCS=50000. PEPTIDES ARE ARRANGED ALONG THE Z-AXIS. THE SURFINP FILE STAYS THE SAME AND A5-L5 FILE IS MODIFIED TO INCLUDE Z-COORDINATES FOR BOTH PEPTIDES. MAKE CHANGES IN VARIABLE INITIALIZATION FOR PRODUCTION RUN\*\*

```

program main
implicit NONE
external INCONFIG,NRGTOT,RNDMOVE
external SUMBINW,RANDOM
external PDBOUT,DISTPP,NRGPP
external DISTW,NRGW
external DISTWMSURF
external DISTWMPEP,DISTPEPSF
external NRGWMPEP,NRGPEPSF
external MOVEPEP,pepenrg
external SUMBINPEP,TORSION

```

\*\*\*\*\*

```

integer ACCMOVEW,ACCMOVEWACC,ACCMOVEWTMP,C,I,ISEED,J,M,K,N
integer NADJUST,NBIN,NMCS,NMCSACC,NMCSOLD
integer NMCSTMP,NMOLW,NPTS,NSAVE,RUNSTAT,STEP
integer GSUM(600,3),NMOL,NSURF
integer ZWSUM(2000,2),NBINZ
integer NATOM,NHB,HB(100),NEND,CEND
integer ATOM(200),AA(200),GROUP(200)
integer ACCTOR,ACCPEP
integer ZPEP1SUM(2000,3),ZPEP2SUM(2000,3)
integer NEIBOURWSUM1(100),NEIBOURWSUM2(100)
integer GRPWSUM1(600,2,100),GRPWSUM2(600,2,100)
integer RESNUM
double precision ACCPEPACC,ACCPEPTMP,AVGACCPEP
double precision AVGACCTOR,ACCTORACC,ACCTORTMP
double precision DELZ,UPPOLD,UPPNEW
double precision GOFZW(2000,2),AW(200)
double precision VOLUMEZ,TMP2
double precision AVGNRG,AVGVW
double precision DELR,DELV,DENS,ESUM,ESUMACC,ESUMACCTMP,LOHW
double precision LMIN,MAXROTW,MAXTRNW,PI
double precision LX,LY,LZ
double precision QFACT,RANDOM,RMIN,RMAX,RXMAX,RXMAX2,TEMP
double precision THHOH,AVGACCW
double precision V,VERR

```

double precision VERRSUM  
 double precision VWNEW,VWOLD,VOLUME,VWATER,VWERR  
 double precision VWERRSUM,VWINI,VWSUM,VWSUMACC,VWSUMACCTMP  
 double precision QSURF,RSURF(600,3)  
 double precision AIIW(3),A12W(3,3),CIIW(3),C6W(3,3)  
 double precision AII(2),CII(2),Q(2)  
 double precision A12WS(3,2),C6WS(3,2),QCONWS(3,2)  
 double precision D2WNEW(3,2000,3),D2WOLD(3,2000,3)  
 double precision GOFR(600,3)  
 double precision QW(3),QCONW(3,3)  
 double precision RW(2000,3,3),RWMNEW(3,3),RWMOLD(3,3)  
 double precision VTOT(0:50001),VTOTPT(100),VW(0:50001)  
 double precision VWPT(100)  
 double precision VWMSURFNEW,VWMSURFOLD  
 double precision D2WSURF1OLD(3,600),D2WSURF1NEW(3,600)  
 double precision D2WSURF2OLD(3,600),D2WSURF2NEW(3,600)  
 double precision PEPCORD1(200,3),PEPCORD2(200,3)  
 double precision QPEP(200),APEP(200),A12PP(200,200),CPEP(200)  
 double precision phi1(20),psi1(20),phi2(20),psi2(20)  
 double precision COM1(3),COM2(3),C6PP(200,200)  
 double precision v1,v2,v3,v4,v5,v6,QCONPP(200,200)  
 double precision A12WP(200,3),C6WP(200,3),QCONWP(200,3)  
 double precision A12PS(200,2),C6PS(200,2),QCONPS(200,2)  
 double precision D2WPEP1OLD(200,2000,3),D2WPEP2OLD(200,2000,3)  
 double precision D2WPEP1NEW(200,2000,3),D2WPEP2NEW(200,2000,3)  
 double precision UWMPEP(2000,2)  
 double precision UPEP1SFNEW,UPEP2SFNEW,UPEP1SFOLD,UPEP2SFOLD  
 double precision D2WMPEP1NEW(200,3),D2WMPEP1OLD(200,3)  
 double precision D2WMPEP2NEW(200,3),D2WMPEP2OLD(200,3)  
 double precision UPEP1WM,UPEP2WM,COM1OLD(3),COM2OLD(3)  
 double precision UPEP1WNEW,UPEP1WOLD,UPEP2WNEW,UPEP2WOLD  
 double precision UWMPEPOLD(2000,2)  
 double precision PEPCORD1OLD(200,3),PEPCORD2OLD(200,3)  
 double precision phi1old(20),psi1old(20),phi2old(20),psi2old(20)  
 double precision MXTPEP,MXRPEP,maxpsi,maxphi  
 double precision D2PEP1SF1NEW(200,600),D2PEP1SF2NEW(200,600)  
 double precision D2PEP2SF1NEW(200,600),D2PEP2SF2NEW(200,600)  
 double precision D2PPOLD(200,200),D2PPNEW(200,200)  
 double precision NEWP1ENRG,OLDP1ENRG,NEWP2ENRG,OLDP2ENRG  
 double precision RWM(3,3)  
 double precision UPEP1WMOLD,UPEP1WMNEW,UPEP2WMOLD,UPEP2WMNEW  
 double precision VPEP,PE1,PE2

```

double precision NEIBOURW1(100),NEIBOURW2(100)
double precision GOFRWP1(600,2,100),GOFRWP2(600,2,100)
double precision COSAVG1,COSSQAVG1,COSAVG2,COSSQAVG2
double precision ZPEP1(2000,3),ZPEP2(2000,3)
double precision COSSQSUM1,COSSUM1,COSSQSUM2,COSSUM2
double precision ETED1(50000),ETED2(50000),ETEDP1(100),ETEDP2(100)
double precision ETED1T,ETED2T,ETED1AVG,ETED2AVG
double precision CMP1T,CMP2T,CMP1AVG,CMP2AVG
double precision CM1(50000),CMP1(100),CM2(50000),CMP2(100)
double precision VSUS(50000),VSOP(50000)
double precision VPS(50000)
double precision VSUSPT(100)
double precision VPSPT(100),VSOPPT(100)
double precision VSUSS,VSOPS
double precision VPSS,VSURSOLI,VRESTI
double precision VSURSOL,VREST,UPEPWI
character CFGINP*15,CFGOUT*15,OUTPUT*15
character TITLE*50,PDB*15
character SURFINP*15,SURFNAME*15
character PEP*15
parameter (PI=3.14159265d+0)
C *****
C ** INPUT DATA.                **
C *****
  open(unit=5,file='input1.inp',status='unknown',
& form='formatted')
  rewind(5)
  read(5,*)
  read(5,*)
  read(5,*) TITLE
  read(5,100) RUNSTAT
  read(5,*) CFGINP
  read(5,*) CFGOUT
  read(5,*) OUTPUT
  read(5,*) PDB
  read(5,110) NMOLW
  if(NMOLW.GT.2000) STOP 'Matrix size too small'
  read(5,120) NMCS
  if(NMCS.GT.50000) STOP 'Matrix size too small'
  read(5,140) TEMP
  read(5,*) NSAVE
  read(5,*) NADJUST

```

```

read(5,*) SURFINP
read(5,*) QSURF
read(5,*) PEP
read(5,*) VPEP
read(5,*) RESNUM
100 format(i1)
110 format(i4)
120 format(i6)
140 format(f6.2)
    open(unit=88,file='phipsi1',status='unknown')
    open(unit=89,file='phipsi2',status='unknown')
    open(unit=7,file=SURFINP,status='unknown',form='formatted')
read(7,*) NSURF
if(NSURF.GT.600) STOP 'Matrix size too small'
do 105 N=1,NSURF
    read(7,222) (RSURF(N,I),I=1,3)
105 continue
222 format(f17.13,2x,f17.13,2x,f17.13)
    read(7,*) SURFNAME
    read(7,*) LX
    read(7,*) LY
    open(unit=9,file=OUTPUT,status='unknown',form='formatted')
C *****
C ** BOX DIMENSIONS & CONSTANTS FOR PCFs.    **
C *****
    LZ=38.00D+0
    NMOL=NMOLW
    DENS=NMOL/(LX*LY*LZ - VPEP)
    write(*,*)LZ,NMOL,NMOLW,DENS
    NPTS=1
    if(NMCS.gt.100) then
        NPTS=NMCS/100
    end if
    MAXTRNW=0.20
    MAXROTW=0.30
    MXTPEP=0.005
    MXRPEP=0.008
    maxpsi=0.002
    maxphi=0.004
    QFACT=1.671d+5/TEMP
    NMCSTMP=0
    ACCMOVEWTMP=0

```

```

ESUMACCTMP=0.d+0
VWSUMACCTMP=0.d+0
NMCSACC=0
ACCMOVEWACC=0
ESUMACC=0.d+0
VWSUMACC=0.d+0
ESUM=0.d+0
VWSUM=0.d+0
UPEPWI=0.0D+0
LMIN=LX
if(LY.LT.LMIN) then
  LMIN=LY
end if
if(LZ.LT.LMIN) then
  LMIN=LZ
end if
RXMAX=LMIN/2.0d+0
RXMAX2=RXMAX*RXMAX
DELR=0.05d+0
DELZ=DELR
NBIN=idint(RXMAX/DELR-0.5d+0)
NBINZ=idint(LZ/DELZ)
ACCMOVEW=0
ACCPEP=0
ACCTOR=0
open(unit=2,file=CFGOUT,status='unknown',form='formatted')
C *****
C ** WATER MODEL PARAMETERS.          **
C ** current model = SPC.              **
C *****
LOHW=1.0d+0
THHOH=109.47d+0*PI/180.0d+0
AIIW(1)=5.110d+0
CIIW(1)=8.248d+0
AIIW(2)=0.d+0
CIIW(2)=0.d+0
AIIW(3)=AIIW(2)
CIIW(3)=CIIW(2)
QW(1)=-0.82d+0
QW(2)=+0.41d+0
QW(3)=QW(2)

```



```

*****
** Surface Parameters *****
*****

  AII(1)=6.344d+0
  CII(1)=10.152d+0
  AII(2)=AII(1)
  CII(2)=CII(1)
  Q(1)=QSURF
  Q(2)=-QSURF
*****

do 1100 I=1,3
  do 1110 J=1,3
    A12W(I,J)=(AIIW(I)*AIIW(J))**6/TEMP
    C6W(I,J)=(CIIW(I)*CIIW(J))**3/TEMP
    QCONW(I,J)=QFACT*QW(I)*QW(J)
  1110 continue
1100 continue

do 1581 I=1,3
  do 1481 J=1,2
    A12WS(I,J)=(AIIW(I)*AII(J))**6/TEMP
    C6WS(I,J)=(CIIW(I)*CII(J))**3/TEMP
    QCONWS(I,J)=QFACT*QW(I)*Q(J)
  1481 continue
1581 continue

C *****
C ** INITIAL CONFIGURATION AND ENERGY.          **
C ** RUNSTAT=0...cold start, call INCONFIG.      **
C ** RUNSTAT=1...warm start, before equilibrium, read in coords.**
C ** RUNSTAT=2...1st data collecting run        **
C ** RUNSTAT=3...collecting data (after 1st collecting run) **
C *****

  if(RUNSTAT.eq.0)then
    call INCONFIG(NMOL,NMOLW,LX,LY,LZ,LOHW,
& THHOH,RW,NATOM,PEPCORD1,PEPCORD2,APEP,
& CPEP,QPEP,AW,AA,ATOM,GROUP,NHB,HB,NEND,CEND,phi1,phi2,psi1,
& psi2,v1,v2,v3,v4,v5,v6,PEP,RESNUM)
    do 1032 I=1,3
      COM1(I)=0.d+0
      COM2(I)=0.d+0
    1032 continue

```

```

do 1006 K=1,NBIN
  do 1016 I=1,3
    GSUM(K,I)=0
1016  continue
  do 1026 I=1,2
    do 1014 N=1,NHB
      GRPWSUM1(K,I,N)=0
      GRPWSUM2(K,I,N)=0
      NEIBOURWSUM1(N)=0
      NEIBOURWSUM2(N)=0
1014  continue
1026  continue
1006  continue

do 8027 N=1,NBINZ
  do I=1,2
    ZWSUM(N,I)=0
  ENDDO
  do 3026 I=1,3
    ZPEP1SUM(N,I)=0
    ZPEP2SUM(N,I)=0
3026  continue
8027  continue
else
  open(unit=1,file=CFGINP,status='unknown',form='formatted')
  read(1,*) NATOM
  do 35500 M=1,NATOM
    read(1,889) AA(M),ATOM(M),GROUP(M),PEPCORD1(M,1),
    & PEPCORD1(M,2),PEPCORD1(M,3),PEPCORD2(M,1),PEPCORD2(M,2),
    & PEPCORD2(M,3),APEP(M),CPEP(M),QPEP(M),AW(M)
35500  continue
889  format(5x,I3,1x,I3,1x,I4,1x,f8.3,f8.3,f8.3,f8.3,f8.3,f8.3,
1    f8.3,f8.3,f8.3,f8.3)
    read(1,*) NHB
    do 35020 M=1,NHB
      read(1,*) HB(M)
35020  continue
    do 1208 I=1,3
      read(1,778)COM1(I),COM2(I)
1208  continue
    read(1,*) NEND
    read(1,*) CEND

```

```

do 1200 M=1,NMOLW
  read(1,*)
do 1210 I=1,3
  read(1,777) (RW(M,C,I),C=1,3)
1210  continue
1200  continue
  do I=1,RESNUM
  read(1,1523)phi1(I),phi2(I),psi1(I),psi2(I)
  enddo
  read(1,1524)v1,v2,v3,v4,v5,v6
1523  format(5x,f8.3,5x,f8.3,5x,f8.3,5x,f8.3)
1524  format(f8.3,f8.3,f8.3,f8.3,f8.3,f8.3)
  read(1,*) MAXTRNW
  read(1,*) MAXROTW
  read(1,*) MXTPEP
  read(1,*) MXRPEP
  read(1,*) maxpsi
  read(1,*) maxphi
  if(RUNSTAT.ne.3)then
  do 1005 K=1,NBIN
  do 1015 I=1,3
    GSUM(K,I)=0
1015  continue
  do 1027 I=1,2
  do 8028 N=1,NHB
    GRPWSUM1(K,I,N)=0
    GRPWSUM2(K,I,N)=0
    NEIBOURWSUM1(N)=0
    NEIBOURWSUM1(N)=0
8028  continue
1027  continue
1005  continue
  COSSUM1=0.d+0
  COSSQSUM1=0.d+0
  COSSUM2=0.d+0
  COSSQSUM2=0.d+0
  do 1001 K=1,NBINZ
  DO I=1,2
    ZWSUM(K,I)=0
  ENDDO
  do 1011 I=1,3
    ZPEP1SUM(K,I)=0

```

```

        ZPEP2SUM(K,I)=0
1011    continue
1001    continue
    else
        read(1,*) NMCSACC
        read(1,*) ACCMOVEWACC
        read(1,*) ESUMACC
        read(1,*) VWSUMACC
        read(1,*) ACCPEPACC
        read(1,*) ACCTORACC
        read(1,*)
        do 1003 K=1,NBIN
            read(1,333) (GSUM(K,I),I=1,3)
1003    continue
        read(1,*)
        do 1028 K=1,NBIN
            do 1029 N=1,NHB
                read(1,334) (GRPWSUM1(K,I,N),I=1,2)
1029    continue
1028    continue
            read(1,*)
            do 10266 K=1,NBIN
                do 1030 N=1,NHB
                    read(1,334) (GRPWSUM2(K,I,N),I=1,2)
1030    continue
10266    continue
                read(1,*)
                do 1031 N=1,NHB
                    read(1,*) NEIBOURWSUM1(N)
1031    continue
                read(1,*)
                do 1033 N=1,NHB
                    read(1,*) NEIBOURWSUM2(N)
1033    continue
                read(1,*)
                do K=1,NBINZ
                    read(1,338) (ZWSUM(K,I),I=1,2)
                enddo
                read(1,*)
                do 1004 K=1,NBINZ
                    do 2478 I=1,3
                        read(1,*) ZPEP1SUM(K,I)

```

```

2478     continue
1004     continue
        do 1007 K=1,NBINZ
            do 2480 I=1,3
                read(1,*) ZPEP2SUM(K,I)
2480     continue
1007     continue
        read(1,*)
        read(1,339) COSSUM1,COSSUM2
        read(1,339) COSSQSUM1,COSSQSUM2
        end if
        close(1)
    endif
333  format(i12,2x,i12,2x,i12,2x,i12,2x,i12,
    &   2x,i12,2x,i12,2x,i12,2x,i12,2x,i12)
334  format(i12,2x,i12,2x,i12)
338  format(i12,2x,i12,2x,i12)
339  format(f8.3,2x,f8.3)

C *****
C ** PEPTIDE-WATER MODEL PARAMETERS.          **
C *****

    do 30107 M=1,NATOM
        do 30108 I=1,3
            A12WP(M,I)=(APEP(M)*AIIW(I)**6/TEMP
            C6WP(M,I)=(CPEP(M)*CIIW(I)**3/TEMP
            QCONWP(M,I)=QFACT*QPEP(M)*QW(I)
30108  continue
30107 continue

    do 30172 M=1,NATOM
        do 30182 I=1,2
            A12PS(M,I)=(APEP(M)*AII(I)**6/TEMP
            C6PS(M,I)=(CPEP(M)*CII(I)**3/TEMP
            QCONPS(M,I)=QFACT*QPEP(M)*Q(I)
30182  continue
30172 continue

C *****
C ** PEPTIDE-PEPTIDE MODEL PARAMETERS.          **
C *****

```

```

do 30173 M=1,NATOM
  do 30183 N=1,NATOM
    A12PP(M,N)=(APEP(M)*APEP(N))**6/TEMP
    C6PP(M,N)=(CPEP(M)*CPEP(N))**3/TEMP
    QCONPP(M,N)=QFACT*QPEP(M)*QPEP(N)
  30183 continue
30173 continue

C**INITIAL ENERGY ESTIMATION*****
  call NRGTOT(V,NMOLW,LX,LY,LZ,RW,A12W,C6W,QCONW,
& A12WS,C6WS,QCONWS,A12WP,C6WP,QCONWP,A12PS,C6PS,QCONPS,
& A12PP,C6PP,QCONPP,NSURF,RSURF,NATOM,PEPCORD1,PEPCORD2,
& D2WPEP1OLD,D2WPEP2OLD,COM1,COM2,UWMPEP,UPEP1SFOLD,UPEP2SFOLD,
& VWINI,VSURSOLI,VRESTI,UPPOLD,UPEP1WOLD,UPEP2WOLD,D2PPOLD)

  call pepenrg(OLDP1ENRG,APEP,CPEP,QPEP,PEPCORD1,NATOM,
1 phi1,psi1,v1,v2,v3,v4,v5,v6,RESNUM)
  call pepenrg(OLDP2ENRG,APEP,CPEP,QPEP,PEPCORD2,NATOM,
1 phi2,psi2,v1,v2,v3,v4,v5,v6,RESNUM)
  write(*,*)V,OLDP1ENRG,OLDP2ENRG'
  write(*,*)V,OLDP1ENRG,OLDP2ENRG
  VTOT(0)=V+OLDP1ENRG+OLDP2ENRG
  V=VTOT(0)
  VWATER=VWINI
  VSUSS=VSURSOLI
  do M=1,NMOLW
    UPEPWI=UPEPWI+UWMPEP(M,1)+UWMPEP(M,2)
  enddo
  VSOPS=UPEPWI
  VPSS=UPEP1SFOLD+UPEP2SFOLD
  VWINI=VWINI/dbl(NMOLW)
  AVGACCW=0.5d+0
  AVGACCPEP=0.5d+0
  AVGACTOR=0.5d+0
  call PDBOUT(RSURF,RW,PEPCORD1,PEPCORD2,'SYSTEM0.pdb',NATOM,NMOLW,
& NSURF,AA,ATOM,GROUP)
C *****
C ** MC STEPS. **
C ** each MC step consists of an attempted move for each and every molecule in the system. Every NADJUST steps,
maximum movements are adjusted for an acceptance rate of 0.50. Every NSAVE steps, configurations and results are
saved for recovery in case the job is killed. **

```

C \*\*\*\*\*

```

do 10000 STEP=1,NMCS
  if(mod(STEP,NADJUST).eq.0) then
    AVGACCW=dbl(ACCMOVEW)/dbl(NMOLW*(STEP-1))
    AVGACCPEP=dbl(ACCPEP)/dbl(2*(STEP-1))
    AVGACCTOR=dbl(ACCTOR)/dbl(2*(STEP-1))
    if (AVGACCW.gt.0.50) then
      MAXTRNW = MAXTRNW * 1.03d+0
      MAXROTW = MAXROTW * 1.03d+0
    else
      MAXTRNW = MAXTRNW / 1.03d+0
      MAXROTW = MAXROTW / 1.03d+0
    end if
    if (AVGACCPEP.gt.0.50) then
      MXTPEP = MXTPEP * 1.1d+0
      MXRPEP = MXRPEP * 1.1d+0
    else
      MXTPEP = MXTPEP / 1.1d+0
      MXRPEP = MXRPEP / 1.1d+0
    end if
    if (AVGACCTOR.gt.0.50) then
      maxpsi = maxpsi * 1.1d+0
      maxphi = maxphi * 1.1d+0
    else
      maxpsi = maxpsi / 1.1d+0
      maxphi = maxphi / 1.1d+0
    end if
    if(mod(STEP,NSAVE).eq.0) then
      NMCSTMP=STEP-1
      ACCMOVEWTMP=ACCMOVEW
      ESUMACCTMP=ESUM
      VWSUMACCTMP=VWSUM
      ACCPEPTMP=ACCPEP
      ACCTORTMP=ACCTOR
      if(RUNSTAT.GE.2) then
        NMCSTMP=NMCSACC+STEP-1
        ACCMOVEWTMP=ACCMOVEWACC+ACCMOVEW
        ESUMACCTMP=ESUMACC+ESUM
        VWSUMACCTMP=VWSUMACC+VWSUM
        ACCPEPTMP=ACCPEPACC+ACCPEP
        ACCTORTMP=ACCTORACC+ACCTOR
      end if
    end if
  end if
end do

```

```

open(unit=3,file='CFG.TMP',status='unknown',form=
&      'formatted')
rewind(3)
do 34000 M=1,NATOM
  write(3,889)AA(M),ATOM(M),GROUP(M),PEPCORD1(M,1),
&      PEPCORD1(M,2),PEPCORD1(M,3),PEPCORD2(M,1),PEPCORD2(M,2),
&      PEPCORD2(M,3),APEP(M),CPEP(M),QPEP(M),AW(M)
34000  continue
write(3,*) NHB
do 35030 M=1,NHB
  write(3,*) HB(M)
35030  continue
do 35031 I=1,3
  write(3,778)COM1(I),COM2(I)
35031  continue
write(3,*) NEND
write(3,*) CEND
do 2105 M=1,NMOLW
  write(3,*)
  do 2115 I=1,3
    write(3,777) (RW(M,C,I),C=1,3)
2115  continue
2105  continue
do I=1,RESNUM
write(3,1522)phi1(I),phi2(I),psi1(I),psi2(I)
enddo
write(3,1525)v1,v2,v3,v4,v5,v6
1522  format(5x,f8.3,5x,f8.3,5x,f8.3,5x,f8.3)
1525  format(f8.3,f8.3,f8.3,f8.3,f8.3,f8.3)
write(3,*) MAXTRNW
write(3,*) MAXROTW
write(3,*) MXTPEP
write(3,*) MXRPEP
write(3,*) maxpsi
write(3,*) maxphi
write(3,*) NMCSTMP
write(3,*) ACCMOVEWTMP
write(3,*) ESUMACCTMP
write(3,*) VWSUMACCTMP
write(3,*) ACCPEPTMP
write(3,*) ACCTORTMP
write(3,*) 'GSUM'

```



```

do 1108 K=1,NBIN
  write(3,333) (GSUM(K,I),I=1,3)
1108  continue
write(3,*) 'GRPSUM-Peptide 1'
do 41000 K=1,NBIN
  do 41100 N=1,NHB
    write(3,334) (GRPWSUM1(K,I,N),I=1,2)
41100  continue
41000  continue
write(3,*) 'GRPSUM-Peptide 2'
do 41001 K=1,NBIN
  do 41101 N=1,NHB
    write(3,334) (GRPWSUM2(K,I,N),I=1,2)
41101  continue
41001  continue
write(3,*) 'NEIBOURSUM-Peptide 1'
do 41200 N=1,NHB
  write(3,*) NEIBOURWSUM1(N)
41200  continue
write(3,*) 'NEIBOURSUM-Peptide 2'
do 41201 N=1,NHB
  write(3,*) NEIBOURWSUM2(N)
41201  continue
write(3,*) 'ZWSUM'
do K=1,NBINZ
  write(3,338) (ZWSUM(K,I),I=1,2)
enddo
write(3,*) 'ZPEPSUM-Peptide 1'
do 1107 K=1,NBINZ
  do 41800 I=1,3
    write(3,*) ZPEP1SUM(K,I)
41800  continue
1107  continue
write(3,*) 'ZPEPSUM-Peptide 1'
do 1109 K=1,NBINZ
  do 41802 I=1,3
    write(3,*) ZPEP2SUM(K,I)
41802  continue
1109  continue
write(3,339) COSSUM1,COSSUM2
write(3,339) COSSQSUM1,COSSQSUM2
write(3,*) 'STEP= ',STEP

```

```

        close(3)
    end if
end if
UPEP1WOLD=0.0D+0
UPEP2WOLD=0.0D+0
do 50010 M=1,NMOLW
    UPEP1WOLD=UPEP1WOLD+UWMPEP(M,1)
    UPEP2WOLD=UPEP2WOLD+UWMPEP(M,2)
    UWMPEPOLD(M,1)=UWMPEP(M,1)
    UWMPEPOLD(M,2)=UWMPEP(M,2)
50010 continue
do 50000 M=1,NATOM
    do 51000 I=1,3
        PEPCORD1OLD(M,I)=PEPCORD1(M,I)
        PEPCORD2OLD(M,I)=PEPCORD2(M,I)
51000 continue
50000 continue
do I=1,RESNUM
    phi1old(i)=phi1(i)
    phi2old(i)=phi2(i)
    psi1old(i)=psi1(i)
    psi2old(i)=psi2(i)
enddo
do 54000 I=1,3
    COM1OLD(I)=COM1(I)
    COM2OLD(I)=COM2(I)
54000 continue
***** peptide movement *****
**peptide-1**

    call MOVEPEP(NATOM,PEPCORD1,COM1,MXTPEP,MXRPEP,
&         LX,LY,LZ,ISEED)
    call DISTPEPSF(D2PEP1SF1NEW,D2PEP1SF2NEW,RSURF,NSURF,
&         PEPCORD1,NATOM,LX,LY,COM1)
    call NRGPEPSF(A12PS,C6PS,QCONPS,UPEP1SFNEW,D2PEP1SF1NEW,
&         D2PEP1SF2NEW,NSURF,NATOM)
    call DISTPP(D2PPNEW,PEPCORD1,PEPCORD2,NATOM,LX,LY,COM1,COM2)
    call NRGPP(D2PPNEW,A12PP,C6PP,QCONPP,UPPNEW,NATOM)
    call pepenrg(NEWP1ENRG,APEP,CPEP,QPEP,PEPCORD1,NATOM,
&         phi1,psi1,v1,v2,v3,v4,v5,v6,RESNUM)

    UPEP1WNEW=0.0D+0

```

```

do 51100 M=1,NMOLW
  do 51200 I=1,3
    do 51300 C=1,3
      RWM(C,I)=RW(M,C,I)
51300   continue
51200   continue
    call DISTWMPEP(D2WMPEP1NEW,PEPCORD1,NATOM,RWM,LX,LY,COM1)
    call NRGWMPEP(D2WMPEP1NEW,A12WP,C6WP,QCONWP,UPEP1WM,NATOM)
    do 51210 N=1,NATOM
      do 51220 I=1,3
        D2WPEP1NEW(N,M,I)=D2WMPEP1NEW(N,I)
51220   continue
51210   continue
        UWMPEP(M,1)=UPEP1WM
        UPEP1WNEW=UPEP1WNEW+UPEP1WM
51100   continue
        DELV=UPEP1WNEW+UPEP1SFNEW+NEWP1ENRG+UPPNEW
        &   -UPEP1WOLD-UPEP1SFOLD-OLDP1ENRG-UPPOLD

        if((DELV.ge.0.0d+0).and.(dexp(-DELV).lt.RANDOM(ISEED)))then
          do 52000 N=1,NATOM
            do 53000 I=1,3
              PEPCORD1(N,I)=PEPCORD1OLD(N,I)
            do 53100 M=1,NMOLW
              D2WPEP1NEW(N,M,I)=D2WPEP1OLD(N,M,I)
53100   continue
53000   continue
52000   continue
            do 54200 N=1,NMOLW
              UWMPEP(N,1)=UWMPEPOLD(N,1)
54200   continue
            do 54100 I=1,3
              COM1(I)=COM1OLD(I)
54100   continue
            call SUMBINPEP(PEPCORD1,LZ,NMOLW,NATOM,
1   RXMAX2,DELZ,DELZ,D2WPEP1NEW,COSSUM1,COSSQSUM1,
1   ZPEP1SUM,NEND,CEND,GRPWSUM1,NEIBOURWSUM1,
1   HB,NHB,COM1)
            else
              V=V+DELV
              VPSS=VPSS+UPEP1SFNEW-UPEP1SFOLD
              VSOPS=VSOPS+UPEP1WNEW-UPEP1WOLD

```

```

ACCPEP=ACCPEP+1
OLDP1ENRG=NEWP1ENRG
do 54110 N=1,NATOM
  do 54120 I=1,NATOM
    D2PPOLD(I,N)=D2PPNEW(I,N)
54120  continue
54110  continue
      call SUMBINPEP(PEPCORD1,LZ,NMOLW,NATOM,
1  RXMAX2,DELR,DELZ,D2WPEP1NEW,COSSUM1,COSSQSUM1,
1  ZPEP1SUM,NEND,CEND,GRPWSUM1,NEIBOURWSUM1,
1  HB,NHB,COM1)
      UPEP1SFOLD=UPEP1SFNEW
      UPPOLD=UPPNEW
      end if
**peptide-2**
      call MOVEPEP(NATOM,PEPCORD2,COM2,MXTPEP,MXRPEP,
&      LX,LY,LZ,ISEED)
      call DISTPEPSF(D2PEP2SF1NEW,D2PEP2SF2NEW,RSURF,NSURF,
&      PEPCORD2,NATOM,LX,LY,COM2)
      call NRGPEPSF(A12PS,C6PS,QCONPS,UPEP2SFNEW,D2PEP2SF1NEW,
&      D2PEP2SF2NEW,NSURF,NATOM)
      call DISTPP(D2PPNEW,PEPCORD1,PEPCORD2,NATOM,LX,LY,COM1,COM2)
      call NRGPP(D2PPNEW,A12PP,C6PP,QCONPP,UPPNEW,NATOM)
      call pepenrg(NEWP2ENRG,APEP,CPEP,QPEP,PEPCORD2,NATOM,
&      phi2,psi2,v1,v2,v3,v4,v5,v6,RESNUM)

      UPEP2WNEW=0.0D+0
      do 54111 M=1,NMOLW
        do 54114 I=1,3
          do 54112 C=1,3
            RWM(C,I)=RW(M,C,I)
54112    continue
54114    continue
          call DISTWMPEP(D2WMPEP2NEW,PEPCORD2,NATOM,RWM,LX,LY,COM2)
          call NRGWMPEP(D2WMPEP2NEW,A12WP,C6WP,QCONWP,UPEP2WM,NATOM)
          do 54130 N=1,NATOM
            do 54132 I=1,3
              D2WPEP2NEW(N,M,I)=D2WMPEP2NEW(N,I)
54132    continue
54130    continue
            UWMPEP(M,2)=UPEP2WM
            UPEP2WNEW=UPEP2WNEW+UPEP2WM

```

```

54111 continue
      DELV=UPEP2WNEW+UPEP2SFNEW+NEWP2ENRG+UPPNEW
&      -UPEP2WOLD-UPEP2SFOLD-OLDP2ENRG-UPPOLD
      if((DELV.ge.0.0d+0).and.(dexp(-DELV).lt.RANDOM(ISEED)))then
      do 54140 N=1,NATOM
        do 54142 I=1,3
          PEPCORD2(N,I)=PEPCORD2OLD(N,I)
        do 54144 M=1,NMOLW
          D2WPEP2NEW(N,M,I)=D2WPEP2OLD(N,M,I)
54144 continue
54142 continue
54140 continue
      do 54150 N=1,NMOLW
        UWMPEP(N,2)=UWMPEPOLD(N,2)
54150 continue
      do 54152 I=1,3
        COM2(I)=COM2OLD(I)
54152 continue
      call SUMBINPEP(PEPCORD2,LZ,NMOLW,NATOM,
1 RXMAX2,DELR,DELZ,D2WPEP2NEW,COSSUM2,COSSQSUM2,
1 ZPEP2SUM,NEND,CEND,GRPWSUM2,NEIBOURWSUM2,
1 HB,NHB,COM2)
      else
      V=V+DELV
      VPSS=VPSS+UPEP2SFNEW-UPEP2SFOLD
      VSOPS=VSOPS+UPEP2WNEW-UPEP2WOLD
      ACCPEP=ACCPEP+1
      OLDP2ENRG=NEWP2ENRG
      do 54154 N=1,NATOM
        do 54156 I=1,NATOM
          D2PPOLD(I,N)=D2PPNEW(I,N)
54156 continue
54154 continue
      call SUMBINPEP(PEPCORD2,LZ,NMOLW,NATOM,
1 RXMAX2,DELR,DELZ,D2WPEP2NEW,COSSUM2,COSSQSUM2,
1 ZPEP2SUM,NEND,CEND,GRPWSUM2,NEIBOURWSUM2,
1 HB,NHB,COM2)
      UPEP2SFOLD=UPEP2SFNEW
      UPPOLD=UPPNEW
      end if

```

```

*****TORSION MOVEMENT*****

```

```

do I=1,3
  COM1OLD(I)=COM1(I)
  COM2OLD(I)=COM2(I)
enddo
do M=1,NATOM
  do I=1,3
    PEPCORD1OLD(M,I)=PEPCORD1(M,I)
    PEPCORD2OLD(M,I)=PEPCORD2(M,I)
  enddo
enddo
do N=1,NATOM
  do I=1,3
  do M=1,NMOLW
    D2WPEP1OLD(N,M,I)=D2WPEP1NEW(N,M,I)
    D2WPEP2OLD(N,M,I)=D2WPEP2NEW(N,M,I)
  enddo
  enddo
  enddo
  UPEP1WOLD=0.0D+0
  UPEP2WOLD=0.0D+0
  do 50017 M=1,NMOLW
    UPEP1WOLD=UPEP1WOLD+UWMPEP(M,1)
    UPEP2WOLD=UPEP2WOLD+UWMPEP(M,2)
    UWMPEPOLD(M,1)=UWMPEP(M,1)
    UWMPEPOLD(M,2)=UWMPEP(M,2)
50017  continue

**TORSIONAL MOVEMENT PEPTIDE-1**
  call TORSION(NATOM,PEPCORD1,maxpsi,maxphi,phi1,psi1,
&  COM1,AW,RESNUM,ISEED)
  call DISTPEPSF(D2PEP1SF1NEW,D2PEP1SF2NEW,RSURF,NSURF,
1      PEPCORD1,NATOM,LX,LY,COM1)
  call NRGPEPSF(A12PS,C6PS,QCONPS,UPEP1SFNEW,D2PEP1SF1NEW,
1      D2PEP1SF2NEW,NSURF,NATOM)
  call DISTPP(D2PPNEW,PEPCORD1,PEPCORD2,NATOM,LX,LY,COM1,COM2)
  call NRGPP(D2PPNEW,A12PP,C6PP,QCONPP,UPPNEW,NATOM)
  call pepenrg(NEWP1ENRG,APEP,CPEP,QPEP,PEPCORD1,NATOM,
1  phi1,psi1,v1,v2,v3,v4,v5,v6,RESNUM)

  UPEP1WNEW=0.0d+0
  do 51103 M=1,NMOLW
    do 51203 I=1,3

```

```

do 51303 C=1,3
  RWM(C,I)=RW(M,C,I)
51303  continue
51203  continue
  call DISTWMPEP(D2WMPEP1NEW,PEPCORD1,NATOM,RWM,LX,LY,COM1)
  call NRGWMPEP(D2WMPEP1NEW,A12WP,C6WP,QCONWP,UPEP1WM,NATOM)
  do 51213 N=1,NATOM
    do 51223 I=1,3
      D2WPEP1NEW(N,M,I)=D2WMPEP1NEW(N,I)
51223  continue
51213  continue
      UWMPEP(M,1)=UPEP1WM
      UPEP1WNEW=UPEP1WNEW+UPEP1WM
51103  continue
      DELV=UPEP1WNEW+UPEP1SFNEW+NEWP1ENRG+UPPNEW
      &  -UPEP1WOLD-UPEP1SFOLD-OLDP1ENRG-UPPOLD
      if((DELV.ge.0.0d+0).and.(dexp(-DELV).lt.RANDOM(ISEED)))then
        do 52111 N=1,NATOM
          do 53111 I=1,3
            PEPCORD1(N,I)=PEPCORD1OLD(N,I)
          do 53112 M=1,NMOLW
            D2WPEP1NEW(N,M,I)=D2WPEP1OLD(N,M,I)
53112  continue
53111  continue
52111  continue
          do I=1,3
            COM1(I)=COM1OLD(I)
          enddo
          do i=1,RESNUM
            phi1(i)=phi1old(i)
            psi1(i)=psi1old(i)
          enddo
          do 54211 N=1,NMOLW
            UWMPEP(N,1)=UWMPEPOLD(N,1)
54211  continue
            call SUMBINPEP(PEPCORD1,LZ,NMOLW,NATOM,RXMAX2,
1  DELR,DELZ,D2WPEP1NEW,COSSUM1,COSSQSUM1,
1  ZPEP1SUM,NEND,CEND,GRPWSUM1,NEIBOURWSUM1,
1  HB,NHB,COM1)
            else
              do i=1,RESNUM
                write(88,*)phi1(i),psi1(i)

```

```

        enddo
        V=V+DELV
        VPSS=VPSS+UPEP1SFNEW-UPEP1SFOLD
        VSOPS=VSOPS+UPEP1WNEW-UPEP1WOLD
        ACCTOR=ACCTOR+1
        call SUMBINPEP(PEPCORD1,LZ,NMOLW,NATOM,RXMAX2,
1  DELR,DELZ,D2WPEP1NEW,COSSUM1,COSSQSUM1,
1  ZPEP1SUM,NEND,CEND,GRPWSUM1,NEIBOURWSUM1,
1  HB,NHB,COM1)
        UPEP1SFOLD=UPEP1SFNEW
        UPPOLD=UPPNEW
                OLDP1ENRG=NEWP1ENRG
        end if
**TORSIONAL MOVEMENT PEPTIDE-2**
        call TORSION(NATOM,PEPCORD2,maxpsi,maxphi,phi2,psi2,
&  COM2,AW,RESNUM,ISEED)
        call DISTPEPSF(D2PEP2SF1NEW,D2PEP2SF2NEW,RSURF,NSURF,
1  PEPCORD2,NATOM,LX,LY,COM2)
        call NRGPEPSF(A12PS,C6PS,QCONPS,UPEP2SFNEW,D2PEP2SF1NEW,
1  D2PEP2SF2NEW,NSURF,NATOM)
        call DISTPP(D2PPNEW,PEPCORD1,PEPCORD2,NATOM,LX,LY,COM1,COM2)
        call NRGPP(D2PPNEW,A12PP,C6PP,QCONPP,UPPNEW,NATOM)
        call pepenrg(NEWP2ENRG,APEP,CPEP,QPEP,PEPCORD2,NATOM,
1  phi2,psi2,v1,v2,v3,v4,v5,v6,RESNUM)

        UPEP2WNEW=0.0d+0
        do 51105 M=1,NMOLW
            do 51205 I=1,3
                do 51305 C=1,3
                    RWM(C,I)=RW(M,C,I)
51305    continue
51205    continue
                call DISTWMPEP(D2WMPEP2NEW,PEPCORD2,NATOM,RWM,LX,LY,COM2)
                call NRGWMPEP(D2WMPEP2NEW,A12WP,C6WP,QCONWP,UPEP2WM,NATOM)
                do 51215 N=1,NATOM
                    do 51225 I=1,3
                        D2WPEP2NEW(N,M,I)=D2WMPEP2NEW(N,I)
51225    continue
51215    continue
                    UWMPEP(M,2)=UPEP2WM
                    UPEP2WNEW=UPEP2WNEW+UPEP2WM
51105    continue

```



```

DELV=UPEP2WNEW+UPEP2SFNEW+NEWP2ENRG+UPPNEW
&   -UPEP2WOLD-UPEP2SFOLD-OLDP2ENRG-UPPOLD
if((DELV.ge.0.0d+0).and.(dexp(-DELV).lt.RANDOM(ISEED)))then
do 52113 N=1,NATOM
  do 53113 I=1,3
    PEPCORD2(N,I)=PEPCORD2OLD(N,I)
  do 53115 M=1,NMOLW
    D2WPEP2NEW(N,M,I)=D2WPEP2OLD(N,M,I)
53115   continue
53113   continue
52113   continue
    do I=1,3
      COM2(I)=COM2OLD(I)
    enddo
    do i=1,RESNUM
      phi2(i)=phi2old(i)
      psi2(i)=psi2old(i)
    enddo
    do 54213 N=1,NMOLW
      UWMPEP(N,2)=UWMPEPOLD(N,2)
54213   continue
      call SUMBINPEP(PEPCORD2,LZ,NMOLW,NATOM,RXMAX2,
1  DELR,DELZ,D2WPEP2NEW,COSSUM2,COSSQSUM2,
1  ZPEP2SUM,NEND,CEND,GRPWSUM2,NEIBOURWSUM2,
1  HB,NHB,COM2)
      else
      do i=1,RESNUM
        write(89,*)phi2(i),psi2(i)
      enddo
      V=V+DELV
      VPSS=VPSS+UPEP2SFNEW-UPEP2SFOLD
      VSOPS=VSOPS+UPEP2WNEW-UPEP2WOLD
      ACCTOR=ACCTOR+1
      call SUMBINPEP(PEPCORD2,LZ,NMOLW,NATOM,RXMAX2,
1  DELR,DELZ,D2WPEP2NEW,COSSUM2,COSSQSUM2,
1  ZPEP2SUM,NEND,CEND,GRPWSUM2,NEIBOURWSUM2,
1  HB,NHB,COM2)
      UPEP2SFOLD=UPEP2SFNEW
      UPPOLD=UPPNEW
      OLDP2ENRG=NEWP2ENRG
    end if
***** Water Movement *****

```

```

do 10100 M=1,NMOLW
  do 11000 I=1,3
    do 11100 C=1,3
      RWMOLD(C,I)=RW(M,C,I)
11100  continue
11000  continue
      call DISTW(D2WOLD,RWMOLD,RW,M,NMOLW,LX,LY,LZ)
      call DISTWMSURF(D2WSURF1OLD,D2WSURF2OLD,RSURF,RWMOLD,
1  NSURF,LX,LY)
      UPEP1WMOLD=UWMPEP(M,1)
      UPEP2WMOLD=UWMPEP(M,2)
      do 55400 N=1,NATOM
        do 55410 I=1,3
          D2WMPEP1OLD(N,I)=D2WPEP1NEW(N,M,I)
          D2WMPEP2OLD(N,I)=D2WPEP2NEW(N,M,I)
55410  continue
55400  continue
          call NRGW(RWMOLD,M,NMOLW,D2WOLD,A12W,C6W,QCONW,
1  A12WS,C6WS,QCONWS,VWOLD,VWMSURFOLD,D2WSURF1OLD,
1  D2WSURF2OLD,NSURF,LX,LY,LZ)
          call RNDMOVE(RWMOLD,RWMNEW,MAXTRNW,MAXROTW,LX,LY,LZ,ISEED)
          do 11200 I=1,3
            do 11300 C=1,3
              RW(M,C,I)=RWMNEW(C,I)
11300  continue
11200  continue
              call DISTW(D2WNEW,RWMNEW,RW,M,NMOLW,LX,LY,LZ)
              call DISTWMSURF(D2WSURF1NEW,D2WSURF2NEW,RSURF,RWMNEW,
1  NSURF,LX,LY)
              call DISTWMPEP(D2WMPEP1NEW,PEPCORD1,NATOM,RWMNEW,LX,LY,COM1)
              call DISTWMPEP(D2WMPEP2NEW,PEPCORD2,NATOM,RWMNEW,LX,LY,COM2)
              call NRGWMPEP(D2WMPEP1NEW,A12WP,C6WP,QCONWP,UPEP1WMNEW,NATOM)
              call NRGWMPEP(D2WMPEP2NEW,A12WP,C6WP,QCONWP,UPEP2WMNEW,NATOM)
              call NRGW(RWMNEW,M,NMOLW,D2WNEW,A12W,C6W,QCONW,
1  A12WS,C6WS,QCONWS,VWNEW,VWMSURFNEW,D2WSURF1NEW,
1  D2WSURF2NEW,NSURF,LX,LY,LZ)
              DELV=VWNEW+VWMSURFNEW+UPEP1WMNEW+UPEP2WMNEW
& -VWOLD-VWMSURFOLD-UPEP1WMOLD-UPEP2WMOLD
              if((DELV.ge.0.0d+0).and.(dexp(-DELV).lt.RANDOM(ISEED)))then
                do 11400 I=1,3
                  do 11500 C=1,3
                    RW(M,C,I)=RWMOLD(C,I)

```

```

11500  continue
11400  continue
      do 53131 N=1,NATOM
          do 53141 I=1,3
              D2WPEP1NEW(N,M,I)=D2WPEP1OLD(N,M,I)
              D2WPEP2NEW(N,M,I)=D2WPEP2OLD(N,M,I)
53141  continue
53131  continue
      call SUMBINW(M,NMOLW,PEPCORD1,RXMAX2,GSUM,DELR,D2WOLD,
&  D2WMPEP1OLD,GRPWSUM1,NEIBOURWSUM1,HB,NHB,LX,LY,LZ,
&  RW,DELZ,ZWSUM)
      call SUMBINW(M,NMOLW,PEPCORD2,RXMAX2,GSUM,DELR,D2WOLD,
&  D2WMPEP2OLD,GRPWSUM2,NEIBOURWSUM2,HB,NHB,LX,LY,LZ,
&  RW,DELZ,ZWSUM)
      else
          do 53130 N=1,NATOM
              do 53140 I=1,3
                  D2WPEP1OLD(N,M,I)=D2WMPEP1NEW(N,I)
                  D2WPEP2OLD(N,M,I)=D2WMPEP2NEW(N,I)
53140  continue
53130  continue
              UWMPEP(M,1)=UPEP1WMNEW
              UWMPEP(M,2)=UPEP2WMNEW
              V=V+DELV
              VWATER=VWATER+VWNEW-VWOLD
              VSUSS=VSUSS+VWMSURFNEW-VWMSURFOLD
              VSOPS=VSOPS+UPEP1WMNEW+UPEP2WMNEW-UPEP1WMOLD-UPEP2WMOLD
              ACCMOVEW=ACCMOVEW+1
              call SUMBINW(M,NMOLW,PEPCORD1,RXMAX2,GSUM,DELR,D2WNEW,
&  D2WMPEP1NEW,GRPWSUM1,NEIBOURWSUM1,HB,NHB,LX,LY,LZ,
&  RW,DELZ,ZWSUM)
              call SUMBINW(M,NMOLW,PEPCORD2,RXMAX2,GSUM,DELR,D2WNEW,
&  D2WMPEP2NEW,GRPWSUM2,NEIBOURWSUM2,HB,NHB,LX,LY,LZ,
&  RW,DELZ,ZWSUM)
              end if
10100 continue
*****

VTOT(STEP)=V
VW(STEP)=VWATER/dbl(NMOLW)
VSUS(STEP)=VSUSS
VSOP(STEP)=VSOPS

```

```

VPS(STEP)=VPSS
ESUM=ESUM+VTOT(STEP)
VWSUM=VWSUM+VW(STEP)
ETED1(STEP)=
& ( (PEPCORD1(94,1)-PEPCORD1(1,1))**2 +
& (PEPCORD1(94,2)-PEPCORD1(1,2))**2 +
& (PEPCORD1(94,3)-PEPCORD1(1,3))**2 )**0.5
ETED2(STEP)=
& ( (PEPCORD2(94,1)-PEPCORD2(1,1))**2 +
& (PEPCORD2(94,2)-PEPCORD2(1,2))**2 +
& (PEPCORD2(94,3)-PEPCORD2(1,3))**2 )**0.5

CM1(STEP)=COM1(3)
CM2(STEP)=COM2(3)
10000 continue
  call NRGTOT(V,NMOLW,LX,LY,LZ,RW,A12W,C6W,QCONW,
& A12WS,C6WS,QCONWS,A12WP,C6WP,QCONWP,A12PS,C6PS,QCONPS,
& A12PP,C6PP,QCONPP,NSURF,RSURF,NATOM,PEPCORD1,PEPCORD2,
& D2WPEP1NEW,D2WPEP2NEW,COM1,COM2,UWMPEP,UPEP1SFNEW,UPEP2SFNEW,
& VWATER,VSURSOL,VREST,UPPNEW,UPEP1WNEW,UPEP2WNEW,D2PPNEW)
  call pepenrg(PE1,APEP,CPEP,QPEP,PEPCORD1,NATOM
1 ,phi1,psi1,v1,v2,v3,v4,v5,v6,RESNUM)
  call pepenrg(PE2,APEP,CPEP,QPEP,PEPCORD2,NATOM
1 ,phi2,psi2,v1,v2,v3,v4,v5,v6,RESNUM)
VTOT(NMCS+1)=V+PE1+PE2
VWATER=VWATER/dble(NMOLW)
NMCSOLD=NMCS
if(RUNSTAT.GE.2) then
  NMCSACC=NMCSACC+NMCS
  ACCMOVEWACC=ACCMOVEWACC+ACCMOVEW
  ACCPEPACC=ACCPEPACC+ACCPEP
  ACCTORACC=ACCTORACC+ACCTOR
  NMCS=NMCSACC
  ACCMOVEW=ACCMOVEWACC
end if
C *****
C ** AVERAGES. **
C ** acceptance ratio. energy. pair correlation functions. **
C *****
  AVGACCW=dble(ACCMOVEW)/dble(NMOLW*NMCS)
  AVGACCPEP=dble(ACCPEP)/dble(NMCS)
  AVGACCTOR=dble(ACCTOR)/dble(NMCS)

```

```

if(RUNSTAT.GE.2)then
  ESUMACC=ESUMACC+ESUM
  VWSUMACC=VWSUMACC+VWSUM
  ESUM=ESUMACC
  VWSUM=VWSUMACC
end if
AVGNRG=ESUM/dble(NMCS)
AVGVW=VWSUM/dble(NMCS)
do 13100 K=1,NBIN
  RMIN=DELR*(db1e(K)-0.5d+0)
  RMAX=RMIN+DELR
  VOLUME=4.0d+0/3.0d+0*PI*(RMAX*RMAX*RMAX-RMIN*RMIN*RMIN)
  VOLUME=VOLUME*DENS*db1e(NMCS)
  GOFR(K,1)=(db1e(GSUM(K,1)))/(VOLUME*db1e(NMOLW))
  GOFR(K,2)=(db1e(GSUM(K,2)))/(4.0d+0*VOLUME*db1e(NMOLW))
  GOFR(K,3)=(db1e(GSUM(K,3)))/(4.0D+0*VOLUME*db1e(NMOLW))
do 13110 N=1,NHB
  NEIBOURW1(N)=db1e(NEIBOURWSUM1(N))/db1e(NMCS*2)
  NEIBOURW2(N)=db1e(NEIBOURWSUM2(N))/db1e(NMCS*2)
  GOFRWP1(K,1,N)=(db1e(GRPWSUM1(K,1,N)))/(VOLUME*2.d+0)
  GOFRWP1(K,2,N)=(db1e(GRPWSUM1(K,2,N)))/(VOLUME*4.d+0)
  GOFRWP2(K,1,N)=(db1e(GRPWSUM2(K,1,N)))/(VOLUME*2.d+0)
  GOFRWP2(K,2,N)=(db1e(GRPWSUM2(K,2,N)))/(VOLUME*4.d+0)
13110 continue
13100 continue
  VOLUMEZ=LX*LY*DE LZ
  TMP2= VOLUMEZ*DENS*NMCS
do 83 K=1,NBINZ
  GOFZW(K,1)=ZWSUM(K,1)/TMP2
  GOFZW(K,2)=ZWSUM(K,2)/(TMP2*2.d+0)
83 continue
do 13101 K=1,NBINZ
  do 13102 I=1,3
    ZPEP1(K,I)=db1e(ZPEP1SUM(K,I))/db1e(NMCS)
    ZPEP2(K,I)=db1e(ZPEP2SUM(K,I))/db1e(NMCS)
13102 continue
13101 continue
  COSAVG1=COSSUM1/NMCS
  COSSQAVG1=(COSSQSUM1*3.d+0-dble(NMCS))/(2.d+0*NMCS)
  COSAVG2=COSSUM2/NMCS
  COSSQAVG2=(COSSQSUM2*3.d+0-dble(NMCS))/(2.d+0*NMCS)
C *****

```

```

C ** OUTPUT DATA. final configuration. acceptance ratios. energies. pair correlation functions.      **
C *****
write(9,*)
write(9,*) '*****'
write(9,*)
write(9,*) 'GENERAL OUTPUT DATA'
write(9,*)
write(9,*) TITLE
write(9,100) RUNSTAT
write(9,*) CFGINP
write(9,*) CFGOUT
write(9,*) OUTPUT
write(9,*) PDB
write(9,110) NMOLW
write(9,120) NMCS
write(9,140) TEMP
write(9,*) NSAVE
write(9,*) NADJUST
write(9,*) SURFINP
write(9,*) QSURF
write(9,*) PEP
write(9,*) VPEP
write(9,*) NSURF
write(9,*) SURFNAME
write(9,*) LX,LY,LZ
write(9,*)
write(9,*) 'AVERAGE WATER ACCEPTANCE RATIO'
write(9,*) AVGACCW
write(9,*) 'Maximum WATER Translational Movement'
write(9,*) MAXTRNW
write(9,*) 'Maximum WATER Rotational Movement'
write(9,*) MAXROTW
write(9,*) 'AVERAGE PEPTIDE MOVEMENT ACCEPTANCE RATIO'
write(9,*) AVGACCPEP
write(9,*) 'AVERAGE PEPTIDE TORSIONAL ACCEPTANCE RATIO'
write(9,*) AVGACCTOR
write(9,*) 'Maximum psi Movement of Peptide'
write(9,*) maxpsi
write(9,*) 'Maximum phi Movement of Peptide'
write(9,*) maxphi
write(9,*)
write(9,*) 'INITIAL ENERGY'

```

```

write(9,*) VTOT(0)
write(9,*) 'Initial Water Energy'
write(9,*) VWINI
write(9,*)
write(9,*) 'ENERGIES (system and water)'
do 3111 STEP=1,NMCSOLD
  if(mod(STEP,NPTS).eq.0)then
    N=STEP/NPTS
    VTOTPT(N)=VTOT(STEP)
    VWPT(N)=VW(STEP)
    ETEDP1(N)=ETED1(STEP)
    ETEDP2(N)=ETED2(STEP)
    CMP1(N)=CM1(STEP)
    CMP2(N)=CM2(STEP)
    VSUSPT(N)=VSUS(STEP)
    VSOPPT(N)=VSOP(STEP)
    VPSPT=VPS(STEP)
  end if
3111 continue
  VERRSUM=0.d+0
  VWERRSUM=0.d+0
  do 3000 N=1,100
    write(9,888) N*NPTS,VTOTPT(N),VWPT(N)
    VERRSUM=VERRSUM+dabs(VTOTPT(N)-AVGNRG)
    VWERRSUM=VWERRSUM+dabs(VWPT(N)-AVGVW)
  3000 continue
888  format('step=',I6,'<E>=',D18.10,'<VW>=',D18.10)
C   do 30005 N=1,100
C     write(9,8885) N*NPTS,VSUSPT(N),VSOPPT(N),VPSPT(N)
C30005 continue
C8885 format('step=',I6,'<VSUS>=',D18.10,
C &   '<VSOP>=',D18.10,'<VPS>=',D18.10)
  VERR=VERRSUM/100.0d+0
  VWERR=VWERRSUM/100.0d+0
  write(9,*)
  write(9,*) 'FINAL ENERGY'
  write(9,*) VTOT(NMCSOLD+1)
  write(9,*) 'Final Water Energy'
  write(9,*) VWATER
  write(9,*) 'AVERAGE ENERGY & UNCERTAINTY'
  write(9,*) AVGNRG,VERR
  write(9,*) 'AVERAGE WATER ENERGY & UNCERTAINTY'

```

```

write(9,*) AVGVW,VWERR
write(9,*)

do N=1,100
  ETED1T=ETED1T+ETEDP1(N)
  ETED2T=ETED2T+ETEDP2(N)
  CMP1T=CMP1T+CMP1(N)
  CMP2T=CMP2T+CMP2(N)
end do

ETED1AVG=ETED1T/100D+0
ETED2AVG=ETED2T/100D+0
CMP1AVG=CMP1T/100D+0
CMP2AVG=CMP2T/100D+0
C  write(9,890) N*NPTS,ETEDP1(N),ETEDP2(N),CMP1(N),CMP2(N)
C  end do
C890 format('STEP=',I6,<ETEDP1>=',D18.10,<ETEDP2>=',D18.10,
C  & '<CMP1>=',D18.10,<CMP2>=',D18.10)
  write(9,895) ETED1AVG,CMP1AVG,ETED2AVG,CMP2AVG
895 format(<ETEDP1AVG>=',D18.10,<CMP1AVG>=',D18.10,
& '<ETEDP2AVG>=',D18.10,<CMP2AVG>=',D18.10)
  write(9,*)
  write(9,*) 'COSAVG1= ',COSAVG1,'COSAVG2= ',COSAVG2
  write(9,*) 'COSSQAVG1= ',COSSQAVG1,'COSSQAVG2= ',COSSQAVG2
  write(9,*)

if(RUNSTAT.gt.1) then
  Write(9,*) 'Closest neighbour number peptide 1'
  do 55110 N=1,NHB
    write(9,*) 'NW(',HB(N),)= ',NEIBOURW1(N)
55110 continue
  Write(9,*) 'Closest neighbour number peptide 2'
  do 55112 N=1,NHB
    write(9,*) 'NW(',HB(N),)= ',NEIBOURW2(N)
55112 continue
  write(9,*) 'WATER DENSITY PROFILE'
  do 92 I=1,2
    write(9,*) 'ZW(',I,')'
    do 91 K=1,NBINZ
      write(9,999) GOFZW(K,I)
91  continue
92  continue
  write(9,*) 'PEPTIDE-1 Z position'

```



```

do 5001 I=1,3
  write(9,*) 'ZPEP(',I,')'
  do 5011 K=1,NBINZ
    write(9,999) ZPEP1(K,I)
5011  continue
5001  continue
  write(9,*) 'PEPTIDE-2 Z position'
  do 5003 I=1,3
    write(9,*) 'ZPEP(',I,')'
    do 5013 K=1,NBINZ
      write(9,999) ZPEP2(K,I)
5013  continue
5003  continue
  write(9,*) 'WATER-WATER PAIR CORRELATION FUNCTIONS'
  write(9,*) '(from r = DELR to RXMAX)'
  do 5000 I=1,3
    write(9,*) 'g(',I,')'
    do 5010 K=1,NBIN
      write(9,999) GOFR(K,I)
5010  continue
5000  continue
999  format(f8.4)
  write(9,*) 'PEPTIDE 1 - WATER PAIR CORRELATION FUNCTIONS'
  write(9,*) '(from r = DELR to RXMAX)'
  do 55000 N=1,NHB
    do 55100 I=1,2
      write(9,*) 'g(',HB(N),-',',I,')'
    do 55010 K=1,NBIN
      write(9,999) GOFRWP1(K,I,N)
55010  continue
55100  continue
55000  continue
  write(9,*) 'PEPTIDE 2 - WATER PAIR CORRELATION FUNCTIONS'
  write(9,*) '(from r = DELR to RXMAX)'
  do 55002 N=1,NHB
    do 55102 I=1,2
      write(9,*) 'g(',HB(N),-',',I,')'
    do 55012 K=1,NBIN
      write(9,999) GOFRWP2(K,I,N)
55012  continue
55102  continue
55002  continue

```

```

end if
close(9)
*****
write(2,*) NATOM
do 33000 M=1,NATOM
  write(2,889) AA(M),ATOM(M),GROUP(M),PEPCORD1(M,1),
1 PEPCORD1(M,2),PEPCORD1(M,3),PEPCORD2(M,1),PEPCORD2(M,2),
1 PEPCORD2(M,3),APEP(M),CPEP(M),QPEP(M),AW(M)
33000 continue
write(2,*) NHB
do 35040 M=1,NHB
  write(2,*) HB(M)
35040 continue
do 35041 I=1,3
  write(2,778)COM1(I),COM2(I)
35041 continue
write(2,*) NEND
write(2,*) CEND
do 2000 M=1,NMOLW
  write(2,*)
do 2010 I=1,3
  write(2,777) (RW(M,C,I),C=1,3)
2010 continue
2000 continue
do I=1,RESNUM
write(2,1528)phi1(I),phi2(I),psi1(I),psi2(I)
enddo
write(2,1529)v1,v2,v3,v4,v5,v6
1528 format(5x,f8.3,5x,f8.3,5x,f8.3,5x,f8.3)
1529 format(f8.3,f8.3,f8.3,f8.3,f8.3,f8.3)
write(2,*) MAXTRNW
write(2,*) MAXROTW
write(2,*) MXTPEP
write(2,*) MXRPEP
write(2,*) maxpsi
write(2,*) maxphi
if(RUNSTAT.gt.1) then
write(2,*) NMCSACC
write(2,*) ACCMOVEWACC
write(2,*) ESUMACC
write(2,*) VWSUMACC
write(2,*) ACCPEPACC

```

```
write(2,*) ACCTORACC
write(2,*) 'GSUM'
do 1008 K=1,NBIN
  write(2,333) (GSUM(K,I),I=1,3)
1008  continue
write(2,*) 'GRPSUM Peptide 1'
do 41500 K=1,NBIN
  do 41600 N=1,NHB
    write(2,334) (GRPWSUM1(K,I,N),I=1,2)
41600  continue
41500  continue
write(2,*) 'GRPSUM Peptide 2'
do 41502 K=1,NBIN
  do 41602 N=1,NHB
    write(2,334) (GRPWSUM2(K,I,N),I=1,2)
41602  continue
41502  continue
write(2,*) 'NEIBOURWSUM Peptide 1'
do 41700 N=1,NHB
  write(2,*) NEIBOURWSUM1(N)
41700  continue
write(2,*) 'NEIBOURWSUM Peptide 2'
do 41702 N=1,NHB
  write(2,*) NEIBOURWSUM2(N)
41702  continue
write(2,*) 'ZWSUM'
do K=1,NBINZ
  write(2,338) (ZWSUM(K,I),I=1,2)
enddo
write(2,*) 'ZPEPSUM Peptide 1'
do 10077 K=1,NBINZ
  do 41710 I=1,3
    write(2,*) ZPEP1SUM(K,I)
41710  continue
10077  continue
write(2,*) 'ZPEPSUM Peptide 2'
do 1009 K=1,NBINZ
  do 41712 I=1,3
    write(2,*) ZPEP2SUM(K,I)
41712  continue
1009  continue
write(2,*)
```

```

write(2,339) COSSUM1,COSSUM2
write(2,339) COSSQSUM1,COSSQSUM2
write(2,*) 'STEP= ',NMCSOLD
end if
close(2)

close(7)
close(88)
777 format(f17.13,2x,f17.13,2x,f17.13)
778 format(f17.13,2x,f17.13)
call PDBOUT(RSURF,RW,PEPCORD1,PEPCORD2,PDB,NATOM,NMOLW,
& NSURF,AA,ATOM,GROUP)
close(7)
stop
end
*****
** RNDMOVE.....subroutine which randomly displaces a molecule.    **
** the displacement is performed as a translational displacement of **
** the molecule coupled with a rotational displacement about the   **
** position of the oxygen atom.                                     **
*****
subroutine RNDMOVE(RMOLD,RMNEW,MAXTRN,MAXROT,LX,LY,LZ,ISEED)
implicit NONE
external MULTMAT,RANDOM
*****
integer I,C,ISEED
double precision LX,LY,LZ,DELC,MAXTRN,MAXROT,THX,THY,THZ
double precision COSX,SINX,COSY,SINY,COSZ,SINZ
double precision ROTX(3,3),ROTY(3,3),ROTZ(3,3),TMP1(3,3)
double precision TMP2(3,3)
double precision RMOLD(3,3),RMNEW(3,3),RHYOLD(3,2),RHYNEW(3,2)
double precision RANDOM
C *****
C ** HYDROGEN COORDINATE MATRIX. **
C ** the positions of the hydrogen atoms are **
C ** defined relative to the oxygen atom. **
C *****
do 1000 I=1,2
do 1010 C=1,3
RHYOLD(C,I)=RMOLD(C,I+1)-RMOLD(C,I)
1010 continue
1000 continue

```

```

C *****
C ** RANDOM ROTATION.          **
C *****

  THX=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MAXROT
  THY=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MAXROT
  THZ=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MAXROT
  COSX=dcos(THX)
  SINX=dsin(THX)
  COSY=dcos(THY)
  SINY=dsin(THY)
  COSZ=dcos(THZ)
  SINZ=dsin(THZ)
  ROTX(1,1)=1.0d+0
  ROTX(1,2)=0.0d+0
  ROTX(1,3)=0.0d+0
  ROTX(2,1)=0.0d+0
  ROTX(2,2)=COSX
  ROTX(2,3)=-SINX
  ROTX(3,1)=0.0d+0
  ROTX(3,2)=SINX
  ROTX(3,3)=COSX
  ROTY(1,1)=COSY
  ROTY(1,2)=0.0d+0
  ROTY(1,3)=-SINY
  ROTY(2,1)=0.0d+0
  ROTY(2,2)=1.0d+0
  ROTY(2,3)=0.0d+0
  ROTY(3,1)=SINY
  ROTY(3,2)=0.0d+0
  ROTY(3,3)=COSY
  ROTZ(1,1)=COSZ
  ROTZ(1,2)=-SINZ
  ROTZ(1,3)=0.0d+0
  ROTZ(2,1)=SINZ
  ROTZ(2,2)=COSZ
  ROTZ(2,3)=0.0d+0
  ROTZ(3,1)=0.0d+0
  ROTZ(3,2)=0.0d+0
  ROTZ(3,3)=1.0d+0
  call MULTMAT(ROTX,ROTY,3,3,3,TMP1)
  call MULTMAT(TMP1,ROTZ,3,3,3,TMP2)

```

```

      call MULTMAT(TMP2,RHYOLD,3,3,2,RHYNEW)
C *****
C ** RANDOM TRANSLATION.          **
C *****

      DELC=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MAXTRN
      RMNEW(1,1)=DELC+RMOLD(1,1)
      RMNEW(1,1)=RMNEW(1,1)-LX*dnint(RMNEW(1,1)/LX)
      RMNEW(1,2)=RMNEW(1,1)+RHYNEW(1,1)
      RMNEW(1,3)=RMNEW(1,1)+RHYNEW(1,2)
      DELC=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MAXTRN*LY/LX
      RMNEW(2,1)=DELC+RMOLD(2,1)
      RMNEW(2,1)=RMNEW(2,1)-LY*dnint(RMNEW(2,1)/LY)
      RMNEW(2,2)=RMNEW(2,1)+RHYNEW(2,1)
      RMNEW(2,3)=RMNEW(2,1)+RHYNEW(2,2)
      DELC=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MAXTRN*LZ/LX
      RMNEW(3,1)=DELC+RMOLD(3,1)
      RMNEW(3,2)=RMNEW(3,1)+RHYNEW(3,1)
      RMNEW(3,3)=RMNEW(3,1)+RHYNEW(3,2)

      return
      end

*****
** NRGTOT.....subroutine which calculates the total potential energy for any given configuration.  for programming
simplicity, the code is written to sum up the potential energies felt by each molecule and then divide this sum by a
factor of 2 since each pair interaction is counted twice.          **
*****

      subroutine NRGTOT(V,NMOLW,LX,LY,LZ,RW,A12W,C6W,
& QCONW,A12WS,C6WS,QCONWS,A12WP,C6WP,QCONWP,
& A12PS,C6PS,QCONPS,A12PP,C6PP,QCONPP,NSURF,RSURF,NATOM,PEPCORD1,
& PEPCORD2,D2WPEP1,D2WPEP2,COM1,COM2,VWMPEP,VPEP1SF,VPEP2SF,VWATER,
& VSURSOL,VREST,VPP,VWPEP1,VWPEP2,D2PP)

      implicit NONE

      external DISTW,NRGW
      external DISTWMSURF
      external DISTWMPEP,DISTPEPSF
      external NRGWMPEP

*****

      integer      M,I,C,NMOLW,NSURF,N,K,NATOM
      double precision V,VW,VREST,LX,LY,LZ,VWATER
      double precision D2W(3,2000,3),RW(2000,3,3),RWM(3,3)
      double precision A12W(3,3),C6W(3,3),A12PP,C6PP,QCONPP
      double precision QCONW(3,3)
      double precision A12WS(3,2),C6WS(3,2)

```

```

double precision QCONWS(3,2)
double precision RSURF(600,3),D2PP(200,200)
double precision D2WSURF1(3,600),D2WSURF2(3,600)
double precision VWMSURF,VWSURF
double precision A12WP(200,3),C6WP(200,3),QCONWP(200,3)
double precision A12PS(200,2),C6PS(200,2),QCONPS(200,2)
double precision PEPCORD1(200,3),PEPCORD2(200,3),COM1(3),COM2(3)
double precision D2WPEP1(200,2000,3),D2WPEP2(200,2000,3)
double precision VWMPEP(2000,2),VPEP1SF,VPEP2SF,VPP
double precision D2WMPEP1(200,3),D2WMPEP2(200,3)
double precision D2PEP1SF1(200,600),D2PEP2SF1(200,600)
double precision D2PEP1SF2(200,600),D2PEP2SF2(200,600)
double precision VPEP1WM,VPEP2WM,VWPEP1,VWPEP2,VSURSOL
C *****
C ** TOTAL ENERGY. the calculation includes all pair interactions between a given molecule and the closest images
of the remaining molecules. **
C *****
V=0.0d+0
VWATER=0.0d+0
VREST=0.0d+0
VWSURF=0.0D+0
VWPEP1=0.0D+0
VWPEP2=0.0D+0
VSURSOL=0.0D+0
**WATER-WATER,WATER-SURFACE AND WATER PEPTIDE
do 1000 M=1,NMOLW
do 1010 I=1,3
do 1020 C=1,3
RWM(C,I)=RW(M,C,I)
1020 continue
1010 continue
call DISTW(D2W,RWM,RW,M,NMOLW,LX,LY,LZ)
call DISTWMSURF(D2WSURF1,D2WSURF2,RSURF,RWM,NSURF,LX,LY)
call DISTWMPEP(D2WMPEP1,PEPCORD1,NATOM,RWM,LX,LY,COM1)
call DISTWMPEP(D2WMPEP2,PEPCORD2,NATOM,RWM,LX,LY,COM2)
call NRGWMPEP(D2WMPEP1,A12WP,C6WP,QCONWP,VPEP1WM,NATOM)
call NRGWMPEP(D2WMPEP2,A12WP,C6WP,QCONWP,VPEP2WM,NATOM)
call NRGW(RWM,M,NMOLW,D2W,A12W,C6W,QCONW,A12WS,C6WS,
1 QCONWS,VW,VWMSURF,D2WSURF1,D2WSURF2,NSURF,LX,LY,LZ)
VWATER=VWATER+VW
VWSURF=VWSURF+VWMSURF
VWPEP1=VWPEP1+VPEP1WM

```

```

VWPEP2=VWPEP2+VPEP2WM
VWMPEP(M,1)=VPEP1WM
VWMPEP(M,2)=VPEP2WM
do 2000 N=1,NATOM
  do 2010 K=1,3
    D2WPEP1(N,M,K)=D2WMPEP1(N,K)
    D2WPEP2(N,M,K)=D2WMPEP2(N,K)
2010  continue
2000  continue
1000  continue
  VWATER=VWATER/2.d+0
**PEPTIDE-PEPTIDE
  call DISTPP(D2PP,PEPCORD1,PEPCORD2,NATOM,LX,LY,COM1,COM2)
  call NRGPP(D2PP,A12PP,C6PP,QCONPP,VPP,NATOM)
**PEPTIDE-SURFACE
  call DISTPEPSF(D2PEP1SF1,D2PEP1SF2,RSURF,NSURF,PEPCORD1,
1      NATOM,LX,LY,COM1)
  call DISTPEPSF(D2PEP2SF1,D2PEP2SF2,RSURF,NSURF,PEPCORD2,
1      NATOM,LX,LY,COM2)
  call NRGPEPSF(A12PS,C6PS,QCONPS,VPEP1SF,D2PEP1SF1,
1      D2PEP1SF2,NSURF,NATOM)
  call NRGPEPSF(A12PS,C6PS,QCONPS,VPEP2SF,D2PEP2SF1,
1      D2PEP2SF2,NSURF,NATOM)
V=VWATER+VWSURF+VWPEP1+VWPEP2+VPP+VPEP1SF+VPEP2SF
VREST=VWATER
VSURSOL=VWSURF
return
end
*****
** DISTW....subroutine which calculates the squared distances between the atoms of the chosen molecule and the
atoms of the closest images of the remaining molecules in the system.      **
*****
  subroutine DISTW(D2,RM,R,M,NMOL,LX,LY,LZ)
  implicit NONE
  integer M,N,I,J,NMOL
  double precision LX,LY,LZ,CORX,CORY
  double precision DELX,DELY,DELZ
  double precision D2(3,2000,3),R(2000,3,3),RM(3,3)
C *****
C ** SQUARED DISTANCES.          **
C ** note that the minimum image convention is **
C ** applied only to oxygen pairs.      **

```



```

C *****
do 1000 N=1,M-1
  DELX=R(N,1,1)-RM(1,1)
  CORX=LX*dnint(DELX/LX)
  DELY=R(N,2,1)-RM(2,1)
  CORY=LY*dnint(DELY/LY)
do 1010 I=1,3
do 1020 J=1,3
  DELX=R(N,1,J)-RM(1,J)-CORX
  DELY=R(N,2,J)-RM(2,J)-CORY
  DELZ=R(N,3,J)-RM(3,J)
  D2(I,N,J)=DELX*DELX+DELY*DELY+DELZ*DELZ
1020 continue
1010 continue
1000 continue
. do 2000 N=M+1,NMOL
  DELX=R(N,1,1)-RM(1,1)
  CORX=LX*dnint(DELX/LX)
  DELY=R(N,2,1)-RM(2,1)
  CORY=LY*dnint(DELY/LY)
do 2010 I=1,3
do 2020 J=1,3
  DELX=R(N,1,J)-RM(1,J)-CORX
  DELY=R(N,2,J)-RM(2,J)-CORY
  DELZ=R(N,3,J)-RM(3,J)
  D2(I,N,J)=DELX*DELX+DELY*DELY+DELZ*DELZ
2020 continue
2010 continue
2000 continue
  return
end
*****
** NRGW.....subroutine which calculates the potential energy "felt" by a chosen molecule.
*****
  subroutine NRGW(RM,M,NMOL,D2,A12,C6,QCON,A12WS,C6WS,QCONWS,
&   VM,VMSURF,D2SURF1,D2SURF2,NSURF,LX,LY,LZ)
  implicit NONE
*****
  integer    M,N,I,J,NMOL,NSURF
  double precision VM,D,D6,D12,LX,LY,LZ
  double precision D2(3,2000,3),A12(3,3),C6(3,3),QCON(3,3),RM(3,3)
  double precision A12WS(3,2),C6WS(3,2),QCONWS(3,2)

```

```

double precision D2SURF1(3,600),D2SURF2(3,600)
double precision VMSURF,DSURF
C *****
C ** ENERGY CALCULATION. the calculation includes all pairs of atoms in the chosen molecule and the closest
images of the remaining molecules. **
C *****
  VM=0.0d+0
  VMSURF=0.0D+0
  do 1000 I=1,3
  do 1010 J=1,3
  do 1100 N=1,M-1
    D=dsqrt(D2(I,N,J))
    D6=D2(I,N,J)*D2(I,N,J)*D2(I,N,J)
    D12=D6*D6
    VM=VM+A12(I,J)/D12-C6(I,J)/D6+QCON(I,J)/D
  1100 continue
  do 1110 N=M+1,NMOL
    D=dsqrt(D2(I,N,J))
    D6=D2(I,N,J)*D2(I,N,J)*D2(I,N,J)
    D12=D6*D6
    VM=VM+A12(I,J)/D12-C6(I,J)/D6+QCON(I,J)/D
  1110 continue
  1010 continue
  1000 continue
  VMSURF=0.0
  do 2000 I=1,3
  do 2100 N=1,NSURF
    DSURF=dsqrt(D2SURF1(I,N))
    D6=D2SURF1(I,N)*D2SURF1(I,N)*D2SURF1(I,N)
    D12=D6*D6
    VMSURF=VMSURF+A12WS(I,1)/D12-C6WS(I,1)/D6
  1    +QCONWS(I,1)/DSURF
    DSURF=dsqrt(D2SURF2(I,N))
    D6=D2SURF2(I,N)*D2SURF2(I,N)*D2SURF2(I,N)
    D12=D6*D6
    VMSURF=VMSURF+A12WS(I,2)/D12-C6WS(I,2)/D6
  1    +QCONWS(I,2)/DSURF
  2100 continue
  2000 continue
  return
end
*****

```

```

** MULTMAT....subroutine which multiplies two matrices.          **
*****
subroutine MULTMAT(A,B,ROW,MID,COL,AB)
implicit NONE
*****
integer    I,J,K,ROW,MID,COL
double precision A(ROW,MID),B(MID,COL),AB(ROW,COL)
C *****
do 1000 I=1,ROW
do 1010 J=1,COL
  AB(I,J)=0.0d+0
do 1020 K=1,MID
  AB(I,J)=AB(I,J)+A(I,K)*B(K,J)
1020 continue
1010 continue
1000 continue
  return
end
*****
** SUMBINW....subroutine which collects the data for the number of atoms in each bin.**
*****
subroutine SUMBINW(M,NMOLW,PEPCORD1,RXMAX2,GSUM,DELR,
1  D2,D2MPEP,GRPSUM,NEIBOURSUM,HB,NHB,LX,LY,LZ,
1  RW,DELZ,ZSUM)
implicit NONE
integer I,J,M,N,BIN,NMOLW,GSUM(600,3)
integer ZSUM(2000,2),C,BINZ,NHB
integer GRPSUM(600,2,100),HB(100),NEIBOURSUM(100)
double precision D,DSQ,RXMAX2,DELR,DISTMIN
double precision D2(3,2000,3)
double precision HALFLZ,LX,LY,LZ
double precision RW(2000,3,3),DELZ,RWM(3,3)
double precision PEPCORD1(200,3),D2MPEP(200,3)
C *****
C ** COLLECTION OF DATA.          **
C *****
DISTMIN=3.3d+0
HALFLZ=LZ/2.d+0
** Density profile
do 2100 I=1,3
do 2200 C=1,3
  RWM(C,I)=RW(M,C,I)

```

```

2200 continue
2100 continue
** Oxygen
  BINZ=idint((RWM(3,1)+HALFLZ)/DELZ)+1
  ZSUM(BINZ,1)=ZSUM(BINZ,1)+1
** Hydrogen
  BINZ=idint((RWM(3,2)+HALFLZ)/DELZ)+1
  ZSUM(BINZ,2)=ZSUM(BINZ,2)+1
  BINZ=idint((RWM(3,3)+HALFLZ)/DELZ)+1
  ZSUM(BINZ,2)=ZSUM(BINZ,2)+1
**PEPTIDE-WATER PCF and nearest neighbour
do 1400 N=1,NHB
  DSQ=D2MPEP(HB(N),1)
  if(DSQ.LE.RXMAX2) then
    D=dsqrt(DSQ)
    if(D.LE.DISTMIN) then
      NEIBOURSUM(N)=NEIBOURSUM(N)+1
    end if
    BIN=idint(D/DELR-0.5d+0)+1
    GRPSUM(BIN,1,N)=GRPSUM(BIN,1,N)+1
  end if
do 1410 J=2,3
  DSQ=D2MPEP(HB(N),J)
  if(DSQ.LE.RXMAX2) then
    D=dsqrt(DSQ)
    BIN=idint(D/DELR-0.5d+0)+1
    GRPSUM(BIN,2,N)=GRPSUM(BIN,2,N)+1
  end if
1410 continue
1400 continue
**Water-Water PCF
do 1100 N=1,M-1
  DSQ=D2(1,N,1)
  if(DSQ.le.RXMAX2)then
    D=dsqrt(DSQ)
    BIN=idint(D/DELR-0.5d+0)+1
    GSUM(BIN,1)=GSUM(BIN,1)+1
  endif
do 1110 J=2,3
  DSQ=D2(1,N,J)
  if(DSQ.le.RXMAX2)then
    D=dsqrt(DSQ)

```

```

        BIN=idint(D/DELR-0.5d+0)+1
        GSUM(BIN,2)=GSUM(BIN,2)+1
    endif
1110  continue
    do 1120 I=2,3
        DSQ=D2(I,N,1)
        if(DSQ.le.RXMAX2)then
            D=dsqrt(DSQ)
            BIN=idint(D/DELR-0.5d+0)+1
            GSUM(BIN,2)=GSUM(BIN,2)+1
        endif
1120  continue
    do 1130 I=2,3
        do 1140 J=2,3
            DSQ=D2(I,N,J)
            if(DSQ.le.RXMAX2)then
                D=dsqrt(DSQ)
                BIN=idint(D/DELR-0.5d+0)+1
                GSUM(BIN,3)=GSUM(BIN,3)+1
            endif
1140  continue
1130  continue
1100  continue
    do 1200 N=M+1,NMOLW
        DSQ=D2(1,N,1)
        if(DSQ.le.RXMAX2)then
            D=dsqrt(DSQ)
            BIN=idint(D/DELR-0.5d+0)+1
            GSUM(BIN,1)=GSUM(BIN,1)+1
        endif
    do 1210 J=2,3
        DSQ=D2(1,N,J)
        if(DSQ.le.RXMAX2)then
            D=dsqrt(DSQ)
            BIN=idint(D/DELR-0.5d+0)+1
            GSUM(BIN,2)=GSUM(BIN,2)+1
        endif
1210  continue
    do 1220 I=2,3
        DSQ=D2(I,N,1)
        if(DSQ.le.RXMAX2)then
            D=dsqrt(DSQ)

```

```

        BIN=idint(D/DELR-0.5d+0)+1
        GSUM(BIN,2)=GSUM(BIN,2)+1
    endif
1220 continue
    do 1230 I=2,3
    do 1240 J=2,3
        DSQ=D2(I,N,J)
        if(DSQ.le.RXMAX2)then
            D=dsqrt(DSQ)
            BIN=idint(D/DELR-0.5d+0)+1
            GSUM(BIN,3)=GSUM(BIN,3)+1
        endif
1240 continue
1230 continue
1200 continue
    return
end
*****
** INCONFIG.....subroutine which sets up the initial configuration of water molecules for a cold start**
*****
    subroutine INCONFIG(NMOL,NMOLW,LX,LY,LZ,LOHW,
    & THHOH,RW,NATOM,PEPCORD1,PEPCORD2,APEP,
    & CPEP,QPEP,AW,AA,ATOM,GROUP,NHB,HB,NEND,CEND,phi1,phi2,psi1,psi2,
    & v1,v2,v3,v4,v5,v6,PEP,RESNUM)
    implicit NONE
*****
    integer I,IX,IY,IZ,N,NMOL,NMOLW,M
    integer FLAGX,FLAGY,FLAGZ,RESNUM
    integer NATOM,NHB,HB(100),NEND,CEND
    integer ATOM(200),AA(200),GROUP(200)
    double precision DL,LX,LY,LZ,DXHY,DYHY,LOHW,THHOH
    double precision TMP1,TMP2,TMP3, MAX2(3),MIN2(3)
    double precision VMAX,VWATER,MAX1(3),MIN1(3)
    double precision RW(2000,3,3),PEPCORD1(200,3)
    double precision PEPCORD2(200,3),APEP(200),CPEP(200),QPEP(200)
    double precision phi1(20),psi1(20),phi2(20),psi2(20)
    double precision v1,v2,v3,v4,v5,v6
    double precision AW(200)
    character PEP*15
    open(unit=10,file=PEP,status='unknown',form='formatted')
    read(10,*)
    read(10,*) NATOM

```

```

if(NATOM.GT.200) STOP 'Matrix size too small'
do 35000 M=1,NATOM
  read(10,889) AA(M),ATOM(M),GROUP(M),PEPCORD1(M,1),PEPCORD1(M,2),
1      APEP(M),CPEP(M),QPEP(M),AW(M)
          PEPCORD2(M,1)=PEPCORD1(M,1)
          PEPCORD2(M,2)=PEPCORD1(M,2)
35000 continue
  read(10,*)
  read(10,*) NHB
  do 35010 M=1,NHB
    read(10,*) HB(M)
35010 continue
889  format(5x,I3,1x,I3,1x,I4,1x,f8.3,f8.3,8X,f8.3,f8.3,f8.3,f8.3)
  read(10,*) NEND
  read(10,*) CEND
  read(10,*)
  do 34700 I=1,2
    read(10,222) MAX1(I),MIN1(I)
    MAX1(I)=MAX1(I)+1.1d+0
    MIN1(I)=MIN1(I)-1.1d+0
          MAX2(I)=MAX1(I)
          MIN2(I)=MIN1(I)
34700 continue
    read(10,222) MAX1(3),MIN1(3)
    read(10,222) MAX2(3),MIN2(3)
    MAX1(3)=MAX1(3)+1.1d+0
    MIN1(3)=MIN1(3)-1.1d+0
    MAX2(3)=MAX2(3)+1.1d+0
    MIN2(3)=MIN2(3)-1.1d+0
222  format(f8.3,1x,f8.3)
  do I=1,RESNUM
    read(10,21)phi1(I),psi1(I)
    phi2(I)=phi1(I)
    psi2(I)=psi1(I)
  enddo
21  format(5x,f8.3,5x,f8.3)
  read(10,22)v1,v2,v3,v4,v5,v6
22  format(f8.3,f8.3,f8.3,f8.3,f8.3,f8.3)
    read(10,*)
    do M=1,NATOM
      read(10,23) PEPCORD1(M,3),PEPCORD2(M,3)
    enddo

```

```

23 format(f8.3,f8.3)
   close(10)
   VMAX=1.d+0
   do 3000 I=1,3
     VMAX=dabs(MAX1(I)-MIN1(I))*VMAX
3000 continue
     VMAX=VMAX+VMAX
   VWATER=LX*LY*LZ-VMAX
   DL=(VWATER/NMOL)**(1.d+0/3.d+0)
C *****
C ** constants and stuff          **
C *****
   DXHY=LOHW*dsin(THHOH/2.0d+0)
   DYHY=LOHW*dcos(THHOH/2.0d+0)
77777 IX=1
   IY=1
   IZ=1
   do 1000 N=1,NMOL
9999   TMP1=(dble(IX)-0.5d+0)*DL-LX/2.0d+0
      if(dabs(TMP1).LT.LX/2.d+0) then
        FLAGX=0
        TMP2=(dble(IY)-0.5d+0)*DL-LY/2.0d+0
        if(dabs(TMP2).LT.LY/2.d+0) then
          FLAGY=0
          TMP3=(dble(IZ)-0.5d+0)*DL-LZ/2.0d+0+1.5d+0
          if(dabs(TMP3).LT.LZ/2.d+0) then
            FLAGZ=0
          else
            write(6,*) 'Wait,configure fails,trying smaller DL'
            DL=DL*0.99d+0
            go to 77777
          end if
        else
          FLAGY=1
          goto 88888
        end if
      else
        FLAGX=1
        goto 99999
      end if
      if(((TMP1.LE.MAX1(1)).and.(TMP1.GE.MIN1(1)))
$ .and.((TMP2.LE.MAX1(2)).and.(TMP2.GE.MIN1(2))))

```



```

$ .and.((TMP3.LE.MAX2(3)).and.(TMP3.GE.MIN1(3)))) then
  IX=IX+1
  goto 9999
else
  RW(N,1,1)=TMP1
  RW(N,2,1)=TMP2
  RW(N,3,1)=TMP3
  RW(N,1,2)=RW(N,1,1)-DXHY
  RW(N,2,2)=RW(N,2,1)+DYHY
  RW(N,3,2)=RW(N,3,1)
  RW(N,1,3)=RW(N,1,1)+DXHY
  RW(N,2,3)=RW(N,2,1)+DYHY
  RW(N,3,3)=RW(N,3,1)
  IX=IX+1
end if
88888 if(FLAGX.EQ.1) then
  IX=1
  IY=1
  IZ=IZ+1
  goto 9999
end if
99999 if(FLAGX.EQ.1) then
  IX=1
  IY=IY+1
  goto 9999
end if
1000 continue
return
end

```

```
*****
```

\*\* RANDOM.....This function generates random numbers in the range 0.0 TO 1.0. It is a modification of the RANDOM subroutine in V.A. Dyck, J.D. Lawson and J.A. Smith, "FORTRAN 77: an introduction to structured problem solving", Prentice-Hall, New Jersey (1984). Modifications include the use of a 97-element array to shuffle the generated random deviates. This is the technique used with the RAN0, RAN1, and RAN2 functions in W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, "Numerical Recipes: The Art of Scientific Computing (FORTRAN Vers.)", Cambridge University Press, Cambridge (1989). This modified generator is used since it is easy to keep track of the current value of the seed (iseed). This is necessary in Monte Carlo simulations so that a simulation may be continued using the same series of random numbers if desired. Note : The constants used in this generator are for use on a 32 bit machine only (e.g. VAX, DecStation). It is not suitable for use on machines with other than 32 bit word sizes. Version 1.0 February, 1992. Trevor Creamer. The code was obtained from the following web site: [http://euripides.gws.uky.edu/~trevor/Public/random\\_f.html](http://euripides.gws.uky.edu/~trevor/Public/random_f.html)

```
*****
```

```

double precision function RANDOM(ISEED)
implicit      NONE
*****
integer      I,ISEED,MBIG1,MBIG2,MBIG3
double precision N1,N2,RSMALL,RAN(97)
logical      LFIRST
data        MBIG1,MBIG2 /843314861,453816693/
data        MBIG3,RSMALL /2147483647,0.4656612d-9/
data        LFIRST,RAN /.true.,97*0.0d+0/
C *****
C ** initialization of shuffling array (1st call) **
C *****
if(LFIRST)then
  write(9,*)
  write(9,*) 'PARAMETERS FOR THE RANDOM # GENERATOR'
  write(9,*) MBIG1,MBIG2
  write(9,*) MBIG3,RSMALL
  do 1000 I=1,97
    ISEED=ISEED*MBIG1+MBIG2
    if(ISEED.lt.0)then
      ISEED=ISEED+MBIG3+1
    endif
    RAN(I)=dble(ISEED)*RSMALL
1000  continue
    LFIRST=.false.
  endif
C *****
C ** generation of a new random deviate using a linear congruential generator **
C *****
  ISEED=ISEED*MBIG1+MBIG2
  if(ISEED.lt.0)then
    ISEED=ISEED+MBIG3+1
  endif
  N1=dble(ISEED)*RSMALL
C *****
C ** use the new deviate to select one from the shuffling array and then replace the selected one in the array with the
newly generated deviate*
C *****
  I=idint(N1*97.0d+0)+1
  N2=RAN(I)
  RAN(I)=N1
  RANDOM=N2

```

```

return
end
*****
** DISTWMSURF....subroutine which calculates the squared distance between the atoms of the chosen molecules and
the sites of the surfaces. **
*****

subroutine DISTWMSURF(D2SURF1,D2SURF2,RSURF,RM,NSURF,LX,LY)
implicit NONE
integer N,I,NSURF
double precision LX,LY,DELX,DELY,DELZ1,DELZ2,CORX,CORY
double precision D2SURF1(3,600),D2SURF2(3,600)
double precision RM(3,3),RSURF(600,3)
do 2000 N=1,NSURF
  DELX=RSURF(N,1)-RM(1,1)
  CORX=LX*dnint(DELX/LX)
  DELY=RSURF(N,2)-RM(2,1)
  CORY=LY*dnint(DELY/LY)
  do 2100 I=1,3
    DELX=RSURF(N,1)-RM(1,I)-CORX
    DELY=RSURF(N,2)-RM(2,I)-CORY
    DELZ1=RSURF(N,3)-RM(3,I)
    DELZ2=-RSURF(N,3)-RM(3,I)
    D2SURF1(I,N)=DELX*DELX+DELY*DELY+DELZ1*DELZ1
    D2SURF2(I,N)=DELX*DELX+DELY*DELY+DELZ2*DELZ2
  2100 continue
2000 continue
return
end
*****
* PDBOUT: this subroutine generates a PDB output file. You can view it through RASMOL. **
*****

subroutine PDBOUT(RSURF,RW,PEPCORD1,PEPCORD2,FILENAME,NATOM,
1      NMOLW,NSURF,AA,ATOM,GROUP)
implicit none
character FILENAME*15
integer NATOM,NMOLW,NSURF
integer M,K,ALA,LYS
integer N,H,CA,CB1,CB2,C,O,CG,CD,CE,NZ,HZ1,HZ2,HZ3
integer AA(200),ATOM(200),GROUP(200)
double precision RSURF(600,3),RW(2000,3,3),PEPCORD1(200,3)
double precision PEPCORD2(200,3)
ALA=1

```

```
LYS=2
C =1
CA =2
CB1=3
CB2=4
CG =5
H =6
N =7
O =8
CD =9
CE =10
NZ =11
HZ1=12
HZ2=13
HZ3=14
open(unit=8,file=FILENAME,status='unknown',form='formatted')
write(8,*)'HEADER'
do 223 M=1,NATOM
  if(ATOM(M).EQ.C) then
    if(AA(M).EQ.ALA) then
      write(8,950) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
    else if(AA(M).EQ.LYS) then
      write(8,937) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
    end if
  else if(ATOM(M).EQ.CA) then
    if(AA(M).EQ.ALA) then
      write(8,951) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
    else if(AA(M).EQ.LYS) then
      write(8,938) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
    end if
  else if(ATOM(M).EQ.CB1) then
    if(AA(M).EQ.ALA) then
      write(8,952) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
    else if(AA(M).EQ.LYS) then
      write(8,939) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
    end if
```

```
else if(ATOM(M).EQ.CB2) then
  if(AA(M).EQ.ALA) then
    write(8,953) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  else if(AA(M).EQ.LYS) then
    write(8,940) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  end if
else if(ATOM(M).EQ.CG) then
  if(AA(M).EQ.ALA) then
    write(8,954) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  else if(AA(M).EQ.LYS) then
    write(8,941) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  end if
else if(ATOM(M).EQ.CD) then
  if(AA(M).EQ.ALA) then
    write(8,955) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  else if(AA(M).EQ.LYS) then
    write(8,942) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  end if
else if(ATOM(M).EQ.CE) then
  if(AA(M).EQ.ALA) then
    write(8,956) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  else if(AA(M).EQ.LYS) then
    write(8,943) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  end if
else if(ATOM(M).EQ.H) then
  if(AA(M).EQ.ALA) then
    write(8,957) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  else if(AA(M).EQ.LYS) then
    write(8,944) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
  end if
else if(ATOM(M).EQ.N) then
  if(AA(M).EQ.ALA) then
```

```
write(8,958) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
else if(AA(M).EQ.LYS) then
write(8,945) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
end if
else if(ATOM(M).EQ.NZ) then
if(AA(M).EQ.ALA) then
write(8,959) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
else if(AA(M).EQ.LYS) then
write(8,946) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
end if
else if(ATOM(M).EQ.O) then
if(AA(M).EQ.ALA) then
write(8,960) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
else if(AA(M).EQ.LYS) then
write(8,947) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
end if
else if(ATOM(M).EQ.HZ1) then
if(AA(M).EQ.ALA) then
write(8,961) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
else if(AA(M).EQ.LYS) then
write(8,948) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
end if
else if(ATOM(M).EQ.HZ2) then
if(AA(M).EQ.ALA) then
write(8,962) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
else if(AA(M).EQ.LYS) then
write(8,949) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
end if
else if(ATOM(M).EQ.HZ3) then
if(AA(M).EQ.ALA) then
write(8,963) M,GROUP(M),PEPCORD1(M,1),
1      PEPCORD1(M,2),PEPCORD1(M,3)
```

```
      else if(AA(M).EQ.LYS) then
        write(8,850) M,GROUP(M),PEPCORD1(M,1),
1         PEPCORD1(M,2),PEPCORD1(M,3)
        end if
      end if
223 continue
      do 230 M=1,NATOM
        if(ATOM(M).EQ.C) then
          if(AA(M).EQ.ALA) then
            write(8,950) M,GROUP(M),PEPCORD2(M,1),
1             PEPCORD2(M,2),PEPCORD2(M,3)
            else if(AA(M).EQ.LYS) then
              write(8,937) M,GROUP(M),PEPCORD2(M,1),
1              PEPCORD2(M,2),PEPCORD2(M,3)
            end if
          else if(ATOM(M).EQ.CA) then
            if(AA(M).EQ.ALA) then
              write(8,951) M,GROUP(M),PEPCORD2(M,1),
1              PEPCORD2(M,2),PEPCORD2(M,3)
            else if(AA(M).EQ.LYS) then
              write(8,938) M,GROUP(M),PEPCORD2(M,1),
1              PEPCORD2(M,2),PEPCORD2(M,3)
            end if
          else if(ATOM(M).EQ.CB1) then
            if(AA(M).EQ.ALA) then
              write(8,952) M,GROUP(M),PEPCORD2(M,1),
1              PEPCORD2(M,2),PEPCORD2(M,3)
            else if(AA(M).EQ.LYS) then
              write(8,939) M,GROUP(M),PEPCORD2(M,1),
1              PEPCORD2(M,2),PEPCORD2(M,3)
            end if
          else if(ATOM(M).EQ.CB2) then
            if(AA(M).EQ.ALA) then
              write(8,953) M,GROUP(M),PEPCORD2(M,1),
1              PEPCORD2(M,2),PEPCORD2(M,3)
            else if(AA(M).EQ.LYS) then
              write(8,940) M,GROUP(M),PEPCORD2(M,1),
1              PEPCORD2(M,2),PEPCORD2(M,3)
            end if
          else if(ATOM(M).EQ.CG) then
            if(AA(M).EQ.ALA) then
              write(8,954) M,GROUP(M),PEPCORD2(M,1),
```

```
1      PEPCORD2(M,2),PEPCORD2(M,3)
      else if(AA(M).EQ.LYS) then
        write(8,941) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        end if
      else if(ATOM(M).EQ.CD) then
        if(AA(M).EQ.ALA) then
          write(8,955) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        else if(AA(M).EQ.LYS) then
          write(8,942) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        end if
      else if(ATOM(M).EQ.CE) then
        if(AA(M).EQ.ALA) then
          write(8,956) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        else if(AA(M).EQ.LYS) then
          write(8,943) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        end if
      else if(ATOM(M).EQ.H) then
        if(AA(M).EQ.ALA) then
          write(8,957) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        else if(AA(M).EQ.LYS) then
          write(8,944) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        end if
      else if(ATOM(M).EQ.N) then
        if(AA(M).EQ.ALA) then
          write(8,958) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        else if(AA(M).EQ.LYS) then
          write(8,945) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        end if
      else if(ATOM(M).EQ.NZ) then
        if(AA(M).EQ.ALA) then
          write(8,959) M,GROUP(M),PEPCORD2(M,1),
1      PEPCORD2(M,2),PEPCORD2(M,3)
        else if(AA(M).EQ.LYS) then
```



```

        write(8,946) M,GROUP(M),PEPCORD2(M,1),
1          PEPCORD2(M,2),PEPCORD2(M,3)
        end if
        else if(ATOM(M).EQ.O) then
            if(AA(M).EQ.ALA) then
                write(8,960) M,GROUP(M),PEPCORD2(M,1),
1          PEPCORD2(M,2),PEPCORD2(M,3)
            else if(AA(M).EQ.LYS) then
                write(8,947) M,GROUP(M),PEPCORD2(M,1),
1          PEPCORD2(M,2),PEPCORD2(M,3)
            end if
        else if(ATOM(M).EQ.HZ1) then
            if(AA(M).EQ.ALA) then
                write(8,961) M,GROUP(M),PEPCORD2(M,1),
1          PEPCORD2(M,2),PEPCORD2(M,3)
            else if(AA(M).EQ.LYS) then
                write(8,948) M,GROUP(M),PEPCORD2(M,1),
1          PEPCORD2(M,2),PEPCORD2(M,3)
            end if
        else if(ATOM(M).EQ.HZ2) then
            if(AA(M).EQ.ALA) then
                write(8,962) M,GROUP(M),PEPCORD2(M,1),
1          PEPCORD2(M,2),PEPCORD2(M,3)
            else if(AA(M).EQ.LYS) then
                write(8,949) M,GROUP(M),PEPCORD2(M,1),
1          PEPCORD2(M,2),PEPCORD2(M,3)
            end if
        else if(ATOM(M).EQ.HZ3) then
            if(AA(M).EQ.ALA) then
                write(8,963) M,GROUP(M),PEPCORD2(M,1),
1          PEPCORD2(M,2),PEPCORD2(M,3)
            else if(AA(M).EQ.LYS) then
                write(8,850) M,GROUP(M),PEPCORD2(M,1),
1          PEPCORD2(M,2),PEPCORD2(M,3)
            end if
        end if
230 continue
        K=NATOM
        do 224 M=1,NMOLW
            K=K+1
            write(8,964) K,9,RW(M,1,1),RW(M,2,1),RW(M,3,1)
            K=K+1

```

```

write(8,965) K,10,RW(M,1,2),RW(M,2,2),RW(M,3,2)
K=K+1
write(8,965) K,10,RW(M,1,3),RW(M,2,3),RW(M,3,3)
224 continue
do 226 M=1,NSURF
K=K+1
write(8,966) K,11,RSURF(M,1),RSURF(M,2),RSURF(M,3)
K=K+1
write(8,966) K,12,RSURF(M,1),RSURF(M,2),-RSURF(M,3)
226 continue
937 format('ATOM ',I5,1x,' C ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
938 format('ATOM ',I5,1x,' CA ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
939 format('ATOM ',I5,1x,' CB1',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
940 format('ATOM ',I5,1x,' CB2',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
941 format('ATOM ',I5,1x,' CG ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
942 format('ATOM ',I5,1x,' CD ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
943 format('ATOM ',I5,1x,' CE ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
944 format('ATOM ',I5,1x,' H ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
945 format('ATOM ',I5,1x,' N ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
946 format('ATOM ',I5,1x,' NZ ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
947 format('ATOM ',I5,1x,' O ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
948 format('ATOM ',I5,1x,' HZ1',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
949 format('ATOM ',I5,1x,' HZ2',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
850 format('ATOM ',I5,1x,' HZ3',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
950 format('ATOM ',I5,1x,' C ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
951 format('ATOM ',I5,1x,' CA ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
952 format('ATOM ',I5,1x,' CB1',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
953 format('ATOM ',I5,1x,' CB2',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
954 format('ATOM ',I5,1x,' CG ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
955 format('ATOM ',I5,1x,' CD ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
956 format('ATOM ',I5,1x,' CE ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
957 format('ATOM ',I5,1x,' H ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
958 format('ATOM ',I5,1x,' N ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
959 format('ATOM ',I5,1x,' NZ ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
960 format('ATOM ',I5,1x,' O ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
961 format('ATOM ',I5,1x,' HZ1',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
962 format('ATOM ',I5,1x,' HZ2',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
963 format('ATOM ',I5,1x,' HZ3',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
964 format('ATOM ',I5,1x,' OW ',1x,'H2O',2x,I4,4x,f8.3,f8.3,f8.3)
965 format('ATOM ',I5,1x,' HW ',1x,'H2O',2x,I4,4x,f8.3,f8.3,f8.3)
966 format('ATOM ',I5,1x,' CH2',1x,'SUF',2x,I4,4x,f8.3,f8.3,f8.3)
close(8)

```

```

return
end
*****
** DISTWMPEP.....subroutine which calculates the squared distances between the atoms of the chosen molecule and
the atoms of the closest images of the remaining molecules in the system.      **
*****

subroutine DISTWMPEP(D2MPEP,PEPCORD,NATOM,RM,LX,LY,COM)
implicit NONE
integer N,I,NATOM
double precision DELX,DELY,DELZ,LX,LY,CORX,CORY
double precision D2MPEP(200,3),RM(3,3),COM(3)
double precision PEPCORD(200,3)
C *****
C ** SQUARED DISTANCES.                **
C ** note that the minimum image convention is applied only to oxygen pairs.      **
C *****

DELX=RM(1,1)-COM(1)
CORX=LX*dnint(DELX/LX)
DELY=RM(2,1)-COM(2)
CORY=LY*dnint(DELY/LY)
do 1010 I=1,3
  do 1020 N=1,NATOM
    DELX=RM(1,I)-PEPCORD(N,1)-CORX
    DELY=RM(2,I)-PEPCORD(N,2)-CORY
    DELZ=RM(3,I)-PEPCORD(N,3)
    D2MPEP(N,I)=DELX*DELX+DELY*DELY+DELZ*DELZ
1020 continue
1010 continue
  return
end
*****
** DISTPP.....subroutine which calculates the squared distances between the atoms of the chosen peptide molecule and
the atoms of the closest images of the remaining peptide molecules in the system.**
*****

subroutine DISTPP(D2PP,PEPCORD1,PEPCORD2,NATOM,LX,LY,COM1,COM2)
implicit NONE
integer N,I,NATOM
double precision DELX,DELY,DELZ,LX,LY,CORX,CORY
double precision D2PP(200,200),COM1(3),COM2(3)
double precision PEPCORD1(200,3),PEPCORD2(200,3)
C *****
C ** SQUARED DISTANCES.                **

```

```

C ** note that the minimum image convention is applied only to COM          **
C *****
  DELX=COM1(1)-COM2(1)
  CORX=LX*dnint(DELX/LX)
  DELY=COM1(2)-COM2(2)
  CORY=LY*dnint(DELY/LY)
  do 1011 I=1,NATOM
    do 1021 N=1,NATOM
      DELX=PEPCORD1(I,1)-PEPCORD2(N,1)-CORX
      DELY=PEPCORD1(I,2)-PEPCORD2(N,2)-CORY
      DELZ=PEPCORD1(I,3)-PEPCORD2(N,3)
      D2PP(N,I)=DELX*DELX+DELY*DELY+DELZ*DELZ
1021 continue
1011 continue
      return
    end
*****
** DISTPEPSF.....subroutine which calculates the squared distance between the atoms of the chosen molecules and the
sites of the surfaces. **
*****
  subroutine DISTPEPSF(D2PEPSF1,D2PEPSF2,RSURF,NSURF,PEPCORD1,NATOM,
1    LX,LY,COM)
  implicit NONE
  integer N,M,NSURF,NATOM
  double precision LX,LY,DELX,DELY,DELZ1,DELZ2,CORX,CORY
  double precision D2PEPSF1(200,600),D2PEPSF2(200,600)
  double precision PEPCORD1(200,3)
  double precision RSURF(600,3),COM(3)
  do 2000 N=1,NSURF
    DELX=RSURF(N,1)-COM(1)
    CORX=LX*dnint(DELX/LX)
    DELY=RSURF(N,2)-COM(2)
    CORY=LY*dnint(DELY/LY)
    do 2100 M=1,NATOM
      DELX=RSURF(N,1)-PEPCORD1(M,1)-CORX
      DELY=RSURF(N,2)-PEPCORD1(M,2)-CORY
      DELZ1=RSURF(N,3)-PEPCORD1(M,3)
      DELZ2=-RSURF(N,3)-PEPCORD1(M,3)
      D2PEPSF1(M,N)=DELX*DELX+DELY*DELY+DELZ1*DELZ1
      D2PEPSF2(M,N)=DELX*DELX+DELY*DELY+DELZ2*DELZ2
2100 continue
2000 continue

```

```

return
end

*****
** NRGPEPSF: subroutine which calculates the potential energy "felt" by a chosen molecule. **
*****

subroutine NRGPEPSF(A12PX,C6PX,QCONPX,VPEPSF,D2PEPSF1,D2PEPSF2,
1      NSURF,NATOM)
implicit NONE
*****

integer      M,N,NSURF,NATOM
double precision D6,D12,VPEPSF,DSURF
double precision D2PEPSF1(200,600),D2PEPSF2(200,600)
double precision A12PX(200,2),C6PX(200,2),QCONPX(200,2)
VPEPSF=0.d+0
do 2000 M=1,NATOM
  do 2100 N=1,NSURF
    DSURF=dsqrt(D2PEPSF1(M,N))
    D6=D2PEPSF1(M,N)**3
    D12=D6*D6
    VPEPSF=VPEPSF+A12PX(M,1)/D12-C6PX(M,1)/D6
1    +QCONPX(M,1)/DSURF
    DSURF=dsqrt(D2PEPSF2(M,N))
    D6=D2PEPSF2(M,N)**3
    D12=D6*D6
    VPEPSF=VPEPSF+A12PX(M,2)/D12-C6PX(M,2)/D6
1    +QCONPX(M,2)/DSURF
2100  continue
2000  continue
  return
end

*****
** NRGWMPEP: subroutine which calculates the potential energy "felt" by a chosen molecule.
*****

subroutine NRGWMPEP(D2MPEP,A12PX,C6PX,QCONPX,VMPEP,NATOM)
implicit NONE
*****

integer      M,I,NATOM
double precision VMPEP,D,D6,D12
double precision A12PX(200,3),C6PX(200,3),QCONPX(200,3)
double precision D2MPEP(200,3)
C *****

```

```

C ** ENERGY CALCULATION.
C ** the calculation includes all pairs of atoms in the chosen molecule and the closest images of the remaining
molecules. **
C *****
  VMPEP=0.d+0
  do 1000 I=1,3
    do 1010 M=1,NATOM
      D=dsqrt(D2MPEP(M,I))
      D6=D2MPEP(M,I)**3
      D12=D6*D6
      VMPEP=VMPEP+A12PX(M,I)/D12-C6PX(M,I)/D6+QCONPX(M,I)/D
1010  continue
1000  continue
      return
    end
*****
** NRGPP: subroutine which calculates the potential energy "felt" by a chosen peptide molecule from the atoms of the
remaining peptides.**
*****
      subroutine NRGPP(D2PP,A12PP,C6PP,QCONPP,VPP,NATOM)
      implicit NONE
*****
      integer      M,I,NATOM
      double precision VPP,D,D6,D12
      double precision A12PP(200,200),C6PP(200,200),QCONPP(200,200)
      double precision D2PP(200,200)
C *****
C ** ENERGY CALCULATION. **
C ** the calculation includes all pairs of atoms in the chosen molecule and the closest images of the remaining
molecules. **
C *****
  VPP=0.d+0
  do 1001 I=1,NATOM
    do 1011 M=1,NATOM
      D=dsqrt(D2PP(M,I))
      D6=D2PP(M,I)**3
      D12=D6*D6
      VPP=VPP+A12PP(M,I)/D12-C6PP(M,I)/D6+QCONPP(M,I)/D
1011  continue
1001  continue
      return
    end

```

```

*****
** MOVEPEP....subroutine which randomly displaces a molecule. the displacement is performed as a translational
displacement of the molecule coupled with a rotational displacement about the position of the oxygen atom. **
*****

subroutine MOVEPEP(NATOM,PEPCORD1,COM,MXTPEP,MXRPEP,
1      LX,LY,LZ,ISEED)
implicit NONE
external MULTMAT,RANDOM
*****

integer      I,M,ISEED,NATOM
double precision LX,LY,LZ,DELC,MXTPEP,MXRPEP,THX,THY,THZ
double precision COSX,SINX,COSY,SINY,COSZ,SINZ
double precision ROTX(3,3),ROTY(3,3),ROTZ(3,3),TMP1(3,3)
double precision TMP2(3,3),COM(3)
double precision PEPCORD1(200,3),RHYOLD(3,200),RHYNEW(3,200)
double precision RANDOM,L(3)

L(1)=LX
L(2)=LY
L(3)=LZ

C *****
C ** HYDROGEN COORDINATE MATRIX.          **
C ** the positions of the hydrogen atoms are defined relative to the oxygen atom.    **
C *****

do 1000 M=1,NATOM
do 1010 I=1,3
RHYOLD(I,M)=PEPCORD1(M,I)-COM(I)
1010 continue
1000 continue

c *****
C ** RANDOM ROTATION.                    **
C *****

THX=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MXRPEP
THY=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MXRPEP
THZ=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MXRPEP
COSX=dcos(THX)
SINX=dsin(THX)
COSY=dcos(THY)
SINY=dsin(THY)
COSZ=dcos(THZ)
SINZ=dsin(THZ)
ROTX(1,1)=1.0d+0
ROTX(1,2)=0.0d+0

```

```

ROTX(1,3)=0.0d+0
ROTX(2,1)=0.0d+0
ROTX(2,2)=COSX
ROTX(2,3)=-SINX
ROTX(3,1)=0.0d+0
ROTX(3,2)=SINX
ROTX(3,3)=COSX
ROTY(1,1)=COSY
ROTY(1,2)=0.0d+0
ROTY(1,3)=-SINY
ROTY(2,1)=0.0d+0
ROTY(2,2)=1.0d+0
ROTY(2,3)=0.0d+0
ROTY(3,1)=SINY
ROTY(3,2)=0.0d+0
ROTY(3,3)=COSY
ROTZ(1,1)=COSZ
ROTZ(1,2)=-SINZ
ROTZ(1,3)=0.0d+0
ROTZ(2,1)=SINZ
ROTZ(2,2)=COSZ
ROTZ(2,3)=0.0d+0
ROTZ(3,1)=0.0d+0
ROTZ(3,2)=0.0d+0
ROTZ(3,3)=1.0d+0
call MULTMAT(ROTX,ROTY,3,3,3,TMP1)
call MULTMAT(TMP1,ROTZ,3,3,3,TMP2)
call MULTMAT(TMP2,RHYOLD,3,3,NATOM,RHYNEW)
C *****
C ** RANDOM TRANSLATION.          **
C *****
do 100 I=1,3
  DELC=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MXTPEP*L(I)/LX
  COM(I)=COM(I)+DELC
  if(I.NE.3) then
    COM(I)=COM(I)-L(I)*dnint(COM(I)/L(I))
  end if
  do 110 M=1,NATOM
    PEPCORD1(M,I)=COM(I)+RHYNEW(I,M)
110  continue
100  continue
return

```



```

end
*****
** SUBROUTINE WHICH CALCULATES THE ENERGY OF THE PEPTIDE MOLECULE **
*****

SUBROUTINE pepenrg(configE,APEP,CPEP,QPEP,PEPCORD1,NATOM,
1 phi,psi,v1,v2,v3,v4,v5,v6,RESNUM)
integer NATOM,M
integer I,RESNUM
double precision PEPCORD1(200,3),APEP(200),CPEP(200),QPEP(200)
double precision QFACTP,TEMP,A12P(200,200),C6P(200,200)
double precision QCONP(200,200)
double precision AP(200,200),CP(200,200)
double precision D2PEP(200,200),DELX,DELY,DELZ
double precision PVPEP,DSURF,D6,D12,VTOR
double precision phi(20),psi(20),configE
double precision v1,v2,v3,v4,v5,v6
c *****
c ** The non-bonded interaction energy calculations **
c *****
c ***** LJ Potential Calculations *****
TEMP=298.0
QFACTP=1.671d+5/TEMP
do 200 M=1,NATOM-1
do 201 I=M+1,NATOM
AP(M,I)=dsqrt( APEP(M)*APEP(I) )
CP(M,I)=dsqrt( CPEP(M)*CPEP(I) )
201 continue
200 continue
do 202 M=1,NATOM-1
do 203 I=M+1,NATOM
A12P(M,I)=(AP(M,I)**12)/TEMP
C6P(M,I)=(CP(M,I)**6)/TEMP
QCONP(M,I)=QFACTP*QPEP(M)*QPEP(I)
203 continue
202 continue
A12P(1,2)=0.0d+0
A12P(1,3)=0.0d+0
A12P(1,4)=0.0d+0
A12P(1,5)=0.0d+0
A12P(2,1)=0.0d+0
A12P(2,3)=0.0d+0
A12P(3,1)=0.0d+0

```

A12P(3,2)=0.0d+0  
A12P(3,4)=0.0d+0  
A12P(3,5)=0.0d+0  
A12P(3,6)=0.0d+0  
A12P(3,7)=0.0d+0  
A12P(4,1)=0.0d+0  
A12P(4,3)=0.0d+0  
A12P(4,5)=0.0d+0  
A12P(5,1)=0.0d+0  
A12P(5,3)=0.0d+0  
A12P(5,4)=0.0d+0  
A12P(5,6)=0.0d+0  
A12P(5,7)=0.0d+0  
A12P(5,8)=0.0d+0  
A12P(5,9)=0.0d+0  
A12P(6,3)=0.0d+0  
A12P(6,5)=0.0d+0  
A12P(6,7)=0.0d+0  
A12P(7,3)=0.0d+0  
A12P(7,5)=0.0d+0  
A12P(7,6)=0.0d+0  
A12P(7,8)=0.0d+0  
A12P(7,9)=0.0d+0  
A12P(7,10)=0.0d+0  
A12P(7,11)=0.0d+0  
A12P(8,5)=0.0d+0  
A12P(8,7)=0.0d+0  
A12P(8,9)=0.0d+0  
A12P(9,5)=0.0d+0  
A12P(9,7)=0.0d+0  
A12P(9,8)=0.0d+0  
A12P(9,10)=0.0d+0  
A12P(9,11)=0.0d+0  
A12P(9,12)=0.0d+0  
A12P(9,13)=0.0d+0  
A12P(10,9)=0.0d+0  
A12P(10,7)=0.0d+0  
A12P(10,11)=0.0d+0  
A12P(11,7)=0.0d+0  
A12P(11,9)=0.0d+0  
A12P(11,10)=0.0d+0  
A12P(11,12)=0.0d+0

A12P(11,13)=0.0d+0  
A12P(11,14)=0.0d+0  
A12P(11,15)=0.0d+0  
A12P(12,11)=0.0d+0  
A12P(12,9)=0.0d+0  
A12P(12,13)=0.0d+0  
A12P(13,9)=0.0d+0  
A12P(13,12)=0.0d+0  
A12P(13,11)=0.0d+0  
A12P(13,14)=0.0d+0  
A12P(13,15)=0.0d+0  
A12P(13,16)=0.0d+0  
A12P(13,17)=0.0d+0  
A12P(14,11)=0.0d+0  
A12P(14,13)=0.0d+0  
A12P(14,15)=0.0d+0  
A12P(15,11)=0.0d+0  
A12P(15,13)=0.0d+0  
A12P(15,14)=0.0d+0  
A12P(15,16)=0.0d+0  
A12P(15,17)=0.0d+0  
A12P(15,18)=0.0d+0  
A12P(15,19)=0.0d+0  
A12P(16,13)=0.0d+0  
A12P(16,15)=0.0d+0  
A12P(16,17)=0.0d+0  
A12P(17,13)=0.0d+0  
A12P(17,15)=0.0d+0  
A12P(17,16)=0.0d+0  
A12P(17,18)=0.0d+0  
A12P(17,19)=0.0d+0  
A12P(17,20)=0.0d+0  
A12P(17,21)=0.0d+0  
A12P(18,15)=0.0d+0  
A12P(18,17)=0.0d+0  
A12P(18,19)=0.0d+0  
A12P(19,15)=0.0d+0  
A12P(19,17)=0.0d+0  
A12P(19,18)=0.0d+0  
A12P(19,20)=0.0d+0  
A12P(19,21)=0.0d+0  
A12P(19,22)=0.0d+0

A12P(19,23)=0.0d+0  
A12P(20,17)=0.0d+0  
A12P(20,19)=0.0d+0  
A12P(20,21)=0.0d+0  
A12P(21,17)=0.0d+0  
A12P(21,19)=0.0d+0  
A12P(21,20)=0.0d+0  
A12P(21,22)=0.0d+0  
A12P(21,23)=0.0d+0  
A12P(21,24)=0.0d+0  
A12P(21,25)=0.0d+0  
A12P(22,19)=0.0d+0  
A12P(22,21)=0.0d+0  
A12P(22,23)=0.0d+0  
A12P(23,19)=0.0d+0  
A12P(23,22)=0.0d+0  
A12P(23,21)=0.0d+0  
A12P(23,24)=0.0d+0  
A12P(23,25)=0.0d+0  
A12P(23,26)=0.0d+0  
A12P(23,27)=0.0d+0  
A12P(24,21)=0.0d+0  
A12P(24,23)=0.0d+0  
A12P(24,25)=0.0d+0  
A12P(25,21)=0.0d+0  
A12P(25,23)=0.0d+0  
A12P(25,24)=0.0d+0  
A12P(25,26)=0.0d+0  
A12P(25,27)=0.0d+0  
A12P(25,28)=0.0d+0  
A12P(25,29)=0.0d+0  
A12P(26,23)=0.0d+0  
A12P(26,25)=0.0d+0  
A12P(26,27)=0.0d+0  
A12P(27,23)=0.0d+0  
A12P(27,25)=0.0d+0  
A12P(27,26)=0.0d+0  
A12P(27,28)=0.0d+0  
A12P(27,29)=0.0d+0  
A12P(27,30)=0.0d+0  
A12P(27,31)=0.0d+0  
A12P(28,25)=0.0d+0

A12P(28,27)=0.0d+0  
A12P(28,29)=0.0d+0  
A12P(29,25)=0.0d+0  
A12P(29,27)=0.0d+0  
A12P(29,28)=0.0d+0  
A12P(29,30)=0.0d+0  
A12P(29,31)=0.0d+0  
A12P(29,32)=0.0d+0  
A12P(29,33)=0.0d+0  
A12P(30,27)=0.0d+0  
A12P(30,29)=0.0d+0  
A12P(30,31)=0.0d+0  
A12P(31,27)=0.0d+0  
A12P(31,29)=0.0d+0  
A12P(31,30)=0.0d+0  
A12P(31,32)=0.0d+0  
A12P(31,33)=0.0d+0  
A12P(31,34)=0.0d+0  
A12P(31,42)=0.0d+0  
A12P(32,29)=0.0d+0  
A12P(32,31)=0.0d+0  
A12P(32,33)=0.0d+0  
A12P(33,29)=0.0d+0  
A12P(33,31)=0.0d+0  
A12P(33,32)=0.0d+0  
A12P(33,34)=0.0d+0  
A12P(33,35)=0.0d+0  
A12P(33,42)=0.0d+0  
A12P(33,43)=0.0d+0  
A12P(33,44)=0.0d+0  
A12P(34,33)=0.0d+0  
A12P(34,31)=0.0d+0  
A12P(34,42)=0.0d+0  
A12P(34,35)=0.0d+0  
A12P(34,36)=0.0d+0  
A12P(35,33)=0.0d+0  
A12P(35,34)=0.0d+0  
A12P(35,36)=0.0d+0  
A12P(35,37)=0.0d+0  
A12P(36,34)=0.0d+0  
A12P(36,35)=0.0d+0  
A12P(36,37)=0.0d+0

A12P(36,38)=0.0d+0  
A12P(37,35)=0.0d+0  
A12P(37,36)=0.0d+0  
A12P(37,38)=0.0d+0  
A12P(37,39)=0.0d+0  
A12P(37,40)=0.0d+0  
A12P(38,36)=0.0d+0  
A12P(38,37)=0.0d+0  
A12P(38,39)=0.0d+0  
A12P(38,40)=0.0d+0  
A12P(38,41)=0.0d+0  
A12P(39,37)=0.0d+0  
A12P(39,38)=0.0d+0  
A12P(39,40)=0.0d+0  
A12P(39,41)=0.0d+0  
A12P(40,37)=0.0d+0  
A12P(40,38)=0.0d+0  
A12P(40,39)=0.0d+0  
A12P(40,41)=0.0d+0  
A12P(41,37)=0.0d+0  
A12P(41,38)=0.0d+0  
A12P(41,39)=0.0d+0  
A12P(41,40)=0.0d+0  
A12P(42,31)=0.0d+0  
A12P(42,33)=0.0d+0  
A12P(42,34)=0.0d+0  
A12P(42,43)=0.0d+0  
A12P(42,44)=0.0d+0  
A12P(42,45)=0.0d+0  
A12P(42,46)=0.0d+0  
A12P(43,33)=0.0d+0  
A12P(43,42)=0.0d+0  
A12P(43,44)=0.0d+0  
A12P(44,33)=0.0d+0  
A12P(44,42)=0.0d+0  
A12P(44,43)=0.0d+0  
A12P(44,45)=0.0d+0  
A12P(44,46)=0.0d+0  
A12P(44,47)=0.0d+0  
A12P(44,55)=0.0d+0  
A12P(45,42)=0.0d+0  
A12P(45,44)=0.0d+0

A12P(45,46)=0.0d+0  
A12P(46,42)=0.0d+0  
A12P(46,44)=0.0d+0  
A12P(46,45)=0.0d+0  
A12P(46,47)=0.0d+0  
A12P(46,48)=0.0d+0  
A12P(46,55)=0.0d+0  
A12P(46,56)=0.0d+0  
A12P(46,57)=0.0d+0  
A12P(47,44)=0.0d+0  
A12P(47,46)=0.0d+0  
A12P(47,48)=0.0d+0  
A12P(47,49)=0.0d+0  
A12P(47,55)=0.0d+0  
A12P(48,46)=0.0d+0  
A12P(48,47)=0.0d+0  
A12P(48,49)=0.0d+0  
A12P(48,50)=0.0d+0  
A12P(49,47)=0.0d+0  
A12P(49,48)=0.0d+0  
A12P(49,50)=0.0d+0  
A12P(49,51)=0.0d+0  
A12P(50,49)=0.0d+0  
A12P(50,48)=0.0d+0  
A12P(50,51)=0.0d+0  
A12P(50,52)=0.0d+0  
A12P(50,53)=0.0d+0  
A12P(51,49)=0.0d+0  
A12P(51,50)=0.0d+0  
A12P(51,52)=0.0d+0  
A12P(51,53)=0.0d+0  
A12P(51,54)=0.0d+0  
A12P(52,50)=0.0d+0  
A12P(52,51)=0.0d+0  
A12P(52,53)=0.0d+0  
A12P(52,54)=0.0d+0  
A12P(53,50)=0.0d+0  
A12P(53,51)=0.0d+0  
A12P(53,52)=0.0d+0  
A12P(53,54)=0.0d+0  
A12P(54,50)=0.0d+0  
A12P(54,51)=0.0d+0

A12P(54,52)=0.0d+0  
A12P(54,53)=0.0d+0  
A12P(55,44)=0.0d+0  
A12P(55,46)=0.0d+0  
A12P(55,47)=0.0d+0  
A12P(55,56)=0.0d+0  
A12P(55,57)=0.0d+0  
A12P(55,58)=0.0d+0  
A12P(56,46)=0.0d+0  
A12P(56,55)=0.0d+0  
A12P(56,57)=0.0d+0  
A12P(57,46)=0.0d+0  
A12P(57,55)=0.0d+0  
A12P(57,56)=0.0d+0  
A12P(57,58)=0.0d+0  
A12P(57,59)=0.0d+0  
A12P(57,60)=0.0d+0  
A12P(57,68)=0.0d+0  
A12P(58,55)=0.0d+0  
A12P(58,57)=0.0d+0  
A12P(58,59)=0.0d+0  
A12P(59,55)=0.0d+0  
A12P(59,57)=0.0d+0  
A12P(59,58)=0.0d+0  
A12P(59,60)=0.0d+0  
A12P(59,61)=0.0d+0  
A12P(59,68)=0.0d+0  
A12P(59,69)=0.0d+0  
A12P(59,70)=0.0d+0  
A12P(60,57)=0.0d+0  
A12P(60,59)=0.0d+0  
A12P(60,68)=0.0d+0  
A12P(60,61)=0.0d+0  
A12P(60,62)=0.0d+0  
A12P(61,59)=0.0d+0  
A12P(61,60)=0.0d+0  
A12P(61,62)=0.0d+0  
A12P(61,63)=0.0d+0  
A12P(62,60)=0.0d+0  
A12P(62,61)=0.0d+0  
A12P(62,63)=0.0d+0  
A12P(62,64)=0.0d+0



A12P(63,61)=0.0d+0  
A12P(63,62)=0.0d+0  
A12P(63,64)=0.0d+0  
A12P(63,65)=0.0d+0  
A12P(63,66)=0.0d+0  
A12P(63,67)=0.0d+0  
A12P(64,65)=0.0d+0  
A12P(64,66)=0.0d+0  
A12P(64,67)=0.0d+0  
A12P(64,62)=0.0d+0  
A12P(64,63)=0.0d+0  
A12P(65,63)=0.0d+0  
A12P(65,64)=0.0d+0  
A12P(65,66)=0.0d+0  
A12P(65,67)=0.0d+0  
A12P(66,63)=0.0d+0  
A12P(66,64)=0.0d+0  
A12P(66,65)=0.0d+0  
A12P(66,67)=0.0d+0  
A12P(67,63)=0.0d+0  
A12P(67,64)=0.0d+0  
A12P(67,65)=0.0d+0  
A12P(67,66)=0.0d+0  
A12P(68,57)=0.0d+0  
A12P(68,59)=0.0d+0  
A12P(68,60)=0.0d+0  
A12P(68,69)=0.0d+0  
A12P(68,70)=0.0d+0  
A12P(68,71)=0.0d+0  
A12P(68,72)=0.0d+0  
A12P(69,59)=0.0d+0  
A12P(69,68)=0.0d+0  
A12P(69,70)=0.0d+0  
A12P(70,59)=0.0d+0  
A12P(70,68)=0.0d+0  
A12P(70,69)=0.0d+0  
A12P(70,71)=0.0d+0  
A12P(70,72)=0.0d+0  
A12P(70,73)=0.0d+0  
A12P(70,81)=0.0d+0  
A12P(71,68)=0.0d+0  
A12P(71,70)=0.0d+0

A12P(71,72)=0.0d+0  
A12P(72,68)=0.0d+0  
A12P(72,70)=0.0d+0  
A12P(72,71)=0.0d+0  
A12P(72,81)=0.0d+0  
A12P(72,82)=0.0d+0  
A12P(72,83)=0.0d+0  
A12P(72,73)=0.0d+0  
A12P(72,74)=0.0d+0  
A12P(73,70)=0.0d+0  
A12P(73,72)=0.0d+0  
A12P(73,81)=0.0d+0  
A12P(73,74)=0.0d+0  
A12P(73,75)=0.0d+0  
A12P(74,72)=0.0d+0  
A12P(74,73)=0.0d+0  
A12P(74,75)=0.0d+0  
A12P(74,76)=0.0d+0  
A12P(75,73)=0.0d+0  
A12P(75,74)=0.0d+0  
A12P(75,76)=0.0d+0  
A12P(75,77)=0.0d+0  
A12P(76,75)=0.0d+0  
A12P(76,74)=0.0d+0  
A12P(76,77)=0.0d+0  
A12P(76,78)=0.0d+0  
A12P(76,79)=0.0d+0  
A12P(76,80)=0.0d+0  
A12P(77,75)=0.0d+0  
A12P(77,76)=0.0d+0  
A12P(77,78)=0.0d+0  
A12P(77,79)=0.0d+0  
A12P(77,80)=0.0d+0  
A12P(78,76)=0.0d+0  
A12P(78,77)=0.0d+0  
A12P(78,79)=0.0d+0  
A12P(78,80)=0.0d+0  
A12P(79,76)=0.0d+0  
A12P(79,77)=0.0d+0  
A12P(79,78)=0.0d+0  
A12P(79,80)=0.0d+0  
A12P(80,79)=0.0d+0

A12P(80,76)=0.0d+0  
A12P(80,77)=0.0d+0  
A12P(80,78)=0.0d+0  
A12P(81,70)=0.0d+0  
A12P(81,72)=0.0d+0  
A12P(81,73)=0.0d+0  
A12P(81,82)=0.0d+0  
A12P(81,83)=0.0d+0  
A12P(81,84)=0.0d+0  
A12P(82,72)=0.0d+0  
A12P(82,81)=0.0d+0  
A12P(82,83)=0.0d+0  
A12P(82,84)=0.0d+0  
A12P(83,81)=0.0d+0  
A12P(83,82)=0.0d+0  
A12P(83,72)=0.0d+0  
A12P(83,84)=0.0d+0  
A12P(83,85)=0.0d+0  
A12P(83,86)=0.0d+0  
A12P(83,94)=0.0d+0  
A12P(84,81)=0.0d+0  
A12P(84,83)=0.0d+0  
A12P(84,85)=0.0d+0  
A12P(85,81)=0.0d+0  
A12P(85,83)=0.0d+0  
A12P(85,84)=0.0d+0  
A12P(85,86)=0.0d+0  
A12P(85,87)=0.0d+0  
A12P(85,94)=0.0d+0  
A12P(85,95)=0.0d+0  
A12P(86,87)=0.0d+0  
A12P(86,88)=0.0d+0  
A12P(86,83)=0.0d+0  
A12P(86,85)=0.0d+0  
A12P(86,94)=0.0d+0  
A12P(87,88)=0.0d+0  
A12P(87,89)=0.0d+0  
A12P(87,85)=0.0d+0  
A12P(87,86)=0.0d+0  
A12P(88,86)=0.0d+0  
A12P(88,87)=0.0d+0  
A12P(88,89)=0.0d+0

A12P(88,90)=0.0d+0  
A12P(89,87)=0.0d+0  
A12P(89,88)=0.0d+0  
A12P(89,90)=0.0d+0  
A12P(89,91)=0.0d+0  
A12P(89,92)=0.0d+0  
A12P(89,93)=0.0d+0  
A12P(90,88)=0.0d+0  
A12P(90,89)=0.0d+0  
A12P(90,91)=0.0d+0  
A12P(90,92)=0.0d+0  
A12P(90,93)=0.0d+0  
A12P(91,89)=0.0d+0  
A12P(91,90)=0.0d+0  
A12P(91,92)=0.0d+0  
A12P(91,93)=0.0d+0  
A12P(92,90)=0.0d+0  
A12P(92,91)=0.0d+0  
A12P(92,93)=0.0d+0  
A12P(92,89)=0.0d+0  
A12P(93,89)=0.0d+0  
A12P(93,90)=0.0d+0  
A12P(93,91)=0.0d+0  
A12P(93,92)=0.0d+0  
A12P(94,95)=0.0d+0  
A12P(94,85)=0.0d+0  
A12P(94,86)=0.0d+0  
A12P(95,94)=0.0d+0  
A12P(95,85)=0.0d+0

\*\*\*\*\*

C6P(1,2)=0.0d+0  
C6P(1,3)=0.0d+0  
C6P(1,4)=0.0d+0  
C6P(1,5)=0.0d+0  
C6P(2,1)=0.0d+0  
C6P(2,3)=0.0d+0  
C6P(3,1)=0.0d+0  
C6P(3,2)=0.0d+0  
C6P(3,4)=0.0d+0  
C6P(3,5)=0.0d+0  
C6P(3,6)=0.0d+0  
C6P(3,7)=0.0d+0

C6P(4,1)=0.0d+0  
C6P(4,3)=0.0d+0  
C6P(4,5)=0.0d+0  
C6P(5,1)=0.0d+0  
C6P(5,3)=0.0d+0  
C6P(5,4)=0.0d+0  
C6P(5,6)=0.0d+0  
C6P(5,7)=0.0d+0  
C6P(5,8)=0.0d+0  
C6P(5,9)=0.0d+0  
C6P(6,3)=0.0d+0  
C6P(6,5)=0.0d+0  
C6P(6,7)=0.0d+0  
C6P(7,3)=0.0d+0  
C6P(7,5)=0.0d+0  
C6P(7,6)=0.0d+0  
C6P(7,8)=0.0d+0  
C6P(7,9)=0.0d+0  
C6P(7,10)=0.0d+0  
C6P(7,11)=0.0d+0  
C6P(8,5)=0.0d+0  
C6P(8,7)=0.0d+0  
C6P(8,9)=0.0d+0  
C6P(9,5)=0.0d+0  
C6P(9,7)=0.0d+0  
C6P(9,8)=0.0d+0  
C6P(9,10)=0.0d+0  
C6P(9,11)=0.0d+0  
C6P(9,12)=0.0d+0  
C6P(9,13)=0.0d+0  
C6P(10,9)=0.0d+0  
C6P(10,7)=0.0d+0  
C6P(10,11)=0.0d+0  
C6P(11,7)=0.0d+0  
C6P(11,9)=0.0d+0  
C6P(11,10)=0.0d+0  
C6P(11,12)=0.0d+0  
C6P(11,13)=0.0d+0  
C6P(11,14)=0.0d+0  
C6P(11,15)=0.0d+0  
C6P(12,11)=0.0d+0  
C6P(12,9)=0.0d+0

C6P(12,13)=0.0d+0  
C6P(13,9)=0.0d+0  
C6P(13,12)=0.0d+0  
C6P(13,11)=0.0d+0  
C6P(13,14)=0.0d+0  
C6P(13,15)=0.0d+0  
C6P(13,16)=0.0d+0  
C6P(13,17)=0.0d+0  
C6P(14,11)=0.0d+0  
C6P(14,13)=0.0d+0  
C6P(14,15)=0.0d+0  
C6P(15,11)=0.0d+0  
C6P(15,13)=0.0d+0  
C6P(15,14)=0.0d+0  
C6P(15,16)=0.0d+0  
C6P(15,17)=0.0d+0  
C6P(15,18)=0.0d+0  
C6P(15,19)=0.0d+0  
C6P(16,13)=0.0d+0  
C6P(16,15)=0.0d+0  
C6P(16,17)=0.0d+0  
C6P(17,13)=0.0d+0  
C6P(17,15)=0.0d+0  
C6P(17,16)=0.0d+0  
C6P(17,18)=0.0d+0  
C6P(17,19)=0.0d+0  
C6P(17,20)=0.0d+0  
C6P(17,21)=0.0d+0  
C6P(18,15)=0.0d+0  
C6P(18,17)=0.0d+0  
C6P(18,19)=0.0d+0  
C6P(19,15)=0.0d+0  
C6P(19,17)=0.0d+0  
C6P(19,18)=0.0d+0  
C6P(19,20)=0.0d+0  
C6P(19,21)=0.0d+0  
C6P(19,22)=0.0d+0  
C6P(19,23)=0.0d+0  
C6P(20,17)=0.0d+0  
C6P(20,19)=0.0d+0  
C6P(20,21)=0.0d+0  
C6P(21,17)=0.0d+0

C6P(21,19)=0.0d+0  
C6P(21,20)=0.0d+0  
C6P(21,22)=0.0d+0  
C6P(21,23)=0.0d+0  
C6P(21,24)=0.0d+0  
C6P(21,25)=0.0d+0  
C6P(22,19)=0.0d+0  
C6P(22,21)=0.0d+0  
C6P(22,23)=0.0d+0  
C6P(23,19)=0.0d+0  
C6P(23,22)=0.0d+0  
C6P(23,21)=0.0d+0  
C6P(23,24)=0.0d+0  
C6P(23,25)=0.0d+0  
C6P(23,26)=0.0d+0  
C6P(23,27)=0.0d+0  
C6P(24,21)=0.0d+0  
C6P(24,23)=0.0d+0  
C6P(24,25)=0.0d+0  
C6P(25,21)=0.0d+0  
C6P(25,23)=0.0d+0  
C6P(25,24)=0.0d+0  
C6P(25,26)=0.0d+0  
C6P(25,27)=0.0d+0  
C6P(25,28)=0.0d+0  
C6P(25,29)=0.0d+0  
C6P(26,23)=0.0d+0  
C6P(26,25)=0.0d+0  
C6P(26,27)=0.0d+0  
C6P(27,23)=0.0d+0  
C6P(27,25)=0.0d+0  
C6P(27,26)=0.0d+0  
C6P(27,28)=0.0d+0  
C6P(27,29)=0.0d+0  
C6P(27,30)=0.0d+0  
C6P(27,31)=0.0d+0  
C6P(28,25)=0.0d+0  
C6P(28,27)=0.0d+0  
C6P(28,29)=0.0d+0  
C6P(29,25)=0.0d+0  
C6P(29,27)=0.0d+0  
C6P(29,28)=0.0d+0

C6P(29,30)=0.0d+0  
C6P(29,31)=0.0d+0  
C6P(29,32)=0.0d+0  
C6P(29,33)=0.0d+0  
C6P(30,27)=0.0d+0  
C6P(30,29)=0.0d+0  
C6P(30,31)=0.0d+0  
C6P(31,27)=0.0d+0  
C6P(31,29)=0.0d+0  
C6P(31,30)=0.0d+0  
C6P(31,32)=0.0d+0  
C6P(31,33)=0.0d+0  
C6P(31,34)=0.0d+0  
C6P(31,42)=0.0d+0  
C6P(32,29)=0.0d+0  
C6P(32,31)=0.0d+0  
C6P(32,33)=0.0d+0  
C6P(33,29)=0.0d+0  
C6P(33,31)=0.0d+0  
C6P(33,32)=0.0d+0  
C6P(33,34)=0.0d+0  
C6P(33,35)=0.0d+0  
C6P(33,42)=0.0d+0  
C6P(33,43)=0.0d+0  
C6P(33,44)=0.0d+0  
C6P(34,33)=0.0d+0  
C6P(34,31)=0.0d+0  
C6P(34,42)=0.0d+0  
C6P(34,35)=0.0d+0  
C6P(34,36)=0.0d+0  
C6P(35,33)=0.0d+0  
C6P(35,34)=0.0d+0  
C6P(35,36)=0.0d+0  
C6P(35,37)=0.0d+0  
C6P(36,34)=0.0d+0  
C6P(36,35)=0.0d+0  
C6P(36,37)=0.0d+0  
C6P(36,38)=0.0d+0  
C6P(37,35)=0.0d+0  
C6P(37,36)=0.0d+0  
C6P(37,38)=0.0d+0  
C6P(37,39)=0.0d+0



C6P(37,40)=0.0d+0  
C6P(38,36)=0.0d+0  
C6P(38,37)=0.0d+0  
C6P(38,39)=0.0d+0  
C6P(38,40)=0.0d+0  
C6P(38,41)=0.0d+0  
C6P(39,37)=0.0d+0  
C6P(39,38)=0.0d+0  
C6P(39,40)=0.0d+0  
C6P(39,41)=0.0d+0  
C6P(40,37)=0.0d+0  
C6P(40,38)=0.0d+0  
C6P(40,39)=0.0d+0  
C6P(40,41)=0.0d+0  
C6P(41,37)=0.0d+0  
C6P(41,38)=0.0d+0  
C6P(41,39)=0.0d+0  
C6P(41,40)=0.0d+0  
C6P(42,31)=0.0d+0  
C6P(42,33)=0.0d+0  
C6P(42,34)=0.0d+0  
C6P(42,43)=0.0d+0  
C6P(42,44)=0.0d+0  
C6P(42,45)=0.0d+0  
C6P(42,46)=0.0d+0  
C6P(43,33)=0.0d+0  
C6P(43,42)=0.0d+0  
C6P(43,44)=0.0d+0  
C6P(44,33)=0.0d+0  
C6P(44,42)=0.0d+0  
C6P(44,43)=0.0d+0  
C6P(44,45)=0.0d+0  
C6P(44,46)=0.0d+0  
C6P(44,47)=0.0d+0  
C6P(44,55)=0.0d+0  
C6P(45,42)=0.0d+0  
C6P(45,44)=0.0d+0  
C6P(45,46)=0.0d+0  
C6P(46,42)=0.0d+0  
C6P(46,44)=0.0d+0  
C6P(46,45)=0.0d+0  
C6P(46,47)=0.0d+0

C6P(46,48)=0.0d+0  
C6P(46,55)=0.0d+0  
C6P(46,56)=0.0d+0  
C6P(46,57)=0.0d+0  
C6P(47,44)=0.0d+0  
C6P(47,46)=0.0d+0  
C6P(47,48)=0.0d+0  
C6P(47,49)=0.0d+0  
C6P(47,55)=0.0d+0  
C6P(48,46)=0.0d+0  
C6P(48,47)=0.0d+0  
C6P(48,49)=0.0d+0  
C6P(48,50)=0.0d+0  
C6P(49,47)=0.0d+0  
C6P(49,48)=0.0d+0  
C6P(49,50)=0.0d+0  
C6P(49,51)=0.0d+0  
C6P(50,49)=0.0d+0  
C6P(50,48)=0.0d+0  
C6P(50,51)=0.0d+0  
C6P(50,52)=0.0d+0  
C6P(50,53)=0.0d+0  
C6P(51,49)=0.0d+0  
C6P(51,50)=0.0d+0  
C6P(51,52)=0.0d+0  
C6P(51,53)=0.0d+0  
C6P(51,54)=0.0d+0  
C6P(52,50)=0.0d+0  
C6P(52,51)=0.0d+0  
C6P(52,53)=0.0d+0  
C6P(52,54)=0.0d+0  
C6P(53,50)=0.0d+0  
C6P(53,51)=0.0d+0  
C6P(53,52)=0.0d+0  
C6P(53,54)=0.0d+0  
C6P(54,50)=0.0d+0  
C6P(54,51)=0.0d+0  
C6P(54,52)=0.0d+0  
C6P(54,53)=0.0d+0  
C6P(55,44)=0.0d+0  
C6P(55,46)=0.0d+0  
C6P(55,47)=0.0d+0

C6P(55,56)=0.0d+0  
C6P(55,57)=0.0d+0  
C6P(55,58)=0.0d+0  
C6P(56,46)=0.0d+0  
C6P(56,55)=0.0d+0  
C6P(56,57)=0.0d+0  
C6P(57,46)=0.0d+0  
C6P(57,55)=0.0d+0  
C6P(57,56)=0.0d+0  
C6P(57,58)=0.0d+0  
C6P(57,59)=0.0d+0  
C6P(57,60)=0.0d+0  
C6P(57,68)=0.0d+0  
C6P(58,55)=0.0d+0  
C6P(58,57)=0.0d+0  
C6P(58,59)=0.0d+0  
C6P(59,55)=0.0d+0  
C6P(59,57)=0.0d+0  
C6P(59,58)=0.0d+0  
C6P(59,60)=0.0d+0  
C6P(59,61)=0.0d+0  
C6P(59,68)=0.0d+0  
C6P(59,69)=0.0d+0  
C6P(59,70)=0.0d+0  
C6P(60,57)=0.0d+0  
C6P(60,59)=0.0d+0  
C6P(60,68)=0.0d+0  
C6P(60,61)=0.0d+0  
C6P(60,62)=0.0d+0  
C6P(61,59)=0.0d+0  
C6P(61,60)=0.0d+0  
C6P(61,62)=0.0d+0  
C6P(61,63)=0.0d+0  
C6P(62,60)=0.0d+0  
C6P(62,61)=0.0d+0  
C6P(62,63)=0.0d+0  
C6P(62,64)=0.0d+0  
C6P(63,61)=0.0d+0  
C6P(63,62)=0.0d+0  
C6P(63,64)=0.0d+0  
C6P(63,65)=0.0d+0  
C6P(63,66)=0.0d+0

C6P(63,67)=0.0d+0  
C6P(64,65)=0.0d+0  
C6P(64,66)=0.0d+0  
C6P(64,67)=0.0d+0  
C6P(64,62)=0.0d+0  
C6P(64,63)=0.0d+0  
C6P(65,63)=0.0d+0  
C6P(65,64)=0.0d+0  
C6P(65,66)=0.0d+0  
C6P(65,67)=0.0d+0  
C6P(66,63)=0.0d+0  
C6P(66,64)=0.0d+0  
C6P(66,65)=0.0d+0  
C6P(66,67)=0.0d+0  
C6P(67,63)=0.0d+0  
C6P(67,64)=0.0d+0  
C6P(67,65)=0.0d+0  
C6P(67,66)=0.0d+0  
C6P(68,57)=0.0d+0  
C6P(68,59)=0.0d+0  
C6P(68,60)=0.0d+0  
C6P(68,69)=0.0d+0  
C6P(68,70)=0.0d+0  
C6P(68,71)=0.0d+0  
C6P(68,72)=0.0d+0  
C6P(69,59)=0.0d+0  
C6P(69,68)=0.0d+0  
C6P(69,70)=0.0d+0  
C6P(70,59)=0.0d+0  
C6P(70,68)=0.0d+0  
C6P(70,69)=0.0d+0  
C6P(70,71)=0.0d+0  
C6P(70,72)=0.0d+0  
C6P(70,73)=0.0d+0  
C6P(70,81)=0.0d+0  
C6P(71,68)=0.0d+0  
C6P(71,70)=0.0d+0  
C6P(71,72)=0.0d+0  
C6P(72,68)=0.0d+0  
C6P(72,70)=0.0d+0  
C6P(72,71)=0.0d+0  
C6P(72,81)=0.0d+0

C6P(72,82)=0.0d+0  
C6P(72,83)=0.0d+0  
C6P(72,73)=0.0d+0  
C6P(72,74)=0.0d+0  
C6P(73,70)=0.0d+0  
C6P(73,72)=0.0d+0  
C6P(73,81)=0.0d+0  
C6P(73,74)=0.0d+0  
C6P(73,75)=0.0d+0  
C6P(74,72)=0.0d+0  
C6P(74,73)=0.0d+0  
C6P(74,75)=0.0d+0  
C6P(74,76)=0.0d+0  
C6P(75,73)=0.0d+0  
C6P(75,74)=0.0d+0  
C6P(75,76)=0.0d+0  
C6P(75,77)=0.0d+0  
C6P(76,75)=0.0d+0  
C6P(76,74)=0.0d+0  
C6P(76,77)=0.0d+0  
C6P(76,78)=0.0d+0  
C6P(76,79)=0.0d+0  
C6P(76,80)=0.0d+0  
C6P(77,75)=0.0d+0  
C6P(77,76)=0.0d+0  
C6P(77,78)=0.0d+0  
C6P(77,79)=0.0d+0  
C6P(77,80)=0.0d+0  
C6P(78,76)=0.0d+0  
C6P(78,77)=0.0d+0  
C6P(78,79)=0.0d+0  
C6P(78,80)=0.0d+0  
C6P(79,76)=0.0d+0  
C6P(79,77)=0.0d+0  
C6P(79,78)=0.0d+0  
C6P(79,80)=0.0d+0  
C6P(80,79)=0.0d+0  
C6P(80,76)=0.0d+0  
C6P(80,77)=0.0d+0  
C6P(80,78)=0.0d+0  
C6P(81,70)=0.0d+0  
C6P(81,72)=0.0d+0

C6P(81,73)=0.0d+0  
C6P(81,82)=0.0d+0  
C6P(81,83)=0.0d+0  
C6P(81,84)=0.0d+0  
C6P(82,72)=0.0d+0  
C6P(82,81)=0.0d+0  
C6P(82,83)=0.0d+0  
C6P(82,84)=0.0d+0  
C6P(83,81)=0.0d+0  
C6P(83,82)=0.0d+0  
C6P(83,72)=0.0d+0  
C6P(83,84)=0.0d+0  
C6P(83,85)=0.0d+0  
C6P(83,86)=0.0d+0  
C6P(83,94)=0.0d+0  
C6P(84,81)=0.0d+0  
C6P(84,83)=0.0d+0  
C6P(84,85)=0.0d+0  
C6P(85,81)=0.0d+0  
C6P(85,83)=0.0d+0  
C6P(85,84)=0.0d+0  
C6P(85,86)=0.0d+0  
C6P(85,87)=0.0d+0  
C6P(85,94)=0.0d+0  
C6P(85,95)=0.0d+0  
C6P(86,87)=0.0d+0  
C6P(86,88)=0.0d+0  
C6P(86,83)=0.0d+0  
C6P(86,85)=0.0d+0  
C6P(86,94)=0.0d+0  
C6P(87,88)=0.0d+0  
C6P(87,89)=0.0d+0  
C6P(87,85)=0.0d+0  
C6P(87,86)=0.0d+0  
C6P(88,86)=0.0d+0  
C6P(88,87)=0.0d+0  
C6P(88,89)=0.0d+0  
C6P(88,90)=0.0d+0  
C6P(89,87)=0.0d+0  
C6P(89,88)=0.0d+0  
C6P(89,90)=0.0d+0  
C6P(89,91)=0.0d+0

```

C6P(89,92)=0.0d+0
C6P(89,93)=0.0d+0
C6P(90,88)=0.0d+0
C6P(90,89)=0.0d+0
C6P(90,91)=0.0d+0
C6P(90,92)=0.0d+0
C6P(90,93)=0.0d+0
C6P(91,89)=0.0d+0
C6P(91,90)=0.0d+0
C6P(91,92)=0.0d+0
C6P(91,93)=0.0d+0
C6P(92,90)=0.0d+0
C6P(92,91)=0.0d+0
C6P(92,93)=0.0d+0
C6P(92,89)=0.0d+0
C6P(93,89)=0.0d+0
C6P(93,90)=0.0d+0
C6P(93,91)=0.0d+0
C6P(93,92)=0.0d+0
C6P(94,95)=0.0d+0
C6P(94,85)=0.0d+0
C6P(94,86)=0.0d+0
C6P(95,94)=0.0d+0
C6P(95,85)=0.0d+0
do 204 M=1,NATOM-1
  do 205 I=M+1,NATOM
    DELX=PEPCORD1(I,1)-PEPCORD1(M,1)
    DELY=PEPCORD1(I,2)-PEPCORD1(M,2)
    DELZ=PEPCORD1(I,3)-PEPCORD1(M,3)
    D2PEP(M,I)=DELX*DELX+DELY*DELY+DELZ*DELZ
205  continue
204  continue
  PVPEP=0.d+0
  do 206 M=1,NATOM-1
    do 207 I=M+1,NATOM
      DSURF=dsqrt(D2PEP(M,I))
      D6=(DSURF)**6
      D12=(DSURF)**12
      PVPEP=PVPEP + A12P(M,I)/D12 - C6P(M,I)/D6
1    + QCONP(M,I)/DSURF
207  continue
206  continue

```

```

c      write(*,*)'Total Potential '
c      write(*,208) PVPEP
c 208   format(f15.3)
c *****
c ** Torsional Energy(VTOR) Calculations **
c *****
      VTOR=0.0d+0
      do i=1,RESNUM
        VTOR=VTOR + 0.5d+0*v1*(1.0d+0 - dcos(phi(i)))
1       + 0.5d+0*v2*(1.0d+0 - dcos(2.0d+0*phi(i)))
1       + 0.5d+0*v3*(1.0d+0 + dcos(3.0d+0*phi(i)))
      enddo
      do j=1,RESNUM
        VTOR=VTOR + 0.5d+0*v4*(1.0d+0 - dcos(psi(j)))
1       + 0.5d+0*v5*(1.0d+0 - dcos(2.0d+0*psi(j)))
1       + 0.5d+0*v6*(1.0d+0 + dcos(3.0d+0*psi(j)))
      enddo
C      write(*,*)'VTOR',VTOR
      CLOSE(10)
      configE=VTOR+PVPEP
c      write(*,*)'Total Conformational energy of peptide'
c      write(*,*)'ConfigE=',configE
      RETURN
      END
*****
** SUMBINPEP..subroutine which collects the data for the number of atoms in each bin. **
*****
      subroutine SUMBINPEP(PEPCORD1,LZ,NMOLW,NATOM,RXMAX2,
1 DELR,DELZ,D2WPEP,COSSUM,COSSQSUM,ZPEPSUM,NEND,CEND,GRPWSUM,
1 NEIBOURWSUM,HB,NHB,COM)
      implicit NONE
      integer J,N,M,BIN,NMOLW,NATOM,BINZ,NHB,NEND,CEND
      integer ZPEPSUM(2000,3)
      integer GRPWSUM(600,2,100),HB(100),NEIBOURWSUM(100)
      double precision D,DSQ,RXMAX2,DELR,DELZ,LZ,HALFLZ
      double precision DISTMINW
      double precision PEPCORD1(200,3),COM(3),COSSQ
      double precision D2WPEP(200,2000,3)
      double precision DELTX,DELTY,DELTAZ,COSSUM,COSSQSUM,LNC
C *****
C ** COLLECTION OF DATA.          **
C *****

```



```

HALFLZ=LZ/2.d+0
DISTMINW=3.3d+0
** Z profile of 1st, last, and COM positions
** 1st atom
  BINZ=idint((PEPCORD1(1,3)+HALFLZ)/DELZ)+1
  ZPEPSUM(BINZ,1)=ZPEPSUM(BINZ,1)+1
** COM
  BINZ=idint((COM(3)+HALFLZ)/DELZ)+1
  ZPEPSUM(BINZ,2)=ZPEPSUM(BINZ,2)+1
** last atom
  BINZ=idint((PEPCORD1(NATOM,3)+HALFLZ)/DELZ)+1
  ZPEPSUM(BINZ,3)=ZPEPSUM(BINZ,3)+1
**Order Parameter (from NEND to CEND)
  DELTX=PEPCORD1(NEND,1)-PEPCORD1(CEND,1)
  DELTY=PEPCORD1(NEND,2)-PEPCORD1(CEND,2)
  DELTZ=PEPCORD1(NEND,3)-PEPCORD1(CEND,3)
  LNC=dsqrt(DELTX*DELTX+DELTY*DELTY+DELTZ*DELTZ)
  COSSQ=DELTZ/LNC
  COSSUM=COSSUM+COSSQ
  COSSQ=COSSQ*COSSQ
  COSSQSUM=COSSQSUM+COSSQ
**PEPTIDE-WATER PCF and nearest neighbor
do 1500 M=1,NMOLW
do 1400 N=1,NHB
  DSQ=D2WPEP(HB(N),M,1)
  if(DSQ.LE.RXMAX2) then
    D=dsqrt(DSQ)
    if(D.LE.DISTMINW) then
      NEIBOURWSUM(N)=NEIBOURWSUM(N)+1
    end if
    BIN=idint(D/DELR-0.5d+0)+1
    GRPWSUM(BIN,1,N)=GRPWSUM(BIN,1,N)+1
  end if
do 1410 J=2,3
  DSQ=D2WPEP(HB(N),M,J)
  if(DSQ.LE.RXMAX2) then
    D=dsqrt(DSQ)
    BIN=idint(D/DELR-0.5d+0)+1
    GRPWSUM(BIN,2,N)=GRPWSUM(BIN,2,N)+1
  end if
1410  continue
1400  continue

```

1500 continue

return

end

\*\*\*\*\*

\*\* TORSION ..... subroutine to give the torsional rotation by changing coordinates of the peptide atoms

\*\*\*\*\*

subroutine TORSION(NATOM,PEPCORD1,maxpsi,maxphi,phi,psi,

& COM,AW,RESNUM,ISEED)

integer ISEED,NATOM,RESNUM

integer k1,k2,k3,n3,n4,k4,N

double precision PEPCORD1(200,3)

double precision COM(3),CM(3),AWS,AW(200)

double precision RANDOM

double precision u(20,3),s(3),w(3),x(3),maxpsi,cpsi(20)

double precision phi(20),psi(20),magu(20),magv(20)

double precision v(20,3),t(3),y(3),z(3),maxphi,cphi(20)

c u vector is to be used as reference vector for psi angle (about Calpha - C) rotation

c there are 8 Calpha - C bonds, so 8 u vectors

do I=1,3

CM(I)=0.0D+0

enddo

AWS=0.0D+0

do i=1,RESNUM

cpsi(i)=(2.0d+0\*RANDOM(ISEED)-1.0d+0)\*maxpsi

cphi(i)=(2.0d+0\*RANDOM(ISEED)-1.0d+0)\*maxphi

enddo

do i=1,RESNUM

phi(i)=phi(i) + cphi(i)

psi(i)=psi(i) + cpsi(i)

enddo

do 45 k1=1,RESNUM

if (k1 .le. 5) then

u(k1,1)=PEPCORD1(6\*k1-1,1) - PEPCORD1(6\*k1-3,1)

u(k1,2)=PEPCORD1(6\*k1-1,2) - PEPCORD1(6\*k1-3,2)

u(k1,3)=PEPCORD1(6\*k1-1,3) - PEPCORD1(6\*k1-3,3)

else

u(k1,1)=PEPCORD1((13\*(k1-3))+3,1) - PEPCORD1((13\*(k1-3))-6,1)

u(k1,2)=PEPCORD1((13\*(k1-3))+3,2) - PEPCORD1((13\*(k1-3))-6,2)

u(k1,3)=PEPCORD1((13\*(k1-3))+3,3) - PEPCORD1((13\*(k1-3))-6,3)

endif

c magnitude of reference vector u

magu(k1)=sqrt((u(k1,1))\*\*2 + (u(k1,2))\*\*2 + (u(k1,3))\*\*2)

```

45  continue
c v vector is to be used as reference vector for phi angle (about N - Calpha) rotation
c there are 8 N - Calpha bonds, so 8 v vectors
  do 47 k2=1,RESNUM
  if (k2 .le. 5) then
    v(k2,1)=PEPCORD1(6*k2-3,1) - PEPCORD1(6*k2-5,1)
    v(k2,2)=PEPCORD1(6*k2-3,2) - PEPCORD1(6*k2-5,2)
    v(k2,3)=PEPCORD1(6*k2-3,3) - PEPCORD1(6*k2-5,3)
  else
    v(k2,1)=PEPCORD1((13*(k2-3))-6,1) - PEPCORD1((13*(k2-3))-8,1)
    v(k2,2)=PEPCORD1((13*(k2-3))-6,2) - PEPCORD1((13*(k2-3))-8,2)
    v(k2,3)=PEPCORD1((13*(k2-3))-6,3) - PEPCORD1((13*(k2-3))-8,3)
  endif
c magnitude of reference vector v
  magv(k2)=sqrt((v(k2,1))**2 + (v(k2,2))**2 + (v(k2,3))**2)
47  continue
c FOR PSI (Calpha-C) rotation calculations for atom coordinates when the right
c bond C - N is rotated, all the atoms on the right are mapped
  do 49 k3=1,RESNUM
  IF (K3 .LE. 5) THEN
    n3=6*k3-1
  else
    n3=13*(k3-3)+3
  endif
  do 50 N=n3+1,NATOM
    s(1)=PEPCORD1(N,1)-PEPCORD1(n3,1)
    s(2)=PEPCORD1(N,2)-PEPCORD1(n3,2)
    s(3)=PEPCORD1(N,3)-PEPCORD1(n3,3)
    w(1)=(1.0d+0/magu(k3))*(u(k3,2)*s(3) - u(k3,3)*s(2) )
    w(2)=(1.0d+0/magu(k3))*(u(k3,3)*s(1) - u(k3,1)*s(3) )
    w(3)=(1.0d+0/magu(k3))*(u(k3,1)*s(2) - u(k3,2)*s(1) )
    x(1)=(1.0d+0/magu(k3))*(u(k3,2)*w(3) - u(k3,3)*w(2) )
    x(2)=(1.0d+0/magu(k3))*(u(k3,3)*w(1) - u(k3,1)*w(3) )
    x(3)=(1.0d+0/magu(k3))*(u(k3,1)*w(2) - u(k3,2)*w(1) )
    PEPCORD1(N,1)=PEPCORD1(N,1) + w(1)*dsin(cpsi(k3))
1   + x(1)*(1.0d+0-dcos(cpsi(k3)))
    PEPCORD1(N,2)=PEPCORD1(N,2) + w(2)*dsin(cpsi(k3))
1   + x(2)*(1.0d+0-dcos(cpsi(k3)))
    PEPCORD1(N,3)=PEPCORD1(N,3) + w(3)*dsin(cpsi(k3))
1   + x(3)*(1.0d+0-dcos(cpsi(k3)))
50  continue
49  continue

```

c calculations for atom coordinates when the right bond C - N is rotated, all the atoms  
c on the right are mapped

```

do 51 k4=1,RESNUM
  IF (k4 .LE. 5) THEN
    n4=6*k4-3
  else
    n4=13*(k4-3)-6
  endif
do 52 I=n4+1,NATOM
  t(1)=PEPCORD1(I,1)-PEPCORD1(n4,1)
  t(2)=PEPCORD1(I,2)-PEPCORD1(n4,2)
  t(3)=PEPCORD1(I,3)-PEPCORD1(n4,3)
  y(1)=(1.0d+0/magv(k4))*(v(k4,2)*t(3) - v(k4,3)*t(2) )
  y(2)=(1.0d+0/magv(k4))*(v(k4,3)*t(1) - v(k4,1)*t(3) )
  y(3)=(1.0d+0/magv(k4))*(v(k4,1)*t(2) - v(k4,2)*t(1) )
  z(1)=(1.0d+0/magv(k4))*(v(k4,2)*y(3) - v(k4,3)*y(2) )
  z(2)=(1.0d+0/magv(k4))*(v(k4,3)*y(1) - v(k4,1)*y(3) )
  z(3)=(1.0d+0/magv(k4))*(v(k4,1)*y(2) - v(k4,2)*y(1) )
  PEPCORD1(I,1)=PEPCORD1(I,1) + y(1)*dsin(cphi(k4))
1  + z(1)*(1d+0-dcos(cphi(k4)))
  PEPCORD1(I,2)=PEPCORD1(I,2) + y(2)*dsin(cphi(k4))
1  + z(2)*(1d+0-dcos(cphi(k4)))
  PEPCORD1(I,3)=PEPCORD1(I,3) + y(3)*dsin(cphi(k4))
1  + z(3)*(1d+0-dcos(cphi(k4)))
52 continue
51 continue
do I=1,3
do M=1,NATOM
  CM(I)=CM(I)+PEPCORD1(M,I)*AW(M)
enddo
enddo
do M=1,NATOM
  AWS=AWS+AW(M)
enddo
do I=1,3
  COM(I)=CM(I)/AWS
enddo
return
end

```

## APPENDIX B

### INSTRUCTIONS TO RUN THE EXPLICIT MODEL CODE

The code 2pws.f simulates two identical peptides in bulk water (of explicit molecules) with two charged surfaces along the xy-plane on opposite sides of the box. The simulation is run by splitting the required total number of MC simulation steps in sets of no more than 40000 MC steps each. After completing a set, the input file required for the next set is generated and the simulation is continued. The simulation uses an input file for a run set, which is associated with system configuration files and output files.

**Input data file (inputx.inp):**

Each input data file describes the variables and file names for a simulation run set. They are named as 'inputx.inp', where 'x' denotes the serial number of the simulation run set, like 'input1.inp' for the first set of simulation run. The contents in an input data file (say inputx.inp) are described below,

0 or 1 or 2: Status of simulation set run (0-first run, 1-continuation, 2-Production).

'CFGx-1.out': Input configuration file from previous run set (x-1).

'CFGx.out': Output configuration file for next run set (x+1).

'outx.out': Output result file for the running simulation set (x).

'SYSTEM0.pdb': RasMol visualization file for the initial system configuration of the running simulation set (read only for first run, i.e. if x=1).

'SYSTEMx.pdb': RasMol visualization file for the final system configuration of the running simulation set (x).

NMOLW: Number of explicit water molecules in the simulation box, calculated to maintain 1 g/cm<sup>3</sup> of water density inside the simulation box.

NMCS: Number of Monte Carlo Steps in the simulation run set.

TEMP: Temperature of simulation run (selected 298K).

NSAVE: Simulation step interval between temporary variable storage (selected 500).

NADJUST: Simulation step interval between adjusting limits of molecular movement (selected 25).

'SURF.INP': Surface configuration file carrying data on the number of surface atoms, their coordinates and simulation box dimensions.

QSURF: Charge on the surface (selected 0.3 e)

'A5-I5.cod': Peptide configuration file generated by processing the data file obtained with ALCHEMY using GROMOS software, to put all the hydrogens in place which were not present in the original file. With all the hydrogens in place (H on N of –CONH and H of side chains of lysine) it is then processed again to get the center-of-mass at the origin to begin with. This final file is named as 'A5-L5.cod'. We have five ALA residues and five LYS residues in the peptide, with each LYS residue carrying a charge of '+1' while ALA is neutral.

VPEP: Peptide volume.

RESNUM: Number of peptide residues (10 for the selected peptide model).

For the first run set, the user needs to generate the input file (input1.inp) and the configuration files (A5-L5.cod and SURF.INP) for the selected simulation set-up, then for all other input files (for subsequent run sets) the user needs to only change names of the input configuration file, output configuration file, output file and RasMol file. Thus input data files for all simulation run sets required were created. Then before each

simulation run set (say set x), the only update needed in the FOTRAN code is to change the input file name (to inputx.inp); then compile the code and run.



APPENDIX C

IMPLICIT MODEL SIMULATION  
PROGRAM CODE

c\*Corrected Amol-Fuzz PEPTIDE\_IMP-WATER\_SURFACE code\*\*\*\*\*RUN:2.2(Correction for random initialization for peptide positions) MAX NMOLP = 50; MAX NATOM = 200; MAX NMCS = 50000;MAX RESNUM =20 MAX NSURF = 1000 Corrections to NCA and NCB variables\*\*\*\*\*

```

program main
implicit NONE
external INCONFIG,NRGTOT
external RANDOM
external PDBOUT
external DISTPEPSF,NRGPEPSF
external DISTPP,NRGPP
external MOVEPEP,pepenrg,HPenrg
external SUMBINPEP,TORSION,UHP_est,HBenrg_intra,HBenrg_inter
*****

integer I,ISEED,M,K,N,NMOLP,P,a,b
integer NADJUST,NMCS,NMCSACC,NMCSOLD
integer NMCSTMP,NPTS,NSAVE,RUNSTAT,STEP
integer NSURF
integer NBINZ,NCA(20),NCB(20)
integer NATOM,NEND,CEND
integer ATOM(200),AA(200),GROUP(200)
integer ACCTOR,ACCPEP
integer ZPEPSUM(50,2000,3)
integer RESNUM
double precision ACCPEPACC,ACCPEPTMP,AVGACCPEP
double precision AVGACCTOR,ACCTORACC,ACCTORTMP
double precision DELZ,CMM(3),r
double precision AVGNRG
double precision DELR,DELV,ESUM,ESUMACC,ESUMACCTMP
double precision LMIN,PI,kB
double precision LX,LY,LZ
double precision QFACT,RANDOM,RXMAX,TEMP
double precision V,VERR,VT,VTN
double precision VERRSUM
double precision QSURF,RSURF(1000,3)
double precision AII(2),CII(2),Q(2)
double precision VTOT(0:50001),VTOTPT(100)
double precision PEPCORD(50,200,3),APEP(200),CPEP(200)
double precision QPEP(200),AW(200)
double precision phi(50,20),psi(50,20),COM(50,3)
double precision v1,v2,v3,v4,v5,v6

```

```

double precision A12PS(200,2),C6PS(200,2),QCONPS(200,2)
double precision A12PP(200,200),C6PP(200,200),QCONPP(200,200)
double precision UPEPMSFNEW,UPEPMSFOLD(50)
double precision UPMPNEW,UPMPOLD(50)
double precision PEPCORDOLD(200,3),COMOLD(3)
double precision phiold(20),psiold(20)
double precision PMCORD(200,3),COMM(3),phim(20),psim(20)
double precision UHPO,UHBO,UHBIO,UHPN,UHBN,UHBIN,UHPM,UHBM,UHBIM
double precision UHPA(50,20),UHPB(50,20),PUHPA(20),PUHPB(20)
double precision MXTPEP,MXRPEP,maxpsi,maxphi
double precision D2PEPSF1(200,1000),D2PEPSF2(200,1000)
double precision D2PMPEP(200,50,200)
double precision NEWPENRG,OLDPENRG(50)
double precision VPEP,PE,PENTOT,AWS,UPSF,UP,UPMSF,UPMP
double precision COSAVG(50),COSSQAVG(50),COSSQSUM(50),COSSUM(50)
double precision ZPEP(50,2000,3)
double precision ETED(50,50000),ETEDP(50,100)
double precision CM(50,50000),CMP(50,100)
double precision VPS(50,50000),VPEPT(50,50000)
double precision VPSPT(50,100),VPEPTPT(50,100)
double precision VPSS(50),VRESTI,VPEPTP(50),ETEDAVG(50),CMAVG(50)
double precision VREST,UPEPSFI,UPEPPI,UPEPSF,UPEPP
character CFGINP*15,CFGOUT*15,OUTPUT*15
character TITLE*50,PDB*15,PDBI*15
character SURFINP*15,SURFNAME*15
character PEP*15
parameter (PI=3.14159265d+0)
parameter (kB=1.985878d-3)
C *****
C ** INPUT DATA.                **
C *****

open(unit=5,file='input1.inp',status='unknown')
read(5,*)
read(5,*)
read(5,*) TITLE
read(5,100) RUNSTAT
read(5,*) CFGINP
read(5,*) CFGOUT
read(5,*) OUTPUT
read(5,*) PDB
if (RUNSTAT.eq.0) then
  read(5,*) PDBI

```

```

endif
read(5,110) NMOLP
if(NMOLP.gt.50) STOP 'Matrix size too small'
read(5,120) NMCS
if(NMCS.GT.50000) STOP 'Matrix size too small'
read(5,140) TEMP
read(5,*) NSAVE
read(5,*) NADJUST
read(5,*) SURFINP
read(5,*) QSURF
read(5,*) PEP
read(5,*) VPEP
read(5,*) RESNUM
100 format(i1)
110 format(i4)
120 format(i6)
140 format(f6.2)
    open(unit=88,file='phipsi',status='unknown')
    open(unit=7,file=SURFINP,status='unknown',form='formatted')
    read(7,*)
    read(7,*) NSURF
    if(NSURF.GT.1000) STOP 'Matrix size too small'
    do 105 N=1,NSURF
        read(7,222) (RSURF(N,I),I=1,3)
105 continue
222 format(f17.13,2x,f17.13,2x,f17.13)
    read(7,*) SURFNAME
    read(7,*) LX
    read(7,*) LY
    open(unit=9,file=OUTPUT,status='unknown',form='formatted')
C *****
C ** BOX DIMENSIONS & CONSTANTS FOR PCFs.    **
C *****
C**From the excel sheet calculations for NMOLP no of peptides**
    LZ=85.00D+0
C**a is rows along x-axis and b is rows along z-axis*****
    b=5
    r=dbl(NMOLP)/dbl(b)
    a=ceiling(r)
*****
    NPTS=1
    if(NMCS.gt.100) then

```

```

      NPTS=NMCS/100
    end if
    MXTPEP=0.005
    MXRPEP=0.008
    maxpsi=0.005
    maxphi=0.005
    QFACT=1.671d+5/TEMP
    NMCSTMP=0
    ESUMACCTMP=0.d+0
    NMCSACC=0
    ESUMACC=0.d+0
    ESUM=0.d+0
    LMIN=LX
    if(LY.LT.LMIN) then
      LMIN=LY
    end if
    if(LZ.LT.LMIN) then
      LMIN=LZ
    end if
    RXMAX=LMIN/2.0d+0
    DELR=0.05d+0
    DELZ=DELR
    NBINZ=idint(LZ/DELZ)
    ACCPEP=0
    ACCTOR=0
    open(unit=2,file=CFGOUT,status='unknown',form='formatted')
*****
***** Surface Parameters *****
*****
      AII(1)=6.344d+0
      CII(1)=10.152d+0
      AII(2)=AII(1)
      CII(2)=CII(1)
      Q(1)=QSURF
      Q(2)=-QSURF
C *****
C ** INITIAL CONFIGURATION AND ENERGY.          **
C ** RUNSTAT=0...cold start, call INCONFIG.      **
C ** RUNSTAT=1...warm start, before equilibrium, read in coords.**
C ** RUNSTAT=2...1st data collecting run        **
C ** RUNSTAT=3...collecting data (after 1st collecting run) **
C *****

```

```

if(RUNSTAT.eq.0)then
  call INCONFIG(LX,LY,LZ,NMOLP,NATOM,PEPCORD,COM,APEP,
& CPEP,QPEP,AW,AA,ATOM,GROUP,NEND,CEND,phi,psi,
& v1,v2,v3,v4,v5,v6,RESNUM,NCA,NCB)
  do 1006 P=1,NMOLP
    COSSUM(P)=0.d+0
    COSSQSUM(P)=0.d+0
    do 8027 N=1,NBINZ
      do 3026 I=1,3
        ZPEPSUM(P,N,I)=0
3026    continue
8027    continue
1006    continue
  else
    open(unit=1,file=CFGINP,status='unknown',form='formatted')
    rewind(1)
    read(1,*)
    read(1,*) NATOM
    do 35500 M=1,NATOM
      read(1,889) AA(M),ATOM(M),GROUP(M),APEP(M),CPEP(M),QPEP(M),
1    AW(M)
35500    continue
    do 35505 P=1,NMOLP
      do 35510 M=1,NATOM
        read(1,998) PEPCORD(P,M,1),PEPCORD(P,M,2),PEPCORD(P,M,3)
35510    continue
35505    continue
      do 35020 M=1,RESNUM
        read(1,23) NCA(M),NCB(M)
35020    continue
      do 35025 P=1,NMOLP
        read(1,777) (COM(P,I),I=1,3)
35025    continue
        read(1,*) NEND
        read(1,*) CEND
        do 35035 P=1,NMOLP
          do 35036 I=1,RESNUM
            read(1,1523)phi(P,I),psi(P,I)
35036    continue
35035    continue
          read(1,1524)v1,v2,v3,v4,v5,v6
889    format(5x,I3,1x,I3,1x,I4,1x,f8.3,f8.3,f8.3,f8.3)

```

```

998  format(3x,f8.3,f8.3,f8.3)
1523 format(5x,f8.3,5x,f8.3)
1524 format(f8.3,f8.3,f8.3,f8.3,f8.3,f8.3)
23   format(2x,i3,2x,i3)
777  format(f17.13,2x,f17.13,2x,f17.13)
      read(1,*) MXTPEP
      read(1,*) MXRPEP
      read(1,*) maxpsi
      read(1,*) maxphi
      if(RUNSTAT.ne.3)then
      do 1005 P=1,NMOLP
        COSSUM(P)=0.d+0
        COSSQSUM(P)=0.d+0
        do 1001 K=1,NBINZ
          do 1011 I=1,3
            ZPEPSUM(P,K,I)=0
1011      continue
1001      continue
1005      continue
      else
        read(1,*) NMCSACC
        read(1,*) ESUMACC
        read(1,*) ACCPEPACC
        read(1,*) ACCTORACC
        read(1,*)
        do 1003 P=1,NMOLP
          do 1004 K=1,NBINZ
            do 2478 I=1,3
              read(1,*) ZPEPSUM(P,K,I)
2478      continue
1004      continue
              read(1,*) COSSUM(P)
              read(1,*) COSSQSUM(P)
              read(1,*)
1003      continue
            end if
            close(1)
          endif
C *****
C ** PEPTIDE-SURFACE MODEL PARAMETERS.      **
C *****
      do 30172 M=1,NATOM

```

```

do 30182 I=1,2
  A12PS(M,I)=(APEP(M)*AII(I))**6/TEMP
  C6PS(M,I)=(CPEP(M)*CII(I))**3/TEMP
  QCONPS(M,I)=QFACT*QPEP(M)*Q(I)
30182 continue
30172 continue
C *****
C ** PEPTIDE-PEPTIDE MODEL PARAMETERS.      **
C *****

do 30173 M=1,NATOM
  do 30183 N=1,NATOM
    A12PP(M,N)=(APEP(M)*APEP(N))**6/TEMP
    C6PP(M,N)=(CPEP(M)*CPEP(N))**3/TEMP
    QCONPP(M,N)=QFACT*QPEP(M)*QPEP(N)
30183 continue
30173 continue

C*****INITIAL PEPTIDE ENERGY CALCULATION*****
  call NRGTOT(V,LX,LY,LZ,NMOLP,A12PS,C6PS,QCONPS,A12PP,
& C6PP,QCONPP,NSURF,RSURF,NATOM,PEPCORD,COM,UPEPSFI,
& UPEPMSFOLD,UPEPPI,UPMPOLD,VRESTI)
  call UHP_est(UHPA,UHPB,NMOLP,NATOM,RESNUM,PEPCORD,NCA,NCB,
& COM,LX,LY,LZ)
  PENTOT = 0.0d+0
  UHBO = 0.0d+0
  UHBIO = 0.0d+0
  UHPO = 0.0d+0
do 30174 P=1,NMOLP
  do 30176 M=1,NATOM
    do 30178 I=1,3
      PMCORD(M,I)=PEPCORD(P,M,I)
30178 continue
30176 continue
    do 30210 I=1,3
      COMM(I)=COM(P,I)
30210 continue
    do 30186 I=1,RESNUM
      phim(I)=phi(P,I)
      psim(I)=psi(P,I)
      PUHPA(I)=UHPA(P,I)
      PUHPB(I)=UHPB(P,I)
30186 continue

```



```

    call pepenrg(PE,APEP,CPEP,QPEP,PMCORD,NATOM,
1  phim,psim,v1,v2,v3,v4,v5,v6,RESNUM)
    write(*,*)'PEPTIDE  , OLDPENRG'
    write(*,*)P,PE
    PENTOT=PENTOT+PE
    OLDPENRG(P)=PE
    call HPenrg(UHPM,RESNUM,TEMP,PUHPA,PUHPB,PI,kB)
    UHPO = UHPO + UHPM
    call HBenrg_intra(UHBM,PUHPA,RESNUM,NATOM,GROUP,ATOM,
&  PMCORD,TEMP,COMM,LX,LY,LZ,PI,kB)
    UHBO = UHBO + UHBM
    call HBenrg_inter(UHBIM,PUHPA,UHPA,RESNUM,NMOLP,NATOM,
&  GROUP,ATOM,PMCORD,PEPCORD,TEMP,P,COMM,COM,LX,LY,LZ,PI,kB)
    UHBIO = UHBIO + UHBIM
30174 continue
    UHBIO = 0.5d+0*UHBIO
    VT = V+PENTOT+UHPO+UHBO+UHBIO
    VTOT(0)=VT
    do P=1,NMOLP
        VPSS(P)=UPEPMSFOLD(P)
        VPEPTP(P)=UPMPOLD(P)
    end do
    write(*,*)'V-ENERGY PENTOT  UHP   UHB   UHBI  '
    write(*,333)V,PENTOT,UHPO,UHBO,UHBIO
333  format(2x,f8.3,2x,f8.3,2x,f8.3,2x,f8.3,2x,f8.3)
    AVGACCPEP=0.5d+0
    AVGACCTOR=0.5d+0
    if (RUNSTAT.eq.0) then
        call PDBOUT(RSURF,PEPCORD,PDBI,NMOLP,NATOM,NSURF,AA,ATOM,GROUP)
    endif
C *****
C ** MC STEPS. each MC step consists of an attempted move for each and every molecule in the system. Every
NADJUST steps, maximum movements are adjusted for an acceptance rate of 0.50 Every NSAVE steps, configurations
and results are saved for recovery in case the job is killed. *
C *****
    do 10000 STEP=1,NMCS
        if(mod(STEP,NADJUST).eq.0) then
            AVGACCPEP=dble(ACCPEP)/(NMOLP*(STEP-1))
            AVGACCTOR=dble(ACCTOR)/(NMOLP*(STEP-1))
            if (AVGACCPEP.gt.0.50) then
                MXTPEP = MXTPEP * 1.1d+0
                MXRPEP = MXRPEP * 1.1d+0
            end if
        end if
    end do

```

```

else
  MXTPEP = MXTPEP / 1.1d+0
  MXRPEP = MXRPEP / 1.1d+0
end if
if (AVGACCTOR.gt.0.50) then
  maxpsi = maxpsi * 1.1d+0
  maxphi = maxphi * 1.1d+0
else
  maxpsi = maxpsi / 1.1d+0
  maxphi = maxphi / 1.1d+0
end if
if(mod(STEP,NSAVE).eq.0) then
  NMCSTMP=STEP-1
  ESUMACCTMP=ESUM
  ACCPEPTMP=ACCPEP
  ACCTORTMP=ACCTOR
  if(RUNSTAT.GE.2) then
    NMCSTMP=NMCSACC+STEP-1
    ESUMACCTMP=ESUMACC+ESUM
    ACCPEPTMP=ACCPEPACC+ACCPEP
    ACCTORTMP=ACCTORACC+ACCTOR
  end if
  open(unit=3,file='CFG.TMP',status='unknown',form=
&      'formatted')
  rewind(3)
  do 34000 M=1,NATOM
    write(3,889) AA(M),ATOM(M),GROUP(M),APEP(M),CPEP(M),
1      QPEP(M), AW(M)
34000  continue
  do 34005 P=1,NMOLP
    do 34010 M=1,NATOM
      write(3,998) PEPCORD(P,M,1),PEPCORD(P,M,2),PEPCORD(P,M,3)
34010  continue
34005  continue
    do 35030 M=1,RESNUM
      write(3,23) NCA(M),NCB(M)
35030  continue
    do 35039 P=1,NMOLP
      write(3,777) (COM(P,I),I=1,3)
35039  continue
      write(3,*) NEND
      write(3,*) CEND

```

```

do 35038 P=1,NMOLP
do I=1,RESNUM
write(3,1523)phi(P,I),psi(P,I)
enddo
35038 continue
write(3,1525)v1,v2,v3,v4,v5,v6
1525 format(f8.3,f8.3,f8.3,f8.3,f8.3,f8.3)
write(3,*) MXTPEP
write(3,*) MXRPEP
write(3,*) maxpsi
write(3,*) maxphi
write(3,*) NMCSTMP
write(3,*) ESUMACCTMP
write(3,*) ACCPEPTMP
write(3,*) ACCTORTMP
write(3,*) 'STEP= ',STEP
close(3)
end if
end if
***** peptide movement *****
**TRANSLATIONAL(x,y and z direction) AND ROTATIONAL MOVEMEMNT

**PEPTIDE MOVEMENT**
do 50050 P=1,NMOLP
do 50052 M=1,NATOM
do 50054 I=1,3
PEPCORDOLD(M,I)=PEPCORD(P,M,I)
PMCORD(M,I) = PEPCORD(P,M,I)
50054 continue
50052 continue
do 50056 I=1,3
COMOLD(I)=COM(P,I)
COMM(I) = COM(P,I)
50056 continue
call MOVEPEP(NATOM,PMCORD,COMM,MXTPEP,MXRPEP,
& LX,LY,LZ,ISEED)
do 50062 M=1,NATOM
do 50064 I=1,3
PEPCORD(P,M,I) = PMCORD(M,I)
50064 continue
50062 continue
do 50066 I=1,3

```

```

      COM(P,I)=COMM(I)
50066  continue
**TOTAL ENERGY CALCULATION**
      call UHP_est(UHPA,UHPB,NMOLP,NATOM,RESNUM,PEPCORD,NCA,NCB,
&      COM,LX,LY,LZ)
      UPSF = 0.0d+0
      UP = 0.0d+0
      PENTOT = 0.0d+0
      UHPN = 0.0d+0
      UHBN = 0.0d+0
      UHBIN = 0.0d+0
      do 50113 K=1,NMOLP
      do 50115 M=1,NATOM
      do 50117 I=1,3
          PMCORD(M,I) = PEPCORD(K,M,I)
50117  continue
50115  continue
      do 50121 I=1,3
          COMM(I) = COM(K,I)
50121  continue
      do 50123 N=1,RESNUM
          phim(N)=phi(K,N)
          psim(N)=psi(K,N)
          PUHPA(N)=UHPA(K,N)
          PUHPB(N)=UHPB(K,N)
50123  continue
      call DISTPEPSF(D2PEPSF1,D2PEPSF2,RSURF,NSURF,
&      PMCORD,NATOM,LX,LY,COMM)
      call NRGPEPSF(A12PS,C6PS,QCONPS,UPMSF,D2PEPSF1,
&      D2PEPSF2,NSURF,NATOM)
      UPSF = UPSF+UPMSF
      if (K.eq.P) then
          UPEPMSFNEW=UPMSF
      end if
      call DISTPP(D2PMPEP,PMCORD,PEPCORD,K,NMOLP,NATOM,LX,LY,
&      COM,COMM)
      call NRGPP(D2PMPEP,A12PP,C6PP,QCONPP,UPMP,K,NMOLP,NATOM)
      UP = UP+UPMP
      if (K.eq.P) then
          UPMPNEW=UPMP
      end if
      call pepenrg(PE,APEP,CPEP,QPEP,PMCORD,NATOM,

```

```

&      phim,psim,v1,v2,v3,v4,v5,v6,RESNUM)
PENTOT = PENTOT+PE
if (K.eq.P) then
  NEWPENRG=PE
end if
call HPenrg(UHPM,RESNUM,TEMP,PUHPA,PUHPB,PI,kB)
UHPN = UHPN+UHPM
call HBenrg_intra(UHBM,PUHPA,RESNUM,NATOM,GROUP,ATOM,
&      PMCORD,TEMP,COMM,LX,LY,LZ,PI,kB)
UHBN = UHBN + UHBM
call HBenrg_inter(UHBIM,PUHPA,UHPA,RESNUM,NMOLP,NATOM,GROUP
&      ,ATOM,PMCORD,PEPCORD,TEMP,K,COMM,COM,LX,LY,LZ,PI,kB)
UHBIN = UHBIN + UHBIM
50113  continue
UP = 0.5d+0*UP
UHBIN = 0.5d+0*UHBIN
VTN = UPSF+UP+PENTOT+ UHPN + UHBN + UHBIN
DELV = VTN - VT
if((DELV.ge.0.0d+0).and.(dexp(-DELV).lt.RANDOM(ISEED)))then
  do 52000 N=1,NATOM
    do 53000 I=1,3
      PEPCORD(P,N,I)=PEPCORDOLD(N,I)
53000  continue
52000  continue
    do 54100 I=1,3
      COM(P,I)=COMOLD(I)
54100  continue
    call SUMBINPEP(PEPCORD,P,LZ,NATOM,DELZ,COSSUM,COSSQSUM,
1      ZPEPSUM,NEND,CEND,COM)
  else
    VT=VTN
    VPSS(P)=VPSS(P)+UPEPMSFNEW-UPEPMSFOLD(P)
    VPEPTP(P)=VPEPTP(P)+UPMPNEW-UPMPOLD(P)
    ACCPEP=ACCPEP+1
    call SUMBINPEP(PEPCORD,P,LZ,NATOM,DELZ,COSSUM,
1      COSSQSUM,ZPEPSUM,NEND,CEND,COM)
    OLDPENRG(P)=NEWPENRG
    UPEPMSFOLD(P)=UPEPMSFNEW
    UPMPOLD(P)=UPMPNEW
    UHPO=UHPN
    UHBO=UHBN
    UHBIO=UHBIN

```

```

end if
50050 continue
*****TORSION MOVEMENT*****
do 54010 P=1,NMOLP
do 54020 M=1,NATOM
do 54030 I=1,3
PEPCORDOLD(M,I)=PEPCORD(P,M,I)
PMCORD(M,I)=PEPCORD(P,M,I)
54030 continue
54020 continue
do 54050 I=1,3
COMM(I)=COM(P,I)
COMOLD(I)=COM(P,I)
54050 continue
do 54040 I=1,RESNUM
phiold(I)=phi(P,I)
psiold(I)=psi(P,I)
phim(I)=phi(P,I)
psim(I)=psi(P,I)
54040 continue
**PEPTIDE MOVEMENT**
call TORSION(NATOM,PMCORD,maxpsi,maxphi,phim,psim,
& COMM,AW,RESNUM,ISEED)
do 51120 M=1,NATOM
do 51130 I=1,3
PEPCORD(P,M,I) = PMCORD(M,I)
51130 continue
51120 continue
do 51140 I=1,3
COM(P,I)=COMM(I)
51140 continue
do 51150 I=1,RESNUM
phi(P,I)=phim(I)
psi(P,I)=psim(I)
51150 continue
**TOTAL ENERGY CALCULATION**
call UHP_est(UHPA,UHPB,NMOLP,NATOM,RESNUM,PEPCORD,NCA,NCB,
& COM,LX,LY,LZ)
UPSF = 0.0d+0
UP = 0.0d+0
PENTOT = 0.0d+0
UHPN = 0.0d+0

```

```

UHBN = 0.0d+0
UHBIN = 0.0d+0
do 51151 K=1,NMOLP
  do 51153 M=1,NATOM
    do 51155 I=1,3
      PMCORD(M,I) = PEPCORD(K,M,I)
51155    continue
51153  continue
    do 51157 I=1,3
      COMM(I) = COM(K,I)
51157  continue
    do 51159 N=1,RESNUM
      phim(N)=phi(K,N)
      psim(N)=psi(K,N)
      PUHPA(N)=UHPA(K,N)
      PUHPB(N)=UHPB(K,N)
51159  continue
      call DISTPEPSF(D2PEPSF1,D2PEPSF2,RSURF,NSURF,
&      PMCORD,NATOM,LX,LY,COMM)
      call NRGPEPSF(A12PS,C6PS,QCONPS,UPMSF,D2PEPSF1,
&      D2PEPSF2,NSURF,NATOM)
      UPSF = UPSF+UPMSF
      if (K.eq.P) then
        UPEPMSFNEW=UPMSF
      end if
      call DISTPP(D2PMPEP,PMCORD,PEPCORD,K,NMOLP,NATOM,LX,LY,
&      COM,COMM)
      call NRGPP(D2PMPEP,A12PP,C6PP,QCONPP,UPMP,K,NMOLP,NATOM)
      UP = UP+UPMP
      if (K.eq.P) then
        UPMPNEW=UPMP
      end if
      call pepenrg(PE,APEP,CPEP,QPEP,PMCORD,NATOM,
&      phim,psim,v1,v2,v3,v4,v5,v6,RESNUM)
      PENTOT = PENTOT+PE
      if (K.eq.P) then
        NEWPENRG=PE
      end if
      call HPenrg(UHPM,RESNUM,TEMP,PUHPA,PUHPB,PI,kB)
      UHPN = UHPN+UHPM
      call HBenrg_intra(UHBM,PUHPA,RESNUM,NATOM,GROUP,ATOM,
&      PMCORD,TEMP,COMM,LX,LY,LZ,PI,kB)

```

```

UHBN = UHBN + UHBM
call HBenrg_inter(UHBIM,PUHPA,UHPA,RESNUM,NMOLP,NATOM,GROUP
&      ,ATOM,PMCORD,PEPCORD,TEMP,K,COMM,COM,LX,LY,LZ,PI,kB)
UHBIN = UHBIN + UHBIM
51151  continue
UP = 0.5d+0*UP
UHBIN = 0.5d+0*UHBIN
VTN = UPSF+UP+PENTOT+ UHPN + UHBN + UHBIN
DELV = VTN - VT
if((DELV.ge.0.0d+0).and.(dexp(-DELV).lt.RANDOM(ISEED)))then
do 52111 N=1,NATOM
do 53111 I=1,3
PEPCORD(P,N,I)=PEPCORDOLD(N,I)
53111  continue
52111  continue
do I=1,3
COM(P,I)=COMOLD(I)
enddo
do i=1,RESNUM
phi(P,i)=phiold(i)
psi(P,i)=psiold(i)
enddo
call SUMBINPEP(PEPCORD,P,LZ,NATOM,DELZ,COSSUM,COSSQSUM,
1 ZPEPSUM,NEND,CEND,COM)
else
do i=1,RESNUM
write(88,*)phi(P,i),psi(P,i)
enddo
VT=VTN
VPSS(P)=VPSS(P)+UPEPMSFNEW-UPEPMSFOLD(P)
VPEPTP(P)=VPEPTP(P)+UPMPNEW-UPMPOLD(P)
ACCTOR=ACCTOR+1
call SUMBINPEP(PEPCORD,P,LZ,NATOM,DELZ,COSSUM,COSSQSUM,
1 ZPEPSUM,NEND,CEND,COM)
UPEPMSFOLD(P)=UPEPMSFNEW
OLDPENRG(P)=NEWPENRG
UPMPOLD(P)=UPMPNEW
UHPO=UHPN
UHBO=UHBN
UHBIO=UHBIN
end if
4010 continue

```



```

*****
VTOT(STEP)=VT
ESUM=ESUM+VTOT(STEP)
do 10005 P=1,NMOLP
  VPS(P,STEP)=VPSS(P)
  VPEPT(P,STEP)=VPEPTP(P)
  ETED(P,STEP)=
& ((PEPCORD(P,94,1)-PEPCORD(P,1,1))**2 +
& (PEPCORD(P,94,2)-PEPCORD(P,1,2))**2 +
& (PEPCORD(P,94,3)-PEPCORD(P,1,3))**2)**0.5
  CM(P,STEP)=COM(P,3)
10005 continue
10000 continue
C****FINAL PEPTIDE ENERGY CALCULATION*****
  call NRGTOT(V,LX,LY,LZ,NMOLP,
& A12PS,C6PS,QCONPS,A12PP,C6PP,QCONPP,NSURF,RSURF,NATOM,PEPCORD,
& COM,UPEPSF,UPEPMSFOLD,UPEPP,UPMPOLD,VREST)
  call UHP_est(UHPA,UHPB,NMOLP,NATOM,RESNUM,PEPCORD,NCA,NCB,
& COM,LX,LY,LZ)
PENTOT = 0.0d+0
UHBN = 0.0d+0
UHBIN = 0.0d+0
UHPN = 0.0d+0
do 10007 P=1,NMOLP
  do 10008 M=1,NATOM
    do 10012 I=1,3
      PMCORD(M,I)=PEPCORD(P,M,I)
10012 continue
10008 continue
    do 10030 I=1,3
      COMM(I)=COM(P,I)
10030 continue
    do 10022 I=1,RESNUM
      phim(I)=phi(P,I)
      psim(I)=psi(P,I)
      PUHPA(I)=UHPA(P,I)
      PUHPB(I)=UHPB(P,I)
10022 continue
    call pepenrg(PE,APEP,CPEP,QPEP,PMCORD,NATOM
1 ,phim,psim,v1,v2,v3,v4,v5,v6,RESNUM)
    PENTOT = PENTOT + PE
    OLDPENRG(P)=PE

```

```

call HPerg(UHPM,RESNUM,TEMP,PUHPA,PUHPB,PI,kB)
UHPN = UHPN + UHPM
call HBenrg_intra(UHBM,PUHPA,RESNUM,NATOM,GROUP,ATOM,
&   PMCORD,TEMP,COMM,LX,LY,LZ,PI,kB)
UHBN = UHBN + UHBM
call HBenrg_inter(UHBIM,PUHPA,UHPA,RESNUM,NMOLP,NATOM,
&   GROUP,ATOM,PMCORD,PEPCORD,TEMP,P,COMM,COM,LX,LY,LZ,PI,kB)
UHBIN = UHBIN + UHBIM
10007 continue
UHBIN = 0.5d+0*UHBIN
VT = V+PENTOT+UHPN+UHBN+UHBIN
VTOT(NMCS+1)=VT
NMCSOLD=NMCS
if(RUNSTAT.GE.2) then
  NMCSACC=NMCSACC+NMCS
  ACCPEPACC=ACCPEPACC+ACCPEP
  ACCTORACC=ACCTORACC+ACCTOR
  NMCS=NMCSACC
end if
C *****
C ** AVERAGES.                **
C ** acceptance ratio. energy. pair correlation functions.
C *****
  AVGACCPEP=dbl(ACCPEP)/dbl(NMOLP*NMCS)
  AVGACCTOR=dbl(ACCTOR)/dbl(NMOLP*NMCS)
  if(RUNSTAT.ge.2)then
    ESUMACC=ESUMACC+ESUM
    ESUM=ESUMACC
  end if
  AVGNRG=ESUM/dbl(NMCS)
  do 13303 P=1,NMOLP
    do 13101 K=1,NBINZ
      do 13102 I=1,3
        ZPEP(P,K,I)=dbl(ZPEPSUM(P,K,I))/dbl(NMCS)
      13102 continue
    13101 continue
    COSAVG(P)=COSSUM(P)/NMCS
    COSSQAVG(P)=(COSSQSUM(P)*3.d+0-dbl(NMCS))/(2.d+0*NMCS)
  13303 continue
C *****
C ** OUTPUT DATA.                **
C ** final configuration. acceptance ratios. energies. pair correlation functions. **

```

```
C *****
write(9,*)
write(9,*) '*****'
write(9,*)
write(9,*) 'GENERAL OUTPUT DATA'
write(9,*)
write(9,*) TITLE
write(9,100) RUNSTAT
write(9,*) CFGINP
write(9,*) CFGOUT
write(9,*) OUTPUT
write(9,*) PDB
write(9,*)'No of molecules of peptide'
write(9,*) NMOLP
write(9,*)'Total no of MC steps'
write(9,120) NMCS
write(9,140) TEMP
write(9,*) NSAVE
write(9,*) NADJUST
write(9,*) SURFINP
write(9,*) QSURF
write(9,*) PEP
write(9,*) VPEP
write(9,*) NSURF
write(9,*) SURFNAME
write(9,*) LX,LY,LZ
write(9,*)
write(9,*) 'AVERAGE PEPTIDE MOVEMENT ACCEPTANCE RATIO'
write(9,*) AVGACCPEP
write(9,*) 'AVERAGE PEPTIDE TORSIONAL ACCEPTANCE RATIO'
write(9,*) AVGACCTOR
write(9,*) 'Maximum psi Movement of Peptide'
write(9,*) maxpsi
write(9,*) 'Maximum phi Movement of Peptide'
write(9,*) maxphi
write(9,*)
write(9,*) 'INITIAL ENERGY'
write(9,*) VTOT(0)
write(9,*)
write(9,*) 'ENERGIES (system and water)'
do P=1,NMOLP
  ETEDAVG(P)=0.0d+0
```

```

    CMAVG(P)=0.0d+0
  end do
do 3111 STEP=1,NMCSOLD
  if(mod(STEP,NPTS).eq.0)then
    N=STEP/NPTS
    VTOTPT(N)=VTOT(STEP)
    do P=1,NMOLP
      ETEDP(P,N)=ETED(P,STEP)
      CMP(P,N)=CM(P,STEP)
      VPSPT(P,N)=VPS(P,STEP)
      VPEPTPT(P,N)=VPEPT(P,STEP)
      ETEDAVG(P)=ETEDAVG(P)+ETED(P,STEP)
      CMAVG(P)=CMAVG(P)+CM(P,STEP)
    end do
  end if
3111 continue
  if (NMCSOLD.ge.100) then
    do P=1,NMOLP
      ETEDAVG(P)=ETEDAVG(P)/100
      CMAVG(P)=CMAVG(P)/100
    end do
  end if
  VERRSUM=0.d+0
  do 3000 N=1,100
    write(9,888) N*NPTS,VTOTPT(N)
    VERRSUM=VERRSUM+dabs(VTOTPT(N)-AVGNRG)
3000 continue
888 format('step=',I6,'<E>=',D18.10)
C   do 30005 N=1,100
C     write(9,8885) N*NPTS,VPSPT(N)
C30005 continue
C8885 format('step=',I6,'<VPS>=',D18.10)
  VERR=VERRSUM/100.0d+0
  write(9,*)
  write(9,*) 'FINAL ENERGY'
  write(9,*) VTOT(NMCSOLD+1)
  write(9,*) 'AVERAGE ENERGY & UNCERTAINTY'
  write(9,*) AVGNRG,VERR
  write(9,*)
  write(9,*)'Peptide ETED and CM'
C   do 8889 P=1,(NMOLP-1),2
C     do 8887 N=1,100

```

```

C      write(9,890) N*NPTS,P,ETEDP(P,N),P,CMP(P,N),(P+1),
C      &          ETEDP((P+1),N),(P+1),CMP((P+1),N)
C8887 continue
C8889 continue
C      if (mod(NMOLP,2).ne.0)then
C      do 8892 N=1,100
C      write(9,891) N*NPTS,NMOLP,ETEDP(NMOLP,N),NMOLP,CMP(NMOLP,N)
C8892 continue
C      end if
      do 10889 P=1,(NMOLP-1),2
          write(9,1090) P,ETEDAVG(P),P,CMAVG(P),(P+1),ETEDAVG(P+1),(P+1),
          &          CMAVG(P+1)
10889 continue
          if (mod(NMOLP,2).ne.0)then
              write(9,1091) NMOLP,ETEDAVG(NMOLP),NMOLP,CMAVG(NMOLP)
          end if
C890 format('STEP=',I6,'<ETEDP',I6,'>=',D18.10,'<CMP',I6,'>=',D18.10,
C      & ' <ETEDP',I6,'>=',D18.10,'<CMP',I6,'>=',D18.10)
C891 format('STEP=',I6,'<ETEDP',I6,'>=',D18.10,'<CMP',I6,'>=',D18.10)
1090 format('<ETEDAVG_PEP',I6,'>=',D18.10,'<CMAVG_PEP',I6,'>=',
      & D18.10,'<ETEDAVG_PEP',I6,'>=',D18.10,'<CMAVG_PEP',I6,
      & '>=',D18.10)
1091 format('<ETEDAVG_PEP',I6,'>=',D18.10,'<CMAVG_PEP',I6,'>='
      & ',D18.10)
      write(9,*)
      write(9,*)'Peptide order parameter'
      do P=1,NMOLP-1,2
          write(9,895) P,COSAVG(P),COSSQAVG(P),(P+1),COSAVG(P+1),
          & COSSQAVG(P+1)
      end do
      if (mod(NMOLP,2).ne.0)then
          write(9,896) NMOLP,COSAVG(NMOLP),COSSQAVG(NMOLP)
      end if
895 format('<PEPTIDE:',I6,'><COSAVG>=',D18.10,'<COSSQAVG>=',D18.10,3x,
      & '<PEPTIDE:',I6,'><COSAVG>=',D18.10,'<COSSQAVG>=',D18.10,3x)
896 format('<PEPTIDE:',I6,'><COSAVG>=',D18.10,'<COSSQAVG>=',D18.10,3x)
      if (RUNSTAT.gt.1) then
          write(9,*)
          write(9,*) 'PEPTIDE Z position'
          do 4999 P=1,NMOLP-1,2
              do 5001 I=1,3
                  write(9,977) P,I,(P+1),I

```

```

do 5011 K=1,NBINZ
  write(9,996) ZPEP(P,K,I),ZPEP((P+1),K,I)
5011  continue
5001  continue
4999  continue
  if (mod(NMOLP,2).ne.0)then
    do 5002 I=1,3
      write(9,997) NMOLP,I
      do 5012 K=1,NBINZ
        write(9,1099) ZPEP(NMOLP,K,I)
5012  continue
5002  continue
      end if
977  format('PEPTIDE:',I6,'ZPEP',I6,6x,'PEPTIDE:',I6,'ZPEP',I6)
996  format(6x,f8.4,6x,f8.4)
997  format('PEPTIDE:',I6,'ZPEP',I6)
1099 format(6x,f8.4)
    endif
    close(9)
*****

write(2,*)NMOLP
write(2,*) NATOM
do 33000 M=1,NATOM
  write(2,889) AA(M),ATOM(M),GROUP(M),APEP(M),CPEP(M),QPEP(M)
  1 ,AW(M)
33000 continue
  do 33005 P=1,NMOLP
    do 33010 M=1,NATOM
      write(2,998) PEPCORD(P,M,1),PEPCORD(P,M,2),PEPCORD(P,M,3)
33010  continue
33005 continue
    do 35040 M=1,RESNUM
      write(2,23)NCA(M),NCB(M)
35040 continue
    do 35045 P=1,NMOLP
      write(2,777) (COM(P,I),I=1,3)
35045  continue
      write(2,*) NEND
      write(2,*) CEND
      do 35055 P=1,NMOLP
        do 35056 I=1,RESNUM
          write(2,1523)phi(P,I),psi(P,I)

```

35056 continue

35055 continue

write(2,1524)v1,v2,v3,v4,v5,v6

write(2,\*) MXTPEP

write(2,\*) MXRPEP

write(2,\*) maxpsi

write(2,\*) maxphi

if (RUNSTAT.gt.1) then

write(2,\*) NMCSACC

write(2,\*) ESUMACC

write(2,\*) ACCPEPACC

write(2,\*) ACCTORACC

write(2,\*) 'ZPEPSUM'

do 1008 P=1,NMOLP

do 1007 K=1,NBINZ

do 41710 I=1,3

write(2,\*) ZPEPSUM(P,K,I)

41710 continue

1007 continue

write(2,\*) COSSUM(P)

write(2,\*) COSSQSUM(P)

write(2,\*) '\*\*\*\*\*'

1008 continue

write(2,\*) 'STEP= ',NMCSOLD

end if

close(2)

close(7)

close(88)

call PDBOUT(RSURF,PEPCORD,PDB,NMOLP,NATOM,NSURF,AA,ATOM,GROUP)

stop

end

\*\*\*\*\*

\*\* NRGTOT.....subroutine which calculates the total potential energy for any given configuration. for programming simplicity, the code is written to sum up the potential energies felt by each molecule and then divide this sum by a factor of 2 since each pair interaction is counted twice. \*\*

\*\*\*\*\*

subroutine NRGTOT(V,LX,LY,LZ,NMOLP,

& A12PS,C6PS,QCONPS,A12PP,C6PP,QCONPP,NSURF,RSURF,NATOM,PEPCORD,

& COM,VPEPSF,VPMSF,VPEPP,VPMP,VREST)

implicit NONE

external DISTPEPSF

```

external NRGPEPSF
external DISTPP,NRGPP
*****
integer NSURF,NATOM,NMOLP
integer P,M,I
double precision V,VREST,LX,LY,LZ
double precision RSURF(1000,3)
double precision A12PS(200,2),C6PS(200,2),QCONPS(200,2)
double precision A12PP(200,200),C6PP(200,200),QCONPP(200,200)
double precision PEPCORD(50,200,3),COM(50,3)
double precision VPEPSF,VPMSF(50),VPEPP,VPMP(50),VPSF,VP
double precision D2PEPSF1(200,1000),D2PEPSF2(200,1000)
double precision D2PEPP(200,50,200),PMCORD(200,3),COMM(3)
C *****
C ** TOTAL ENERGY.                **
the calculation includes all pair interactions between a given molecule and the closest images of the remaining
molecules. **
C *****
V=0.0d+0
VREST=0.0d+0
VPEPSF=0.0d+0
VPEPP=0.0d+0
do 1004 P=1,NMOLP
  do 61060 M=1,NATOM
    do 61070 I=1,3
      PMCORD(M,I)=PEPCORD(P,M,I)
61070  continue
61060  continue
    do 61090 I=1,3
      COMM(I)=COM(P,I)
61090  continue
** PEPTIDE-PEPTIDE **
  call DISTPP(D2PEPP,PMCORD,PEPCORD,P,NMOLP,NATOM,LX,LY,COM,COMM)
  call NRGPP(D2PEPP,A12PP,C6PP,QCONPP,VP,P,NMOLP,NATOM)
  VPMP(P)=VP
  VPEPP=VPEPP+VP
** PEPTIDE-SURFACE **
  call DISTPEPSF(D2PEPSF1,D2PEPSF2,RSURF,NSURF,PMCORD,
1      NATOM,LX,LY,COMM)
  call NRGPEPSF(A12PS,C6PS,QCONPS,VPSF,D2PEPSF1,
1      D2PEPSF2,NSURF,NATOM)
  VPEPSF = VPEPSF+VPSF

```



```

      VPMSF(P) = VPSF
1004 continue
      VPEPP=0.5*VPEPP
      V=VPEPSF+VPEPP
      VREST=VPEPSF
      return
      end
*****
** DISTPEPSF.....subroutine which calculates the squared distance between the atoms of the chosen molecules and the
sites of the surfaces. **
*****
      subroutine DISTPEPSF(D2PEPSF1,D2PEPSF2,RSURF,NSURF,PMCORD,NATOM,
1      LX,LY,COMM)
      implicit NONE
      integer N,M,NSURF,NATOM
      double precision LX,LY,DELX,DELY,DELZ1,DELZ2,CORX,CORY
      double precision D2PEPSF1(200,1000),D2PEPSF2(200,1000)
      double precision PMCORD(200,3)
      double precision RSURF(1000,3),COMM(3)
      do 2000 N=1,NSURF
          DELX=RSURF(N,1)-COMM(1)
          CORX=LX*dnint(DELX/LX)
          DELY=RSURF(N,2)-COMM(2)
          CORY=LY*dnint(DELY/LY)
          do 2100 M=1,NATOM
              DELX=RSURF(N,1)-PMCORD(M,1)-CORX
              DELY=RSURF(N,2)-PMCORD(M,2)-CORY
              DELZ1=RSURF(N,3)-PMCORD(M,3)
              DELZ2=-RSURF(N,3)-PMCORD(M,3)
              D2PEPSF1(M,N)=DELX*DELX+DELY*DELY+DELZ1*DELZ1
              D2PEPSF2(M,N)=DELX*DELX+DELY*DELY+DELZ2*DELZ2
          2100 continue
      2000 continue
      return
      end
*****
** NRGPEPSF: subroutine which calculates the potential energy "felt" by a chosen molecule. **
*****
      subroutine NRGPEPSF(A12PX,C6PX,QCONPX,VPEPSF,D2PEPSF1,D2PEPSF2,
1      NSURF,NATOM)
      implicit NONE
*****

```

```

integer      M,N,NSURF,NATOM
double precision D6,D12,VPEPSF,DSURF
double precision D2PEPSF1(200,1000),D2PEPSF2(200,1000)
double precision A12PX(200,2),C6PX(200,2),QCONPX(200,2)
VPEPSF=0.d+0
do 2000 M=1,NATOM
  do 2100 N=1,NSURF
    DSURF=dsqrt(D2PEPSF1(M,N))
    D6=D2PEPSF1(M,N)**3
    D12=D2PEPSF1(M,N)**6
    VPEPSF=VPEPSF+A12PX(M,1)/D12-C6PX(M,1)/D6
1    +QCONPX(M,1)/DSURF
    DSURF=dsqrt(D2PEPSF2(M,N))
    D6=D2PEPSF2(M,N)**3
    D12=D2PEPSF2(M,N)**6
    VPEPSF=VPEPSF+A12PX(M,2)/D12-C6PX(M,2)/D6
1    +QCONPX(M,2)/DSURF
2100  continue
2000  continue
      return
      end
*****
** DISTPP.....subroutine which calculates the squared distances between the atoms of the chosen peptide molecule and
the atoms of the closest images of the remaining peptide molecules in the system.  **
*****
      subroutine DISTPP(D2PP,PMCORD,PEPCORD,P,NMOLP,NATOM,LX,LY,
&      COM,COMM)
      implicit NONE
      integer  N,I,NATOM,P,Q,NMOLP
      double precision DELX,DELY,DELZ,LX,LY,CORX,CORY
      double precision D2PP(200,50,200),COM(50,3),COMM(3)
      double precision PEPCORD(50,200,3),PMCORD(200,3)
C *****
C ** SQUARED DISTANCES.          **
C ** note that the minimum image convention is applied only to COM          **
C *****
      do 1004 Q=1,NMOLP
        if(Q.ne.P)then
          DELX=COM(Q,1)-COMM(1)
          CORX=LX*dnint(DELX/LX)
          DELY=COM(Q,2)-COMM(2)
          CORY=LY*dnint(DELY/LY)

```

```

do 1011 I=1,NATOM
  do 1021 N=1,NATOM
    DELX=PEPCORD(Q,N,1)-PMCORD(I,1)-CORX
    DELY=PEPCORD(Q,N,2)-PMCORD(I,2)-CORY
    DELZ=PEPCORD(Q,N,3)-PMCORD(I,3)
    D2PP(I,Q,N)=DELX*DELX+DELY*DELY+DELZ*DELZ
1021    continue
1011    continue
      end if
1004 continue
      return
      end
*****
** NRGPP: subroutine which calculates the potential energy "felt" by a chosen peptide molecule from the atoms of the
remaining peptides.**
*****

      subroutine NRGPP(D2PP,A12PP,C6PP,QCONPP,VPP,P,NMOLP,NATOM)
      implicit NONE
*****

      integer      M,I,NATOM,P,Q,NMOLP
      double precision VPP,D,D6,D12
      double precision A12PP(200,200),C6PP(200,200),QCONPP(200,200)
      double precision D2PP(200,50,200)
C *****
C ** ENERGY CALCULATION.          **
C ** the calculation includes all pairs of atoms in the chosen molecule and the closest images of the remaining
molecules. **
C *****

      VPP=0.0d+0
      do 1001 I=1,NATOM
        do 1025 Q=1,NMOLP
          if (Q.ne.P)then
            do 1011 M=1,NATOM
              D=dsqrt(D2PP(I,Q,M))
              D6=(D2PP(I,Q,M))**3
              D12=(D2PP(I,Q,M))**6
              VPP=VPP+A12PP(I,M)/D12-C6PP(I,M)/D6+QCONPP(I,M)/D
1011          continue
            end if
1025        continue
1001      continue
      return

```

```

end
*****
** MULTMAT.....subroutine which multiplies two matrices.      **
*****

subroutine MULTMAT(A,B,ROW,MID,COL,AB)
implicit NONE
*****

integer I,J,K,ROW,MID,COL
double precision A(ROW,MID),B(MID,COL),AB(ROW,COL)
C *****

do 1000 I=1,ROW
do 1010 J=1,COL
AB(I,J)=0.0d+0
do 1020 K=1,MID
AB(I,J)=AB(I,J)+A(I,K)*B(K,J)
1020 continue
1010 continue
1000 continue
return
end
*****
** INCONFIG.....subroutine which sets up the initial configuration of water molecules for a cold start **
*****

subroutine INCONFIG(LX,LY,LZ,NMOLP,NATOM,PEPCORD,COM,APEP,
& CPEP,QPEP,AW,AA,ATOM,GROUP,NEND,CEND,phi,psi,
& v1,v2,v3,v4,v5,v6,RESNUM,NCA,NCB)
implicit NONE
*****

integer I,M,P
integer RESNUM
integer NCA(20),NCB(20)
integer NATOM,NEND,CEND,NMOLP
integer ATOM(200),AA(200),GROUP(200)
double precision LX,LY,LZ
double precision PEPCORD(50,200,3),APEP(200),CPEP(200),QPEP(200)
double precision COM(50,3)
double precision phi(50,20),psi(50,20),AW(200)
double precision v1,v2,v3,v4,v5,v6
open(unit=10,file='Shake.cod',status='unknown',form='formatted')
rewind(10)
read(10,*)
read(10,*) NATOM

```

```

if(NATOM.GT.200) STOP 'Matrix size too small'
do 35000 M=1,NATOM
  read(10,889)AA(M),ATOM(M),GROUP(M),APEP(M),CPEP(M),QPEP(M),AW(M)
35000 continue
  do 35020 P=1,NMOLP
    do 35030 M=1,NATOM
      read(10,998) PEPCORD(P,M,1),PEPCORD(P,M,2),PEPCORD(P,M,3)
35030  continue
35020 continue
    do 35010 M=1,RESNUM
      read(10,23) NCA(M),NCB(M)
35010 continue
    do 35040 P=1,NMOLP
      read(10,777) (COM(P,I),I=1,3)
35040  continue
      read(10,*) NEND
      read(10,*) CEND
    do 35050 p=1,NMOLP
      do 35060 I=1,RESNUM
        read(10,21)phi(P,I),psi(P,I)
35060  continue
35050 continue
        read(10,22)v1,v2,v3,v4,v5,v6
889  format(5x,I3,1x,I3,1x,I4,1x,f8.3,f8.3,f8.3,f8.3)
998  format(3x,f8.3,f8.3,f8.3)
21   format(5x,f8.3,5x,f8.3)
22   format(f8.3,f8.3,f8.3,f8.3,f8.3,f8.3)
23   format(2x,i3,2x,i3)
777  format(f17.13,2x,f17.13,2x,f17.13)
      close(10)
      return
    end

```

```
*****
```

\*\* RANDOM.....This function generates random numbers in the range 0.0 \*\* TO 1.0. It is a modification of the RANDOM subroutine in V.A. Dyck, J.D. Lawson and J.A. Smith, "FORTRAN 77: an introduction to structured problem solving", Prentice-Hall, New Jersey (1984). Please cut and paste here the subroutine RANDOM(ISEED) that can be found in the explicit code (Appendix A of this thesis). \*\*

```
*****
```

```
*****
```

\* PDBOUT: this subroutine generates a PDB output file. You can view it through RASMOL. \*\*

```
*****
```

```

subroutine PDBOUT(RSURF,PEPCORD,FILENAME,NMOLP,NATOM,NSURF,
&      AA,ATOM,GROUP)
implicit none
character FILENAME*15
integer NATOM,NMOLP,NSURF
integer M,ALA,LYS,P,K
integer N,H,CA,CB1,CB2,C,O,CG,CD,CE,NZ,HZ1,HZ2,HZ3
integer AA(200),ATOM(200),GROUP(200)
double precision RSURF(1000,3),PEPCORD(50,200,3)
ALA=1
LYS=2
C =1
CA =2
CB1=3
CB2=4
CG =5
H =6
N =7
O =8
CD =9
CE =10
NZ =11
HZ1=12
HZ2=13
HZ3=14
open(unit=8,file=FILENAME,status='unknown',form='formatted')
write(8,*)'HEADER'
do 190 P=1,NMOLP
do 223 M=1,NATOM
  if(ATOM(M).EQ.C) then
    if(AA(M).EQ.ALA) then
      write(8,950) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
    else if(AA(M).EQ.LYS) then
      write(8,937) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
    end if
  else if(ATOM(M).EQ.CA) then
    if(AA(M).EQ.ALA) then
      write(8,951) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
    else if(AA(M).EQ.LYS) then

```

```

        write(8,938) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
      end if
      else if(ATOM(M).EQ.CB1) then
        if(AA(M).EQ.ALA) then
          write(8,952) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
        else if(AA(M).EQ.LYS) then
          write(8,939) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
        end if
      else if(ATOM(M).EQ.CB2) then
        if(AA(M).EQ.ALA) then
          write(8,953) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
        else if(AA(M).EQ.LYS) then
          write(8,940) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
        end if
      else if(ATOM(M).EQ.CG) then
        if(AA(M).EQ.ALA) then
          write(8,954) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
        else if(AA(M).EQ.LYS) then
          write(8,941) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
        end if
      else if(ATOM(M).EQ.CD) then
        if(AA(M).EQ.ALA) then
          write(8,955) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
        else if(AA(M).EQ.LYS) then
          write(8,942) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
        end if
      else if(ATOM(M).EQ.CE) then
        if(AA(M).EQ.ALA) then
          write(8,956) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
        else if(AA(M).EQ.LYS) then
          write(8,943) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)

```

```
end if
else if(ATOM(M).EQ.H) then
  if(AA(M).EQ.ALA) then
    write(8,957) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  else if(AA(M).EQ.LYS) then
    write(8,944) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  end if
else if(ATOM(M).EQ.N) then
  if(AA(M).EQ.ALA) then
    write(8,958) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  else if(AA(M).EQ.LYS) then
    write(8,945) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  end if
else if(ATOM(M).EQ.NZ) then
  if(AA(M).EQ.ALA) then
    write(8,959) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  else if(AA(M).EQ.LYS) then
    write(8,946) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  end if
else if(ATOM(M).EQ.O) then
  if(AA(M).EQ.ALA) then
    write(8,960) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  else if(AA(M).EQ.LYS) then
    write(8,947) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  end if
else if(ATOM(M).EQ.HZ1) then
  if(AA(M).EQ.ALA) then
    write(8,961) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  else if(AA(M).EQ.LYS) then
    write(8,948) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
  end if
else if(ATOM(M).EQ.HZ2) then
```



```

    if(AA(M).EQ.ALA) then
      write(8,962) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
    else if(AA(M).EQ.LYS) then
      write(8,949) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
    end if
    else if(ATOM(M).EQ.HZ3) then
      if(AA(M).EQ.ALA) then
        write(8,963) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
      else if(AA(M).EQ.LYS) then
        write(8,850) M,GROUP(M),PEPCORD(P,M,1),
1      PEPCORD(P,M,2),PEPCORD(P,M,3)
      end if
    end if
223 continue
190 continue
    K=NATOM
    do 226 M=1,NSURF
      K=K+1
      write(8,966) K,11,RSURF(M,1),RSURF(M,2),RSURF(M,3)
      K=K+1
      write(8,966) K,12,RSURF(M,1),RSURF(M,2),-RSURF(M,3)
226 continue
937 format('ATOM ',I5,1x,' C ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
938 format('ATOM ',I5,1x,' CA ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
939 format('ATOM ',I5,1x,' CB1',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
940 format('ATOM ',I5,1x,' CB2',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
941 format('ATOM ',I5,1x,' CG ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
942 format('ATOM ',I5,1x,' CD ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
943 format('ATOM ',I5,1x,' CE ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
944 format('ATOM ',I5,1x,' H ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
945 format('ATOM ',I5,1x,' N ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
946 format('ATOM ',I5,1x,' NZ ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
947 format('ATOM ',I5,1x,' O ',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
948 format('ATOM ',I5,1x,' HZ1',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
949 format('ATOM ',I5,1x,' HZ2',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
850 format('ATOM ',I5,1x,' HZ3',1x,'LYS',2x,I4,4x,f8.3,f8.3,f8.3)
950 format('ATOM ',I5,1x,' C ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
951 format('ATOM ',I5,1x,' CA ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
952 format('ATOM ',I5,1x,' CB1',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)

```

```

953 format('ATOM ',I5,1x,'CB2',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
954 format('ATOM ',I5,1x,'CG ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
955 format('ATOM ',I5,1x,'CD ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
956 format('ATOM ',I5,1x,'CE ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
957 format('ATOM ',I5,1x,'H ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
958 format('ATOM ',I5,1x,'N ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
959 format('ATOM ',I5,1x,'NZ ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
960 format('ATOM ',I5,1x,'O ',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
961 format('ATOM ',I5,1x,'HZ1',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
962 format('ATOM ',I5,1x,'HZ2',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
963 format('ATOM ',I5,1x,'HZ3',1x,'ALA',2x,I4,4x,f8.3,f8.3,f8.3)
966 format('ATOM ',I5,1x,'CH2',1x,'SUF',2x,I4,4x,f8.3,f8.3,f8.3)
  close(8)
  return
end
*****
** UHP_est... SUBROUTINE WHICH CALCULATES THE C_alf & C_beta DENSITY **
** PARAMETER FOR EACH OF THE AMINO ACID FOR ALL THE PEPTIDE MOLECULES **
*****
  subroutine UHP_est(UHPA,UHPB,NMOLP,NATOM,RESNUM,PEPCORD,NCA,NCB,
&   COM,LX,LY,LZ)
  implicit none
  integer Q,C,K,i,r,NMOLP,NATOM,NCA(20),NCB(20)
  integer RESNUM
  double precision PEPCORD(50,200,3)
  double precision COM(50,3)
  double precision PPEPA_Ki(3),PPEPB_Ki(3),PPEPA_Qr(3),PPEPB_Qr(3)
  double precision LX,LY,LZ,PI
  double precision sigHP1,sigHP2,UHPATM,UHPBTM
  double precision CORX,CORY,DELX,DELY,DELZ,D
  double precision D2CAPCA(20,50,20),D2CBPCB(20,50,20)
  double precision UHPA(50,20),UHPB(50,20)
  parameter (PI=3.14159265d+0)
  sigHP1 = 3.5d+0
  sigHP2 = 9.5d+0
  do 515 K=1,NMOLP
    do 222 i=1,RESNUM
      do 224 C=1,3
        PPEPA_Ki(C)=PEPCORD(K,NCA(i),C)
        PPEPB_Ki(C)=PEPCORD(K,NCB(i),C)
224      continue
    do 226 Q=1,NMOLP

```

```

DELX=COM(Q,1)-COM(K,1)
CORX=LX*dnint(DELX/LX)
DELY=COM(Q,2)-COM(K,2)
CORY=LY*dnint(DELY/LY)
do 228 r=1,RESNUM
  do 225 C=1,3
    PPEPA_Qr(C)=PEPCORD(Q,NCA(r),C)
    PPEPB_Qr(C)=PEPCORD(Q,NCB(r),C)
225    continue
  if ((Q.eq.K).and.(r.eq.i)) then
    D2CAPCA(i,Q,r)= 0.0d+0
    D2CBPCB(i,Q,r)= 0.0d+0
  else
    DELX=PPEPA_Qr(1)-PPEPA_Ki(1)-CORX
    DELY=PPEPA_Qr(2)-PPEPA_Ki(2)-CORY
    DELZ=PPEPA_Qr(3)-PPEPA_Ki(3)
    D2CAPCA(i,Q,r)=DELX*DELX+DELY*DELY+DELZ*DELZ
    DELX=PPEPB_Qr(1)-PPEPB_Ki(1)-CORX
    DELY=PPEPB_Qr(2)-PPEPB_Ki(2)-CORY
    DELZ=PPEPB_Qr(3)-PPEPB_Ki(3)
    D2CBPCB(i,Q,r)=DELX*DELX+DELY*DELY+DELZ*DELZ
  end if
228  continue
226  continue
222  continue
do 232 i=1,RESNUM
  UHPATM=0.0d+0
  UHPBTM=0.0d+0
  do 234 Q=1,NMOLP
    do 236 r=1,RESNUM
      if ((Q.ne.K).or.((Q.eq.K).and.(r.ne.i))) then
        D=dsqrt(D2CAPCA(i,Q,r))
        if (D.lt.sigHP1) then
          UHPATM = UHPATM + 1d+0
        elseif (D.le.sigHP2) then
          UHPATM=UHPATM+(0.5*(1+ dcos(PI*(D-sigHP1)/(sigHP2-sigHP1))))
        end if
        D=dsqrt(D2CBPCB(i,Q,r))
        if (D.lt.sigHP1) then
          UHPBTM = UHPBTM + 1d+0
        elseif (D.le.sigHP2) then
          UHPBTM=UHPBTM+(0.5*(1+ dcos(PI*(D-sigHP1)/(sigHP2-sigHP1))))

```

```

        end if
        end if
236    continue
234    continue
        UHPA(K,i)=UHPATM
        UHPB(K,i)=UHPBTM
232    continue
515    continue
        return
        end
*****
** HPenrg.....subroutine which calculates the Hydrophobic interactionenergy of the peptide molecule **
*****
        subroutine HPenrg(VHPM,RESNUM,TEMP,UHPA,UHPB,PI,kB)
        implicit none
        integer M,i,mu
        integer RESNUM
        double precision TEMP,kB,PI
        double precision SHPA,SHPB,VHPA,VHPB,VHPM
        double precision etaHPA,etaHPB,epsHPA,epsHPB(20)
        double precision UHPA(20),UHPB(20)
        VHPM = 0d+0
        VHPA = 0d+0
        VHPB = 0d+0
c ** HP PARAMETERS FOR ALA-LYS BLOCK PEPTIDE ***
        etaHPA = 12d+0
        etaHPB = 12d+0
        epsHPA = 1.12d+0/(kB*TEMP)
        do 210 M=1,RESNUM
            if (M.le.5) then
                epsHPB(M)=0d+0
            else
                epsHPB(M)=2.8d+0/(kB*TEMP)
            end if
210    continue
c *****
c ** Hydrophobic Interaction (HP) Energy Calculations ***
c *****
        do 272 I=1,RESNUM
            if (UHPA(I).gt.etaHPA) then
                SHPA = 1d+0
            else

```

```

      SHPA = dcos(PI*(etaHPA-UHPA(I))/(2*etaHPA))
    end if
    VHPA = VHPA + SHPA
272  continue
    VHPA = epsHPA*VHPA
    do 274 mu=1,RESNUM
      if (UHPB(mu).gt.etaHPB) then
        SHPB = 1d+0
      else
        SHPB = dcos(PI*(etaHPB-UHPB(mu))/(2*etaHPB))
      end if
      VHPB = VHPB + (epsHPB(mu)*SHPB)
274  continue
    VHPM = -(VHPA + VHPB)
  return
end
*****
** HBenrg_intra....subroutine which calculates the intra-molecular **
** Hydrogen Bond interaction energy of the selected peptide molecule **
*****
  subroutine HBenrg_intra(VHB_intra,PUHPA,RESNUM,NATOM,GROUP,ATOM,
1      PMCORD,TEMP,COMM,LX,LY,LZ,PI,kB)
  implicit none
  external SHB_est,VHB1_est,PPEPA_est
  integer I,IJ,RESNUM,NATOM,ATOM(200),GROUP(200)
  double precision PMCORD(200,3),PUHPA(20)
  double precision TEMP,kB,PI
  double precision SHB(20),SHBIJ,PMAMi(9,3),PMAMj(9,3)
  double precision VHB1SUM,VHB2SUM
  double precision sigHB,sigHB12,sigHB10,rHB0,epsHB
  double precision etaHBcMX,etaHBcMN,cHBc
  double precision LX,LY,LZ,COMM(3)
  double precision etaHBMX,etaHBMN,VHB_intra,VHB1
  VHB1SUM = 0d+0
  VHB2SUM = 0d+0
  VHB_intra= 0d+0
  sigHB = 2d+0
  rHB0 = 1.43d+0
  epsHB = 2.8d+0/(kB*TEMP)
  etaHBMX = 9d+0
  etaHBMN = 1d+0
  etaHBcMX = 12d+0

```

```

etaHBcMN = 3d+0
cHBc = 0.5d+0
C *****
c ** Hydrogen Bond (HB) Energy Calculations ***
c *****
  call SHB_est(SHB,PUHPA,RESNUM,etaHBMX,etaHBMN,PI)
  sigHB12=sigHB**12d+0
  sigHB10=sigHB**10d+0
**INTRA-MOLECULAR AMINO ACID INTERACTIONS
  do 320 JJ=1,RESNUM-3
  do 330 II=(JJ+3),RESNUM
    SHBIJ=(SHB(II)+SHB(JJ))/2d+0
    call PPEPA_est(PMAMi,NATOM,GROUP,ATOM,PMCORD,II)
    call PPEPA_est(PMAMj,NATOM,GROUP,ATOM,PMCORD,JJ)
** TO CALCULATE THE FIRST TERM OF HB INTERACTION FOR INTRA-MOLECULAR SCENARIO
    call VHB1_est(VHB1,PMAMi,PMAMj,LX,LY,LZ,sigHB12,sigHB10,
    &      COMM,COMM,SHBIJ,II,JJ)
    VHB1SUM= VHB1SUM + VHB1
330  continue
320  continue
** TO CALCULATE THE SECOND TERM OF HB INTERACTION FOR INTRA-MOLECULAR SCENARIO
  call SHB_est(SHB,PUHPA,RESNUM,etaHBcMX,etaHBcMN,PI)
  do 440 i=1,RESNUM
    VHB2SUM=VHB2SUM + SHB(i)
440  continue
  VHB_intra = epsHB*(VHB1SUM+(cHBc*VHB2SUM))
  return
  end
*****
** SHB_est.....subroutine which calculates the switching function for AA local density in Hydrogen Bond interaction
calculations **
*****
  subroutine SHB_est(SHB,PUHPA,RESNUM,etaHBMX,etaHBMN,PI)
  implicit none
  integer i,RESNUM
  double precision SHB(20),PUHPA(20),etaHBMX,etaHBMN,PI
  do 310 i=1,RESNUM
    if (PUHPA(i).gt.etaHBMX) then
      SHB(i)=1d+0
    elseif (PUHPA(i).ge.etaHBMN) then
      SHB(i)=0.5d+0*(1d+0 + dcos(PI*(etaHBMX-PUHPA(i))/
    &      (etaHBMX-etaHBMN)))

```

```

    else
      SHB(i)=0d+0
    end if
310  continue
    return
  end

*****
** HBenrg_inter.....subroutine which calculates the inter-molecular **
** Hydrogen Bond interaction energy of the selected peptide molecule **
*****

  subroutine HBenrg_inter(VHB_inter,PUHPA,UHPA,RESNUM,NMOLP,NATOM,
&    GROUP,ATOM,PMCORD_P,PEPCORD,TEMP,P,COM_P,COM,LX,LY,LZ,
&    PI,kB)
    implicit none
    external SHB_est,VHB1_est,PPEPA_est
    integer i,k,l,P,NMOLP,Q
    integer II,JJ,RESNUM,NATOM,ATOM(200),GROUP(200)
    double precision PMCORD_P(200,3),PUHPA(20),PMCORD_Q(200,3)
    double precision TEMP,kB,PI,PEPCORD(50,200,3),QUHPA(20)
    double precision SHBP(20),SHBIJ,PMAMi(9,3),PMAMj(9,3),SHBQ(20)
    double precision VHB1SUM,UHPA(50,20)
    double precision sigHB,sigHB12,sigHB10,rHB0,epsHB
    double precision LX,LY,LZ,VHB_inter,VHB1
    double precision COM_P(3),COM(50,3),COM_Q(3)
    double precision etaHBMX,etaHBMN
    VHB1SUM = 0d+0
    VHB_inter= 0d+0
    sigHB = 2d+0
    rHB0 = 1.43d+0
    epsHB = 2.8d+0/(kB*TEMP)
    etaHBMX = 9d+0
    etaHBMN = 1d+0
c *****
c ** Hydrogen Bond (HB) Energy Calculations **
c *****

    call SHB_est(SHBP,PUHPA,RESNUM,etaHBMX,etaHBMN,PI)
    sigHB12=sigHB**12d+0
    sigHB10=sigHB**10d+0
**INTER-MOLECULAR AMINO ACID INTERACTIONS
    do 510 JJ=1,RESNUM
      do 520 Q=1,NMOLP
        if (Q.ne.P) then

```

```

do 522 k=1,NATOM
  do 524 l=1,3
    PMCORD_Q(k,l)=PEPCORD(Q,k,l)
524    continue
522    continue
  do 526 l=1,3
    COM_Q(l)=COM(Q,l)
526    continue
  do 530 i=1,RESNUM
    QUHPA(i)=UHPA(Q,i)
530    continue
  call SHB_est(SHBQ,QUHPA,RESNUM,etaHBMX,etaHBMN,PI)
  do 540 II=1,RESNUM
    SHBIJ=(SHBQ(II)+SHBP(JJ))/2
    call PPEPA_est(PMAMj,NATOM,GROUP,ATOM,PMCORD_P,JJ)
    call PPEPA_est(PMAMi,NATOM,GROUP,ATOM,PMCORD_Q,II)
** TO CALCULATE THE FIRST TERM OF HB INTERACTION FOR INTER-MOLECULE SCENARIO
    call VHB1_est(VHB1,PMAMi,PMAMj,LX,LY,LZ,sigHB12,
&      sigHB10,COM_Q,COM_P,SHBIJ,II,JJ)
    VHB1SUM= VHB1SUM + VHB1
540    continue
  endif
520  continue
510  continue
  VHB_inter = epsHB*(VHB1SUM)
  return
end

*****
** PPEPA_est.....subroutine which returns the coordinates of atoms, interacting in HB model, for the selected amino
acids (will be different for different amino acids)      **
*****

subroutine PPEPA_est(PEPAA,NATOM,GROUP,ATOM,PMCORD,AA)
implicit none
integer NATOM,GROUP(200),ATOM(200),AA,C,i,NATOM1,NATOM2
double precision PEPAA(9,3),PMCORD(200,3)
if (AA.le.5) then
  NATOM1 = 1
  NATOM2 = 30
else
  NATOM1 = 31
  NATOM2 = NATOM
endif

```



```
do 340 i=NATOM1,NATOM2
  if (GROUP(i).eq.AA) then
    if (ATOM(i).eq.7) then
      do 350 C=1,3
        PEPAA(1,C)=PMCORD(i,C)
350      continue
    else if (ATOM(i).eq.6) then
      do 360 C=1,3
        PEPAA(2,C)=PMCORD(i,C)
360      continue
    else if (ATOM(i).eq.2) then
      do 370 C=1,3
        PEPAA(3,C)=PMCORD(i,C)
370      continue
    else if (ATOM(i).eq.1) then
      do 372 C=1,3
        PEPAA(4,C)=PMCORD(i,C)
372      continue
    else if (ATOM(i).eq.8) then
      do 374 C=1,3
        PEPAA(5,C)=PMCORD(i,C)
374      continue
    else if (ATOM(i).eq.11) then
      do 376 C=1,3
        PEPAA(6,C)=PMCORD(i,C)
376      continue
    else if (ATOM(i).eq.12) then
      do 378 C=1,3
        PEPAA(7,C)=PMCORD(i,C)
378      continue
    else if (ATOM(i).eq.13) then
      do 380 C=1,3
        PEPAA(8,C)=PMCORD(i,C)
380      continue
    else if (ATOM(i).eq.14) then
      do 382 C=1,3
        PEPAA(9,C)=PMCORD(i,C)
382      continue
    end if
  end if
340 continue
return
```

```

end
*****
** VHB1_est.....subroutine which calculates the first term of the HB interaction energy estimation model **
*****

subroutine VHB1_est(VHB1,PPEPAi,PPEPAj,LX,LY,LZ,sigHB12,sigHB10,
&    COMi,COMj,SHBIJ,II,JJ)
implicit none
integer j,i,flag,II,JJ,LIMi,LIMj
double precision VHB1,PPEPAi(9,3),PPEPAj(9,3)
double precision D2rij,rij,UHBij,sigHB12,sigHB10
double precision DELX,DELY,DELZ,CORX,CORY
double precision rij10,rij12,COMi(3),COMj(3)
double precision LX,LY,LZ,SHBIJ
VHB1 = 0.0d+0
DELX=COMi(1)-COMj(1)
CORX=LX*dnint(DELX/LX)
DELY=COMi(2)-COMj(2)
CORY=LY*dnint(DELY/LY)
if ((II.le.5).and.(JJ.le.5)) then
**INTERACTION BETWEEN ATOMS OF ALA AMINO ACID UNITS**
do 420 j=1,5
do 430 i=1,5
flag=0
DELX=PPEPAi(i,1)-PPEPAj(j,1)-CORX
DELY=PPEPAi(i,2)-PPEPAj(j,2)-CORY
DELZ=PPEPAi(i,3)-PPEPAj(j,3)
D2rij=DELX*DELX+DELY*DELY+DELZ*DELZ
if (D2rij.gt.0.0d+0) then
rij=dsqrt(D2rij)
rij12 = D2rij**6d+0
rij10 = D2rij**5d+0
if(((j.eq.2).and.(i.eq.5)).or.((j.eq.5).and.(i.eq.2)))
&    then
flag=1
else if ((j.eq.1).and.(i.eq.1)) then
flag=2
else if (((j.eq.3).and.(i.eq.1)).or.((j.eq.1).and.
&    (i.eq.3))) then
flag=2
else if ((j.eq.4).and.(i.eq.4)) then
flag=2
else if (((j.eq.4).and.(i.eq.3)).or.((j.eq.3).and.

```

```

&          (i.eq.4))) then
    flag=2
end if
if (flag.eq.1) then
    UHBij=(5d+0*(sigHB12/rij12))-(6d+0*(sigHB10/rij10))
else if (flag.eq.2) then
    UHBij=(3d+0*(sigHB10/rij10))
else
    UHBij=0
end if
VHB1 = VHB1 + (SHBIJ*UHBij)
end if
430    continue
420    continue
    else if ((II.gt.5).and.(JJ.gt.5)) then
**INTERACTION BETWEEN ATOMS OF LYS AMINO ACID UNITS**
    do 425 j=1,9
        do 435 i=1,9
            flag=0
            DELX=PPEPAi(i,1)-PPEPAj(j,1)-CORX
            DELY=PPEPAi(i,2)-PPEPAj(j,2)-CORY
            DELZ=PPEPAi(i,3)-PPEPAj(j,3)
            D2rij=DELX*DELX+DELY*DELY+DELZ*DELZ
            if (D2rij.gt.0.0d+0) then
                rij=dsqrt(D2rij)
                rij12 = D2rij**6d+0
                rij10 = D2rij**5d+0
                if(((j.eq.2).and.(i.eq.5)).or.((j.eq.5).and.(i.eq.2)))
&          then
                    flag=1
                else if(((j.eq.7).and.(i.eq.5)).or.((j.eq.5).and.
&          (i.eq.7))) then
                    flag=1
                else if(((j.eq.8).and.(i.eq.5)).or.((j.eq.5).and.
&          (i.eq.8))) then
                    flag=1
                else if(((j.eq.9).and.(i.eq.5)).or.((j.eq.5).and.
&          (i.eq.9))) then
                    flag=1
                else if ((j.eq.1).and.(i.eq.1)) then
                    flag=2
                else if (((j.eq.3).and.(i.eq.1)).or.((j.eq.1).and.

```

```

&      (i.eq.3)) then
      flag=2
      else if (((j.eq.1).and.(i.eq.6)).or.((j.eq.6).and.
&      (i.eq.1))) then
      flag=2
      else if ((j.eq.4).and.(i.eq.4)) then
      flag=2
      else if (((j.eq.4).and.(i.eq.3)).or.((j.eq.3).and.
&      (i.eq.4))) then
      flag=2
      else if ((j.eq.6).and.(i.eq.6)) then
      flag=2
      else if (((j.eq.3).and.(i.eq.6)).or.((j.eq.6).and.
&      (i.eq.3))) then
      flag=2
      end if
      if (flag.eq.1) then
        UHBij=(5d+0*(sigHB12/rij12)-(6d+0*(sigHB10/rij10))
      else if (flag.eq.2) then
        UHBij=(3d+0*(sigHB10/rij10))
      else
        UHBij=0
      end if
      VHB1 = VHB1 + (SHBIJ*UHBij)
    end if
435   continue
425   continue
  else
**INTERACTION BETWEEN ATOMS OF ALA -> LYS AMINO ACID UNITS**
    if((JJ.le.5).and.(II.gt.5)) then
      LIMj=5
      LIMi=9
    else
      LIMj=9
      LIMi=5
    endif
    do 445 j=1,LIMj
      do 455 i=1,LIMi
        flag=0
        DELX=PPEPAi(i,1)-PPEPAj(j,1)-CORX
        DELY=PPEPAi(i,2)-PPEPAj(j,2)-CORY
        DELZ=PPEPAi(i,3)-PPEPAj(j,3)

```

```

D2rij=DELX*DELX+DELY*DELY+DELZ*DELZ
if (D2rij.gt.0.0d+0) then
  rij=dsqrt(D2rij)
  rij12 = D2rij**6d+0
  rij10 = D2rij**5d+0
  if(((j.eq.2).and.(i.eq.5)).or((j.eq.5).and.(i.eq.2)))
&    then
    flag=1
  else if(((j.eq.7).and.(i.eq.5)).or((j.eq.5).and.
&    (i.eq.7))) then
    flag=1
  else if(((j.eq.8).and.(i.eq.5)).or((j.eq.5).and.
&    (i.eq.8))) then
    flag=1
  else if(((j.eq.9).and.(i.eq.5)).or((j.eq.5).and.
&    (i.eq.9))) then
    flag=1
  else if ((j.eq.1).and.(i.eq.1)) then
    flag=2
  else if (((j.eq.3).and.(i.eq.1)).or((j.eq.1).and.
&    (i.eq.3))) then
    flag=2
  else if (((j.eq.1).and.(i.eq.6)).or((j.eq.6).and.
&    (i.eq.1))) then
    flag=2
  else if ((j.eq.4).and.(i.eq.4)) then
    flag=2
  else if (((j.eq.4).and.(i.eq.3)).or((j.eq.3).and.
&    (i.eq.4))) then
    flag=2
  else if (((j.eq.3).and.(i.eq.6)).or((j.eq.6).and.
&    (i.eq.3))) then
    flag=2
  end if
  if (flag.eq.1) then
    UHBij=(5d+0*(sigHB12/rij12)-(6d+0*(sigHB10/rij10))
  else if (flag.eq.2) then
    UHBij=(3d+0*(sigHB10/rij10))
  else
    UHBij=0
  end if
  VHB1 = VHB1 + (SHBIJ*UHBij)

```

```

        end if
455    continue
445    continue
    endif

    return
    end

*****
** MOVEPEP....subroutine which randomly displaces a molecule. The displacement is performed as a translational
displacement of the molecule coupled with a rotational displacement about the position of the oxygen atom. **
*****

    subroutine MOVEPEP(NATOM,PMCORD,COMM,MXTPEP,MXRPEP,
1        LX,LY,LZ,ISEED)
    implicit NONE
    external  MULTMAT,RANDOM
*****

    integer    I,M,ISEED,NATOM
    double precision LX,LY,LZ,DELC,MXTPEP,MXRPEP,THX,THY,THZ
    double precision COSX,SINX,COSY,SINY,COSZ,SINZ
    double precision ROTX(3,3),ROTY(3,3),ROTZ(3,3),TMP1(3,3)
    double precision TMP2(3,3),COMM(3)
    double precision PMCORD(200,3),RHYOLD(3,200),RHYNEW(3,200)
    double precision RANDOM,L(3)

    L(1)=LX
    L(2)=LY
    L(3)=LZ

C *****
C ** PEPTIDE COORDINATE MATRIX.          **
C ** the positions of the peptide atoms are defined relative to the COM point.    **
C *****

    do 1000 M=1,NATOM
        do 1010 I=1,3
            RHYOLD(I,M)=PMCORD(M,I)-COMM(I)
1010    continue
1000    continue

c *****
C ** RANDOM ROTATION.                    **
C *****

    THX=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MXRPEP
    THY=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MXRPEP
    THZ=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MXRPEP
    COSX=dcos(THX)

```

```

SINX=dsin(THX)
COSY=dcos(THY)
SINY=dsin(THY)
COSZ=dcos(THZ)
SINZ=dsin(THZ)
ROTX(1,1)=1.0d+0
ROTX(1,2)=0.0d+0
ROTX(1,3)=0.0d+0
ROTX(2,1)=0.0d+0
ROTX(2,2)=COSX
ROTX(2,3)=-SINX
ROTX(3,1)=0.0d+0
ROTX(3,2)=SINX
ROTX(3,3)=COSX
ROTY(1,1)=COSY
ROTY(1,2)=0.0d+0
ROTY(1,3)=-SINY
ROTY(2,1)=0.0d+0
ROTY(2,2)=1.0d+0
ROTY(2,3)=0.0d+0
ROTY(3,1)=SINY
ROTY(3,2)=0.0d+0
ROTY(3,3)=COSY
ROTZ(1,1)=COSZ
ROTZ(1,2)=-SINZ
ROTZ(1,3)=0.0d+0
ROTZ(2,1)=SINZ
ROTZ(2,2)=COSZ
ROTZ(2,3)=0.0d+0
ROTZ(3,1)=0.0d+0
ROTZ(3,2)=0.0d+0
ROTZ(3,3)=1.0d+0
call MULTMAT(ROTX,ROTY,3,3,3,TMP1)
call MULTMAT(TMP1,ROTZ,3,3,3,TMP2)
call MULTMAT(TMP2,RHYOLD,3,3,NATOM,RHYNEW)
C *****
C ** RANDOM TRANSLATION.          **
C *****
do 100 I=1,3
  DELC=(2.0d+0*RANDOM(ISEED)-1.0d+0)*MXTPEP*L(I)/LX
  COMM(I)=COMM(I)+DELC
  if(I.ne.3) then

```

```

      COMM(I)=COMM(I)-L(I)*dnint(COMM(I)/L(I))
    end if
    do 110 M=1,NATOM
      PMCORD(M,I)=COMM(I)+RHYNEW(I,M)
110  continue
100  continue
      return
    end
*****
** SUBROUTINE WHICH CALCULATES THE ENERGY OF THE PEPTIDE MOLECULE *****
*****
      SUBROUTINE pepenrg(configE,APEP,CPEP,QPEP,PEPCORD,NATOM,
1  phi,psi,v1,v2,v3,v4,v5,v6,RESNUM)
      integer NATOM,M
      integer I,RESNUM
      double precision PEPCORD(200,3),APEP(200),CPEP(200),QPEP(200)
      double precision QFACTP,TEMP,A12P(200,200),C6P(200,200)
      double precision QCONP(200,200)
      double precision AP(200,200),CP(200,200)
      double precision D2PEP(200,200),DELX,DELY,DELZ
      double precision PVPEP,D,D6,D12,VTOR
      double precision phi(20),psi(20),configE
      double precision v1,v2,v3,v4,v5,v6
c *****
c ** The non-bonded interaction energy calculations **
c *****
c *****
c ***** LJ Potential Calculations *****
c *****
      TEMP=298.0
      QFACTP=1.671d+5/TEMP
      do 200 M=1,NATOM-1
      do 201 I=M+1,NATOM
        AP(M,I)=dsqrt( APEP(M)*APEP(I) )
        CP(M,I)=dsqrt( CPEP(M)*CPEP(I) )
201  continue
200  continue
      do 202 M=1,NATOM-1
      do 203 I=M+1,NATOM
        A12P(M,I)=(AP(M,I)**12)/TEMP
        C6P(M,I)=(CP(M,I)**6)/TEMP
        QCONP(M,I)=QFACTP*QPEP(M)*QPEP(I)

```



```

203   continue
202   continue
C ** This section of this document is identical to the section of the explicit code starting on Page 98 of this thesis and
therefore has been deleted. Cut and paste here from Page 98 Line yy to Page 120 Line zz. **
      do 204 M=1,NATOM-1
        do 205 I=M+1,NATOM
          DELX=PEPCORD(I,1)-PEPCORD(M,1)
          DELY=PEPCORD(I,2)-PEPCORD(M,2)
          DELZ=PEPCORD(I,3)-PEPCORD(M,3)
          D2PEP(M,I)=DELX*DELX+DELY*DELY+DELZ*DELZ
205   continue
204   continue
      PVPEP=0.d+0
      do 206 M=1,NATOM-1
        do 207 I=M+1,NATOM
          D=dsqrt(D2PEP(M,I))
          D6=(D2PEP(M,I))**3
          D12=(D2PEP(M,I))**6
          PVPEP=PVPEP + A12P(M,I)/D12 - C6P(M,I)/D6
1      + QCONP(M,I)/D
207   continue
206   continue
c      write(*,*)'Total Potential '
c      write(*,208) PVPEP
c 208   format(f15.3)
c *****
c ** Torsional Energy(VTOR) Calculations **
c *****
      VTOR=0.0d+0
      do i=1,RESNUM
        VTOR=VTOR + 0.5d+0*v1*(1.0d+0 - dcos(phi(i)))
1      + 0.5d+0*v2*(1.0d+0 - dcos(2.0d+0*phi(i)))
1      + 0.5d+0*v3*(1.0d+0 + dcos(3.0d+0*phi(i)))
      enddo
      do j=1,RESNUM
        VTOR=VTOR + 0.5d+0*v4*(1.0d+0 - dcos(psi(j)))
1      + 0.5d+0*v5*(1.0d+0 - dcos(2.0d+0*psi(j)))
1      + 0.5d+0*v6*(1.0d+0 + dcos(3.0d+0*psi(j)))
      enddo
C      write(*,*)'VTOR',VTOR
      CLOSE(10)
      configE=VTOR+PVPEP

```

```

c      write(*,*)'Total Conformational energy of peptide'
c      write(*,*)'ConfigE=',configE
      RETURN
      END
*****
** SUMBINPEP..subroutine which collects the data for the number of atoms in each bin. **
*****

      subroutine SUMBINPEP(PEPCORD,P,LZ,NATOM,
1 DELZ,COSSUM,COSSQSUM,ZPEPSUM,NEND,CEND,COM)
      implicit NONE
      integer NATOM,BINZ,NEND,CEND
      integer ZPEPSUM(50,2000,3),P
      double precision DELZ,LZ,HALFLZ
      double precision PEPCORD(50,200,3),COM(50,3),COSSQ
      double precision DELTX,DELTZ,DELTZ,COSSUM(50),COSSQSUM(50),LNC
C *****
C ** COLLECTION OF DATA.          **
C *****
      HALFLZ=LZ/2.d+0
** Z profile of 1st, last, and COM positions
** 1st atom
      BINZ=idint((PEPCORD(P,1,3)+HALFLZ)/DELZ)+1
      ZPEPSUM(P,BINZ,1)=ZPEPSUM(P,BINZ,1)+1
** COM
      BINZ=idint((COM(P,3)+HALFLZ)/DELZ)+1
      ZPEPSUM(P,BINZ,2)=ZPEPSUM(P,BINZ,2)+1
** last atom
      BINZ=idint((PEPCORD(P,NATOM,3)+HALFLZ)/DELZ)+1
      ZPEPSUM(P,BINZ,3)=ZPEPSUM(P,BINZ,3)+1
**Order Parameter (from NEND to CEND)
      DELTX=PEPCORD(P,NEND,1)-PEPCORD(P,CEND,1)
      DELTY=PEPCORD(P,NEND,2)-PEPCORD(P,CEND,2)
      DELTZ=PEPCORD(P,NEND,3)-PEPCORD(P,CEND,3)
      LNC=dsqrt(DELTX*DELTX+DELTZ*DELTZ+DELTZ*DELTZ)
      COSSQ=DELTZ/LNC
      COSSUM(P)=COSSUM(P)+COSSQ
      COSSQ=COSSQ*COSSQ
      COSSQSUM(P)=COSSQSUM(P)+COSSQ
      return
      end
*****
** TORSION ..... subroutine to give the torsional rotation by changing coordinates of the peptide atoms

```

```
*****
```

```

subroutine TORSION(NATOM,PMCORD,maxpsi,maxphi,phim,psim,
& COMM,AW,RESNUM,ISEED)
integer ISEED,NATOM,RESNUM
integer k1,k2,k3,n3,n4,k4,N
double precision PMCORD(200,3)
double precision RANDOM
double precision u(20,3),s(3),w(3),x(3),maxpsi,cpsi(20)
double precision phim(20),psim(20),magu(20),magv(20)
double precision v(20,3),t(3),y(3),z(3),maxphi,cphi(20)
double precision CM(3),AWS,COMM(3),AW(200)
c u vector is to be used as reference vector for psi angle (about Calpha - C) rotation
c there are 8 Calpha - C bonds, so 8 u vectors
do I=1,3
CM(I)=0.0D+0
enddo
AWS=0.0D+0
do i=1,RESNUM
cpsi(i)=(2.0d+0*RANDOM(ISEED)-1.0d+0)*maxpsi
cphi(i)=(2.0d+0*RANDOM(ISEED)-1.0d+0)*maxphi
enddo
do i=1,RESNUM
phim(i)=phim(i) + cphi(i)
psim(i)=psim(i) + cpsi(i)
enddo
do 45 k1=1,RESNUM
if (k1 .le. 5) then
u(k1,1)=PMCORD(6*k1-1,1) - PMCORD(6*k1-3,1)
u(k1,2)=PMCORD(6*k1-1,2) - PMCORD(6*k1-3,2)
u(k1,3)=PMCORD(6*k1-1,3) - PMCORD(6*k1-3,3)
else
u(k1,1)=PMCORD((13*(k1-3))+3,1) - PMCORD((13*(k1-3))-6,1)
u(k1,2)=PMCORD((13*(k1-3))+3,2) - PMCORD((13*(k1-3))-6,2)
u(k1,3)=PMCORD((13*(k1-3))+3,3) - PMCORD((13*(k1-3))-6,3)
endif
c magnitude of reference vector u
magu(k1)=sqrt((u(k1,1))**2 + (u(k1,2))**2 + (u(k1,3))**2)
45 continue
c v vector is to be used as reference vector for phi angle (about N - Calpha) rotation
c there are 8 N - Calpha bonds, so 8 v vectors
do 47 k2=1,RESNUM
if (k2 .le. 5) then

```

```

v(k2,1)=PMCORD(6*k2-3,1) - PMCORD(6*k2-5,1)
v(k2,2)=PMCORD(6*k2-3,2) - PMCORD(6*k2-5,2)
v(k2,3)=PMCORD(6*k2-3,3) - PMCORD(6*k2-5,3)
else
v(k2,1)=PMCORD((13*(k2-3))-6,1) - PMCORD((13*(k2-3))-8,1)
v(k2,2)=PMCORD((13*(k2-3))-6,2) - PMCORD((13*(k2-3))-8,2)
v(k2,3)=PMCORD((13*(k2-3))-6,3) - PMCORD((13*(k2-3))-8,3)
endif
c magnitude of reference vector v
magv(k2)=sqrt((v(k2,1))**2 + (v(k2,2))**2 + (v(k2,3))**2)
47 continue
c FOR PSI (Alpha-C) rotation calculations for atom coordinates when the right
c bond C - N is rotated, all the atoms on the right are mapped
do 49 k3=1,RESNUM
IF (K3 .LE. 5) THEN
n3=6*k3-1
else
n3=13*(k3-3)+3
endif
do 50 N=n3+1,NATOM
s(1)=PMCORD(N,1)-PMCORD(n3,1)
s(2)=PMCORD(N,2)-PMCORD(n3,2)
s(3)=PMCORD(N,3)-PMCORD(n3,3)
w(1)=(1.0d+0/magu(k3))*(u(k3,2)*s(3) - u(k3,3)*s(2) )
w(2)=(1.0d+0/magu(k3))*(u(k3,3)*s(1) - u(k3,1)*s(3) )
w(3)=(1.0d+0/magu(k3))*(u(k3,1)*s(2) - u(k3,2)*s(1) )
x(1)=(1.0d+0/magu(k3))*(u(k3,2)*w(3) - u(k3,3)*w(2) )
x(2)=(1.0d+0/magu(k3))*(u(k3,3)*w(1) - u(k3,1)*w(3) )
x(3)=(1.0d+0/magu(k3))*(u(k3,1)*w(2) - u(k3,2)*w(1) )
PMCORD(N,1)=PMCORD(N,1) + w(1)*dsin(cpsi(k3))
1 + x(1)*(1.0d+0-dcos(cpsi(k3)))
PMCORD(N,2)=PMCORD(N,2) + w(2)*dsin(cpsi(k3))
1 + x(2)*(1.0d+0-dcos(cpsi(k3)))
PMCORD(N,3)=PMCORD(N,3) + w(3)*dsin(cpsi(k3))
1 + x(3)*(1.0d+0-dcos(cpsi(k3)))
50 continue
49 continue
c calculations for atom coordinates when the right bond C - N is rotated, all the atoms
c on the right are mapped
do 51 k4=1,RESNUM
IF (k4 .LE. 5) THEN
n4=6*k4-3

```

```

else
n4=13*(k4-3)-6
endif
do 52 I=n4+1,NATOM
t(1)=PMCORD(I,1)-PMCORD(n4,1)
t(2)=PMCORD(I,2)-PMCORD(n4,2)
t(3)=PMCORD(I,3)-PMCORD(n4,3)
y(1)=(1.0d+0/magv(k4))*(v(k4,2)*t(3) - v(k4,3)*t(2) )
y(2)=(1.0d+0/magv(k4))*(v(k4,3)*t(1) - v(k4,1)*t(3) )
y(3)=(1.0d+0/magv(k4))*(v(k4,1)*t(2) - v(k4,2)*t(1) )
z(1)=(1.0d+0/magv(k4))*(v(k4,2)*y(3) - v(k4,3)*y(2) )
z(2)=(1.0d+0/magv(k4))*(v(k4,3)*y(1) - v(k4,1)*y(3) )
z(3)=(1.0d+0/magv(k4))*(v(k4,1)*y(2) - v(k4,2)*y(1) )
PMCORD(I,1)=PMCORD(I,1) + y(1)*dsin(cphi(k4))
1  + z(1)*(1d+0-dcos(cphi(k4)))
PMCORD(I,2)=PMCORD(I,2) + y(2)*dsin(cphi(k4))
1  + z(2)*(1d+0-dcos(cphi(k4)))
PMCORD(I,3)=PMCORD(I,3) + y(3)*dsin(cphi(k4))
1  + z(3)*(1d+0-dcos(cphi(k4)))
52 continue
51 continue
do I=1,3
do M=1,NATOM
CM(I)=CM(I)+PMCORD(M,I)*AW(M)
enddo
enddo
do M=1,NATOM
AWS=AWS+AW(M)
enddo
do I=1,3
COMM(I)=CM(I)/AWS
enddo
return
end

```

## APPENDIX D

### INSTRUCTIONS TO RUN IMPLICIT MODEL CODE

The implicit model codes `pw_imp.f` (without surface) and `pws_imp.f` (with surface) can simulate up to 50 identical peptides (with a maximum of 20 residues each) in a pre-defined simulation box. Like the explicit model code, the simulation is run by splitting the required total number of MC simulation steps in sets of no more than 40,000 MC steps each. After completing a set, the input file required for the next set is generated and the simulation is continued. The simulation uses an input file for a run set, which is associated with system configuration files and output files.

For the implicit model codes, the input data file (`inputx.inp`) which describes the variables and file names for a simulation run set are written in similar fashion as that for the explicit model. The only difference is the absence of definitions for water molecules in the simulation box. All specific parameters for the HB/HP interaction model have to be set in their respective subroutines in the program code before the simulation run. Thus for running each set of MC simulations the only change to be made in the code is the input data file name for the next set of run. The code is then compiled and submitted for execution.

It should be noted that the attached implicit code to simulate multi-peptide-surface system takes the initial configuration from the output of the computational “shake” simulation. The configuration file (`CFGxx.out`) of the final set of computational “shake” simulations should be renamed as “shake.cod”, which will be read in the `INCONFIG( )` subroutine to assign initial configurations to the peptides. This subroutine should be modified if the peptides should be arranged in order inside the simulation box during initialization; where the peptide coordinates and parameters are read from the original peptide configuration file (for our model, the “A5-L5.cod” file).

## APPENDIX E

### EXPLICIT MODEL RESULTS



## .1. Finite peptide concentrations

The diblock peptide used in this work consists of five alanine (ALA) residues followed by five lysine (LYS) residues. The residues were numbered as 1 to 5 for the first block of ALA and 6 to 10 for the LYS block for sake of identification. ALA is a nonpolar amino acid whereas LYS is a polar amino acid carrying a positive charge at pH 7. The selection of this peptide is based on the different hydrophobicities of the two amino acids. The peptide has an energy minimized  $\alpha$ -helix conformation, for initial state, that was generated by using Alchemy software platform. All peptide atoms were treated explicitly except for the methyl groups,  $\text{CH}_x$  ( $x = 1-3$ ), which were treated as united atoms. The peptide has a volume of  $1200 \text{ \AA}^3$  and an overall charge of  $+5e$ .

All water molecules were treated explicitly using the simple point charge (SPC) model. For the selected box size ( $27.748 \text{ \AA} \times 24.030 \text{ \AA} \times 38.00 \text{ \AA}$ ), the number of water molecules was adjusted to maintain the desired solvent density to  $1 \text{ g/cm}^3$  at a temperature of 298 K. The volume used in calculating solvent density was equal to the volume of the box minus the total volume occupied by the two peptides.

Two surfaces with opposing, but equal, charge density ( $0.10168 \text{ e/ \AA}^2$ ) were placed at the two ends of the simulation box perpendicular to z-axis. Each surface is comprised of 226 united  $\text{CH}_2$  atoms (with LJ parameters  $\sigma = 3.964 \text{ \AA}$  and  $\epsilon = 70.49 \text{ K}$ ), arranged in face-centered cubic (fcc) 111 crystal lattice structure, with a lattice constant of  $6.111 \text{ \AA}$ .

The interaction between any pair of sites, whether solute or solvent, is the spherically symmetric Lennard-Jones (LJ) plus a Coulomb term,

$$u_{\alpha\beta} = \left\{ \frac{A_{\alpha\beta}}{r} \right\}^{12} - \left\{ \frac{C_{\alpha\beta}}{r} \right\}^6 + \frac{e^2 q_\alpha q_\beta}{4\pi\epsilon_0} \left\{ \frac{1}{r} \right\} \quad (\text{E.1.1})$$

Where,  $A_{\alpha\beta}$  and  $C_{\alpha\beta}$  are the repulsive and dispersive coefficients of the LJ potential,  $r$  is the distance between atoms  $\alpha$  and  $\beta$ ,  $q_\alpha$  and  $q_\beta$  are the partial charges on atoms  $\alpha$  and  $\beta$  respectively and  $\epsilon_0$  is the permittivity.

The contribution to the energy of the system by peptide conformation, consists of a nonbonded torsional energy along with the LJ and electrostatic interaction energies, as described above. The torsional energy is represented through a four term Fourier series,

$$E_{\text{torsion}} = \left(\frac{V_1}{2}\right)(1 + \cos \theta) + \left(\frac{V_2}{2}\right)(1 - \cos 2\theta) + \left(\frac{V_3}{2}\right)(1 + \cos 3\theta) \quad (\text{E.1.2})$$

Where,  $\theta$  is the torsion angle (either  $\phi$ , rotation about N-C $^\alpha$ , or  $\psi$ , rotation about C-C $^\alpha$ ). The coefficients,  $V_x$  ( $x=1-3$ ), values are as per Table E.1.1.

Table E.1.1 Coefficients for torsional interactions

Coefficient	For $\phi$ (rotation about N-C $^\alpha$ )	For $\psi$ (rotation about C $^\alpha$ -C)
$V_1$	0.000	1.849
$V_2$	0.000	0.950
$V_3$	1.498	0.840

The system's energy calculations include peptide-peptide, peptide-water, peptide-surface, water-water and water-surface interactions along with peptide intra-molecular interactions. All interactions were calculated using periodic boundary conditions using the minimum image convention and negating any edge effects.

Canonical ensemble MCMC simulations were carried out with the magnitudes of the maximum translational and rotational movements set separately for water and peptide molecules. They were adjusted every 25 steps throughout the simulation to maintain a 50% movement acceptance ratio. This was also applied to the torsional movements for the peptide molecules. Each simulation step consists of  $N_{\text{water}} + N_{\text{peptide}}$  movements ( $N_{\text{water}}$  and  $N_{\text{peptide}}$  are the number of water and peptide molecules respectively). In one step, a

molecule was subject to random translational, rotational or torsional (only for the peptide molecule) movement. Then the system's energy with the resulting conformation is estimated. The difference between the energy of the resulting conformation and the energy of the current conformation is subjected to the Metropolis criteria, to accept or reject the attempted move of the molecule. Movement for peptide-1 is followed by peptide-2 and then by each of the water molecules in a single simulation step. The simulation was run for 1.272 million steps to equilibration.

Figure E.1.1 shows the system's potential energy (PE) vs. the number of MC steps (NMCS). After 0.9 million MC steps the system energy approaches a minimum value and attains the equilibrium state.

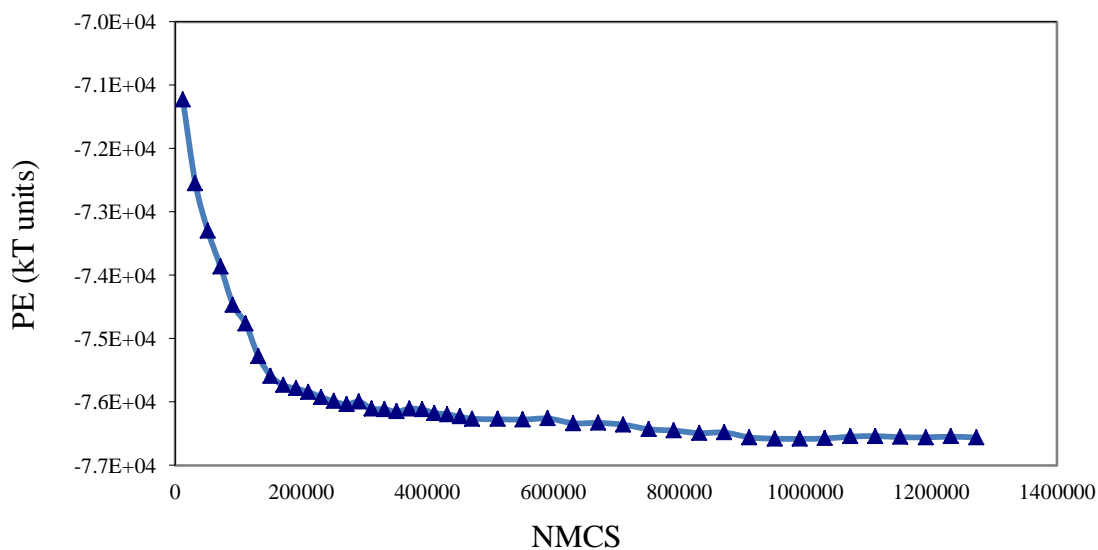


Figure E.1.1 Energy profile for 2-peptide(helix)-water-surface system.

Figures E.1.2 and E.1.3 show the z-location of the peptide's center of mass (COM) and peptide's end-to-end distance (dl), respectively, as functions of the NMCS. Here, the end-to-end distance is defined as the ensemble average of the distance separating the N-atom of the N-terminus ALA residue and the C<sup>α</sup> atom of the C-terminus LYS residue.

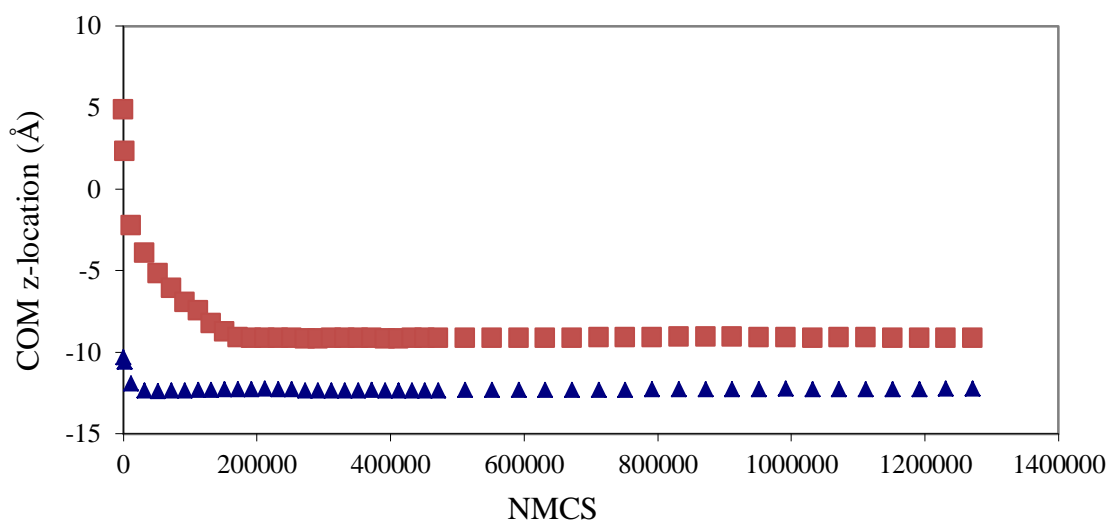


Figure E.1.2 COM z-location profile for 2-peptide(helix)-water-surface system

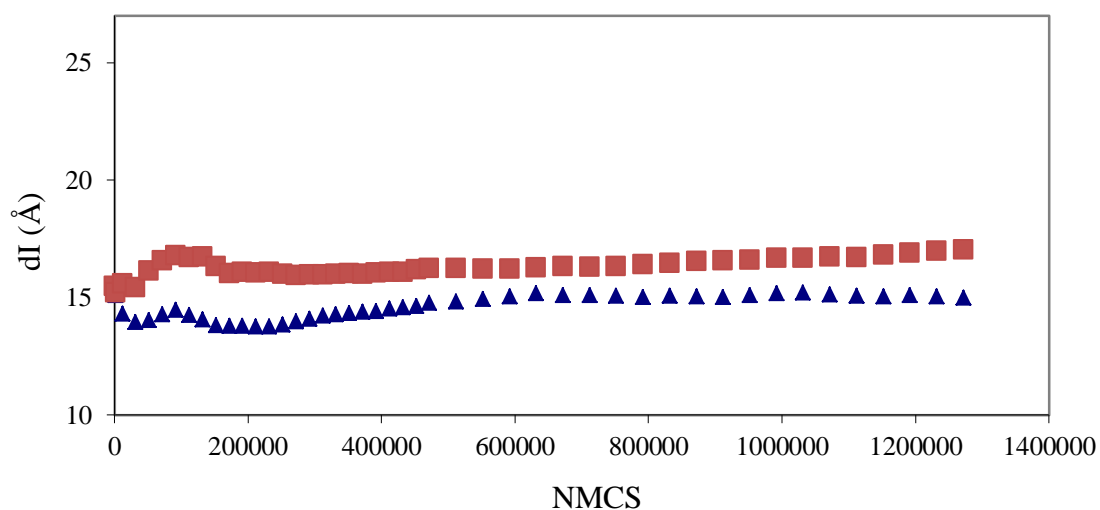


Figure E.1.3 Peptide end-to-end distance profile for 2-peptide(helix)-water-surface system.

Figure E.1.4 shows snapshots of the initial and final configuration states of the system. In Figure E1.4 (b) all water molecules were removed to ease the visualization of the peptides. As part of initialization, depicted in Figure E.1.4 (a), the water molecules were orderly arranged around the peptides, where for a molecule of water the oxygen atom is red and the hydrogen atoms are white. Different amino acids of the diblock

peptide model are displayed in different color shades: the ALA block end is dark blue, the LYS block end is orange, and the methyl groups of the surfaces grey. This color code is kept consistent for all our RasMol “Ball & Stick” graphic visualizations.

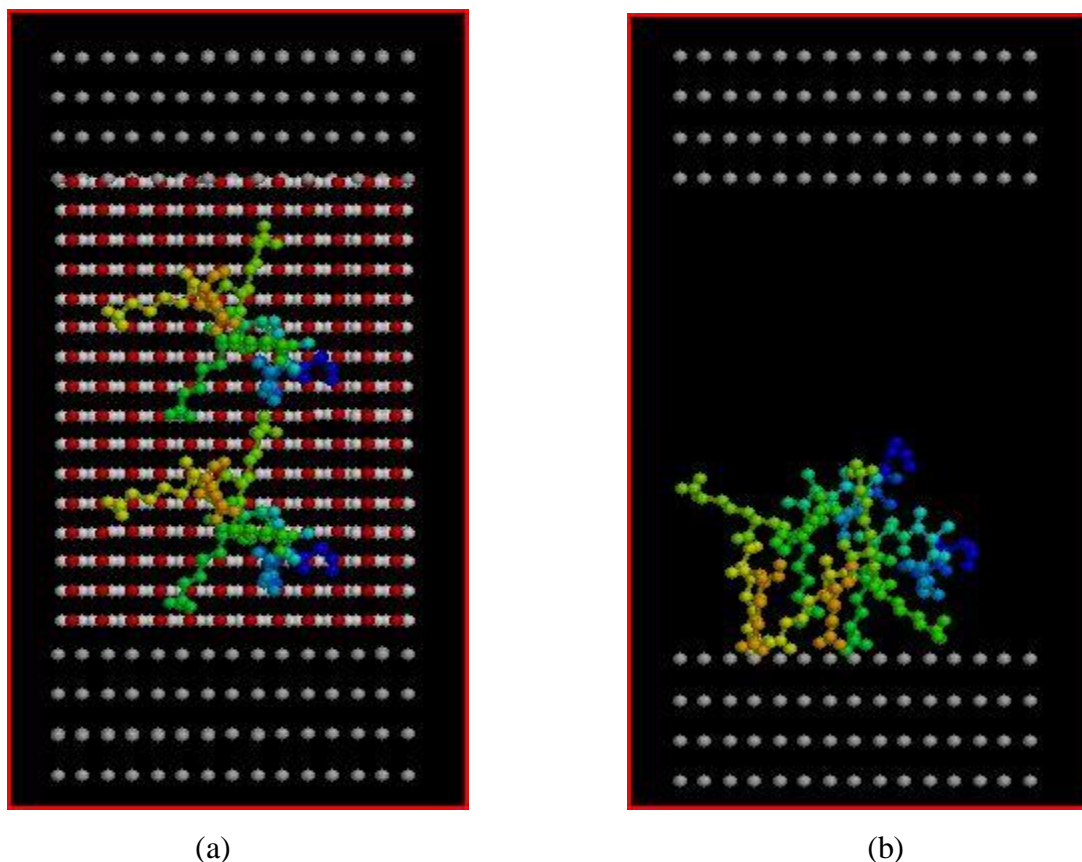


Figure E.1.4 Snapshots of 2-peptide(helix)-water-surface system configuration using RasMol “Ball & Stick” graphics. (a) Initial state. (b) Final state.

The two peptides laid out along  $z$ -direction initially, with peptide-1 closer to the surface of complimentary electrostatic charge (negative charge) located at  $-19 \text{ \AA}$  on the  $z$ -axis, thus approaching the surface before peptide-2, which does so after 0.15 million MC steps. After reaching the surface, the peptide molecules lay flat on the surface to maximize the electrostatic interactions with the opposite charged surface. Both peptides changed their secondary structure upon adsorption (Figure E.1.3). The two peptides follow different paths in the end-to-end distance plot, although they were identical initially.

## E.2 Mixing

Figures E.2.1 and E.2.2 show energy and end-to-end distance profiles, respectively, after 0.135 million MC steps. The initial and final system configuration snapshots are shown in Figures E.2.3 (a) and (b). For the sake of clarity the water molecules are not displayed in Figure E.2.3 (b).

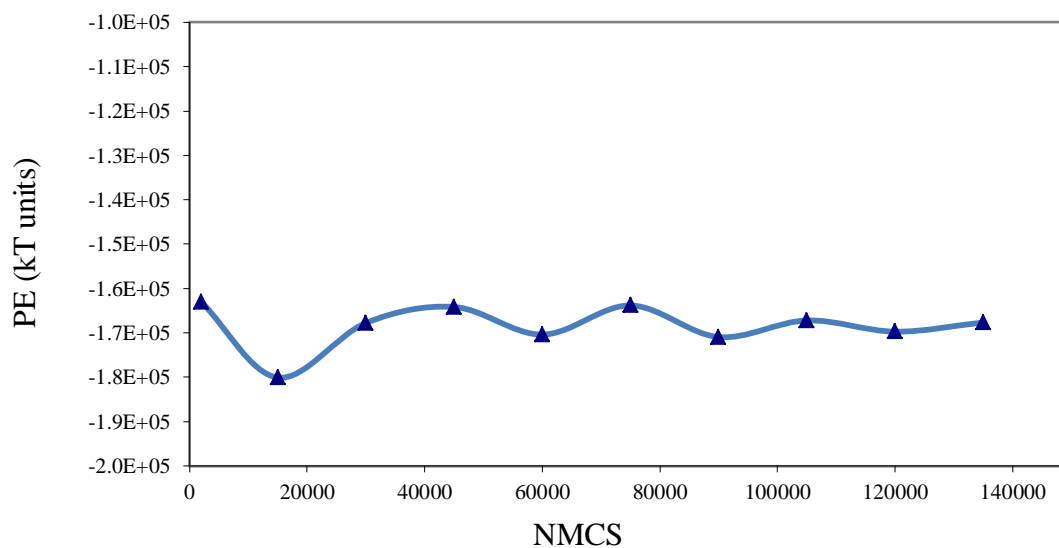


Figure E.2.1 Potential energy vs the number of Monte Carlo steps. Two peptides in explicit water in between surfaces of alternating polarity.

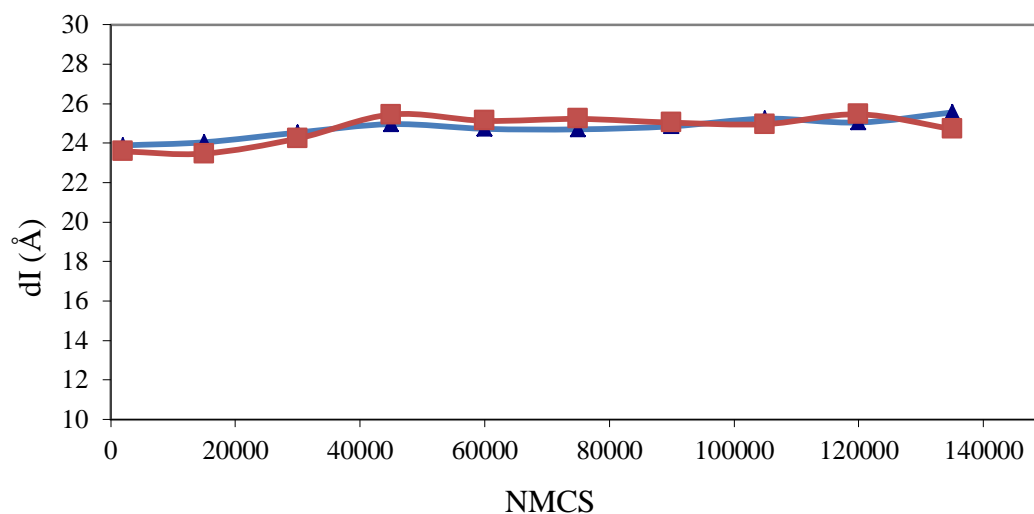


Figure E.2.2 Peptide end-to-end distance vs. the number of Monte Carlo steps. Two peptides in explicit water between two surfaces of alternating polarity.

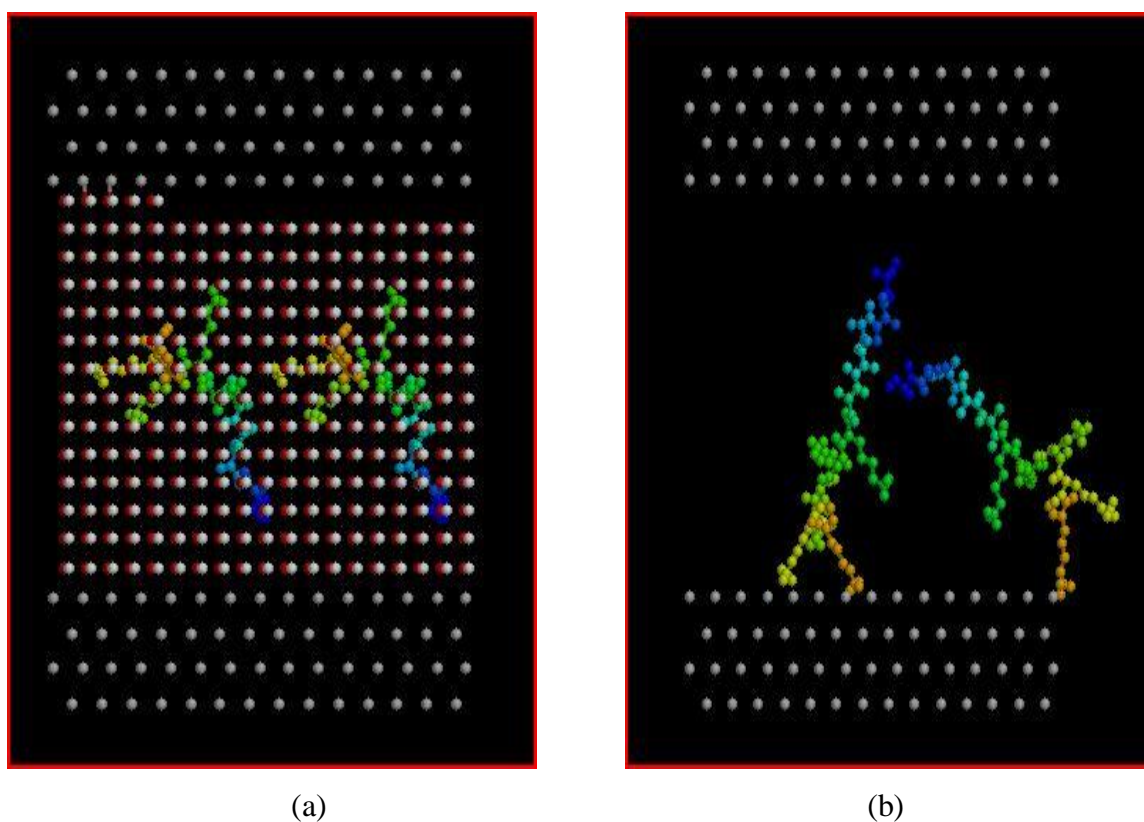


Figure E.2.3 RasMol "Ball & Stick" graphic view of initial and final system configuration. Two peptides in explicit water between two surfaces of alternating polarity.

**BIBLIOGRAPHY**

- [1] Fink, A. L. Protein aggregation: folding aggregates, inclusion bodies and amyloid. *Folding & Design* 3:R9-R23; 1998.
- [2] True, H. L., Lindquist, S. L. A yeast prion provides a mechanism for genetic variation and phenotypic diversity. *Nature*. 407:477-483; 2000.
- [3] Dobson, C. M. Principles of protein folding, misfolding and aggregation. *Seminars in Cell & Dev. Biol.* 15:3-16; 2004.
- [4] Feng, Y. W., Ooishi, A., Honda, S. Aggregation factor analysis for protein formulation by a systematic approach using FTIR, SEC and design of experiments techniques. *J. of Pharm. and Biomed. Anal.* 57:143-152; 2012.
- [5] Cellar, T., Douma, R., Huebner, A., Prausnitz, J.; Blanch, H. Kinetic studies of protein L aggregation and disaggregation. *Biophys. Chem.* 125:350-359; 2007.
- [6] Chi, E. Y., Krishnan, S.; Randolph, T. W., Carpenter, J. F. Physical Stability of Proteins in Aqueous Solution: Mechanism and Driving Forces in Nonnative Protein Aggregation. *Pharmaceutical Res.* 20:1325-1336; 2003.
- [7] Taylor, D. R., Hooper, N. M. Role of lipid rafts in the processing of the pathogenic prion and Alzheimer's amyloid- $\beta$  proteins. *Seminars in Cell & Dev. Biol.* 18:638-648; 2007.
- [8] Giacomelli, C. E., Norde, W. Conformational Changes of the Amyloid  $\beta$ -Peptide (1-40) Adsorbed on Solid Surfaces. *Macromol. Biosci.* 5:401-407; 2005.
- [9] Aisenbrey, C., Borowik, T., Bystrom, R., Bokvist, M., Lindstrom, F., Misiak, H., Sani, M.A., Grobner, G. How is protein aggregation in amyloidogenic diseases modulated by biological membranes? *Eur. Biophys. J.* 37:247-255; 2008.
- [10] Wang, W., Nema, S., Teagarden, D. Protein aggregation - Pathways and influencing factors. *Intern. J. Pharmaceutics* 390:89-99; 2010.
- [11] Sluzky, V., Tamada, J. A., Klibanov, A. M., Langer, R. Kinetics of insulin aggregation in aqueous solutions upon agitation in the presence of hydrophobic surfaces. *Appl. Biol. Sci.* 88:9377-9381; 1991.
- [12] McMasters, M. J., Hammer, R. P., McCarley, R. L. Surface-Induced Aggregation of Beta Amyloid Peptide by  $\omega$ -Substituted Alkanethiol Monolayers Supported on Gold. *Langmuir* 21:4464-4470; 2004.



- [13] Radford, S. E. Protein folding: Progress made and promises ahead. *TIBS* 25:611-618; 2000.
- [14] Karplus, M. The Levinthal paradox: yesterday and today. *Folding and design* 2:S69-S75; 1997.
- [15] Roberts, C. J. Non-Native Protein Aggregation Kinetics. *Biotechnology and bioengineering* 98:927-938; 2007.
- [16] Kunal, M.N. Formation of amyloid fibrils by diblock peptides. M.S. Thesis, Missouri S&T, 2012.
- [17] Matysiak, S., Clementi, C. Optimal combination of theory and experiment for the characterization of the protein folding landscape of S6: How far can a minimalist model go? *J. Mol. Biol.* 343:235-248; 2004.
- [18] Song, D., Forciniti, D. Monte Carlo simulations of peptide adsorption on solid surfaces. *J. of Chem. Phys.* 115:8089-8100; 2001.
- [19] Mungikar, A., Forciniti, D. Molecular Simulations and Neutron Reflectivity studies of Peptides at Solid/liquid Interfaces. *ChemPhysChem (Angewandte Chemie)*, 3: 993-999; 2002.
- [20] Mungikar, A., Forciniti, D. Conformational Changes of Peptides at Solid/Liquid Interfaces: A Monte Carlo Study. *Biomacromolecules*, 5, 2147-2159; 2004.
- [21] Mungikar, A., Forciniti, D. Effect of co-solvents on the adsorption of peptides at the solid-liquid interface. *Biomacromolecules*, 7: 239-251; 2006.
- [22] Takada, S., Luthey-Schulten, Z., Wolynes, P. G. Folding dynamics with nonadditive forces: A simulation study of a designed helical protein and a random heteropolymer. *J. of chem. Phys.* 110:11616-11629; 1999.

## VITA

Rehman Fazeem was born in Kannur, in the state of Kerala, India. He completed bachelors degree in Chemical Engineering from University of Kerala, Trivandrum, India, in May 2007. The same year in September he joined Engineers India Ltd. (EIL), New Delhi, as an Engineer in their Heat and Mass Transfer Division. In EIL, Rehman worked on designing and engineering processes and equipment for off-shore/on-shore gas processing facilities.

After resigning from EIL, Rehman joined Missouri S & T (MST), Rolla, MO, USA, in January 2011 to pursue M.S. degree in Chemical Engineering. He worked as a Graduate Assistant in MST, contributing in teaching and researching.

During M.S. degree, Rehman also worked as an intern in Mynah Technologies LLC, Chesterfield, MO, USA. For nine months in Mynah Technologies, starting August 2012, he worked on Control System Testing and Operator Training using real-time dynamic simulation platform of MiMiC software.